# Table of Contents

## Xamarin Controls (Home Page)


**Grid** ..... 17

- Installation and Demos ..... 19
  - Create a Solution with the DevExpress Grid Component in Visual Studio for Mac ..... 20
  - Create a Solution with the DevExpress Grid Component in Visual Studio ..... 26
  - Grid Demos ..... 29
- Getting Started ..... 30
  - Lesson 1 - Bind a Grid to Data ..... 31
  - Lesson 2 - Create Grid Columns ..... 38
  - Lesson 3 - Add New Item Row ..... 41
  - Lesson 4 - Group Data ..... 43
  - Lesson 5 - Calculate Summaries ..... 45
  - Lesson 6 - Sort Data ..... 48
  - Lesson 7 - Filter Data ..... 51
- Grid Elements ..... 53
  - Columns ..... 54
    - Columns Overview ..... 55
    - Identify and Access Columns ..... 57
    - Create Columns and Bind Them to Data Fields ..... 58
    - Unbound Columns ..... 65
    - Fixed Columns ..... 70
  - Rows ..... 72
    - Rows Overview ..... 73
    - Identify and Access Rows ..... 75
  - Cells ..... 77
- Themes ..... 78
- Conditional Formatting ..... 79
  - Predefined Format Names ..... 83
- End-User Interaction ..... 85
  - Column Chooser ..... 86
  - Row Edit Modes ..... 87
  - End-User Restrictions ..... 93
- Examples ..... 95
  - Conditional Formatting ..... 96
    - How to: Highlight Top Cell Values in a Column ..... 97
    - How to: Apply a Color Scale Format to a Column ..... 100
    - How to: Apply a Data Bar Format to a Column ..... 103
    - How to: Apply an Icon Set Format to a Column ..... 106
    - How to: Apply a Format Based on an Expression ..... 109
    - How to: Highlight Column Values that Match a Condition ..... 112
    - How to: Set Conditional Formatting Rules in XAML ..... 115
  - Themes ..... 116
    - How to: Apply Themes ..... 117
    - How to: Customize Themes ..... 118
    - How to: Customize Refresh Indicators ..... 121
  - How to: Create Grid Columns for Data of Different Types ..... 123
  - How to: Implement a PickerColumn ..... 129
  - How to: Bind a GridControl to a DataTable ..... 136
  - How to: Change Cell Appearance ..... 140
  - How to: Implement Swipe Buttons for Data Rows ..... 142
  - How to: Implement Pull-to-Refresh ..... 144
  - How to: Implement Load-More ..... 148
  - How to: Customize Popup Menu ..... 153


*© 2018 Developer Express Inc.*

How to: Implement a Custom Row Edit Form ... 155
How to: Format Column Values Using Standard Format Strings ... 163
How to: Implement the Search Functionality ... 164
How to: Use a TemplateColumn for Showing and Editing Time Values ... 168
How to: Check a SwitchColumn by a Single Tap ... 171
Tips and Tricks ... 173
Build Android Applications for Release ... 174
Density-Independent Pixels ... 175

API Reference ... 176
DevExpress.Data ... 176
AutoFilterCondition Enumeration ... 177
ColumnFilterMode Enumeration ... 178
ColumnGroupInterval Enumeration ... 179
ColumnSortMode Enumeration ... 181
ColumnSortOrder Enumeration ... 182
UnboundColumnType Enumeration ... 183
DevExpress.Export ... 184
CustomizeCellEventArgs Class ... 186
CustomizeCellEventArgs Members ... 187
CustomizeCellEventArgs Constructor ... 189
CustomizeCellEventArgs Properties ... 190
ColumnFieldName Property ... 192
Formatting Property ... 193
Hyperlink Property ... 194
Value Property ... 195
CustomizeCellEventHandler Delegate ... 196
CustomizeDataAreaEventArgs Class ... 197
CustomizeDataAreaEventArgs Members ... 198
CustomizeDataAreaEventArgs Constructor ... 199
CustomizeDataAreaEventArgs Properties ... 200
Context Property ... 201
CustomizeDataAreaEventHandler Delegate ... 202
CustomizeFooterEventArgs Class ... 203
CustomizeFooterEventArgs Members ... 204
CustomizeFooterEventArgs Constructor ... 205
CustomizeFooterEventArgs Properties ... 206
Context Property ... 207
CustomizeFooterEventHandler Delegate ... 208
CustomizeHeaderEventArgs Class ... 209
CustomizeHeaderEventArgs Members ... 210
CustomizeHeaderEventArgs Constructor ... 211
CustomizeHeaderEventArgs Properties ... 212
Context Property ... 213
CustomizeHeaderEventHandler Delegate ... 214
CustomizeSheetEventArgs Class ... 215
CustomizeSheetEventArgs Members ... 216
CustomizeSheetEventArgs Constructor ... 217
CustomizeSheetEventArgs Properties ... 218
Context Property ... 219
CustomizeSheetEventHandler Delegate ... 220
ExportTarget Enumeration ... 221
GroupState Enumeration ... 222
DevExpress.Mobile.Core ... 223
VisibilityState Enumeration ... 224
DevExpress.Mobile.Core.ConditionalFormatting ... 225
ConditionRule Enumeration ... 226
TopBottomRule Enumeration ... 227

© 2018 Developer Express Inc.

DevExpress.Mobile.DataGrid ... 228
AutoGenerateColumnsMode Enumeration ... 232
AutoGeneratingColumnEventArgs Class ... 233
AutoGeneratingColumnEventArgs Members ... 234
AutoGeneratingColumnEventArgs Constructor ... 235
AutoGeneratingColumnEventArgs Properties ... 236
Column Property ... 237
AutoGeneratingColumnEventHandler Delegate ... 238
CellData Class ... 239
CellData Members ... 240
CellData Constructor ... 241
CellData Properties ... 242
DisplayText Property ... 243
Index Property ... 244
IsSelected Property ... 245
Source Property ... 246
Value Property ... 253
CellIndex Structure ... 254
CellIndex Members ... 255
CellIndex Constructor ... 256
CellIndex Properties ... 257
FieldName Property ... 258
InvalidIndex Property ... 259
IsValid Property ... 260
RowHandle Property ... 261
CellIndex Methods ... 262
Equals Method ... 263
Equals Method (object) ... 263
GetHashCode Method ... 264
ColorScaleFormatCondition Class ... 265
ColorScaleFormatCondition Members ... 268
ColorScaleFormatCondition Constructor ... 271
ColorScaleFormatCondition Properties ... 272
Format Property ... 275
ColumnContentAlignment Enumeration ... 276
CustomizeCellDisplayTextEventArgs Class ... 277
CustomizeCellDisplayTextEventArgs Members ... 278
CustomizeCellDisplayTextEventArgs Constructor ... 279
CustomizeCellDisplayTextEventArgs Properties ... 280
DisplayText Property ... 281
Index Property ... 282
Source Property ... 283
Value Property ... 284
CustomizeCellDisplayTextEventHandler Delegate ... 285
CustomizeCellEventArgs Class ... 286
CustomizeCellEventArgs Members ... 290
CustomizeCellEventArgs Properties ... 291
BackgroundColor Property ... 292
DisplayText Property ... 293
FieldName Property ... 294
ForeColor Property ... 297
Handled Property ... 300
IsSelected Property ... 303
RowHandle Property ... 306
Value Property ... 307
CustomizeCellEventHandler Delegate ... 311
CustomSummaryEventArgs Class ... 312
CustomSummaryEventArgs Members ... 313

© 2018 Developer Express Inc.

CustomSummaryEventArgs Constructor ... 315
CustomSummaryEventArgs Properties ... 316
FieldName Property ... 318
FieldValue Property ... 319
GroupRowHandle Property ... 321
IsGroupSummary Property ... 322
IsTotalSummary Property ... 323
Row Property ... 324
SummaryProcess Property ... 325
TotalValue Property ... 327
TotalValueReady Property ... 328
CustomSummaryEventArgs Methods ... 329
GetValue Method ... 330
CustomSummaryEventHandler Delegate ... 331
CustomSummaryProcess Enumeration ... 332
DataBarFormatCondition Class ... 333
DataBarFormatCondition Members ... 336
DataBarFormatCondition Constructor ... 339
DataBarFormatCondition Properties ... 340
Format Property ... 343
DateColumn Class ... 344
DateColumn Members ... 347
DateColumn Constructor ... 352
DateColumn Properties ... 353
IsNullable Property ... 358
EditingRowAction Enumeration ... 359
EditValuesContainer Class ... 360
EditValuesContainer Members ... 368
EditValuesContainer Constructor ... 369
EditValuesContainer Properties ... 370
IsNewRow Property ... 371
RowData Property ... 379
Values Property ... 380
EditValuesContainer Events ... 389
PropertyChanged Event ... 390
EditValuesContainer Methods ... 391
ContainsValueForField Method ... 392
GetBindingPath Method ... 393
Raise Method ... 402
ExpressionConditionBase Class ... 403
ExpressionConditionBase Members ... 404
ExpressionConditionBase Properties ... 406
ApplyToRow Property ... 408
Expression Property ... 411
Format Property ... 414
FixedStyle Enumeration ... 415
FormatCondition Class ... 416
FormatCondition Members ... 419
FormatCondition Constructor ... 422
FormatCondition Properties ... 423
Value1 Property ... 426
Value2 Property ... 429
ValueRule Property ... 430
FormatConditionBase Class ... 433
FormatConditionBase Members ... 436
FormatConditionBase Properties ... 438
FieldName Property ... 440
Owner Property ... 442

© 2018 Developer Express Inc.

PredefinedFormatName Property ... 443
TypeName Property ... 445
FormatConditionCollection Class ... 446
FormatConditionCollection Members ... 448
FormatConditionCollection Constructor ... 450
FormatConditionCollection Properties ... 451
Owner Property ... 453
FormatConditionCollection Methods ... 454
GetInfoByFieldName Method ... 456
GridColumn Class ... 457
GridColumn Members ... 458
GridColumn Constructor ... 462
GridColumn Properties ... 463
ActualCaption Property ... 467
ActualWidth Property ... 468
AllowAutoFilter Property ... 469
AllowGroup Property ... 470
AllowSort Property ... 471
AutoFilterCondition Property ... 472
AutoFilterValue Property ... 473
Caption Property ... 474
ColumnFilterMode Property ... 477
ContentAlignment Property ... 478
DisplayFormat Property ... 479
FieldName Property ... 481
FixedStyle Property ... 484
GroupInterval Property ... 485
HeaderTemplate Property ... 486
ImmediateUpdateAutoFilter Property ... 487
IsGrouped Property ... 488
IsReadOnly Property ... 489
IsUnbound Property ... 490
IsVisible Property ... 491
MinWidth Property ... 492
SerializationTypeName Property ... 493
SortIndex Property ... 494
SortMode Property ... 495
SortOrder Property ... 496
UnboundExpression Property ... 497
UnboundType Property ... 500
Width Property ... 503
GridColumnCollection Class ... 504
GridColumnCollection Members ... 507
GridColumnCollection Constructor ... 509
GridColumnCollection Properties ... 510
Item Property ... 512
Item Property [string] ... 512
GridColumnCollection Events ... 513
ColumnPropertyChanged Event ... 514
GridColumnCollection Methods ... 515
GetColumnByFieldName Method ... 517
GridColumnDataEventArgs Class ... 518
GridColumnDataEventArgs Members ... 519
GridColumnDataEventArgs Properties ... 520
Column Property ... 521
EditableRowData Property ... 523
IsGetData Property ... 524
IsSetData Property ... 526

© 2018 Developer Express Inc.

RowData Property ... 527
Source Property ... 529
Value Property ... 531
GridColumnDataEventHandler Delegate ... 534
GridColumnSummary Class ... 535
GridColumnSummary Members ... 539
GridColumnSummary Constructor ... 541
GridColumnSummary Constructor ... 542
GridColumnSummary Constructor (string) ... 543
GridColumnSummary Properties ... 544
DisplayFormat Property ... 546
FieldName Property ... 550
IgnoreNullValues Property ... 553
IsValid Property ... 554
SummaryValue Property ... 555
Type Property ... 556
GridColumnSummary Methods ... 559
Clone Method ... 560
GetSummaryValueAsync Method ... 561
Reset Method ... 562
GridControl Class ... 563
GridControl Members ... 570
GridControl Constructor ... 584
GridControl Properties ... 585
ActualFilterString Property ... 599
AllowDeleteRows Property ... 600
AllowEditRows Property ... 601
AllowGroup Property ... 602
AllowGroupCollapse Property ... 603
AllowHorizontalScrollingVirtualization Property ... 604
AllowResizeColumns Property ... 605
AllowSort Property ... 606
AutoFilterPanelHeight Property ... 607
AutoFilterPanelVisibility Property ... 608
AutoGenerateColumnsMode Property ... 610
ColumnHeadersHeight Property ... 611
ColumnHeadersVisibility Property ... 612
Columns Property ... 613
ColumnsAutoWidth Property ... 620
EditFormContent Property ... 621
FilterExpression Property ... 630
FilterPanelHeight Property ... 632
FilterPanelVisibility Property ... 633
FilterString Property ... 634
FormatConditions Property ... 635
GroupsInitiallyExpanded Property ... 637
GroupSummaries Property ... 638
HighlightMenuTargetElements Property ... 642
HorizontalScrollOffset Property ... 645
IsColumnChooserEnabled Property ... 646
IsColumnMenuEnabled Property ... 648
IsGrouped Property ... 649
IsGroupRowMenuEnabled Property ... 650
IsLoadedInVisualTree Property ... 651
IsLoadMoreEnabled Property ... 652
IsPullToRefreshEnabled Property ... 656
IsReadOnly Property ... 661
IsRowCellMenuEnabled Property ... 662

© 2018 Developer Express Inc.

IsTotalSummaryMenuEnabled Property ... 663
ItemsSource Property ... 664
LeftSwipeButtons Property ... 665
LoadMoreCommand Property ... 667
NewItemRowVisibility Property ... 671
OptionsExportCsv Property ... 673
OptionsExportXls Property ... 674
OptionsExportXlsx Property ... 675
PullToRefreshCommand Property ... 676
RightSwipeButtons Property ... 681
RowCount Property ... 683
RowEditMode Property ... 684
RowHeight Property ... 685
RowTapCommand Property ... 686
SelectedDataObject Property ... 687
SelectedRowHandle Property ... 688
SortMode Property ... 689
SwipeButtonCommand Property ... 690
TotalSummaries Property ... 691
TotalSummaryHeight Property ... 695
TotalSummaryVisibility Property ... 696
VisibleColumns Property ... 697
GridControl Events ... 698
AutoGeneratingColumn Event ... 700
CalculateCustomSummary Event ... 701
CustomizeCell Event ... 703
CustomizeCellDisplayText Event ... 707
CustomUnboundColumnData Event ... 708
EndRowEdit Event ... 711
GroupRowCollapsed Event ... 712
GroupRowCollapsing Event ... 713
GroupRowExpanded Event ... 714
GroupRowExpanding Event ... 715
HorizontalScrollOffsetChanged Event ... 716
InitNewRow Event ... 717
LoadMore Event ... 718
PopupMenuCustomization Event ... 721
PullToRefresh Event ... 724
RowTap Event ... 725
SelectionChanged Event ... 726
SwipeButtonClick Event ... 727
SwipeButtonShowing Event ... 729
GridControl Methods ... 731
ClearFilter Method ... 735
ClearGrouping Method ... 736
ClearSorting Method ... 737
CloseEditor Method ... 738
CollapseAllGroups Method ... 739
CollapseGroupRow Method ... 740
DeleteRow Method ... 741
Dispose Method ... 744
ExpandAllGroups Method ... 745
ExpandGroupRow Method ... 746
ExportToExcel Method ... 747
FindRow Method ... 748
FindRowByValue Method ... 749
FindRowByValue Method (GridColumn, object) ... 749
FindRowByValue Method (string, object) ... 749

© 2018 Developer Express Inc.

GetCellDisplayText Method ... 751
GetCellDisplayText Method (int, GridColumn) ... 751
GetCellDisplayText Method (int, string) ... 751
GetCellValue Method ... 753
GetCellValue Method (int, string) ... 753
GetCellValue Method (int, GridColumn) ... 755
GetGroupInfo Method ... 756
GetGroupRowValue Method ... 757
GetRow Method ... 758
GetRow Method (int) ... 758
GetRowHandle Method ... 759
GetService Method ... 760
GetService<T> Method ... 761
GetSourceRowIndex Method ... 762
GroupBy Method ... 763
GroupBy Method (GridColumn) ... 763
GroupBy Method (string) ... 763
HideColumnChooser Method ... 765
HideColumnChooser Method ... 765
HideColumnChooser Method (bool) ... 767
IsGroupCollapsed Method ... 770
IsGroupRow Method ... 771
LoadInVisualTree Method ... 772
OpenEditor Method ... 773
Redraw Method ... 774
RefreshData Method ... 775
RestoreLayoutFromStream Method ... 776
RestoreLayoutFromXml Method ... 777
SaveLayoutToStream Method ... 778
SaveLayoutToXml Method ... 779
ScrollToRow Method ... 780
SetCellValue Method ... 781
SetCellValue Method (int, GridColumn, object) ... 781
SetCellValue Method (int, string, object) ... 781
ShowColumnChooser Method ... 783
SortBy Method ... 785
SortBy Method (GridColumn) ... 785
SortBy Method (string) ... 786
SortBy Method (GridColumn, ColumnSortOrder) ... 786
SortBy Method (GridColumn, ColumnSortOrder, int) ... 787
GridPopupMenuType Enumeration ... 788
GridSortMode Enumeration ... 789
HorizontalScrollOffsetEventArgs Class ... 790
HorizontalScrollOffsetEventArgs Members ... 791
HorizontalScrollOffsetEventArgs Constructor ... 792
HorizontalScrollOffsetEventArgs Properties ... 793
HorizontalOffset Property ... 794
HorizontalScrollOffsetEventHandler Delegate ... 795
IconSetFormatCondition Class ... 796
IconSetFormatCondition Members ... 799
IconSetFormatCondition Constructor ... 802
IconSetFormatCondition Properties ... 803
Format Property ... 806
IEditableRowData Interface ... 807
IEditableRowData Members ... 808
IEditableRowData Methods ... 809
SetFieldValue Method ... 810
IGroupInfo Interface ... 811

© 2018 Developer Express Inc.

IGroupInfo Members ... 812
IGroupInfo Properties ... 813
FieldName Property ... 814
FirstRowHandle Property ... 815
GroupRowHandle Property ... 816
IsCollapsed Property ... 817
LastRowHandle Property ... 818
RowCount Property ... 819
Summaries Property ... 820
Value Property ... 821
ImageColumn Class ... 822
ImageColumn Members ... 828
ImageColumn Constructor ... 833
IndicatorFormatConditionBase Class ... 834
IndicatorFormatConditionBase Members ... 835
IndicatorFormatConditionBase Properties ... 837
MaxValue Property ... 839
MinValue Property ... 840
SelectiveExpression Property ... 841
InitNewRowEventArgs Class ... 842
InitNewRowEventArgs Members ... 843
InitNewRowEventArgs Constructor ... 844
InitNewRowEventArgs Properties ... 845
EditableRowData Property ... 846
InitNewRowEventHandler Delegate ... 847
IRowData Interface ... 848
IRowData Members ... 849
IRowData Properties ... 850
DataObject Property ... 851
RowHandle Property ... 852
IRowData Methods ... 854
GetFieldValue Method ... 855
GetFieldValueGeneric<T> Method ... 856
NumberColumn Class ... 857
NumberColumn Members ... 860
NumberColumn Constructor ... 865
PickerColumn Class ... 866
PickerColumn Members ... 873
PickerColumn Constructor ... 879
PickerColumn Properties ... 880
DisplayMember Property ... 886
ItemsSource Property ... 893
ValueMember Property ... 900
PopupMenuEventArgs Class ... 907
PopupMenuEventArgs Members ... 909
PopupMenuEventArgs Constructor ... 910
PopupMenuEventArgs Properties ... 911
Column Property ... 912
Menu Property ... 913
MenuType Property ... 915
RowHandle Property ... 917
PopupMenuEventHandler Delegate ... 918
RowAllowEventArgs Class ... 919
RowAllowEventArgs Members ... 920
RowAllowEventArgs Constructor ... 921
RowAllowEventArgs Properties ... 922
Allow Property ... 923
RowAllowEventHandler Delegate ... 924

© 2018 Developer Express Inc.

RowEditingEventArgs Class ... 925
RowEditingEventArgs Members ... 926
RowEditingEventArgs Constructor ... 927
RowEditingEventArgs Properties ... 928
Action Property ... 929
RowEditingEventHandler Delegate ... 930
RowEditMode Enumeration ... 931
RowEventArgs Class ... 933
RowEventArgs Members ... 934
RowEventArgs Constructor ... 935
RowEventArgs Properties ... 936
RowHandle Property ... 937
RowEventHandler Delegate ... 939
RowTapEventArgs Class ... 940
RowTapEventArgs Members ... 941
RowTapEventArgs Constructor ... 942
RowTapEventArgs Properties ... 943
FieldName Property ... 944
RowTapEventHandler Delegate ... 945
SourceRowEventArgs Class ... 946
SourceRowEventArgs Members ... 947
SourceRowEventArgs Constructor ... 948
SourceRowEventArgs Properties ... 949
SourceRowIndex Property ... 950
SourceRowEventHandler Delegate ... 951
SummaryType Enumeration ... 952
SwipeButtonEventArgs Class ... 953
SwipeButtonEventArgs Members ... 955
SwipeButtonEventArgs Constructor ... 956
SwipeButtonEventArgs Properties ... 957
ButtonInfo Property ... 958
SwipeButtonEventHandler Delegate ... 960
SwipeButtonInfo Class ... 961
SwipeButtonInfo Members ... 963
SwipeButtonInfo Constructor ... 964
SwipeButtonInfo Properties ... 965
AutoCloseOnTap Property ... 966
BackgroundColor Property ... 967
ButtonName Property ... 969
Caption Property ... 971
TextColor Property ... 973
Width Property ... 974
SwipeButtonShowingEventArgs Class ... 976
SwipeButtonShowingEventArgs Members ... 978
SwipeButtonShowingEventArgs Constructor ... 979
SwipeButtonShowingEventArgs Properties ... 980
IsVisible Property ... 981
SwipeButtonShowingEventHandler Delegate ... 983
SwitchColumn Class ... 984
SwitchColumn Members ... 987
SwitchColumn Constructor ... 993
SwitchColumn Properties ... 994
TextOff Property ... 1000
TextOn Property ... 1002
TextUndefined Property ... 1004
TemplateColumn Class ... 1005
TemplateColumn Members ... 1012
TemplateColumn Constructor ... 1018

© 2018 Developer Express Inc.

TemplateColumn Properties ... 1019
DisplayTemplate Property ... 1025
EditTemplate Property ... 1031
TextColumn Class ... 1032
TextColumn Members ... 1035
TextColumn Constructor ... 1040
ThemeFontAttributes Class ... 1041
ThemeFontAttributes Members ... 1044
ThemeFontAttributes Constructor ... 1045
ThemeFontAttributes Properties ... 1048
Attributes Property ... 1049
Color Property ... 1050
Family Property ... 1051
FontObject Property ... 1052
Size Property ... 1053
ThemeFontAttributes Methods ... 1054
ApplyToLabel Method ... 1055
Equals Method ... 1056
Equals Method (object) ... 1056
FontSizeFromNamedSize Method ... 1057
GetHashCode Method ... 1060
TopBottomRuleFormatCondition Class ... 1061
TopBottomRuleFormatCondition Members ... 1064
TopBottomRuleFormatCondition Constructor ... 1067
TopBottomRuleFormatCondition Properties ... 1068
Rule Property ... 1071
Threshold Property ... 1074
DevExpress.Mobile.DataGrid.Theme ... 1077
DarkTheme Class ... 1079
DarkTheme Members ... 1080
DarkTheme Constructor ... 1082
IAndroidRefreshCustomizer Interface ... 1083
IAndroidRefreshCustomizer Members ... 1085
IAndroidRefreshCustomizer Properties ... 1086
BarColor Property ... 1087
IAutoFilterPanelCustomizer Interface ... 1089
IAutoFilterPanelCustomizer Members ... 1090
IAutoFilterPanelCustomizer Properties ... 1091
BackgroundColor Property ... 1092
BorderColor Property ... 1093
IAutoFilterPanelCustomizer Methods ... 1094
LocateContentInPanel Method ... 1095
ICellCustomizer Interface ... 1096
ICellCustomizer Members ... 1097
ICellCustomizer Properties ... 1098
BackgroundColor Property ... 1099
BorderColor Property ... 1100
Font Property ... 1101
HighlightColor Property ... 1102
Padding Property ... 1106
SelectionColor Property ... 1107
SelectionFontColor Property ... 1108
ICellCustomizer Methods ... 1109
CreateCheckedImageSource Method ... 1110
CreateUnassignedImageSource Method ... 1111
CreateUncheckedImageSource Method ... 1112
LocateCellViewInRow Method ... 1113
IColumnChooserCustomizer Interface ... 1114

© 2018 Developer Express Inc.

IColumnChooserCustomizer Members ... 1115
IColumnChooserCustomizer Properties ... 1116
LabelFont Property ... 1117
IDialogFormCustomizer Interface ... 1118
IDialogFormCustomizer Members ... 1119
IDialogFormCustomizer Properties ... 1120
BackgroundColor Property ... 1121
CaptionBackgroundColor Property ... 1122
CaptionFont Property ... 1123
FullScreenBackgroundColor Property ... 1124
FullScreenBorderColor Property ... 1125
FullScreenCaptionFont Property ... 1126
PanelPadding Property ... 1127
IEditorCustomizer Interface ... 1128
IEditorCustomizer Members ... 1129
IEditorCustomizer Properties ... 1130
EditorBackgroundColor Property ... 1131
Font Property ... 1132
FormEditorsFont Property ... 1133
UnfocusedCellHighlighting Property ... 1134
IEditorCustomizer Methods ... 1135
LocateCellViewInRow Method ... 1136
IFilterPanelCustomizer Interface ... 1137
IFilterPanelCustomizer Members ... 1138
IFilterPanelCustomizer Properties ... 1139
BackgroundColor Property ... 1140
BorderColor Property ... 1141
Font Property ... 1142
PanelPadding Property ... 1143
IFilterPanelCustomizer Methods ... 1144
CreateClearFilterImageSource Method ... 1145
LocateContentInPanel Method ... 1146
IGridControlCustomizer Interface ... 1147
IGridControlCustomizer Members ... 1148
IGridControlCustomizer Properties ... 1149
BackgroundColor Property ... 1150
BorderColor Property ... 1151
BorderThickness Property ... 1152
PopupMenuHighlightColor Property ... 1153
IGroupCustomizer Interface ... 1154
IGroupCustomizer Members ... 1155
IGroupCustomizer Properties ... 1156
BackgroundColor Property ... 1157
Font Property ... 1158
Padding Property ... 1159
IGroupCustomizer Methods ... 1160
CreateArrowRightImageSource Method ... 1161
LocateViewInRow Method ... 1162
IHeaderCustomizer Interface ... 1163
IHeaderCustomizer Members ... 1164
IHeaderCustomizer Properties ... 1165
BackgroundColor Property ... 1166
BorderColor Property ... 1167
Font Property ... 1168
Padding Property ... 1169
IHeaderCustomizer Methods ... 1170
CreateArrowDownImageSource Method ... 1171
CreateArrowNoneImageSource Method ... 1172

© 2018 Developer Express Inc.

CreateArrowUpImageSource Method ... 1173
LocateHeaderViewInPanel Method ... 1174
IIOSRefreshCustomizer Interface ... 1175
IIOSRefreshCustomizer Members ... 1177
IIOSRefreshCustomizer Properties ... 1178
RefreshColor Property ... 1179
ILoadMoreViewCustomizer Interface ... 1181
ILoadMoreViewCustomizer Members ... 1182
ILoadMoreViewCustomizer Properties ... 1183
ActivityViewColor Property ... 1184
BackgroundColor Property ... 1185
BorderColor Property ... 1186
Font Property ... 1187
ILoadMoreViewCustomizer Methods ... 1188
LocateViewInRow Method ... 1189
INewItemRowCustomizer Interface ... 1190
INewItemRowCustomizer Members ... 1193
INewItemRowCustomizer Properties ... 1194
BackgroundColor Property ... 1195
BorderColor Property ... 1198
Font Property ... 1201
INewItemRowCustomizer Methods ... 1204
LocateCellViewInRow Method ... 1205
ISaveCancelButtonsPanelCustomizer Interface ... 1206
ISaveCancelButtonsPanelCustomizer Members ... 1207
ISaveCancelButtonsPanelCustomizer Properties ... 1208
BackgroundColor Property ... 1209
BorderColor Property ... 1210
Font Property ... 1211
IThemeChangingHandler Interface ... 1212
IThemeChangingHandler Members ... 1213
IThemeChangingHandler Methods ... 1214
OnThemeChanged Method ... 1215
ITotalSummaryCustomizer Interface ... 1216
ITotalSummaryCustomizer Members ... 1219
ITotalSummaryCustomizer Properties ... 1220
BackgroundColor Property ... 1221
BorderColor Property ... 1224
Font Property ... 1227
Padding Property ... 1230
ITotalSummaryCustomizer Methods ... 1233
LocateSummaryViewInPanel Method ... 1234
IWinPhoneRefreshCustomizer Interface ... 1235
IWinPhoneRefreshCustomizer Members ... 1237
IWinPhoneRefreshCustomizer Properties ... 1238
BarColor Property ... 1239
LightTheme Class ... 1241
LightTheme Members ... 1242
LightTheme Constructor ... 1244
ThemeBase Class ... 1245
ThemeBase Members ... 1248
ThemeBase Properties ... 1250
AndroidRefreshCustomizer Property ... 1252
AutoFilterPanelCustomizer Property ... 1253
CellCustomizer Property ... 1254
ColumnChooserCustomizer Property ... 1257
DialogFormCustomizer Property ... 1258
EditorCustomizer Property ... 1259

© 2018 Developer Express Inc.

FilterPanelCustomizer Property ... 1260
GridControlCustomizer Property ... 1261
GroupCustomizer Property ... 1262
HeaderCustomizer Property ... 1263
IOSRefreshCustomizer Property ... 1266
LoadMoreViewCustomizer Property ... 1267
Name Property ... 1268
NewItemRowCustomizer Property ... 1269
SaveCancelButtonsPanelCustomizer Property ... 1272
TotalSummaryCustomizer Property ... 1273
WinPhoneRefreshCustomizer Property ... 1276
ThemeBase Methods ... 1277
CreateDefaultFont Method ... 1278
CreateDefaultFont Method (Color) ... 1278
CreateDefaultFont Method (Color, FontAttributes) ... 1278
CreateDefaultFontForFormEditors Method ... 1280
CreateDefaultPadding Method ... 1281
ThemeManager Class ... 1282
ThemeManager Members ... 1285
ThemeManager Properties ... 1286
Theme Property ... 1287
ThemeName Property ... 1290
ThemeManager Events ... 1291
ThemeChanged Event ... 1292
ThemeManager Methods ... 1293
AddThemeChangedHandler Method ... 1294
RefreshTheme Method ... 1295
Themes Class ... 1298
Themes Members ... 1301
Themes Constructor ... 1302
DevExpress.Mobile.Export ... 1303
CsvExportOptions Class ... 1304
CsvExportOptions Members ... 1305
CsvExportOptions Constructor ... 1307
CsvExportOptions Constructor ... 1308
CsvExportOptions Constructor (string, Encoding) ... 1309
CsvExportOptions Properties ... 1310
WritePreamble Property ... 1312
CsvExportOptions Events ... 1313
CustomizeCell Event ... 1314
CustomizeDataArea Event ... 1315
CustomizeFooter Event ... 1316
CustomizeHeader Event ... 1317
CustomizeSheet Event ... 1318
ExportProgress Event ... 1319
CsvExportOptions Methods ... 1320
Assign Method ... 1321
DataAwareExportOptionsBase Class ... 1322
DataAwareExportOptionsBase Members ... 1323
DataAwareExportOptionsBase Properties ... 1326
AllowGrouping Property ... 1329
AllowLookupValues Property ... 1330
AllowSortingAndFiltering Property ... 1331
GroupState Property ... 1332
ShowColumnHeaders Property ... 1333
ShowGroupSummaries Property ... 1334
ShowTotalSummaries Property ... 1335
DataAwareExportOptionsBase Events ... 1336

© 2018 Developer Express Inc.

CustomizeCell Event ... 1337
CustomizeDataArea Event ... 1338
CustomizeFooter Event ... 1339
CustomizeHeader Event ... 1340
CustomizeSheet Event ... 1341
ExportProgress Event ... 1342
DataAwareExportOptionsBase Methods ... 1343
Assign Method ... 1344
ExportOptionsBase Class ... 1345
ExportOptionsBase Members ... 1346
ExportOptionsBase Methods ... 1348
Assign Method ... 1349
TextExportOptionsBase Class ... 1350
TextExportOptionsBase Members ... 1351
TextExportOptionsBase Properties ... 1353
Encoding Property ... 1355
Separator Property ... 1356
TextExportOptionsBase Methods ... 1357
Assign Method ... 1358
XlsExportOptions Class ... 1359
XlsExportOptions Members ... 1360
XlsExportOptions Constructor ... 1364
XlsExportOptionsBase Class ... 1365
XlsExportOptionsBase Members ... 1366
XlsExportOptionsBase Properties ... 1368
ExportHyperlinks Property ... 1370
RawDataMode Property ... 1371
SheetName Property ... 1372
ShowGridLines Property ... 1373
XlsExportOptionsBase Methods ... 1374
Assign Method ... 1375
XlsxExportOptions Class ... 1376
XlsxExportOptions Members ... 1377
XlsxExportOptions Constructor ... 1381
DevExpress.Utils ... 1382
DefaultBoolean Enumeration ... 1383

© 2018 Developer Express Inc.

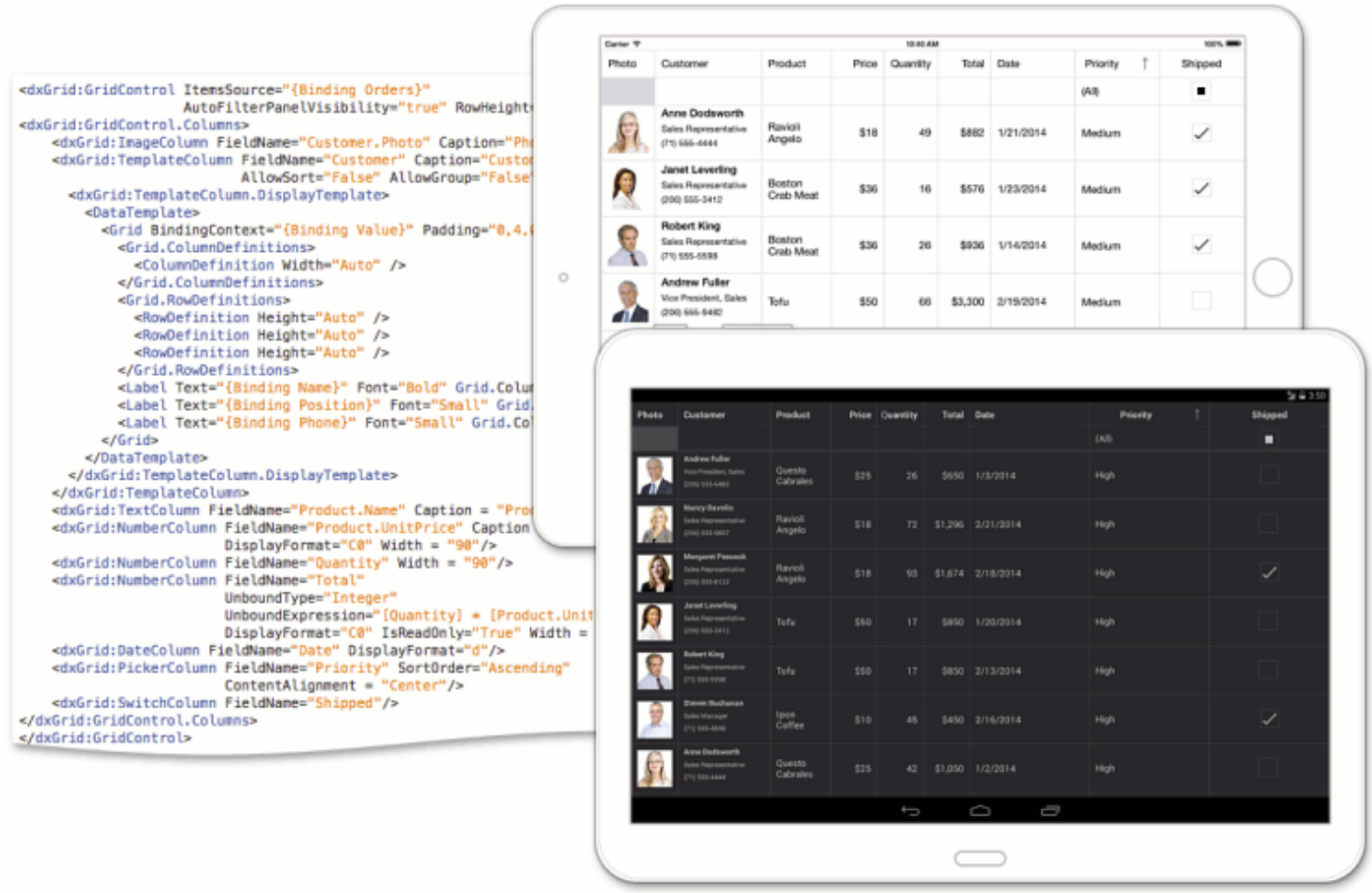
# Xamarin Controls

Xamarin Controls

### Overview

DevExpress controls for Xamarin.Forms allow you to easily extend the functionality of your iOS and Android applications created with Xamarin.Forms. With this technology and DevExpress controls, you can build feature-rich native mobile applications using a shared C# code and XAML markup.

The image below illustrates how the DevExpress Grid control defined in a shared XAML file is rendered for iOS and Android.




### System Requirements

You can create both iOS and Android applications with DevExpress UI controls for Xamarin.Forms on the OS X or Windows operating system. The following development tools are required :

- **OS X Development Machine**
  - Xamarin Studio
  - Xcode (for iOS development)
- **Windows Development Machine**
  - Xamarin Studio (allows Android development only) or Visual Studio supporting Xamarin development
  - An additional networked OS X machine (for iOS development)

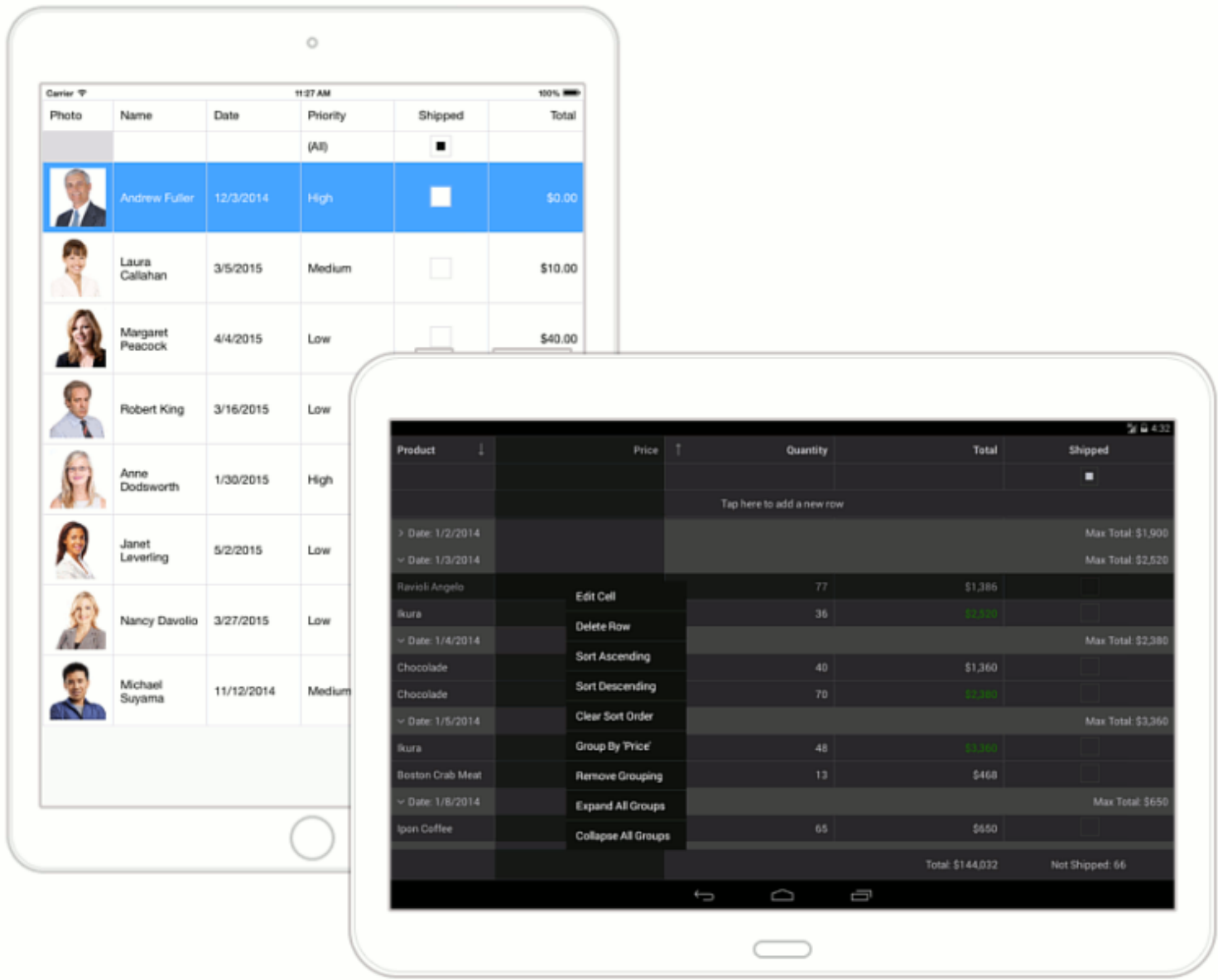
An Introduction to Xamarin.Forms contains information on the versions of supported operating systems and the development tools required to develop Xamarin.Forms applications.

© 2018 Developer Express Inc.

# Grid

Xamarin Controls > Grid

The DevExpress Grid for Xamarin is a powerful editing and data shaping component for working with tabular data on modern mobile devices that utilize the iOS or Android operating systems. It supports the Xamarin.Forms technology and allows you to create native mobile applications in C#. The GridControl enables a flexible UI with a wide-variety of features, allowing your end-users to easily and intuitively manage large amounts of data. It also exposes a comprehensive set of properties, methods and events that can help you develop feature-rich applications.




**Features'**

- Binding to various data sources
- Columns for displaying and editing data of different types (images, text, numbers, dates, etc.)
- Cell templates
- Auto generate columns for data source fields
- Calculated (unbound) columns
- Sorting (including sorting by multiple columns)
- Grouping
- Filtering (auto filter panel, multi-column filtering expressions)

© 2018 Developer Express Inc.

- Data summaries (total and group summaries, five predefined aggregate functions and special event to calculate summaries using custom rules)
- Pull-to-refresh functionality
- Load-more functionality
- Conditional formatting
- Custom cell appearances
- Columns resizing
- New item row
- In-place data editing
- Built-in popup menus
- Column chooser
- Swipe buttons
- Localization
- Theming
- Export to Excel
- Save and restore the grid's layout

**See Also**
Getting Started

© 2018 Developer Express Inc.

# Installation and Demos



This section describes how to download the DevExpress Grid component from the Xamarin Component Store and include it in your solution in Xamarin Studio or Visual Studio, and provides information on available demos demonstrating the Grid's main features. Topics in this section:

- Create a Solution with the DevExpress Grid Component in Visual Studio for Mac
- Create a Solution with the DevExpress Grid Component in Visual Studio
- Grid Demos

© 2018 Developer Express Inc.

# Create a Solution with the DevExpress Grid Component in Visual Studio for Mac

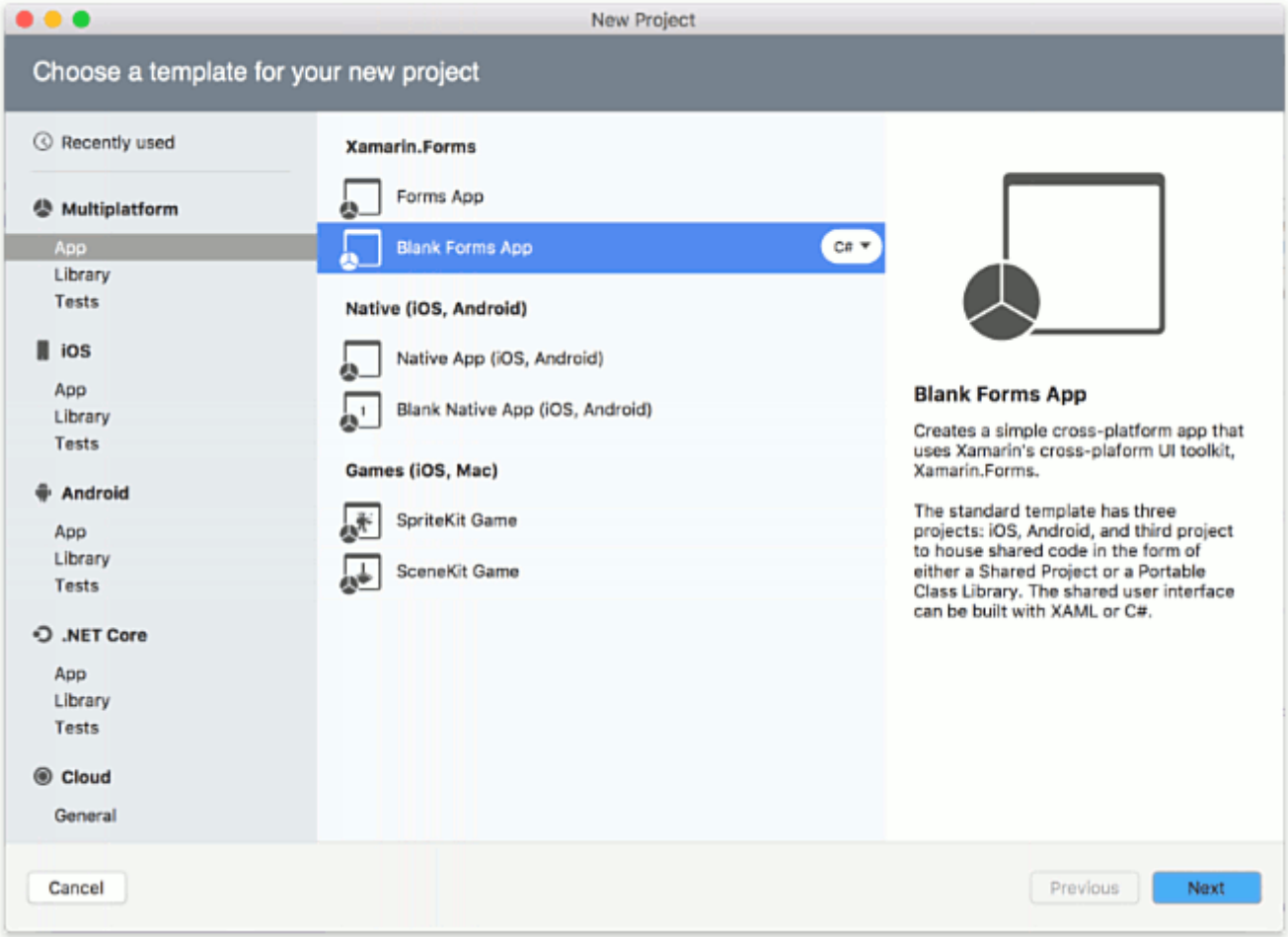


This topic explains how to create a new Xamarin.Forms cross-platform solution in Visual Studio for Mac, and how to add the DevExpress Grid component to your application using the DevExpress Grid NuGet package.


- Create a Solution
- Add the DevExpress Grid Component to a Solution
- Initialize the DevExpress Grid Component
- Add Namespaces
- Create a Grid Instance

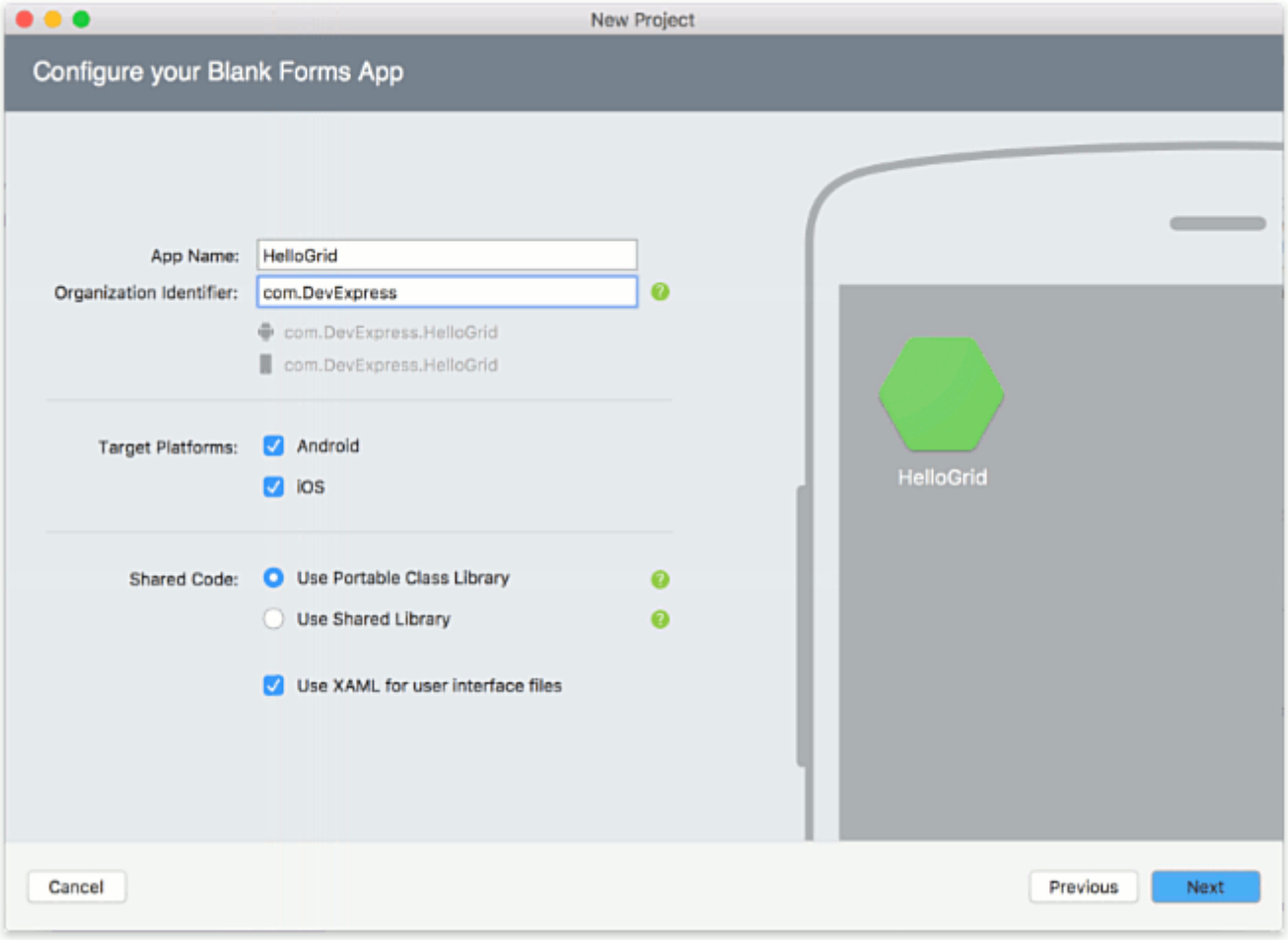

### Create a Solution

1. Start Visual Studio for Mac and select **File | New Solution...** in the main menu. In the invoked **New Project** dialog, switch to the **Multiplatform | App** section and select the **Blank Forms App** template. Click **Next**.



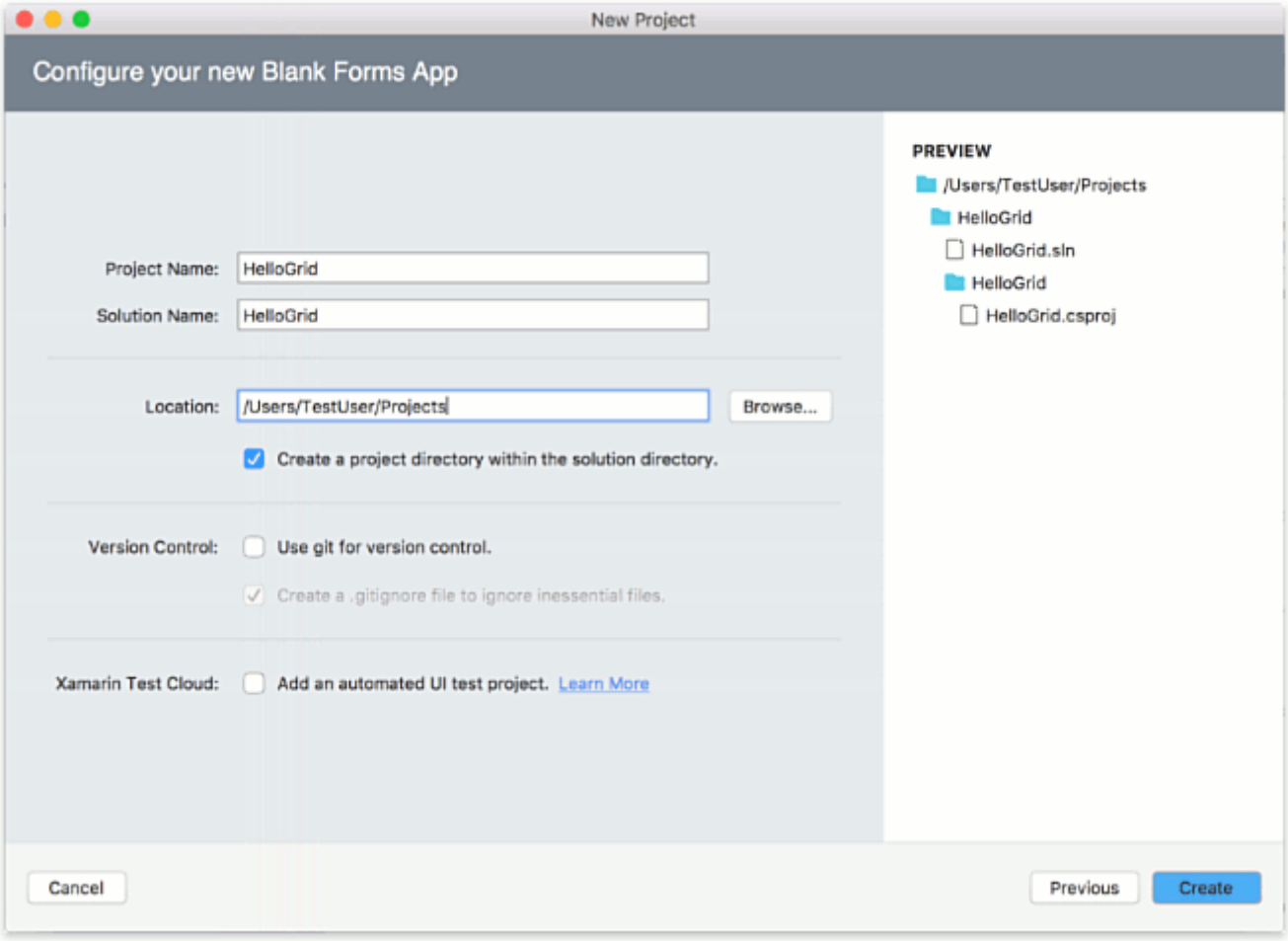

2. Name the application *HelloGrid*, select both Android and iOS target platforms, and specify whether the solution uses a portable or shared library to share code. Click **Next**.

© 2018 Developer Express Inc.




3. Specify the solution's name and location. Click **Create**.

© 2018 Developer Express Inc.




The newly created solution contains the following projects:

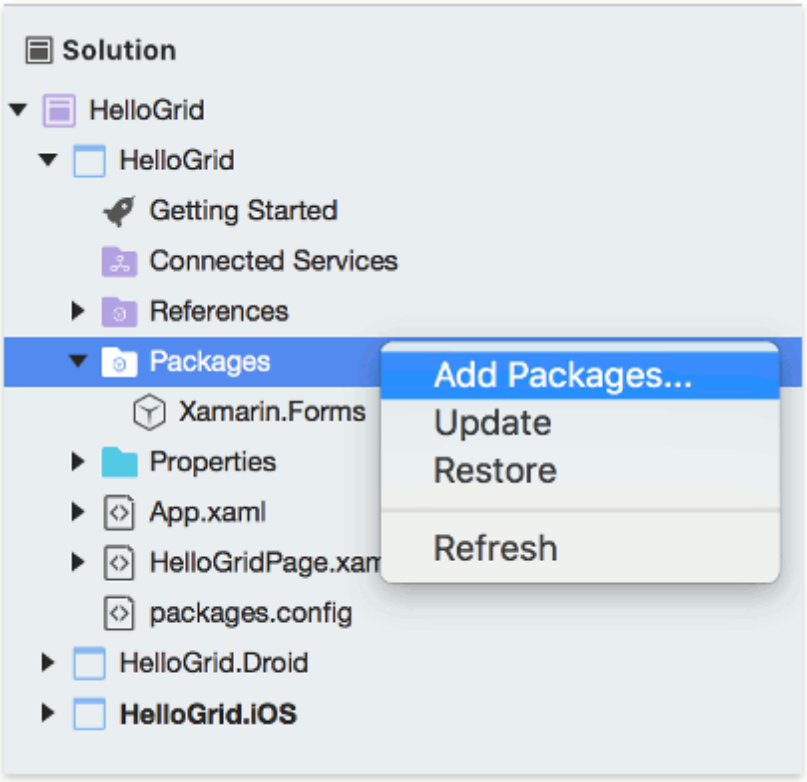
- HelloGrid (portable or shared) - A class library that contains the shared code and UI;
- HelloGrid.Droid - A project that contains Android specific code and serves as the Android application's entry point;
- HelloGrid.iOS - A project that contains iOS specific code and serves as the iOS application's entry point.

### Add the DevExpress Grid Component to a Solution

Add the **DevExpress Grid** NuGet package from NuGet.org to the solution's projects:

1. In a project, right-click the **Packages** folder and select **Add Packages...**

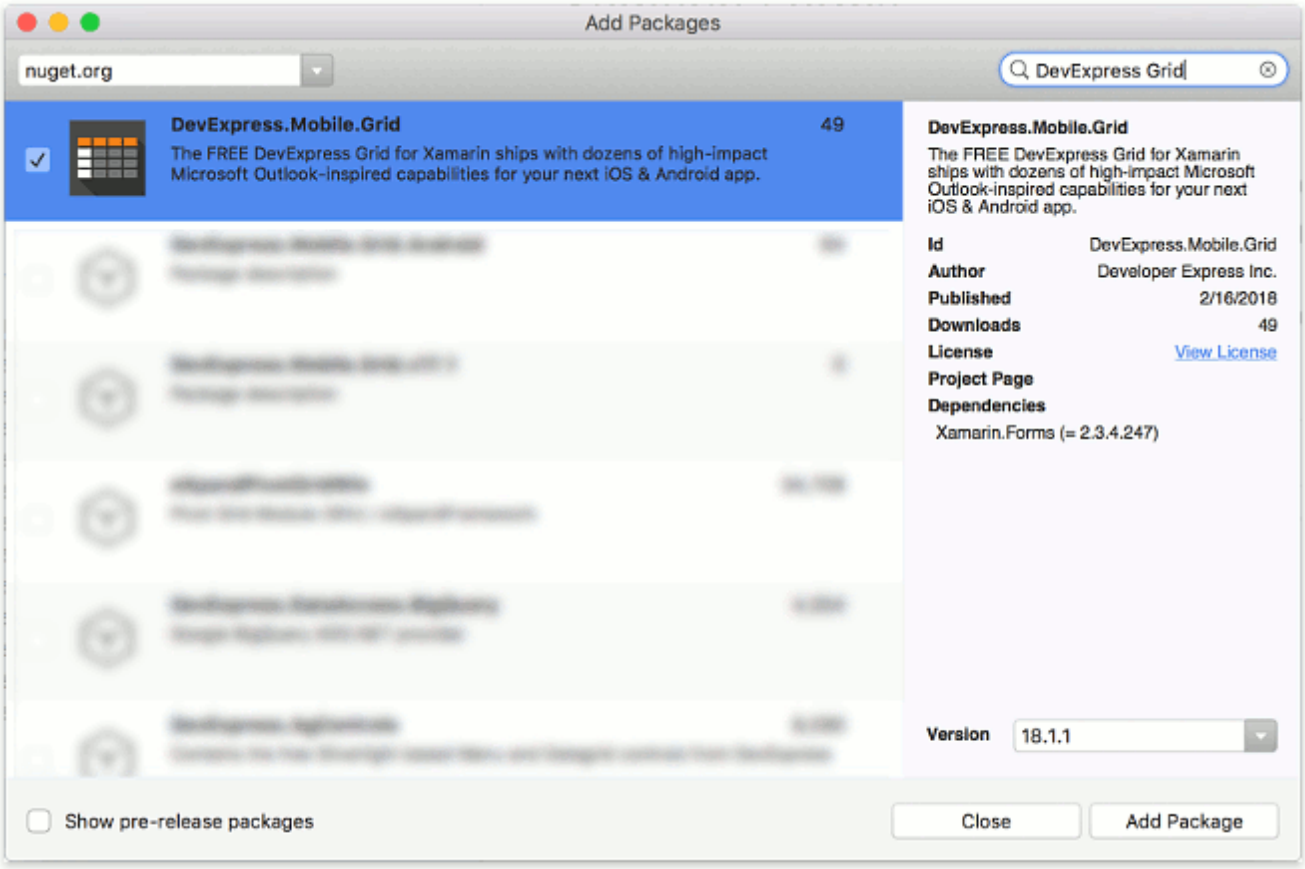
© 2018 Developer Express Inc.




2. In the invoked **Add Packages** dialog, search for **DevExpress.Mobile.Grid** in the *nuget.org* package source and click **Add Package**.

| **Important** | |
|---|---|
| Add Xamarin.Forms version *2.3.4.247* to the project before you add the DevExpress Grid package. The DevExpress Grid component supports this Xamarin.Forms version only. | |

© 2018 Developer Express Inc.




The following assembly references are automatically added to projects after the package is installed:

| Project | Assembly |
|---|---|
| Android | DevExpress.Mobile.Core.Android.v18.1.dll<br>DevExpress.Mobile.Grid.Android.v18.1.dll |
| iOS | DevExpress.Mobile.Core.iOS.v18.1.dll<br>DevExpress.Mobile.Grid.iOS.v18.1.dll |
| PCL<br>(when using a portable project) | DevExpress.Mobile.Grid.v18.1.dll<br>DevExpress.Mobile.Core.v18.1.dll |

When using a shared library, manually add the **DevExpress.Mobile.Core.v18.1.dll** and the **DevExpress.Mobile.Grid.v18.1.dll** assembly references to Android and iOS projects.

All DevExpress Grid libraries are copied to your solution folder:

*{solution location}\packages\DevExpress.Mobile.Grid.18.1.1\lib*

### Initialize the DevExpress Grid Component

After referencing libraries, add the following initialization code to the Android (*MainActivity.cs*) and iOS (*AppDelegate.cs*) projects:

**C#**

```
DevExpress.Mobile.Forms.Init ();
```

### Add Namespaces

Add the following namespace to a XAML markup or C# file in which a Grid instance is created:

**C#**

```
xmlns:dxGrid="clr-namespace:DevExpress.Mobile.DataGrid;assembly=DevExpress.Mobile.Grid.v18.1"
```

© 2018 Developer Express Inc.

**C#**

```
using DevExpress.Mobile.DataGrid;
```

### Create a Grid Instance

You can create a GridControl instance using XAML markup...

**C#**

```
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="HelloGrid.MainPage"
             xmlns:dxGrid="clr-namespace:DevExpress.Mobile.DataGrid;assembly=DevExpress.Mobile.Grid.v18.1
    <dxGrid:GridControl x:Name="grid">
    </dxGrid:GridControl>
</ContentPage>
```

...or C# code.

**C#**

```
using System;
using Xamarin.Forms;
using DevExpress.Mobile.DataGrid;
namespace HelloGrid
{
    public class App : Application
    {
        public App ()
        {
            GridControl grid = new GridControl ();
            MainPage = new ContentPage {
                Content = grid,
            };
        }
    }
}
```

The Getting Started section provides an example with detailed instructions on how to create an application with a grid bound to a data source and adjust various settings.

© 2018 Developer Express Inc.

# Create a Solution with the DevExpress Grid Component in Visual Studio



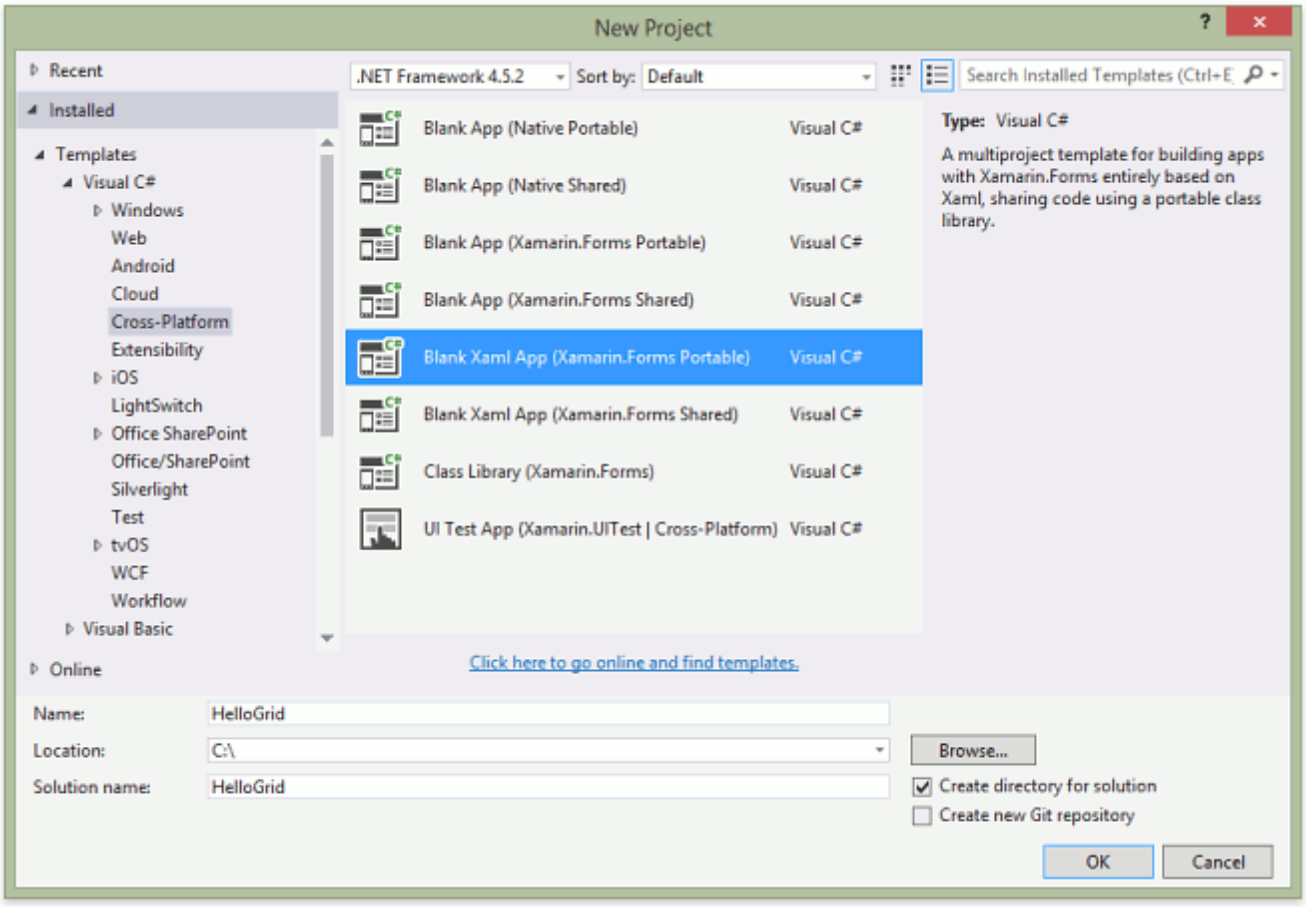
This topic explains how to create a new Xamarin.Forms cross-platform solution in Visual Studio, and how to add the DevExpress Grid component to your application using the DevExpress Grid NuGet package.


- Create a Solution
- Add the DevExpress Grid Component to a Solution
- Initialize the DevExpress Grid Component
- Add Namespaces
- Create a Grid Instance


| Note | |
|---|---|
| Visual Studio 2015 projects can use the DevExpress Grid component version 18.1.1 | |

### Create a Solution

Start Visual Studio and select **File | New | Project...** in the main menu. In the invoked **New Project** dialog, switch to the **Templates | Visual C# | Cross-Platform** section and select the solution template (Xamarin.Forms Portable or Xamarin.Forms Shared). Name the solution *HelloGrid* and click **OK**.




The newly created solution contains the following projects:

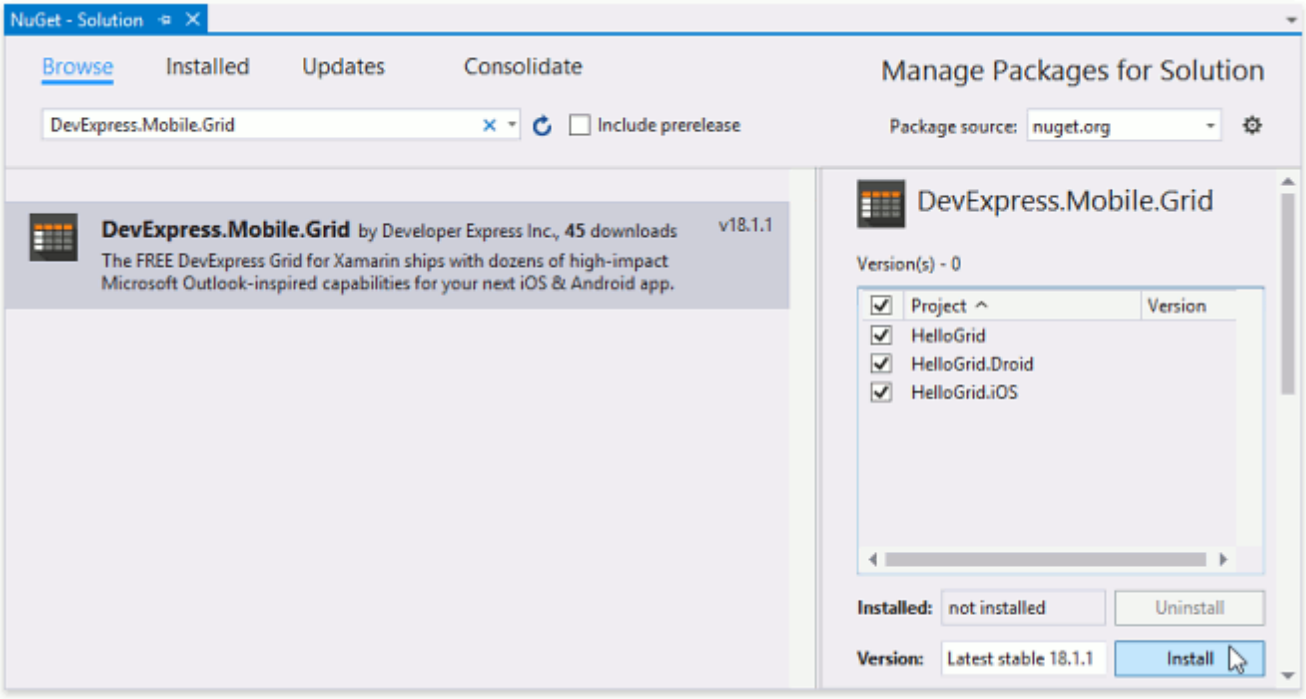
- HelloGrid (Portable or Shared) - A class library that contains the shared code and UI;
- HelloGrid.Droid - A project that contains Android specific code and serves as the Android application's entry point;
- HelloGrid.iOS - A project that contains iOS specific code and serves as the iOS application's entry point.

### Add the DevExpress Grid Component to a Solution

© 2018 Developer Express Inc.

Install the **DevExpress Grid** NuGet package from NuGet.org in one of the following ways:

- NuGet Package Manager UI
  1. Select **Tools | NuGet Package Manager | Manage NuGet Packages for Solution...** in Visual Studio's main menu, or right-click the solution in **Solution Explorer** and select **Manage NuGet Packages for Solution...**
  2. Search for **DevExpress.Mobile.Grid** in the **nuget.org** package source, select all the solution's projects and click **Install**.




- Package Manager Console
  1. Select **Tools | NuGet Package Manager | Package Manager Console** in Visual Studio's main menu.
  2. Run the following command for each project in the console's **Default Project** drop-down list:

*Install-Package DevExpress.Mobile.Grid -Version 18.1.1*

The following assembly references are automatically added to projects after the package is installed:

| Project | Assembly |
|---|---|
| Android | DevExpress.Mobile.Core.Android.v18.1.dll<br>DevExpress.Mobile.Grid.Android.v18.1.dll |
| iOS | DevExpress.Mobile.Core.iOS.v18.1.dll<br>DevExpress.Mobile.Grid.iOS.v18.1.dll |
| PCL<br>(when using a portable project) | DevExpress.Mobile.Grid.v18.1.dll<br>DevExpress.Mobile.Core.v18.1.dll |

When using the **Xamarin.Forms Shared** solution template, manually add the **DevExpress.Mobile.Core.v18.1.dll** and the **DevExpress.Mobile.Grid.v18.1.dll** assembly references to Android and iOS projects.

All DevExpress Grid libraries are copied to your solution folder:

*{solution location}\packages\DevExpress.Mobile.Grid.18.1.1\lib*

### Initialize the DevExpress Grid Component

After referencing libraries, add the following initialization code to the Android (*MainActivity.cs*) and iOS (*AppDelegate.cs*) projects:

**C#**

```
DevExpress.Mobile.Forms.Init ();
```

© 2018 Developer Express Inc.

### Add Namespaces

Add the following namespace to a XAML markup or C# file in which a Grid instance is created:

**C#**

```
xmlns:dxGrid="clr-namespace:DevExpress.Mobile.DataGrid;assembly=DevExpress.Mobile.Grid.v18.1"
```

**C#**

```
using DevExpress.Mobile.DataGrid;
```

### Create a Grid Instance

You can create a GridControl instance using XAML markup...

**Xaml**

```
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="HelloGrid.MainPage"
             xmlns:dxGrid="clr-namespace:DevExpress.Mobile.DataGrid;assembly=DevExpress.Mobile.Grid.v18.1"
    <dxGrid:GridControl x:Name="grid">
    </dxGrid:GridControl>
</ContentPage>
```

...or C# code.

**C#**

```
using System;
using Xamarin.Forms;
using DevExpress.Mobile.DataGrid;
namespace HelloGrid
{
    public class App : Application
    {
        public App ()
        {
            GridControl grid = new GridControl ();
            MainPage = new ContentPage {
                Content = grid,
            };
        }
    }
}
```

The Getting Started section provides an example with detailed instructions on how to create an application with a grid bound to a data source and adjust various settings.

© 2018 Developer Express Inc.

# Grid Demos



The DevExpress Grid for Xamarin.Forms ships with examples that demonstrate the component's key features. You can access these sample projects from the DevExpress Code Examples database.

The table below lists examples which can help you to get started with the Grid component and explore its main features.

| Sample | Description |
|---|---|
| Getting Started | Provides an example of how to integrate the DevExpress Grid component into a project, bind it to a data source and use its API to implement the required functionality. You can replicate this application by performing steps from the Getting Started tutorial of the current documentation. |
| Grid Feature Demos | An application that demonstrates the functionality the DevExpress Grid component provides. Run the application and explore grid features by switching between them using the navigation pane at the left. Buttons at the bottom allow you to review each feature description, as well as the C# code-behind and XAML markup used to implement the functionality. |

© 2018 Developer Express Inc.

# Getting Started



Lessons in this section allow you to get started quickly with the DevExpress Grid component. These examples provide you with step-by-step instructions on how to create a Xamarin.Forms cross platform application for the Android and iOS mobile operating systems using the GridControl and demonstrate the control's basic functionality.

To replicate the current tutorial, you need the OS X operating system, as well as Xamarin Studio and Xcode installed. For details on requirements, see An Introduction to Xamarin.Forms.

This tutorial consists of the following lessons.

- Lesson 1 - Bind a Grid to Data
- Lesson 2 - Create Grid Columns
- Lesson 3 - Add New Item Row
- Lesson 4 - Group Data
- Lesson 5 - Calculate Summaries
- Lesson 6 - Sort Data
- Lesson 7 - Filter Data

Once you perform all of these steps, you will get the same application as the Getting Started sample (the *HelloGrid* application) shipped with the DevExpress Grid component.

© 2018 Developer Express Inc.

# Lesson 1 - Bind a Grid to Data



This document describes how to bind a grid to data created at runtime. To learn more about this procedure, you can also watch a video below.

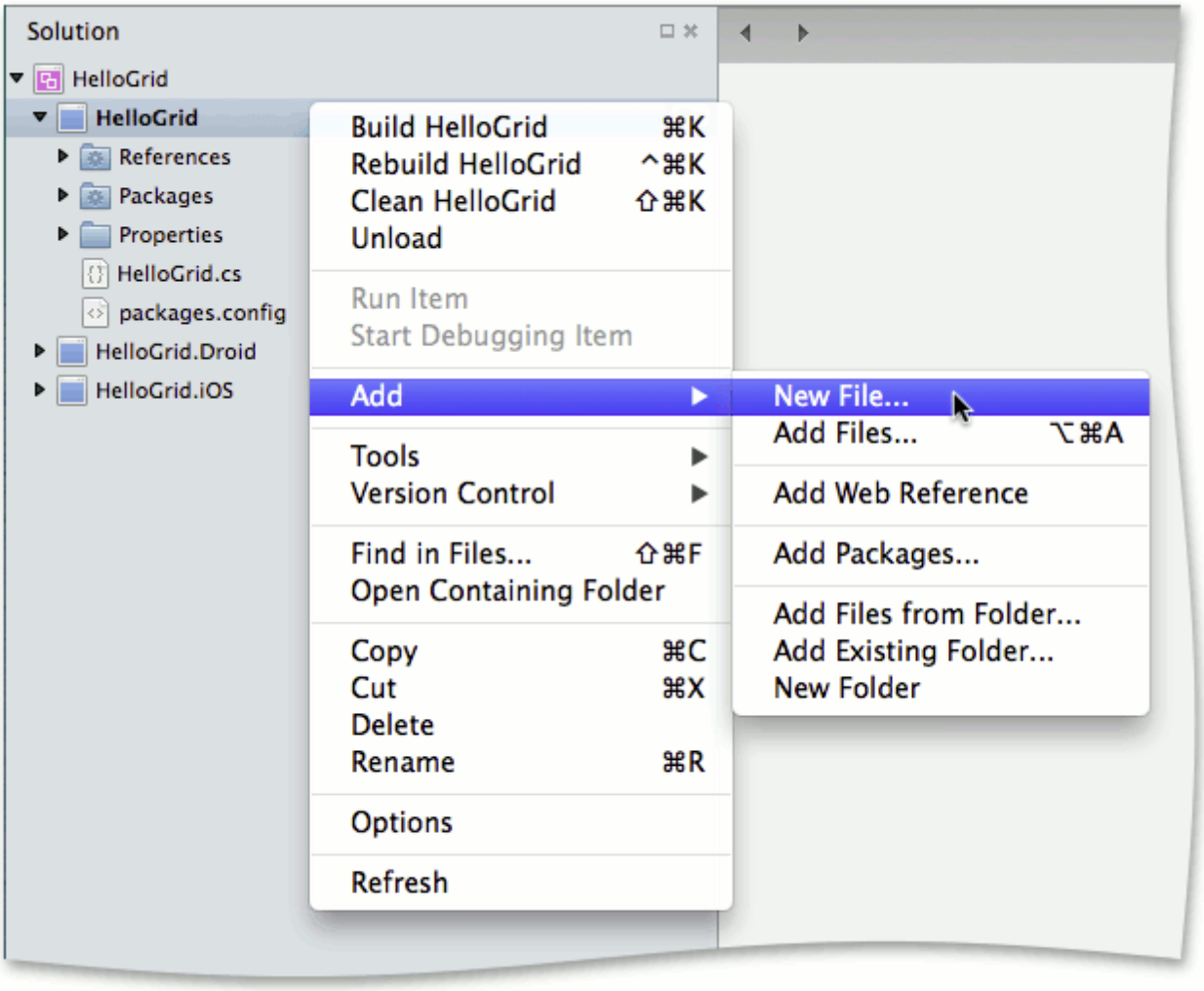
The topic includes the following sections.


- Create a Solution with a Grid
- Prepare a Data Source
- Bind a Grid to Data
- Result


**Create a Solution with a Grid**

1. Create a new Xamarin.Forms cross-platform solution (*HelloGrid*) in Xamarin Studio and include the DevExpress Grid component.
2. Add a new content page to the PCL project. To do this, right-click the project, and select **Add | New File...**. In the invoked dialog, switch to the **Forms** section and select **Forms ContentPage Xaml**. Enter the name *MainPage*.

© 2018 Developer Express Inc.




3. Open the *App.cs* file and modify the **GetMainPage** method to set the newly created page as the start page of the application.

**C#**

```
public class App : Application {
    public App() {
        this.MainPage = GetMainPage();
    }
    public static Page GetMainPage () {
        return new MainPage ();
    }
}
```

4. Add a grid to the page at design time (by using the XAML markup in the *MainPage.xaml* file).

**Xaml**

```
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="HelloGrid.MainPage"
    xmlns:dxGrid="clr-namespace:DevExpress.Mobile.DataGrid;assembly=DevExpress.Mobile.Grid.v18.1">
    <dxGrid:GridControl x:Name="grid">
    </dxGrid:GridControl>
</ContentPage>
```

### Prepare a Data Source

The GridControl does not support the unbound mode in which records would be added directly to a grid. Instead, records must be added to a bound data source and then appear in the grid. In this example, the grid will be bound to data created at runtime.

© 2018 Developer Express Inc.

Declare the **Order** class encapsulating an individual data record. Its public properties (**Date**, **Shipped**, **Product** and **Quantity**) will be data source fields.

| C# | |
|---|---|

© 2018 Developer Express Inc.

```
public class Order : ModelObject {
    DateTime date;
    bool shipped;
    Product product;
    int quantity;
    public Order() {
        this.date = DateTime.Today;
        this.shipped = false;
        this.product = new Product ("", 0);
        this.Quantity = 0;
    }
    public Order(DateTime date, bool shipped, Product product, int quantity) {
        this.date = date;
        this.shipped = shipped;
        this.product = product;
        this.quantity = quantity;
    }
    public DateTime Date {
        get { return date; }
        set { if (date != value) {
                date = value;
                RaisePropertyChanged("Date");}}
    }
    public bool Shipped {
        get { return shipped; }
        set { if(shipped != value) {
                shipped = value;
                RaisePropertyChanged("Shipped");}}
    }
    public Product Product {
        get { return product; }
        set { if (product != value) {
                product = value;
                RaisePropertyChanged ("Product");}}
    }
    public int Quantity {
        get { return quantity; }
        set { if (quantity != value) {
                quantity = value;
                RaisePropertyChanged ("Quantity");}}
    }
}
public class Product : ModelObject {
    string name;
    int unitPrice;
    public Product(string name, int unitPrice) {
        this.name = name;
        this.unitPrice = unitPrice;
    }
    public string Name {
        get { return name; }
        set { name = value; }
    }
    public int UnitPrice{
        get { return unitPrice; }
        set { unitPrice = value; }
    }
}
public class ModelObject : INotifyPropertyChanged {
    public event PropertyChangedEventHandler PropertyChanged;
    protected void RaisePropertyChanged(string name) {
        if (PropertyChanged != null)
            PropertyChanged(this, new PropertyChangedEventArgs(name));
    }
}
```

© 2018 Developer Express Inc.

A collection of **Order** objects will be the data source for the grid. This collection is returned by the **Orders** property of the **TestOrdersRepository** class.

**C#**

```
public abstract class OrdersRepository {
    readonly DevExpress.Mobile.Core.Containers.BindingList<Order> orders;
    public OrdersRepository() {
        this.orders = new DevExpress.Mobile.Core.Containers.BindingList<Order>();
    }
    public DevExpress.Mobile.Core.Containers.BindingList<Order> Orders {
        get { return orders; }
    }
}
public class TestOrdersRepository : OrdersRepository {
    const int orderCount = 100;
    readonly List<Product> products;
    readonly Random random;
    public TestOrdersRepository() : base() {
        this.random = new Random((int)DateTime.Now.Ticks);
        this.products = new List<Product>();
        GenerateProducts();
        for (int i = 0; i < orderCount; i++)
            Orders.Add(GenerateOrder(i));
    }
    Order GenerateOrder(int number) {
        Order order = new Order(new DateTime(2014, 1, 1).AddDays(random.Next(0, 60)),
            number % 3 == 0, RandomItem<Product>(products), random.Next(1, 100));
        return order;
    }
    T RandomItem<T>(IList<T> list) {
        int index = (int)(random.NextDouble() * 0.99 * (list.Count));
        return list[index];
    }
    void GenerateProducts() {
        products.Add(new Product("Tofu", 50));
        products.Add(new Product("Chocolade", 34));
        products.Add(new Product("Ikura", 70));
        products.Add(new Product("Chai", 3));
        products.Add(new Product("Boston Crab Meat", 36));
        products.Add(new Product("Ravioli Angelo", 18));
        products.Add(new Product("Ipon Coffee", 10));
        products.Add(new Product("Questo Cabrales", 25));
    }
}
```

### Bind a Grid to Data

Set **BindingContext** for the content page to a **TestOrdersRepository** class instance in the *MainPage.xaml.cs* file, as shown below.

**C#**

```
namespace HelloGrid {
    public partial class MainPage : ContentPage {
        public MainPage () {
            InitializeComponent ();
            TestOrdersRepository model = new TestOrdersRepository ();
            BindingContext = model;
        }
    }
}
```

To bind the grid to a data source, assign the order collection object (**TestOrdersRepository.Orders**) to the GridControl.ItemsSource property.

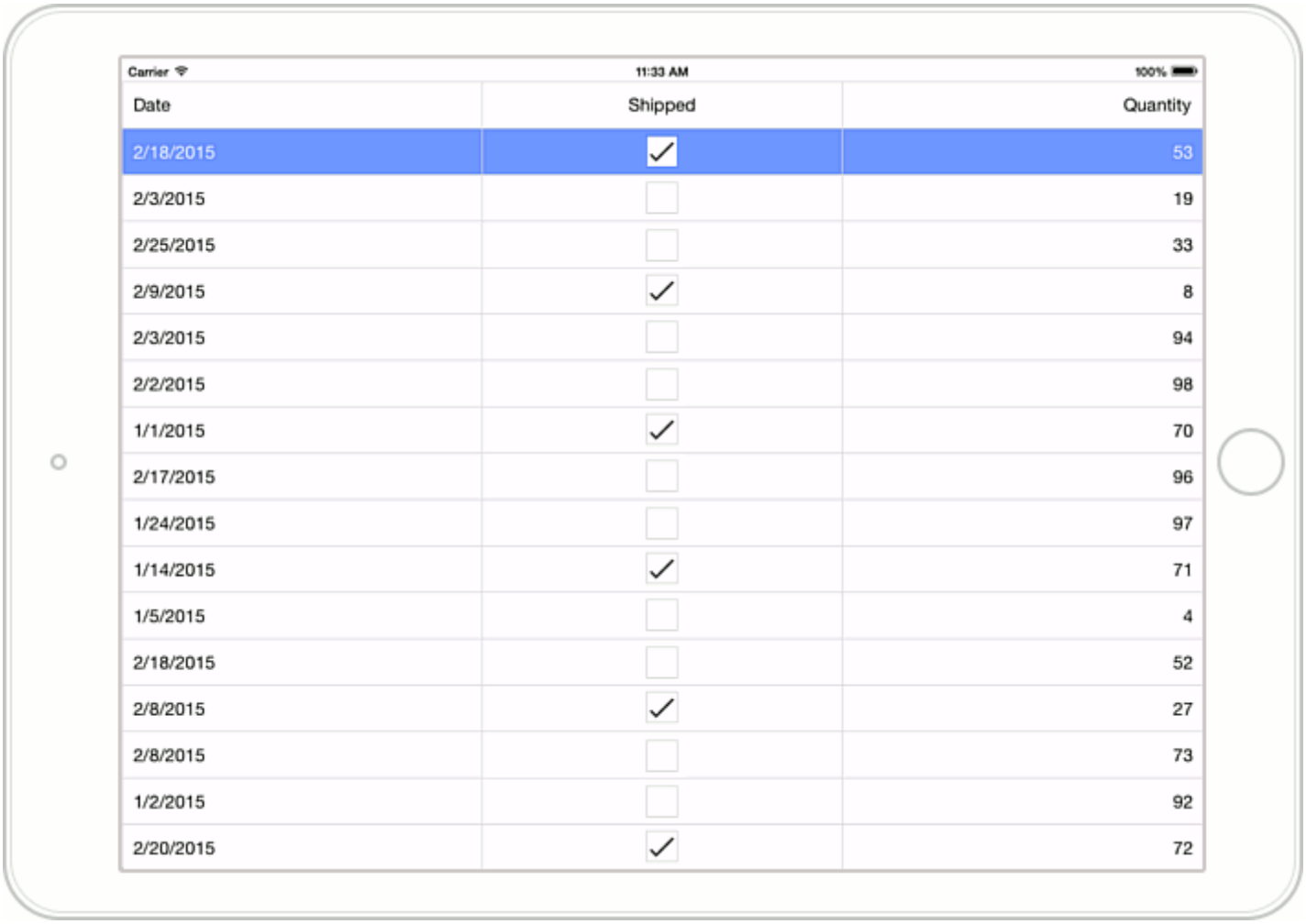
**Xaml**

© 2018 Developer Express Inc.

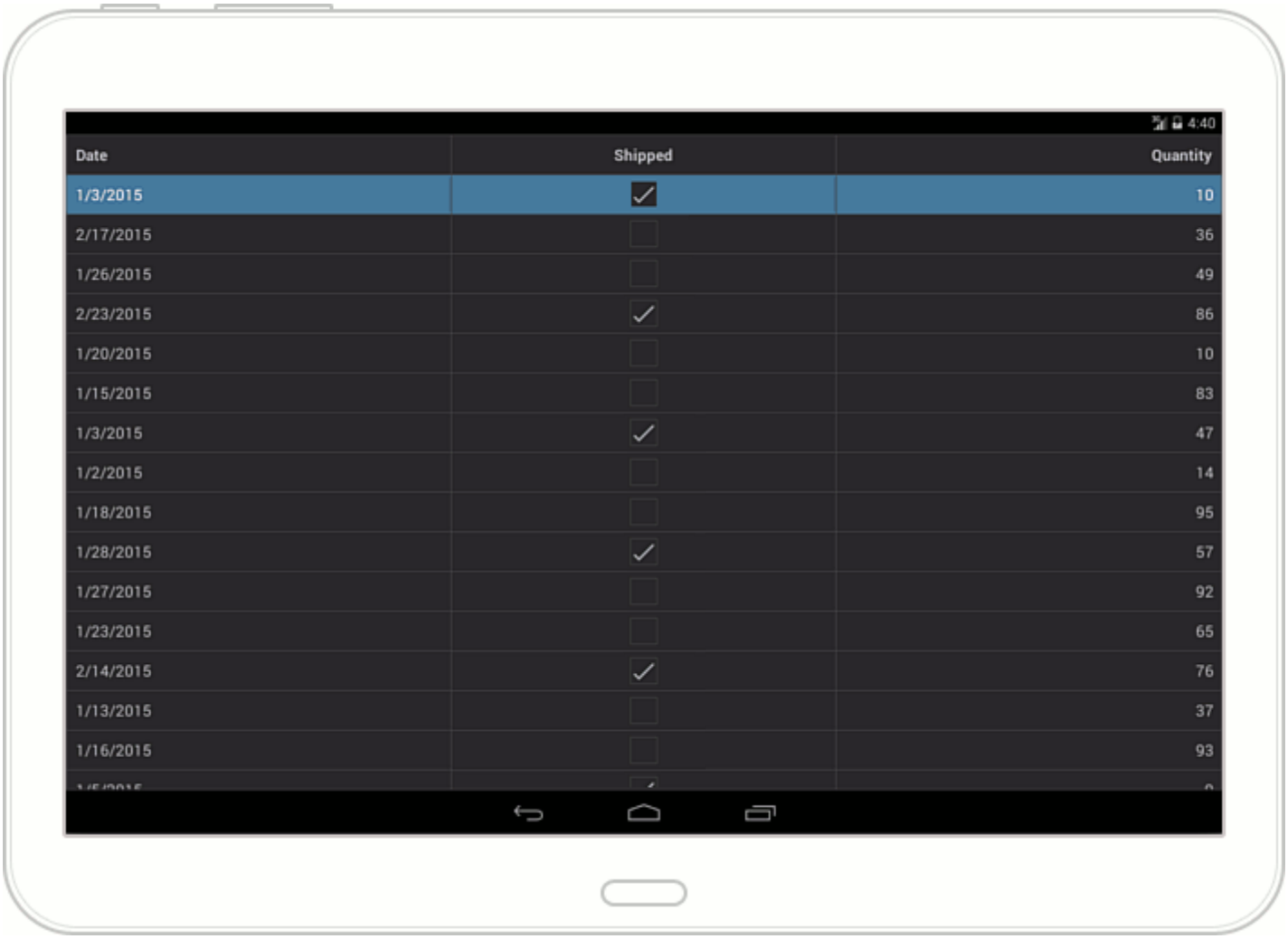
```
<dxGrid:GridControl x:Name="grid" ItemsSource="{Binding Orders}">
</dxGrid:GridControl>
```

**Result**

By default, the grid automatically generates columns for data source fields, except *complex fields*. In the current example, complex fields are fields of a product related to an order (Product.Name and Product.UnitPrice). The sequence of displaying columns within the grid is the same as the sequence of fields in the data source.




© 2018 Developer Express Inc.




Go on to Lesson 2 of the current Getting Started tutorial to learn how to manage a collection of grid columns (bind columns to complex fields, create an unbound column to display calculated values, adjust column settings and set the column display sequence in the grid).

© 2018 Developer Express Inc.

# Lesson 2 - Create Grid Columns



This document is the second step of the Getting Started tutorial, and explains how to work with the DevExpress Grid for Xamarin. It contains an overview of the GridControl's columns and provides an example on how to specify columns for the grid created in Lesson 1.

### Grid Columns Overview

In GridControl, data fields are represented as columns and records are displayed as rows. So, to display data, a grid should contain columns bound to fields of the underlying data source.

Grid columns are stored in the GridControl.Columns collection. An individual column is specified by the TextColumn, NumberColumn, DateColumn, SwitchColumn or ImageColumn object (a GridColumn descendant) depending on the type of data displayed in this column. These objects provide a set of properties for adjusting column settings (for example, the data display format, width, caption displayed in a column header, visibility, etc.).

There are two ways to add columns to a grid and bind them to data fields.

- Create columns for all fields of the bound data source automatically.

By default, the grid automatically generates columns based on the bound data source - one column for one field of a data source object, except for fields of nested objects. The order of columns in the grid is the same as the order of fields in the data source. You can prevent columns from being automatically created or set another mode of auto generating columns, using the GridControl.AutoGenerateColumnsMode property.

- Create columns and bind them to fields manually.

You can manually specify a collection of grid columns. To do this, create appropriate column objects, bind each column to the corresponding data field using the GridColumn.FieldName property and add columns to the GridControl.Columns collection in the order you want to show them in the grid.

In addition to columns bound to specific fields of a data source, you can create **unbound columns** to display data values calculated according to a formula based on values of other columns. To create an unbound column in the grid, add an appropriate column object to the GridControl.Columns collection and set the following column properties.

- GridColumn.FieldName - a unique string that does not match any field name in the grid's underlying data source.
- GridColumn.UnboundExpression - a formula (string expression) to automatically evaluate values for the column.
- GridColumn.UnboundType - the type of data the column is supposed to display (Boolean, DateTime, Decimal, Integer, String or Object).

### Create Grid Columns - Example

Open the *HelloGrid* solution created in Lesson 1 of the current Getting Started tutorial and add columns to the grid to display the following information on orders.

- **Product Name** and **Unit Price**

TextColumn and NumberColumn objects specifying columns bound to complex fields - the Name and UnitPrice properties of the order's related Product entity. These columns are not generated automatically.

- **Quantity**

A NumberColumn object specifying a column bound to the order's Quantity property.

- **Total**

A NumberColumn object specifying an unbound column that displays the amount of each order according to the expression: Quantity*UnitPrice.

- **Date**

A DateColumn object specifying a column bound to the order's Date property.

- **Shipped**

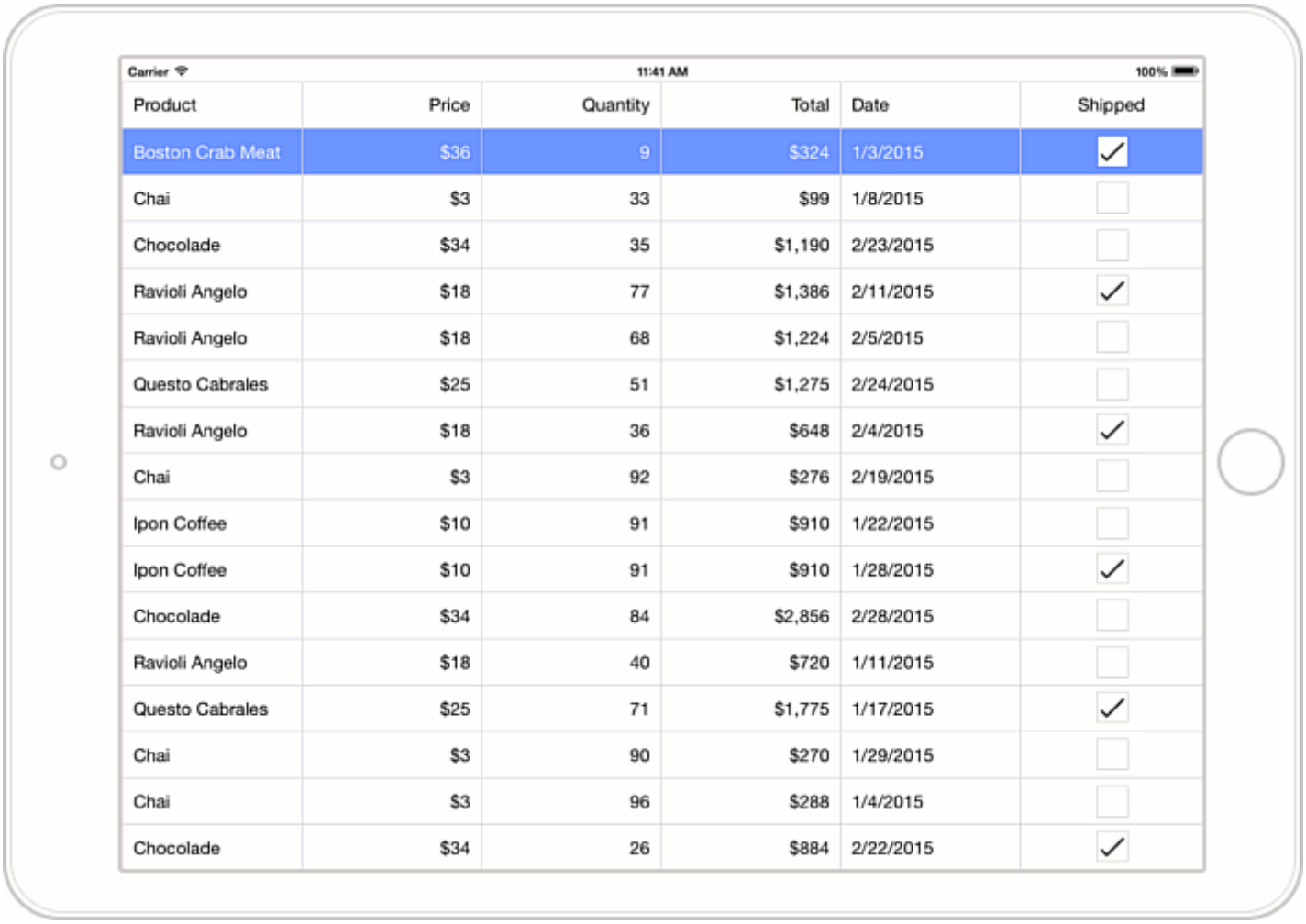
A SwitchColumn object specifying a column bound to the order's Shipped property.

| **Xaml** | |
|---|---|

© 2018 Developer Express Inc.

```
<dxGrid:GridControl x:Name="grid" ItemsSource="{Binding Orders}">
    <dxGrid:GridControl.Columns>
        <dxGrid:TextColumn FieldName="Product.Name" Caption="Product" Width="170" />
        <dxGrid:NumberColumn FieldName="Product.UnitPrice" Caption="Price" DisplayFormat="C0"/>
        <dxGrid:NumberColumn FieldName="Quantity"/>
        <dxGrid:NumberColumn FieldName="Total"
                             UnboundType="Integer" UnboundExpression="[Quantity] * [Product.UnitPrice]"
                             IsReadOnly="True" DisplayFormat="C0"/>
        <dxGrid:DateColumn FieldName="Date" DisplayFormat="d"/>
        <dxGrid:SwitchColumn FieldName="Shipped" />
    </dxGrid:GridControl.Columns>
</dxGrid:GridControl>
```

As a result, you will have a grid that displays all necessary information on orders retrieved from the bound data source.




© 2018 Developer Express Inc.

Go on to Lesson 3 of the current Getting Started tutorial to learn how to provide end-users with the capability to add new records to the grid.

© 2018 Developer Express Inc.

# Lesson 3 - Add New Item Row



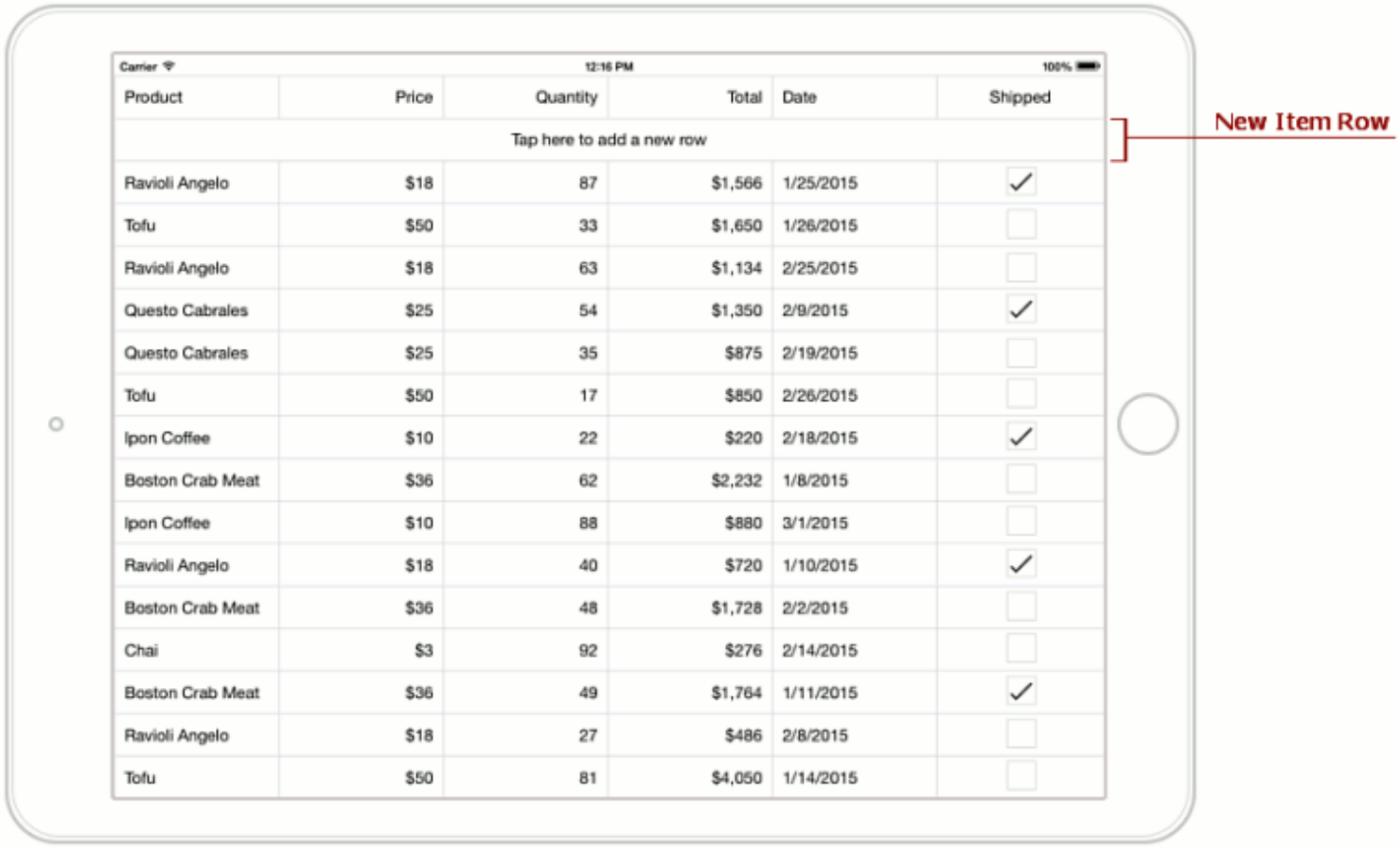
To allow end-users to add new records, show the **New Item Row** in the grid.

Open the *HelloGrid* solution created in Lesson 2 of the current Getting Started tutorial and set the GridControl.NewItemRowVisibility property to **true**.

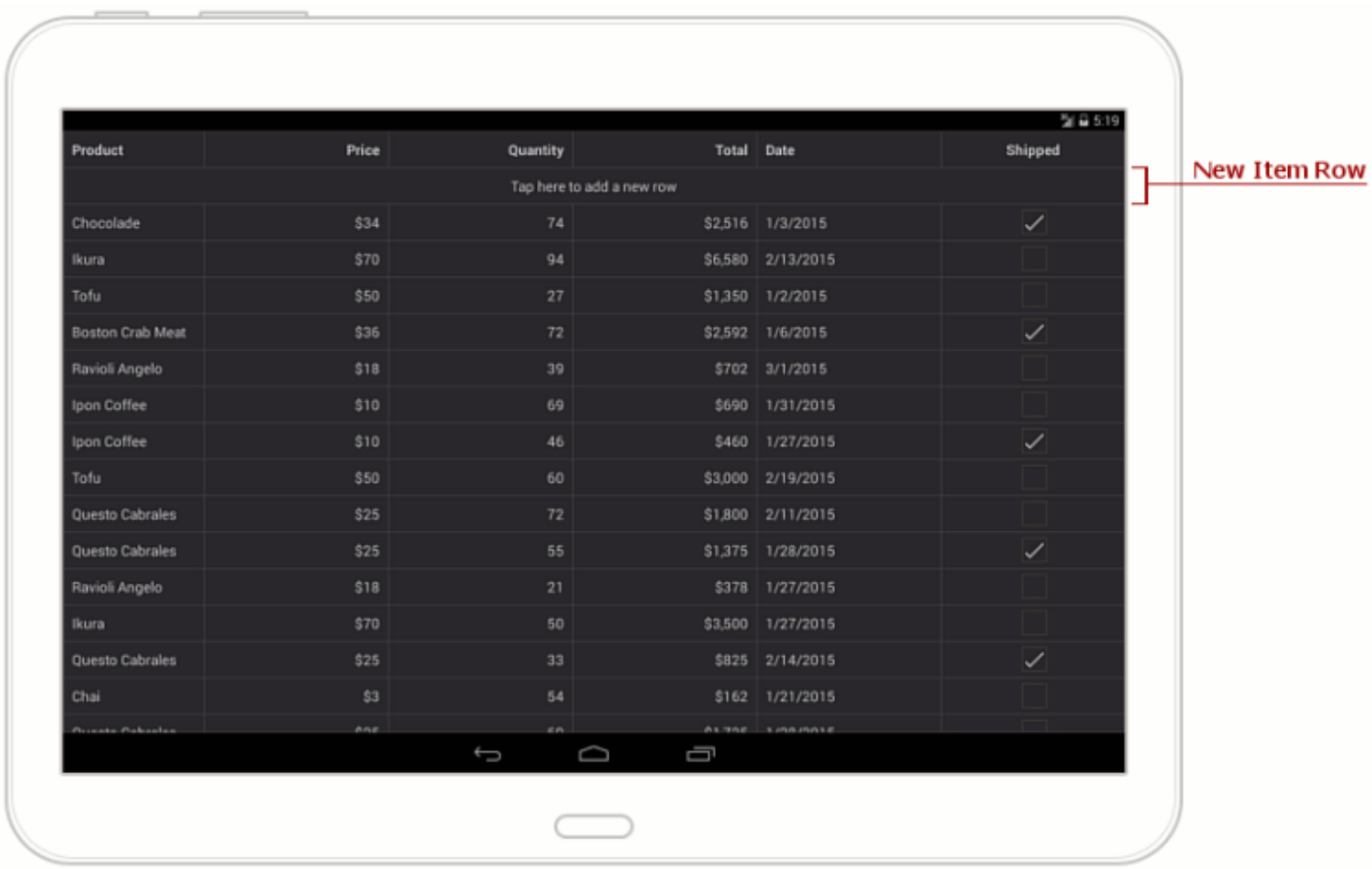
**Xaml**

```
<dxGrid:GridControl x:Name="grid" ItemsSource="{Binding Orders}"
                    NewItemRowVisibility="true" >
    <!-- ... -->
</dxGrid:GridControl>
```

The New Item Row will be shown above grid's rows. When an end-user taps it, a row editor (an in-place editor or edit form, depending on the row edit mode set in the grid) is invoked. After an end-user sets required cell values in this editor and taps **Apply** (or **Done**), a new row is added.




© 2018 Developer Express Inc.




Go on to Lesson 4 of the current Getting Started tutorial to learn how to group data in the grid.

© 2018 Developer Express Inc.

# Lesson 4 - Group Data



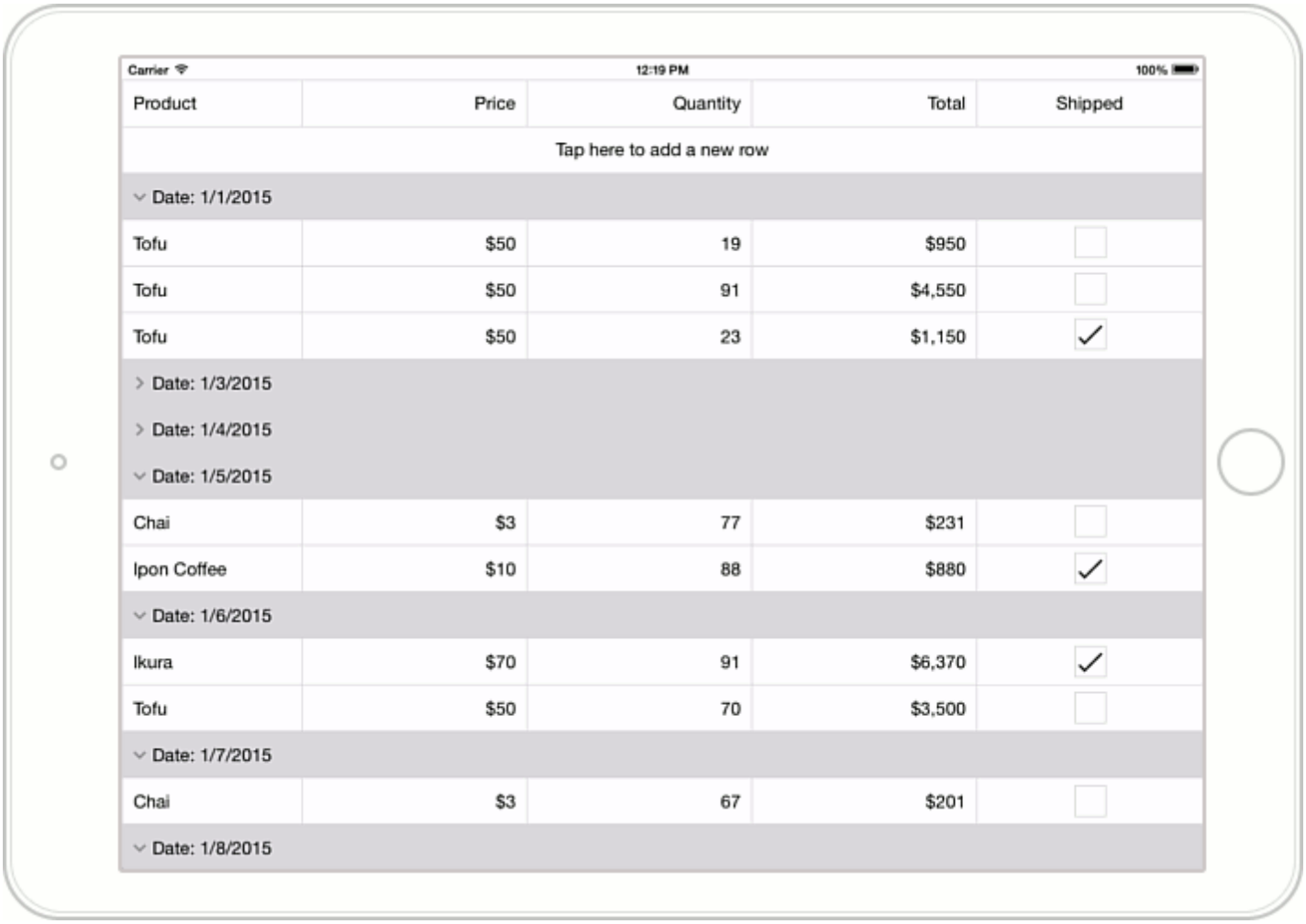
In the GridControl, you can group data by a specified column.

Open the *HelloGrid* solution created in Lesson 3 of the current Getting Started tutorial and group orders displayed in the grid by dates. To do this, set the **Date** column's GridColumn.IsGrouped property to **true** and the GridColumn.GroupInterval property to ColumnGroupInterval.Date.

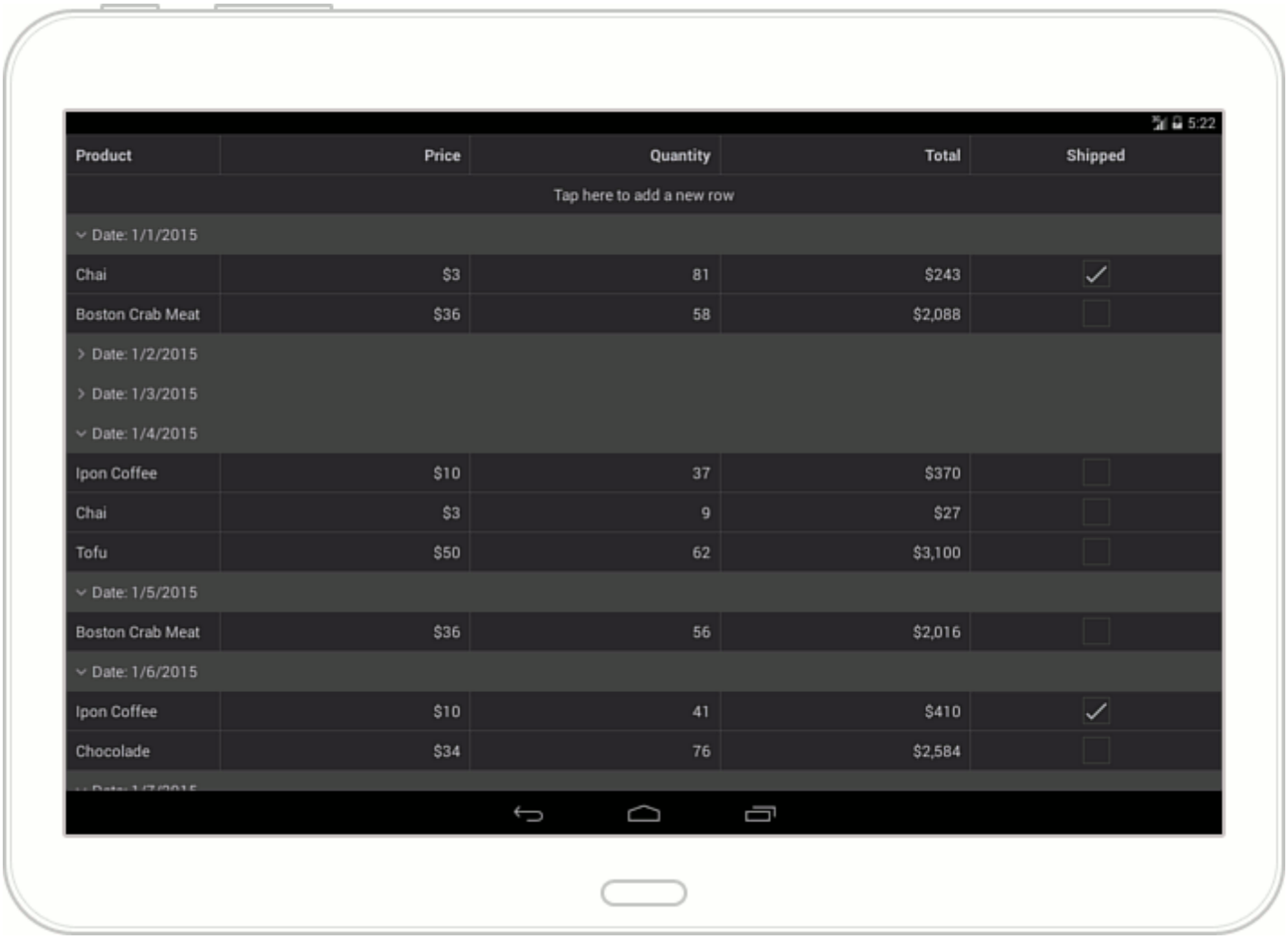
**Xaml**

```
<dxGrid:GridControl x:Name="grid" ItemsSource="{Binding Orders}"
                    NewItemRowVisibility="true" >
    <dxGrid:GridControl.Columns>
        <!-- ... -->
        <dxGrid:DateColumn FieldName="Date" DisplayFormat="d"
                           IsGrouped="true" GroupInterval="Date"/>
        <!-- ... -->
    </dxGrid:GridControl.Columns>
</dxGrid:GridControl>
```

As a result, grid data rows that have identical dates in the **Date** column are arranged into groups. Each group is identified by a group row. An end-user can tap group rows to expand or collapse them.




© 2018 Developer Express Inc.




Go on to Lesson 5 of the current Getting Started tutorial to learn how to calculate and display data summaries (group and total) in the grid.

© 2018 Developer Express Inc.

# Lesson 5 - Calculate Summaries



This document contains an overview of the GridControl's data summaries and provides an example on how to calculate total and group summaries for orders in the grid created in Lesson 4 of the current Getting Started tutorial.

### Data Summaries Overview

GridControl allows you to display brief information about groups of rows or particular columns. For example, you can display the sum of values or the maximum value, etc. Such information is called a summary.

Two summary types are supported.

- **Total Summary**

An aggregate function value calculated over all rows in a column and displayed under this column.

- **Group Summaries**

An aggregate function value calculated over all rows in a group and displayed in a group row.

Summaries (group and total) are specified by GridColumnSummary objects. You can specify the field against whose values the summary is calculated (GridColumnSummary.FieldName), the aggregate function (GridColumnSummary.Type) and the summary value format (GridColumnSummary.DisplayFormat). Total summaries are stored in the GridControl.TotalSummaries collection. Group summaries are stored in the GridControl.GroupSummaries collection.

Summaries (group and total) provide five predefined aggregate functions.

- *Count* - the number of data rows.
- *Max* and *Min* - the maximum and minimum values.
- *Sum* and *Average* - the sum and the average values.

You can also calculate summaries using your own custom rules. To do this, set the GridColumnSummary.Type property to SummaryType.Custom, and handle the GridControl.CalculateCustomSummary event.

### Create Data Summaries - Example

Open the *HelloGrid* solution created in Lesson 4 of the current Getting Started tutorial and create the following data summaries.

- Group summary to display the maximum **Total** value for each group of records.
- Total summary to calculate the sum of values by the whole **Total** column.
- Custom total summary to count a number of orders whose value in the **Shipped** column is **false** (to count orders that are not shipped).

You can add data summaries in XAML ...

**Xaml**

```
<dxGrid:GridControl x:Name="grid" ItemsSource="{Binding Orders}"
                    NewItemRowVisibility="true"
                    CalculateCustomSummary="OnCalculateCustomSummary">
    <!-- ... -->
    <dxGrid:GridControl.GroupSummaries>
        <dxGrid:GridColumnSummary FieldName="Total" Type="Max"/>
    </dxGrid:GridControl.GroupSummaries>
      <dxGrid:GridControl.TotalSummaries>
          <dxGrid:GridColumnSummary FieldName="Total" Type="Sum"
                                    DisplayFormat= "Total: {0:C0}"/>
          <dxGrid:GridColumnSummary FieldName="Shipped" Type="Custom"
                                    DisplayFormat= "Not Shipped: {0}"/>
      </dxGrid:GridControl.TotalSummaries>
</dxGrid:GridControl>
```

**C#**

© 2018 Developer Express Inc.

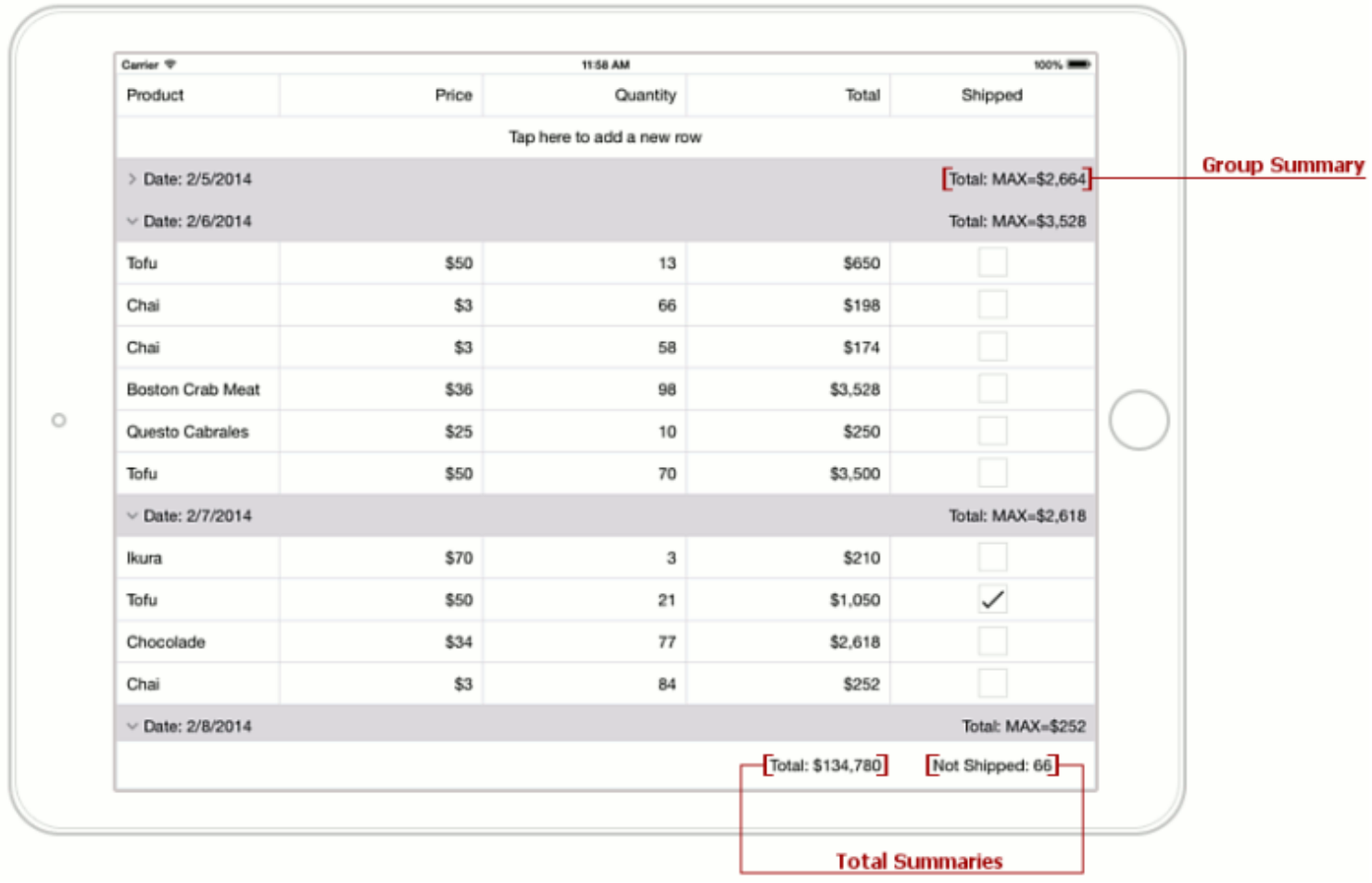
```
int count;
// ...
void OnCalculateCustomSummary(object sender, CustomSummaryEventArgs e) {
    if (e.FieldName.ToString () == "Shipped")
        if (e.IsTotalSummary){
            if (e.SummaryProcess == CustomSummaryProcess.Start) {
                count = 0;
            }
            if (e.SummaryProcess == CustomSummaryProcess.Calculate) {
                if (!(bool)e.FieldValue)
                    count++;
                e.TotalValue = count;
            }
        }
}
```

... or in code.

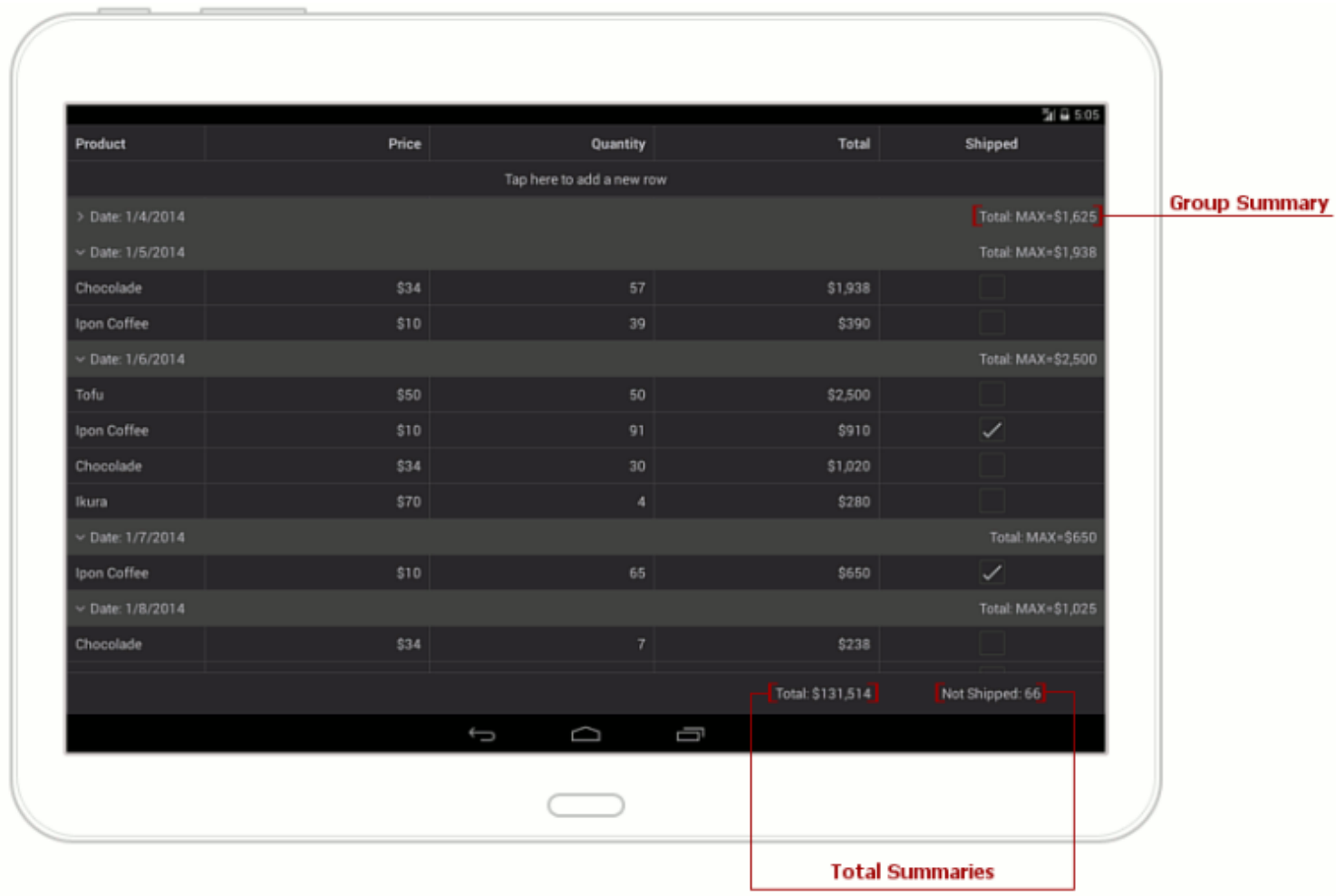
**C#**

```
grid.GroupSummaries.Add (new GridColumnSummary {
    FieldName = "Total",
    Type = SummaryType.Max});
grid.TotalSummaries.Add (new GridColumnSummary () {
    FieldName = "Total",
    Type = SummaryType.Sum,
    DisplayFormat = "Total: {0:C0}"});
```

As a result, group summaries are displayed in group rows and total summaries are displayed in a total summary panel at the bottom of the grid. The end-user can create or remove a group or total summary by tapping the corresponding item in a menu that appears when they touch and hold a group row or summary panel.




© 2018 Developer Express Inc.




Go on to Lesson 6 of the current Getting Started tutorial to learn how to sort data in the grid.

© 2018 Developer Express Inc.

# Lesson 6 - Sort Data

Xamarin Controls > Grid > Getting Started > Lesson 6 - Sort Data

By default, data in a GridControl can be sorted by a single column. To apply sorting by the specified column, set the GridColumn.SortOrder property to ColumnSortOrder.Ascending or ColumnSortOrder.Descending.

You can also sort data records by multiple columns. To do this, set the GridControl.SortMode property to GridSortMode.Multiple, and specify GridColumn.SortOrder for each column to be sorted. To specify the sorting sequence applied to columns, assign the GridColumn.SortIndex of these columns.

To prevent end-users from sorting data by values of specific columns, use the GridColumn.AllowSort property.

Open the *HelloGrid* solution created in Lesson 5 of the current Getting Started tutorial and apply the following sort settings.

- Sort orders displayed in the grid by the **Product.Name** column first and then by the **Quantity** column.
- Disable sorting by the **Shipped** column.

You can specify grid sorting settings in XAML...

**Xaml**

```
<dxGrid:GridControl x:Name="grid" ItemsSource="{Binding Orders}"
                     NewItemRowVisibility="true"
                    CalculateCustomSummary="OnCalculateCustomSummary"
                    SortMode="Multiple">
    <dxGrid:GridControl.Columns>
        <dxGrid:TextColumn FieldName="Product.Name" Caption="Product" Width="170"
                           SortOrder="Descending" SortIndex="0"/>
        <!-- ... -->
        <dxGrid:NumberColumn FieldName="Quantity"
                             SortOrder="Ascending" SortIndex="1"/>
        <!-- ... -->
        <dxGrid:SwitchColumn FieldName="Shipped" AllowSort="False"/>
    </dxGrid:GridControl.Columns>
</dxGrid:GridControl>
```

... or C# code.

**C#**

```
grid.SortMode = GridSortMode.Multiple;
grid.Columns ["Product.Name"].SortOrder = DevExpress.Data.ColumnSortOrder.Descending;
grid.Columns ["Product.Name"].SortIndex = 0;
grid.Columns ["Quantity"].SortOrder = DevExpress.Data.ColumnSortOrder.Ascending;
grid.Columns ["Quantity"].SortIndex = 1;
grid.Columns ["Shipped"].AllowSort = DevExpress.Utils.DefaultBoolean.False;
```

As a result, orders in the grid are sorted by products first and by amount of product units second. End-users can sort data in the grid by tapping a column header or tapping **Sort Ascending** or **Sort Descending** in a menu that appears when they touch and hold a column cell or header.

© 2018 Developer Express Inc.

© 2018 Developer Express Inc.

Go on to Lesson 7 of the current Getting Started tutorial to learn how to filter data in the grid.

© 2018 Developer Express Inc.

# Lesson 7 - Filter Data



The GridControl supports an automatic filtering functionality that displays a subset of data records, which meet a particular criteria. You can filter data against single or multiple columns.

To apply a filter to a column, use the GridColumn.AutoFilterValue property. The type of the comparison operator used to create filter conditions for the current column is specified by the GridColumn.AutoFilterCondition property. You can also use the GridControl.FilterExpression and GridControl.FilterString property to specify a grid filter expression.

End-users can filter data by columns using the auto-filter panel.

Open the *HelloGrid* solution created in Lesson 6 of the current Getting Started tutorial and configure the auto-filter panel for the grid.

- Set the GridControl.AutoFilterPanelVisibility property to **true** to display the auto-filter panel within the grid.
- Set the **Shipped** column's GridColumn.AllowAutoFilter property to **false** to prevent end-users from filtering data by this column's values using the auto-filter panel.

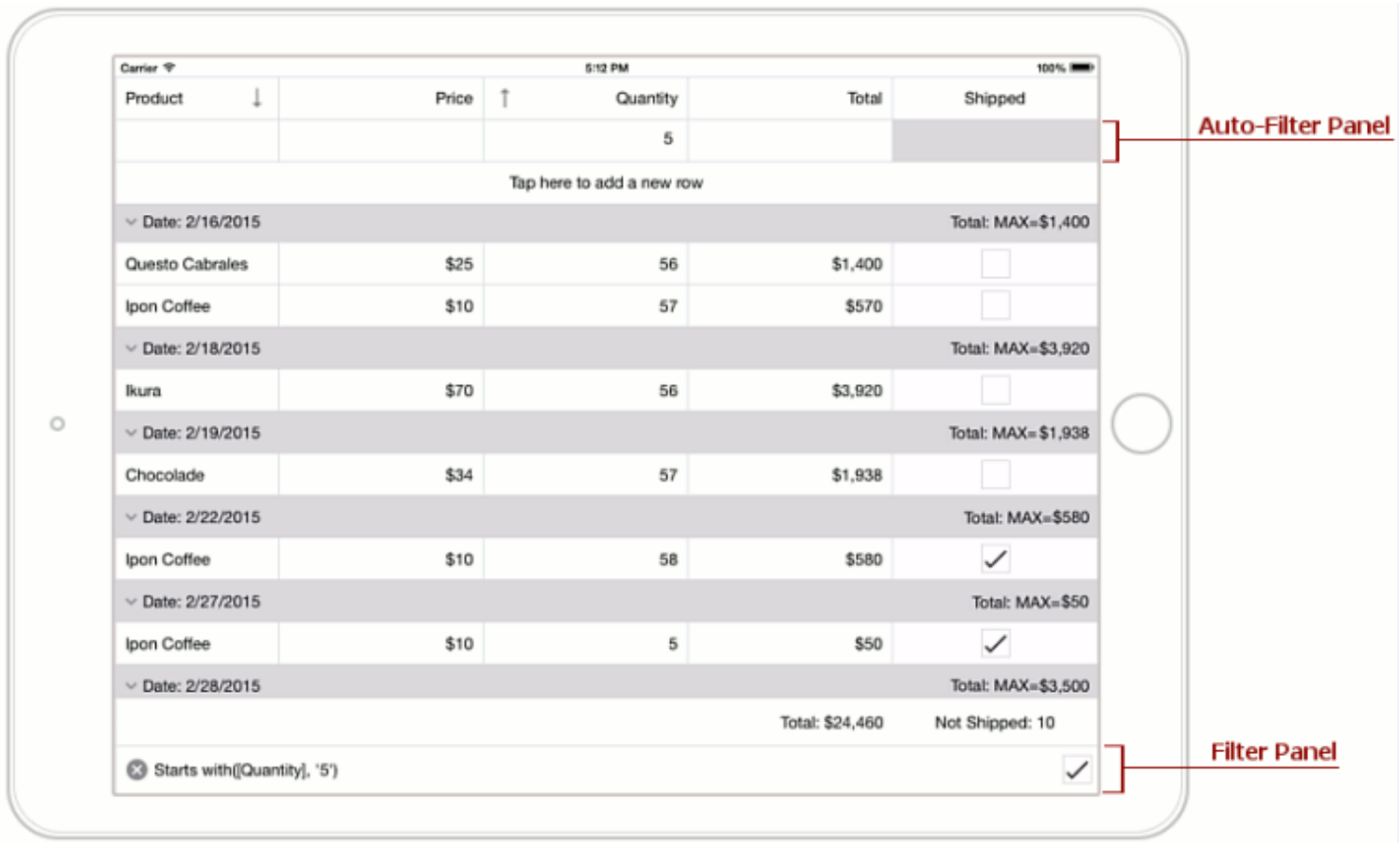
**Xaml**

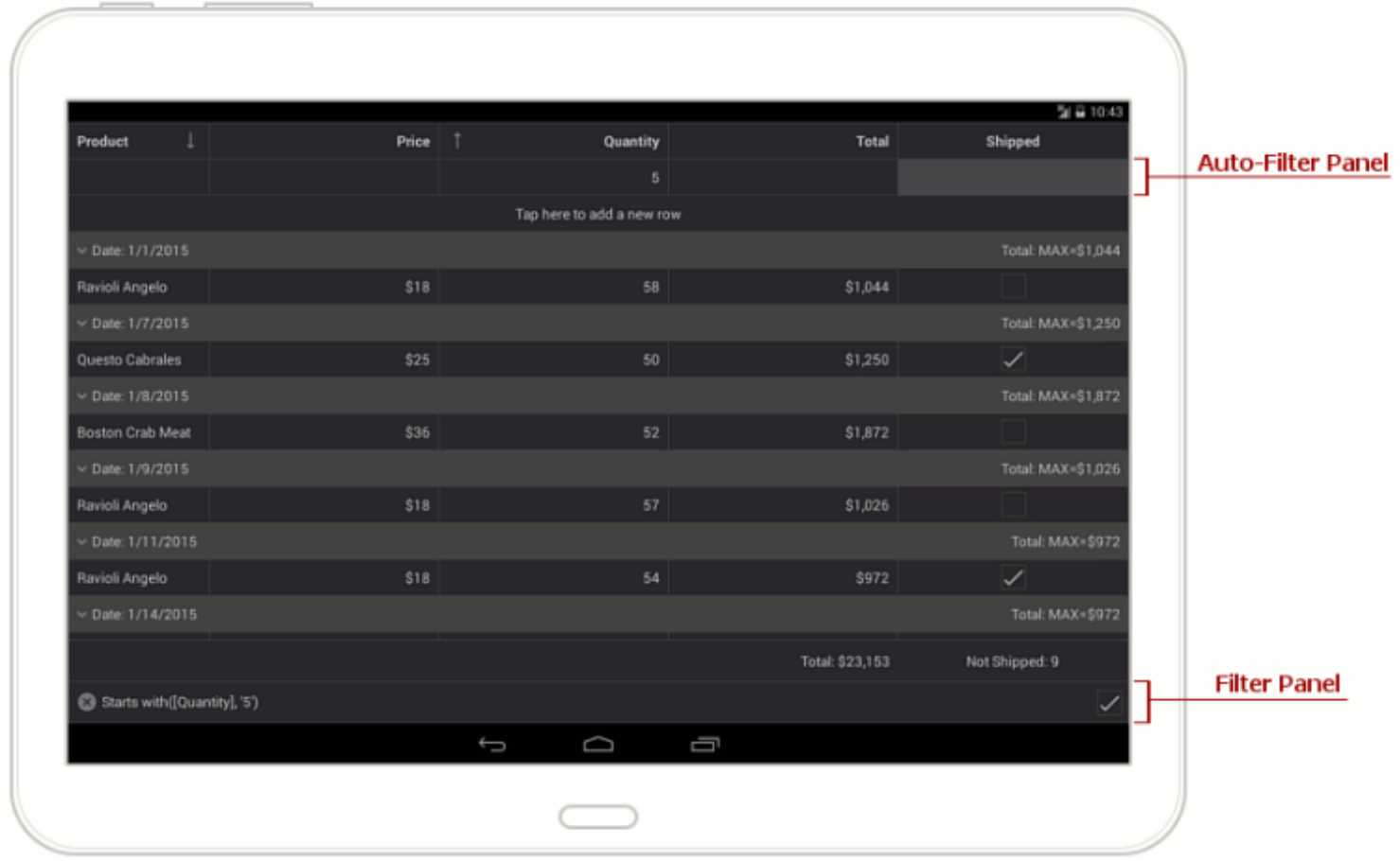
```
<dxGrid:GridControl x:Name="grid" ItemsSource="{Binding Orders}"
                    NewItemRowVisibility="true"
                    CalculateCustomSummary="OnCalculateCustomSummary"
                    SortMode="Multiple" AutoFilterPanelVisibility="true">
    <dxGrid:GridControl.Columns>
        <!-- ... -->
        <dxGrid:SwitchColumn FieldName="Shipped" AllowSort="False" AllowAutoFilter="false"/>
    </dxGrid:GridControl.Columns>
</dxGrid:GridControl>
```

As a result, an auto-filter panel is displayed at the top of the grid. When an end-user types text into this panel□s cell, data is filtered on the fly by values of the corresponding column.

After a filter has been applied, the filter panel appears by default at the bottom of the grid. This panel displays a string that represents a filter criteria currently applied in the grid (GridControl.ActualFilterString) and contains buttons allowing end-users to disable/enable and clear the filter. You can control the visibility of this panel using the GridControl.FilterPanelVisibility property.

© 2018 Developer Express Inc.




© 2018 Developer Express Inc.

# Grid Elements



The topics in this section provide basic information on GridControl's columns, rows and cells.

- Columns
- Rows
- Cells

© 2018 Developer Express Inc.

# Columns

Xamarin Controls > Grid > Grid Elements > Columns

In this section:

- Columns Overview
- Identify and Access Columns
- Create Columns and Bind Them to Data Fields
- Unbound Columns
- Fixed Columns

© 2018 Developer Express Inc.

# Columns Overview

Xamarin Controls > Grid > Grid Elements > Columns > Columns Overview

GridControl is a data-aware control designed to display tabular information from the bound data source. Data sources present this information using data fields and records. In GridControl, data fields are represented as columns, and records are displayed as rows. So, to display data, a grid should contain columns bound to fields of the underlying data source. In addition to columns bound to data fields, a grid can include unbound columns to display data values calculated according to a formula based on values of other columns.

Grid columns are stored in the GridControl.Columns collection. An individual column is specified by a GridColumn class descendant that corresponds to the type of data displayed in this column. The following column types are supported.

| Column Object | Description |
|---|---|
| TextColumn | A grid column to display and edit text values. |
| NumberColumn | A grid column to display and edit numeric values. |
| DateColumn | A grid column to display and edit date values. |
| SwitchColumn | A grid column to display Boolean values and select between two states. |
| ImageColumn | A column type that allows you to display images. |
| PickerColumn | A grid column allowing an end-user to edit a cell value by selecting an item from the predefined set. |
| TemplateColumn | A column type that allows you to define a custom template for column cells. |

Any column object provides a set of properties to adjust column settings (for example, to set the data display format - GridColumn.DisplayFormat, column width - GridColumn.Width, column visibility - GridColumn.IsVisible, column content alignment - GridColumn.ContentAlignment, etc.), as well as to manage and shape grid data (for example, to sort - GridColumn.SortOrder, GridColumn.SortIndex, GridColumn.SortMode, group - GridColumn.IsGrouped, GridColumn.GroupInterval, or filter data by column values - GridColumn.AutoFilterCondition, GridColumn.ColumnFilterMode, GridColumn.ImmediateUpdateAutoFilter, etc.).

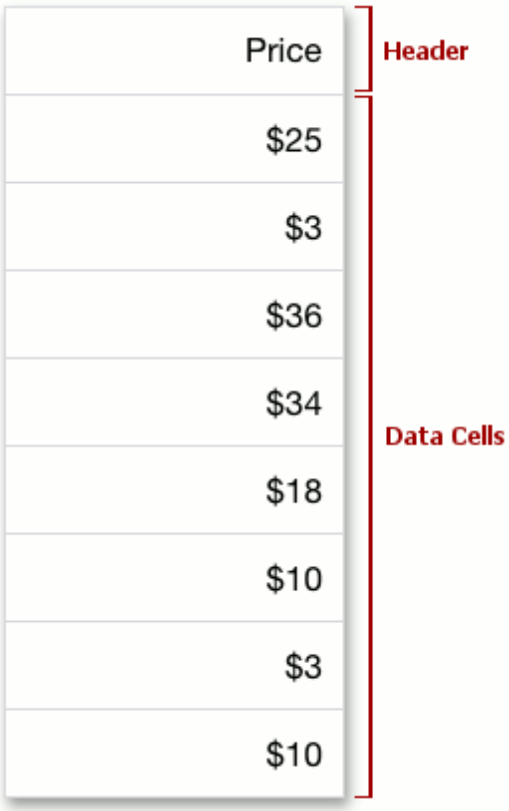
Each column consists of the following elements.

- A *column header* that displays the column caption and identifies the column for end-users. By default, a friendly caption is generated for a column based on its GridColumn.FieldName property. Friendly captions are generated by adding SPACE characters between parts of the field name starting with uppercase letters. For example, if the field name is "CustomerName", the display caption will be "Customer Name". If you need to replace the automatically created column caption with you own text, assign the corresponding string to the GridColumn.Caption property. To obtain a column caption as it is displayed in a column header, use the GridColumn.ActualCaption property.

To hide column headers, set the GridControl.ColumnHeadersVisibility property to **false**.

- *Data cells* that display values from the corresponding field in the bound data source, or calculated values if a column is unbound.

© 2018 Developer Express Inc.




**End-User Capabilities**

By default, an end-user can manage grid columns in the following ways.

- Select columns to be displayed or hidden in the grid using the Column Chooser.
- Resize column widths with the pinch and spread gestures.
- Sort data by column values by tapping a column header or tapping **Sort Ascending** or **Sort Descending** in a menu that appears when an end-user touches and holds a column cell or header.
- Group data by column values by tapping the corresponding item in a menu that appears when an end-user touches and holds a column.
- Filter data on the fly by column values using the auto-filter panel.
- Calculate data summaries (group and total) for columns.

**See Also**

Create Columns and Bind Them to Data Fields
Unbound Columns
Identify and Access Columns
Rows Overview

© 2018 Developer Express Inc.

# Identify and Access Columns



When working with GridControl, you may need to write code that accesses particular column properties or that passes column objects as method parameters, etc. You may also require column objects for event parameters. This topic provides information on how to access and identify columns programmatically.

### Identify Columns

The table below lists properties and XAML attributes that allow you to identify grid columns.

| Identifiers | Description |
|---|---|
| x:Name | Uniquely identifies a column so that it can be accessed from code-behind or general code.<br>**Xaml**<br>`<dxg:TextColumn x:Name="colProduct" FieldName="ProductName"/>` |
| GridColumn.FieldName | Specifies the name of a data source field to which a column is bound. |

### Access Columns

If a column is created at design time, you can specify its unique name using the **x:Name** attribute. In this instance, a column can be accessed by its name from code-behind:

**Xaml**

```
<dxGrid:GridControl.Columns>
    <dxGrid:TextColumn x:Name="colProduct" FieldName="ProductName"/>
    <dxGrid:NumberColumn x:Name="colUnitPrice" FieldName="UnitPrice"/>
</dxGrid:GridControl.Columns>
```

**C#**

```
colUnitPrice.AllowSort = false;
```

Since columns are contained within the grid's GridControl.Columns collection, an individual column can be accessed by the name of the field to which the required column is bound, or using indexed notation:

**Xaml**

```
<dxg:GridControl.Columns>
    <dxg:TextColumn FieldName="ProductName"/>
    <dxg:NumberColumn FieldName="UnitPrice"/>
</dxg:GridControl.Columns>
```

**C#**

```
TextColumn colProduct = grid.Columns["ProductName"];
NumberColumn colPrice = grid.Columns[1];
```

The column collection also provides the GridColumnCollection.GetColumnByFieldName method to access columns by field names.

To provide a column as a parameter for the following methods, you can pass a column object or string that specifies a column field name.

- GridControl.GetCellValue/GridControl.SetCellValue - Obtain or change a cell value.
- GridControl.FindRowByValue - Gets a row which contains the specified value in the specified column.
- GridControl.GroupBy - Groups data by the specified column.
- GridControl.SortBy - Sorts data by values in the specified column.

**See Also**
Identify and Access Rows

© 2018 Developer Express Inc.

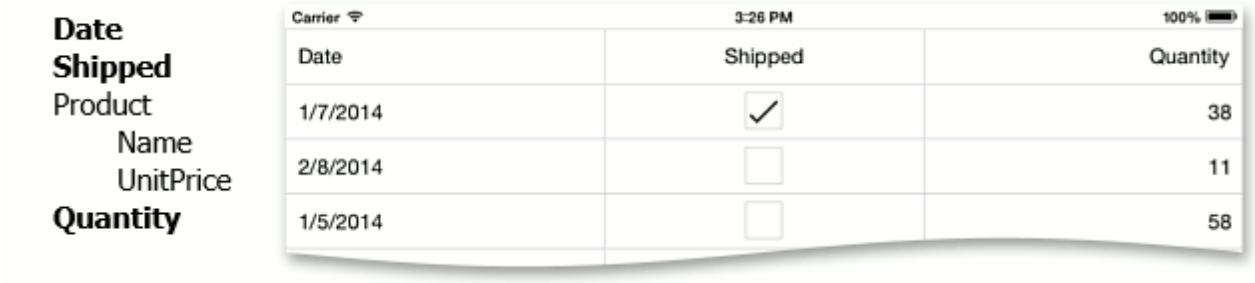
# Create Columns and Bind Them to Data Fields



To work with a GridControl, bind it to a data source. To do this, you need to create columns and bind them to data fields. There are two ways to add columns to a grid and associate them with data fields of the underlying data source.

- Create columns for all fields of the bound data source automatically.

By default, a GridControl instance automatically generates columns based on the bound data source - one column for one field of a data source object, except complex fields. In terms of objects, a complex field can be a field of the nested object. A complex field name is constructed in the "String1.String2" form. For example, in Lesson 1 of the Getting Started tutorial, a grid displays orders. Columns for the *Date*, *Shipped* and *Quantity* properties of an *Order* object are generated automatically, however columns for fields of a related *Product* are not created (you can create columns bound to these complex fields in the grid manually by using the "Product.Name" and "Product.UnitPrice" field names, as shown in Lesson 2).




The order of columns generated automatically in the grid is the same as the order of fields in the data source. You can prevent columns from being automatically created or set another mode of auto generating columns, using the GridControl.AutoGenerateColumnsMode property.

- Create columns and bind them to fields manually.

You can manually specify a collection of grid columns. To do this, create corresponding column objects, bind each column to the corresponding data field using the GridColumn.FieldName property and add columns to the GridControl.Columns collection in the order you want to show them in the grid. You can manage the grid's collection of columns in XAML or C# code.

**Create and Bind Columns in XAML**

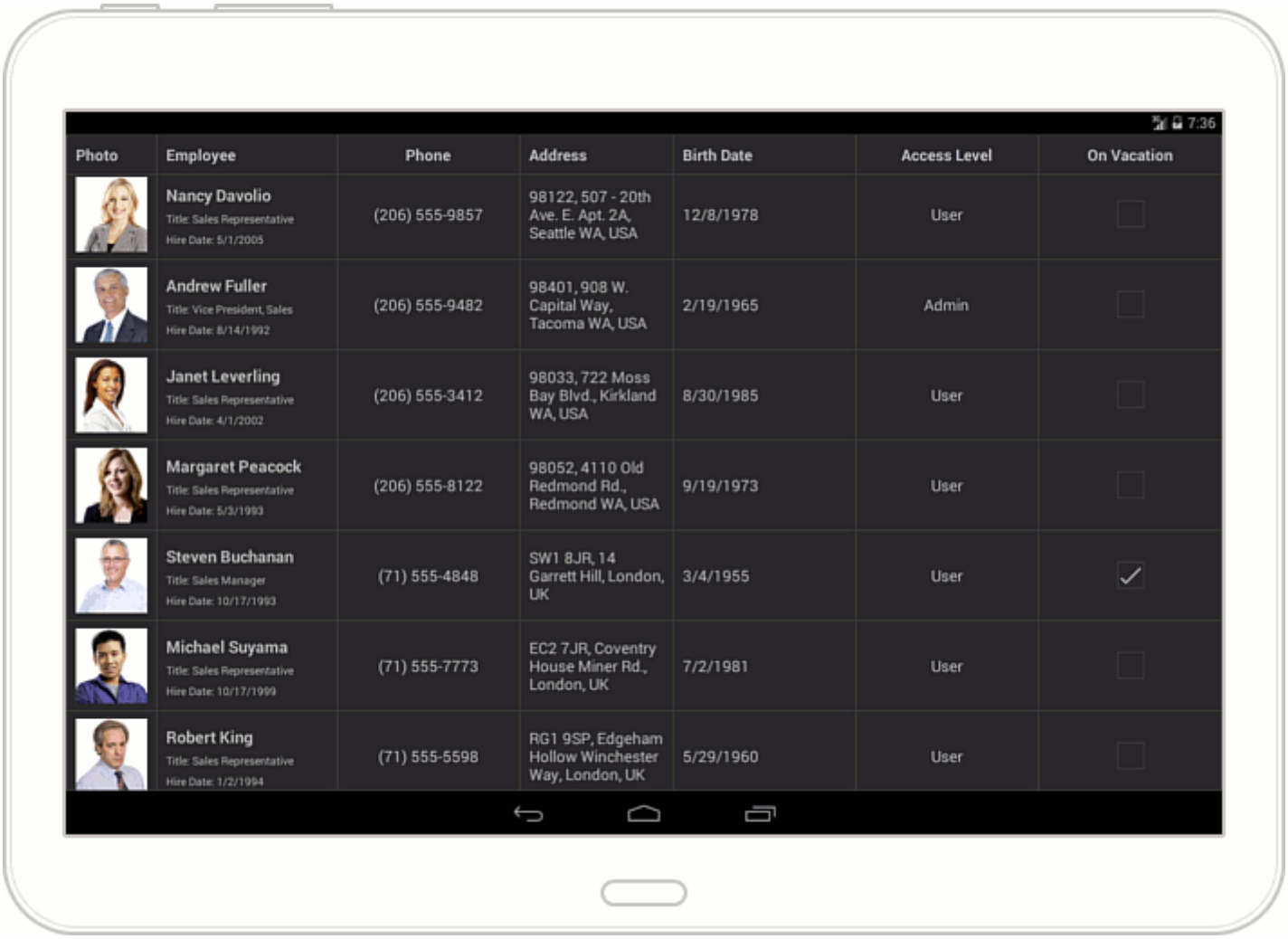
| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T208848. | |

This example shows how to add columns to GridControl to display and edit data of different types. The grid is bound to a collection of **Employee** objects. Each **Employee** object contains an employee's photo (image), name, position, phone, address (strings), hire and birth dates (DateTime values), a value that specifies an employee's access level, and a Boolean value indicating whether an employee is on vacation.

© 2018 Developer Express Inc.

| Photo | Employee | Phone | Address | Birth Date | Access Level | On Vacation |
|---|---|---|---|---|---|---|
| | **Nancy Davolio**<br>Title: Sales Representative<br>Hire Date: 5/1/2005 | (206) 555-9857 | 98122, 507 - 20th Ave. E. Apt. 2A, Seattle WA, USA | 12/8/1978 | User | ☐ |
| | **Andrew Fuller**<br>Title: Vice President, Sales<br>Hire Date: 8/14/1992 | (206) 555-9482 | 98401, 908 W. Capital Way, Tacoma WA, USA | 2/19/1965 | Admin | ☐ |
| | **Janet Leverling**<br>Title: Sales Representative<br>Hire Date: 4/1/2002 | (206) 555-3412 | 98033, 722 Moss Bay Blvd., Kirkland WA, USA | 8/30/1985 | User | ☐ |
| | **Margaret Peacock**<br>Title: Sales Representative<br>Hire Date: 5/3/1993 | (206) 555-8122 | 98052, 4110 Old Redmond Rd., Redmond WA, USA | 9/19/1973 | User | ☐ |
| | **Steven Buchanan**<br>Title: Sales Manager<br>Hire Date: 10/17/1993 | (71) 555-4848 | SW1 8JR, 14 Garrett Hill, London, UK | 3/4/1955 | User | ☑ |
| | **Michael Suyama**<br>Title: Sales Representative<br>Hire Date: 10/17/1999 | (71) 555-7773 | EC2 7JR, Coventry House Miner Rd., London, UK | 7/2/1981 | User | ☐ |
| | **Robert King**<br>Title: Sales Representative<br>Hire Date: 1/2/1994 | (71) 555-5598 | RG1 9SP, Edgeham Hollow Winchester Way, London, UK | 5/29/1960 | User | ☐ |

© 2018 Developer Express Inc.




Grid columns are stored in the GridControl.Columns collection. An individual column is specified by a GridColumn descendant, which is appropriate to the type of data contained in this column. In this example, to display information on employees and allow end-users to edit it, the following columns are created in the grid.

| Grid Column | Description |
|---|---|
| Photo<br>ImageColumn | This column displays employee photos which are images added to a project as embedded resources. |
| Employee<br>TemplateColumn | A cell in this column displays three employee properties: Name, Position and HireDate. A cell appearance is defined via a template. Each cell contains three Xamarin.Forms.Label elements within a Xamarin.Forms.Grid. Each Label element is bound to a property of the **Employee** class.<br><br>A cell template data context is specified by the CellData object. Its CellData.Value property provides access to a value of a data field assigned to the column's GridColumn.FieldName property. In this example, a column cell displays not only this field value but also values of two more fields. So, it is required to specify the whole data row as a binding source for the template. To do this, set the BindingContext property of the Grid to CellData.Source. Note that BindingContext specified for the Grid is inherited by all its children. This means that all Label elements have the same BindingContext as the Grid, and you can specify |

© 2018 Developer Express Inc.

| | |
|---|---|
| | their simple bindings to properties of that object (Employee).<br><br>Use the GridColumn.AllowSort, GridColumn.AllowGroup and GridColumn.IsReadOnly properties to prevent end-users from sorting and grouping data in the grid by this column and disable data editing. |
| Phone<br>NumberColumn | The employee's **Phone** property is of the string type. The keyboard allowing text input is automatically displayed when an end-user activates this column cell to edit an employee's phone. |
| Address<br>TextColumn | This column is bound to the employee's **Address** string property. The keyboard allowing text input is automatically displayed when an end-user activates this column cell to edit an employee's address. |
| Birth Date<br>DateColumn | This column displays and allows editing employee birth days. The GridColumn.DisplayFormat property specifies the format of displaying dates. |
| Access Level<br>PickerColumn | The employee's **Access** property is of the **AccessLevel** type which is an enumeration with values **Admin** and **User**. To allow editing an employee's access level by selecting one of two enumeration values, this property is bound to the grid's picker column. |
| On Vacation<br>SwitchColumn | The employee's **OnVacation** property is of the Boolean type, so it is bound to the grid's switch column which allows editing a cell value by selecting between two states. |

**C#**

```
 (Employee.cs)
public enum AccessLevel {
    Admin,
    User
}
public class Employee : ModelObject {
    string name;
    public string Name {
        get { return name; }
        set {
            name = value;
            if (Photo == null)
                Photo = ImageSource.FromResource (value.Replace (" ", String.Empty));
        }
    }
    public Employee(string name) {
        this.Name = name;
    }
    public ImageSource Photo { get; set; }
    public DateTime BirthDate { get; set; }
    public DateTime HireDate { get; set; }
    public string Position { get; set; }
    public string Address { get; set; }
    public string Phone { get; set; }
    public AccessLevel Access { get; set; }
    public bool OnVacation { get; set; }
}
```

**C#**

© 2018 Developer Express Inc.

```
 (EmployeesRepository.cs)
public abstract class EmployeesRepository {
    readonly ObservableCollection<Employee> employees;
    public EmployeesRepository() {
        this.employees = new ObservableCollection<Employee>();
    }
    public ObservableCollection<Employee> Employees {
        get { return employees; }
    }
}
public class TestEmployeesRepository : EmployeesRepository {
    public TestEmployeesRepository() : base() {
        GenerateCustomers();
    }
    void GenerateCustomers() {
        Employees.Add(
            new Employee("Nancy Davolio") {
                BirthDate = new DateTime(1978, 12, 8),
                HireDate = new DateTime(2005, 5, 1),
                Position = "Sales Representative",
                Address = "98122, 507 - 20th Ave. E. Apt. 2A, Seattle WA, USA",
                Phone = "(206) 555-9857",
                Access = AccessLevel.User,
                OnVacation = false
            }
        );
        Employees.Add(
            new Employee("Andrew Fuller") {
                BirthDate = new DateTime(1965, 2, 19),
                HireDate = new DateTime(1992, 8, 14),
                Position = "Vice President, Sales",
                Address = "98401, 908 W. Capital Way, Tacoma WA, USA",
                Phone = "(206) 555-9482",
                Access = AccessLevel.Admin,
                OnVacation = false
            }
        );
        Employees.Add(
            new Employee("Janet Leverling") {
                BirthDate = new DateTime(1985, 8, 30),
                HireDate = new DateTime(2002, 4, 1),
                Position = "Sales Representative",
                Address = "98033, 722 Moss Bay Blvd., Kirkland WA, USA",
                Phone = "(206) 555-3412",
                Access = AccessLevel.User,
                OnVacation = false
            }
        );
        Employees.Add(
            new Employee("Margaret Peacock") {
                BirthDate = new DateTime(1973, 9, 19),
                HireDate = new DateTime(1993, 5, 3),
                Position = "Sales Representative",
                Address = "98052, 4110 Old Redmond Rd., Redmond WA, USA",
                Phone = "(206) 555-8122",
                Access = AccessLevel.User,
                OnVacation = false
            }
        );
        Employees.Add(
            new Employee("Steven Buchanan") {
                BirthDate = new DateTime(1955, 3, 4),
                HireDate = new DateTime(1993, 10, 17),
                Position = "Sales Manager",
                Address = "SW1 8JR, 14 Garrett Hill, London, UK",
```

© 2018 Developer Express Inc.

```
                Phone = "(71) 555-4848",
                Access = AccessLevel.User,
                OnVacation = true
            }
        );
        Employees.Add(
            new Employee("Michael Suyama") {
                BirthDate = new DateTime(1981, 7, 2),
                HireDate = new DateTime(1999, 10, 17),
                Position = "Sales Representative",
                Address = "EC2 7JR, Coventry House Miner Rd., London, UK",
                Phone = "(71) 555-7773",
                Access = AccessLevel.User,
                OnVacation = false
            }
        );
        Employees.Add(
            new Employee("Robert King") {
                BirthDate = new DateTime(1960, 5, 29),
                HireDate = new DateTime(1994, 1, 2),
                Position = "Sales Representative",
                Address = "RG1 9SP, Edgeham Hollow Winchester Way, London, UK",
                Phone = "(71) 555-5598",
                Access = AccessLevel.User,
                OnVacation = false
            }
        );
        Employees.Add(
            new Employee("Laura Callahan") {
                BirthDate = new DateTime(1985, 1, 9),
                HireDate = new DateTime(2004, 3, 5),
                Position = "Inside Sales Coordinator",
                Address = "98105, 4726 - 11th Ave. N.E., Seattle WA, USA",
                Phone = "(206) 555-1189",
                Access = AccessLevel.User,
                OnVacation = true
            }
        );
        Employees.Add(
            new Employee("Anne Dodsworth") {
                BirthDate = new DateTime(1980, 1, 27),
                HireDate = new DateTime(2004, 11, 15),
                Position = "Sales Representative",
                Address = "WG2 7LT, 7 Houndstooth Rd., London, UK",
                Phone = "(71) 555-4444",
                Access = AccessLevel.User,
                OnVacation = false
            }
        );
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl ItemsSource="{Binding Employees}" RowHeight="100">
    <dxGrid:GridControl.Columns>
        <dxGrid:ImageColumn FieldName="Photo" Width="100" />
        <dxGrid:TemplateColumn FieldName="Name" Caption="Employee" Width="
                               IsReadOnly="true" AllowGroup="False" AllowS
          <dxGrid:TemplateColumn.DisplayTemplate>
            <DataTemplate>
              <Grid BindingContext="{Binding Source}" Padding="0,15,0,15">
                <Grid.RowDefinitions>
                  <RowDefinition Height="Auto" />
                  <RowDefinition Height="Auto" />
                  <RowDefinition Height="Auto" />
                </Grid.RowDefinitions>
                <Label Text="{Binding Name}" Font="Bold, 18" Grid.Row="0"
                <Label Text="{Binding Position, StringFormat = 'Title: {0}
                <Label Text="{Binding HireDate, StringFormat = 'Hire Date:
              </Grid>
            </DataTemplate>
          </dxGrid:TemplateColumn.DisplayTemplate>
        </dxGrid:TemplateColumn>
        <dxGrid:NumberColumn FieldName="Phone" ContentAlignment="Center"/>
        <dxGrid:TextColumn FieldName="Address" Width="170"/>
        <dxGrid:DateColumn FieldName="BirthDate" DisplayFormat="d"/>
        <dxGrid:PickerColumn FieldName="Access" Caption="Access Level" Con
        <dxGrid:SwitchColumn FieldName="OnVacation"/>
    </dxGrid:GridControl.Columns>
</dxGrid:GridControl>
```

### Create and Bind Columns in C#

The following example shows how to create columns and bind them to data fields using C# code.

**C#**

```
using DevExpress.Mobile.DataGrid;
// ...
GridControl grid = new GridControl ();
grid.Columns.Add (new TextColumn (){FieldName = "Product.Name", Caption =
grid.Columns.Add (new NumberColumn (){FieldName = "Product.UnitPrice", Cap
grid.Columns.Add (new NumberColumn (){FieldName = "Quantity"});
grid.Columns.Add (new DateColumn (){FieldName = "Date", DisplayFormat="d"
grid.Columns.Add (new SwitchColumn (){FieldName = "Shipped"});
```

### See Also

Unbound Columns
Identify and Access Columns

© 2018 Developer Express Inc.

# Unbound Columns

Xamarin Controls > Grid > Grid Elements > Columns > Unbound Columns

GridControl supports bound and unbound columns. Bound columns obtain their data from data fields of the grid's underlining data source. Unbound columns are not bound to any field of the data source. These columns must be populated with data manually.

To create an unbound column in the grid, add an appropriate column object to the GridControl.Columns collection and set the following column properties.

| Property | Description |
|---|---|
| GridColumn.FieldName | Set this property to a unique string. This value must not match field names of other columns and it must not match field names existing in the GridControl's underlying data source. |
| GridColumn.UnboundType | Set this property to a value that identifies the type of data the column is supposed to display (Boolean, DateTime, Decimal, Integer, String or Object). Do not set the column's **UnboundType** property to UnboundColumnType.Bound, as this would identify the column as bound. |

After a column is created, use one of the following two approaches to implement logic for populating a column with data.

- Assign a formula (string expression) to the GridColumn.UnboundExpression property to automatically evaluate values for an unbound column. Expressions allow you to calculate values based on values of other columns. You can use constants, various functions and operators in expressions.
- Handle the GridControl.CustomUnboundColumnData event that allows you to implement custom logic of any complexity for populating unbound columns with data.

Use also this event (but not expressions), if you need to save the changes made by end-users in unbound columns. Event's GridColumnDataEventArgs.IsGetData and GridColumnDataEventArgs.IsSetData parameters determine the event's current mode of operation.

- **Display Unbound Data**

If the **IsGetData** parameter is set to **true** (and consequently the **IsSetData** parameter is set to **false**), the event handler should supply data for an unbound column. A value should be assigned to the GridColumnDataEventArgs.Value parameter according to the row currently being processed.

- **Save Changes**

If the **IsSetData** parameter is set to **true** (and consequently the **IsGetData** parameter is set to **false**), the event has been fired as a result of data modifications within a grid. In this case, the **Value** parameter contains modified data and should be stored for further use.

There is no difference between working with bound and unbound columns. You can sort, group, display summaries and filter unbound columns in the same manner as bound columns.

By default, data editing is enabled in unbound column cells. However, you may wish to prevent end-users from editing unbound columns. To make a column read-only, set the GridColumn.IsReadOnly property to **true**.

**Note**

The GridControl cannot operate with only unbound columns. It must be bound to a data source using its GridControl.ItemsSource property.

### Example - Implement Unbound Columns Using Expressions

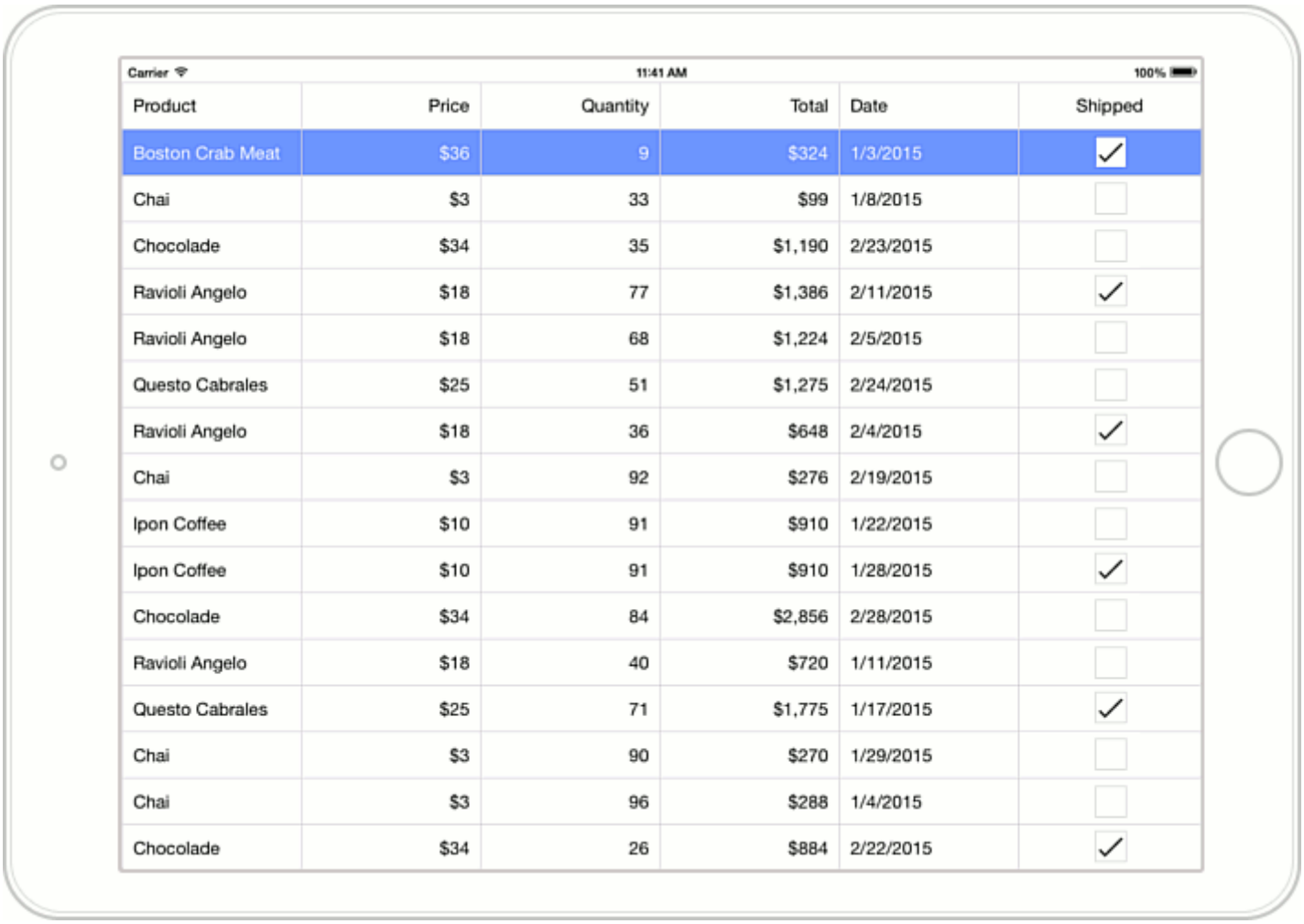
**Show Me**

A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T178764.

This example shows how to create and customize grid columns for displaying and editing data of different types (text, numbers, dates and Boolean values). The specified collection contains columns bound to the data source fields (*Product.Name*, *Product.UnitPrice*, *Quantity*, *Date* and *Shipped*) and one unbound column (*Total*) displaying data values calculated according to a formula based on the values of other columns.

© 2018 Developer Express Inc.

The image below illustrates the result.




© 2018 Developer Express Inc.

**Xaml**

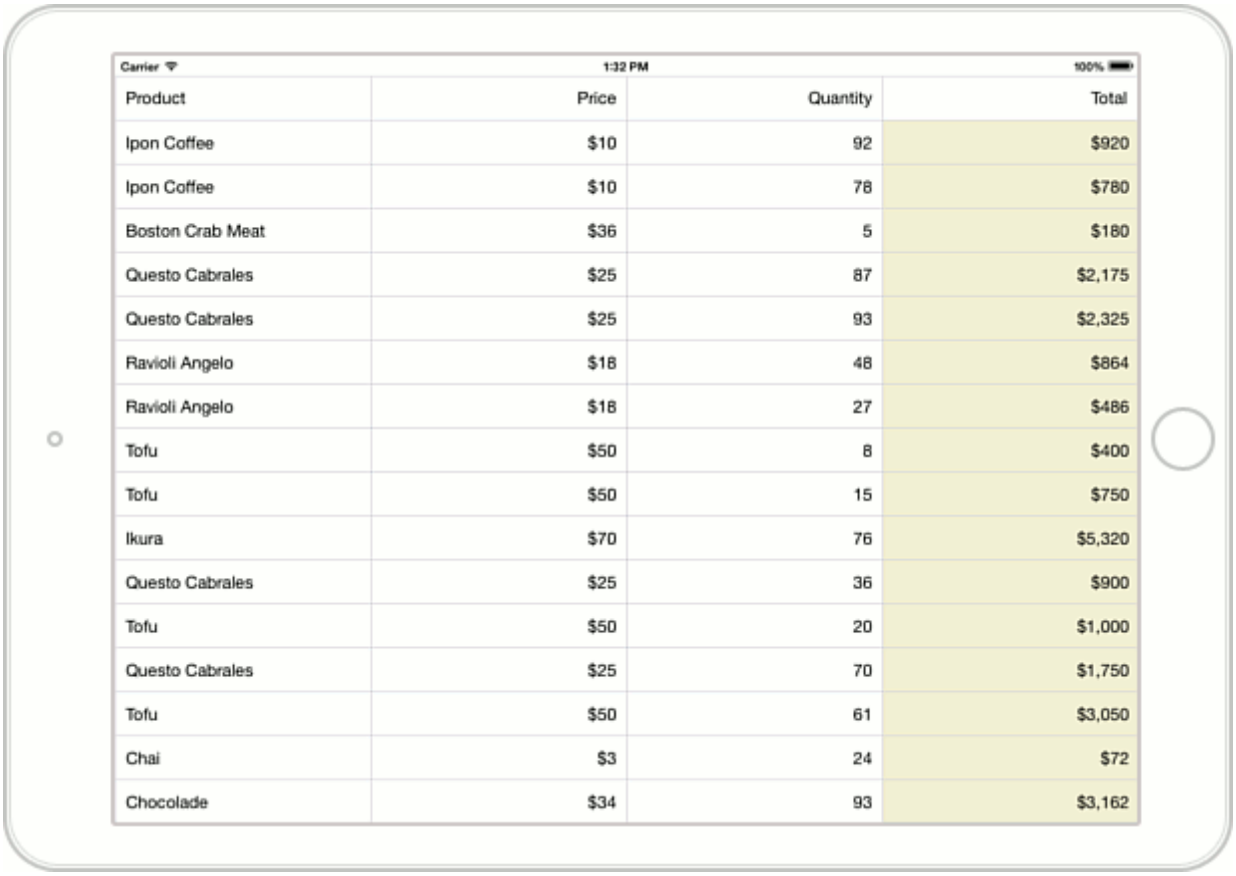
```
 (MainPage.xaml)
<dxGrid:GridControl.Columns>
        <dxGrid:TextColumn FieldName="Product.Name" Caption = "Product" Width = "170"
                           SortOrder = "Descending" SortIndex = "0"/>
        <dxGrid:NumberColumn FieldName="Product.UnitPrice" Caption = "Price" DisplayFormat="C0"/>
        <dxGrid:NumberColumn FieldName="Quantity"
                             SortOrder = "Ascending" SortIndex = "1"/>
        <dxGrid:NumberColumn FieldName="Total"
                             UnboundType="Integer" UnboundExpression="[Quantity] * [Product.UnitPrice]"
                             DisplayFormat="C0" IsReadOnly="True"/>
        <dxGrid:DateColumn FieldName="Date" DisplayFormat="d"
                           IsGrouped = "true" GroupInterval = "Date"/>
        <dxGrid:SwitchColumn FieldName="Shipped" AllowSort = "False" AllowAutoFilter="false"/>
    </dxGrid:GridControl.Columns>
```

**Example - Implement Unbound Columns Using Event**

Assume that the GridControl instance is bound to a collection of orders. Each order has the "Product", "UnitPrice"and "Quantity" fields. The example below shows how to add an unbound column to the grid to display the amount of each order according to the expression: UnitPrice*Quantity.

The result is displayed below:

© 2018 Developer Express Inc.



| Product | Price | Quantity | Total |
|---|---|---|---|
| Ipon Coffee | $10 | 92 | $920 |
| Ipon Coffee | $10 | 78 | $780 |
| Boston Crab Meat | $36 | 5 | $180 |
| Questo Cabrales | $25 | 87 | $2,175 |
| Questo Cabrales | $25 | 93 | $2,325 |
| Ravioli Angelo | $18 | 48 | $864 |
| Ravioli Angelo | $18 | 27 | $486 |
| Tofu | $50 | 8 | $400 |
| Tofu | $50 | 15 | $750 |
| Ikura | $70 | 76 | $5,320 |
| Questo Cabrales | $25 | 36 | $900 |
| Tofu | $50 | 20 | $1,000 |
| Questo Cabrales | $25 | 70 | $1,750 |
| Tofu | $50 | 61 | $3,050 |
| Chai | $3 | 24 | $72 |
| Chocolade | $34 | 93 | $3,162 |

**Xaml**

```
<dxGrid:GridControl ItemsSource="{Binding Orders}"
                    CustomUnboundColumnData="OnCustomUnboundColumnData"
                    CustomizeCell="OnCustomizeCell">
    <dxGrid:GridControl.Columns>
         <dxGrid:TextColumn FieldName="Product.Name" Caption="Product"/>
         <dxGrid:NumberColumn FieldName="Product.UnitPrice" Caption="Price" DisplayFormat="C0"/>
         <dxGrid:NumberColumn FieldName="Quantity" />
         <dxGrid:NumberColumn FieldName="Total" IsReadOnly="True"
                              UnboundType="Decimal" DisplayFormat="C0"/>
    </dxGrid:GridControl.Columns>
</dxGrid:GridControl>
```

**C#**

```
// Returns the total for a specific row.
decimal getTotalValue(GridControl grid, int rowHandle) {
    decimal unitPrice = Convert.ToDecimal(grid.GetCellValue(rowHandle, "Product.UnitPrice"));
    decimal quantity = Convert.ToDecimal(grid.GetCellValue(rowHandle, "Quantity"));
    return unitPrice * quantity;
}
// Provides data for the Total column.
void OnCustomUnboundColumnData(object sender, GridColumnDataEventArgs e) {
    if (e.Column.FieldName == "Total" && e.IsGetData)
        e.Value = getTotalValue (e.Source, e.RowData.RowHandle);
}
// Customizes the appearance settings.
void OnCustomizeCell(CustomizeCellEventArgs e) {
if (e.FieldName == "Total") {
        e.BackgroundColor = Color.FromRgb(247, 240, 213);
        e.Handled = true;
    }
}
```

© 2018 Developer Express Inc.

**See Also**
Create Columns and Bind Them to Data Fields
Identify and Access Columns

© 2018 Developer Express Inc.

# Fixed Columns

Xamarin Controls > Grid > Grid Elements > Columns > Fixed Columns

GridControl allows you to anchor (fix) columns to the left or right edge, so they are not horizontally scrolled with the grid. This can be useful when columns should always be displayed onscreen, regardless of scrolling.

**Note**

Using horizontal scrolling only makes sense when the auto-column width feature is disabled (that is when the GridControl.ColumnsAutoWidth property is set to **false**) and the total width of columns exceeds the grid's width. Otherwise, fixed columns and horizontal scrolling are disabled.

On the image below, the **Photo** and **Name** columns are fixed to the left and the **Total** column is fixed to the right.

| Photo | Name | Position | Phone | Ema | Total |
|---|---|---|---|---|---|
| | Laura Callahan | Inside Sales Coordinator | (206) 555-1189 | Laur | $0.00 |
| | Steven Buchanan | Sales Manager | (71) 555-4848 | Stev | $10.00 |
| | Nancy Davolio | Sales Representative | (206) 555-9857 | Nan | $40.00 |
| | Margaret Peacock | Sales Representative | (206) 555-8122 | Marg | $90.00 |
| | Nancy Davolio | Sales Representative | (206) 555-9857 | Nan | $160.00 |
| | Michael | Sales | (71) 555-7773 | Mich | $250.00 |

To fix a column position, use its GridColumn.FixedStyle property. By default, this property is set to FixedStyle.None and a column is not fixed. Set the **FixedStyle** property to FixedStyle.Left or FixedStyle.Right to anchor the column to the left or right grid edge, respectively.

Unfixed columns can be horizontally scrolled by end-users. Also, you can scroll a grid from the code using the GridControl.HorizontalScrollOffset property. It allows you to set (or get) a scrolling offset for the grid. The code below sets the **Position** column's width as a scrolling offset.

© 2018 Developer Express Inc.

**C#**

```
using DevExpress.Mobile.DataGrid;
    // ...
    grid.HorizontalScrollOffset = grid.Columns["Position"].Width;
    // ...
```

**Tip**

To speed up the grid's performance, enable the GridControl.AllowHorizontalScrollingVirtualization option. If this option is enabled, a grid renders only those columns that are displayed onscreen. Each time a grid is horizontally scrolled, visible (displayed onscreen) columns are rendered again. Otherwise, if the **AllowHorizontalScrollingVirtualization** property is set to **false**, the grid renders all columns.

© 2018 Developer Express Inc.

# Rows



In this section:


- Rows Overview
- Identify and Access Rows


© 2018 Developer Express Inc.

# Rows Overview

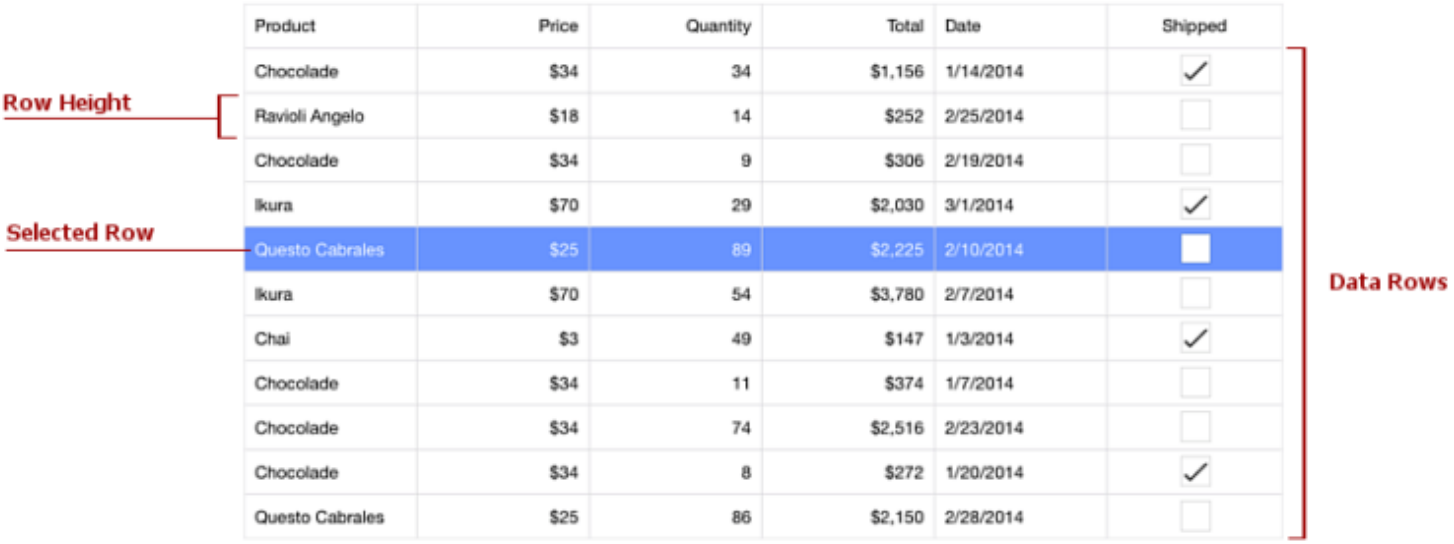
Xamarin Controls > Grid > Grid Elements > Rows > Rows Overview

A GridControl presents its data with the help of rows and columns. This document provides an overview of row types available in the grid. To get detailed information on grid columns, review the Columns section.

### Data Rows

A data source represents tabular information using data fields and records. A GridControl bound to a data source displays data fields as columns, and records as data rows. These rows are displayed one after another. Record values are arranged horizontally within them.

All rows have the same height. You can set it manually, using the GridControl.RowHeight property.




| Product | Price | Quantity | Total | Date | Shipped |
|---|---|---|---|---|---|
| Chocolade | $34 | 34 | $1,156 | 1/14/2014 | ✓ |
| Ravioli Angelo | $18 | 14 | $252 | 2/25/2014 | |
| Chocolade | $34 | 9 | $306 | 2/19/2014 | |
| Ikura | $70 | 29 | $2,030 | 3/1/2014 | ✓ |
| Questo Cabrales | $25 | 89 | $2,225 | 2/10/2014 | |
| Ikura | $70 | 54 | $3,780 | 2/7/2014 | |
| Chai | $3 | 49 | $147 | 1/3/2014 | ✓ |
| Chocolade | $34 | 11 | $374 | 1/7/2014 | |
| Chocolade | $34 | 74 | $2,516 | 2/23/2014 | |
| Chocolade | $34 | 8 | $272 | 1/20/2014 | ✓ |
| Questo Cabrales | $25 | 86 | $2,150 | 2/28/2014 | |

Each row is identified by a unique integer value - row handle. Data row handles start from zero. You can access data values in a row using the GridControl.GetCellValue, GridControl.SetCellValue and GridControl.GetRow methods with the passed handle of the required data row.

When an end-user taps a data row in the grid, this row is selected. To set a selected row programmatically, use the GridControl.SelectedRowHandle method.

### Group Rows

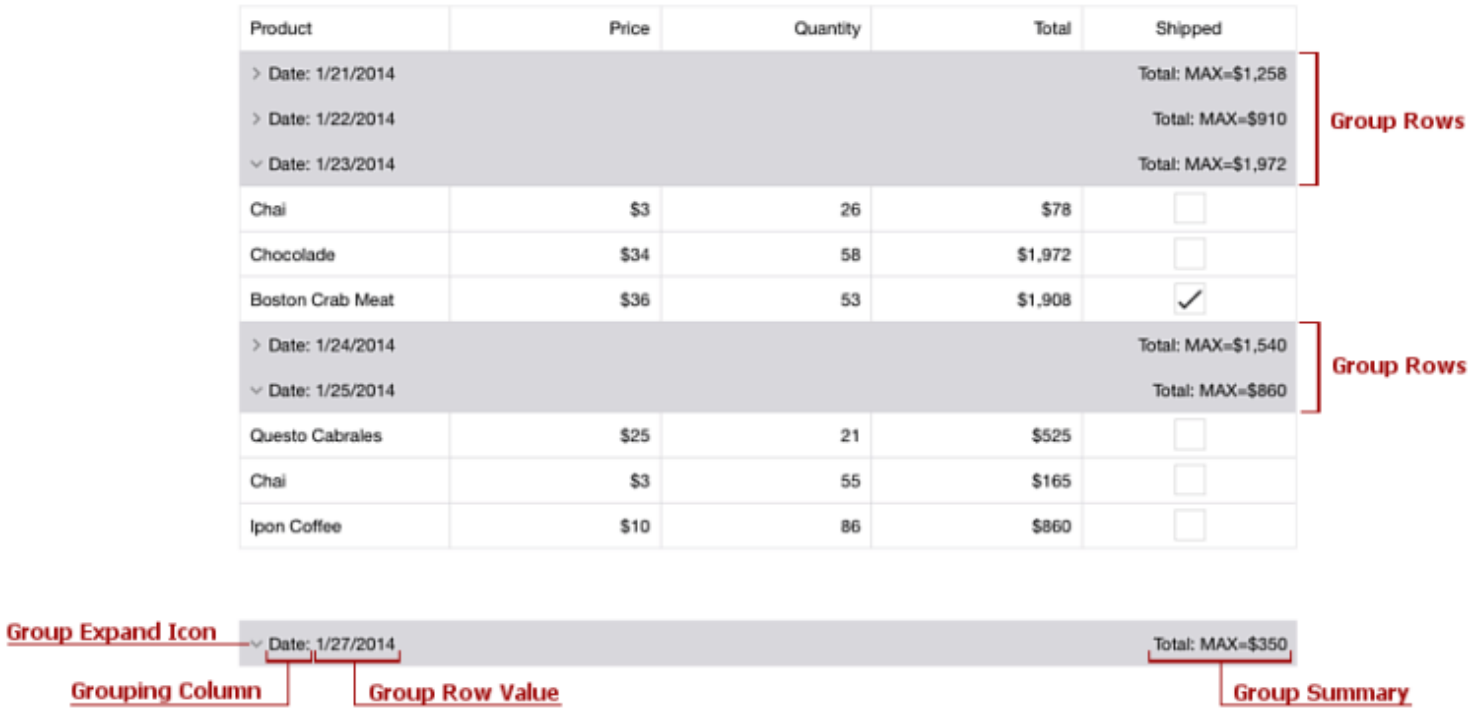
When grouping is applied in a GridControl, the grid displays group rows. These rows function to separate data row groups and can also be expanded and collapsed to show and hide data rows that belong to a group.

Rows are always sorted against the grouping column. If data is grouped against a column which is not sorted, the grid will automatically apply sorting to the column in ascending order.

A group row displays the following elements:

- Group Expand Icon - Indicates whether the group is expanded or collapsed.
- Grouping Column - A caption of a column by whose values data is grouped.
- Group Row Value - A data value of a grouping column for which the group is created.
- Group Summary - A group summary value.

© 2018 Developer Express Inc.




Each group row is identified by a unique integer value - row handle. Group row handles are negative (starting from **-1**). You can obtain detailed information on an individual data group using the GridControl.GetGroupInfo method with the corresponding group row handle passed as a parameter. This method returns an object implementing the IGroupInfo interface that provides a set of properties to get information on a particular data group. For example, the following table lists properties that you can use to get information displayed in the specified group row.

| Property | Description |
|---|---|
| IGroupInfo.FieldName | Gets the field name of the column by which data is grouped. |
| IGroupInfo.Value | Gets the value for which the group is created (a value which is displayed in a group row). |
| IGroupInfo.Summaries | Gets the list of objects that contain information on group summaries calculated for the data group and displayed within the group row. |
| IGroupInfo.IsCollapsed | Indicates whether the group is collapsed. |

When an end-user taps a group row, this group is collapsed or expanded. You can disable this functionality by setting the GridControl.AllowGroupCollapse property to **false**. To collapse or expand group rows programmatically, use the GridControl.CollapseGroupRow, GridControl.CollapseAllGroups, GridControl.ExpandGroupRow or GridControl.ExpandAllGroups method. The GridControl.GroupsInitiallyExpanded property allows you to specify whether groups should appear expanded or collapsed immediately after a grouping is applied in the grid.

Before a group row is collapsed or expanded, the grid raises the GridControl.GroupRowCollapsing or GridControl.GroupRowExpanding event, allowing you to cancel the corresponding operation. After a group row has been collapsed or expanded, the grid notifies you by raising the GridControl.GroupRowCollapsed or GridControl.GroupRowExpanded event.

When an end-user taps and holds a group row, a context menu appears allowing end-users to manage groups (calculate group summaries, collapse and expand all groups at once, and remove grouping from the grid). To prevent this menu from being shown, set the GridControl.IsGroupRowMenuEnabled property to **false**.

**See Also**
Identify and Access Rows
Columns

© 2018 Developer Express Inc.

# Identify and Access Rows



When using GridControl, you may need to perform operations on rows using code. For instance, to focus (select) a specific row in code, you first need to get the corresponding row identifier and assign it to the GridControl.SelectedRowHandle property. Alternatively, you may wish to identify the type of the row or row value that is currently hovered. In these cases, you will need to know how to identify rows within the grid.

### Identify Rows - Row Handles

Rows in a GridControl are identified by unique integer values - **row handles**. A grid associates its rows with these values using the following rules:

- Each row has a row handle which uniquely identifies it, regardless of whether a row is currently visible (a row might not be visible due to scrolling, and it can also be hidden within a collapsed group).
- Data row handles start from zero. The zero row handle is assigned to the first data row within a grid. In this case, "first" implies the visual row order, but does not necessarily refer to the record's position within the bound data source. Then, successive integer values are assigned to additional data rows. The second data row handle is **1**, the third data row handle is **2**, etc. Note that modifying the row order changes the handles assigned to rows. This occurs, for example, when grid data is sorted or grouped. As a result, a specific row handle may refer to different rows at different times.
- Group row handles are negative, starting from **-1**. Again, group row handles change when rows are reordered.

| Product | Price | Amount | Total | Row Handles |
|---|---|---|---|---|
| Date: 1/1/2015 | | | | -1 |
| Boston Crab Meat | $36 | 13 | $468 | 0 |
| Date: 1/2/2015 | | | | -2 |
| Chai | $3 | 69 | $207 | 1 |
| Questo Cabrales | $25 | 72 | $1,800 | 2 |
| Chocolade | $34 | 47 | $1,598 | 3 |
| Date: 1/3/2015 | | | | -3 |
| Questo Cabrales | $25 | 29 | $725 | 4 |
| Tofu | $50 | 46 | $2,300 | 5 |
| Date: 1/4/2015 | | | | -4 |
| Chai | $3 | 17 | $51 | 6 |

### Obtain Row Handles

| Member | Description |
|---|---|
| GridControl.FindRowByValue | Searches for the value in the column and returns the handle of the corresponding row. |
| GridControl.GetRowHandle | Returns the handle of the row which corresponds to the specified record in the data source. |
| GridControl.SelectedRowHandle | Gets or sets the selected row handle. This is a bindable property. |

### Use Row Handles

Row handles are used by GridControl to reference its rows. For instance, the GridControl.SelectedRowHandle property specifies the selected row by its handle. The GridControl.SelectionChanged event passes the currently selected row handle as a

© 2018 Developer Express Inc.

parameter. When accessing (GridControl.GetCellValue) or changing (GridControl.SetCellValue) cell values, you need to provide the appropriate row handle.

To obtain information on an individual group of rows, use the GridControl.GetGroupInfo and GridControl.GetGroupRowValue method with the appropriate group row handle passed as a parameter. To collapse or expand a group of rows, call the GridControl.CollapseGroupRow or GridControl.ExpandGroupRow method with the specified group row handle. When collapsing or expanding a group row, the GridControl.GroupRowCollapsing, GridControl.GroupRowCollapsed, GridControl.GroupRowExpanding and GridControl.GroupRowExpanded events also identify the affected group row by its handle.

If you need to programmatically scroll the grid to the specified row or delete a row, pass required row handle to the GridControl.ScrollToRow or GridControl.DeleteRow method, respectively.

### Obtain Data Rows and Row Indexes in the Data Source

Row handles reflect the visual order of rows in a grid, and these may change as rows change their positions. In many cases, the best solution for referring to specific rows is to use objects that represent records in the bound data source. To refer to a particular row in a data source, you can use a row's index in a data source. To obtain a row's list index by specifying its handle, use the GridControl.GetSourceRowIndex method. To obtain a row object that corresponds to a row with the specified handle, use GridControl.GetRow.

### See Also

Identify and Access Columns

© 2018 Developer Express Inc.

# Cells

Xamarin Controls > Grid > Grid Elements > Cells

GridControl displays data using columns and rows. These elements correspond to bound data source fields and records respectively. Each row displays data cells arranged in a column layout. In other words, data cells are located at the intersection of rows and columns. Each data cell corresponds to a field value within a record in a data source. Cells are used to display and edit data.

The grid provides a set of methods allowing you to obtain and modify its cell values, regardless of whether end-users are allowed to edit data. The required cell is identified by specifying its owning row and column. The row is specified by its handle. The column can be specified by its name or field name. For detailed information on how to obtain row handles and identify columns, see Identify and Access Rows and Identify and Access Columns.

### Get Cell Values

GridControl provides the following methods for obtaining cell values.

| Method | Description |
|---|---|
| GridControl.GetCellValue | Returns the value of the specified data cell. |
| GridControl.GetGroupRowValue, IGroupInfo.Value | Returns the value for which the group is created (a value which is displayed in a group row). |
| GridControl.GetCellDisplayText | Returns the text displayed within the specified cell. |

It is also possible to get cell values using the functionality provided by the grid's data source. To do this, call the GridControl.GetRow method. It returns an IRowData object providing access to a grid's row data. Then you can use this object's IRowData.GetFieldValue method to get values of the specified fields within the processed row, or use the IRowData.DataObject property to access an object representing a data source's record and then call its methods to retrieve field values.

### Set Cell Values

To set a value of the specified cell, use the GridControl.SetCellValue method.

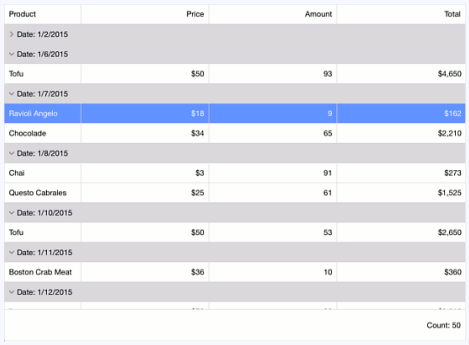
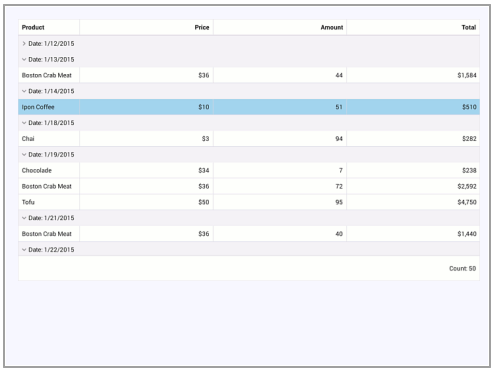
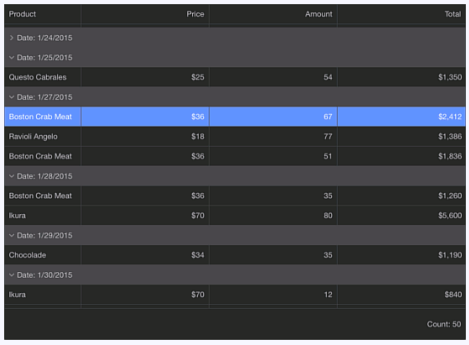
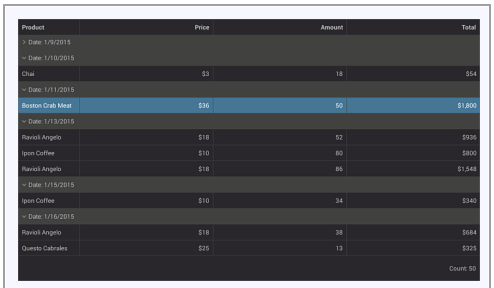
To invoke an in-place editor programmatically, call the GridControl.OpenEditor method. When an end-user closes an in-place editor to apply or cancel changed cell values, the GridControl.EndRowEdit event is raised.

© 2018 Developer Express Inc.

# Themes

Xamarin Controls > Grid > Themes Overview

DevExpress provides the **Light** and **Dark** built-in themes that you can apply to GridControl to change its appearance.

| Theme | iOS | Android |
|---|---|---|
| Light<br>(Default for iOS) |  |  |
| Dark<br>(Default for Android) |  |  |

### Apply Predefined Theme

The Themes class stores available theme names. To change the theme, assign **Themes** class's appropriate field to the ThemeManager.ThemeName property.

**C#**

```
using Xamarin.Forms;
using DevExpress.Mobile.DataGrid.Theme;
    // ...
    ThemeManager.ThemeName = Themes.Dark;
    // ...
```

### Customize Applied Theme

To customize the applied theme, use the ThemeBase class. It provides you with various customizers that allow you to change a theme's appearance and apply it to the grid. You can use these customizers to change background colors, fonts and others attributes of the desired grid element (header, group row and so on). For more information, see the How to: Customize Themes document.

**Important**

To apply the customization, call the ThemeManager.RefreshTheme method.

To track the moment the theme changes, either handle the ThemeManager.ThemeChanged event or use the IThemeChangingHandler.OnThemeChanged method.

### Define a Custom Theme

To define a custom grid theme, you can either inherit from the LightTheme or DarkTheme class, or manually implement an appropriate interface.

© 2018 Developer Express Inc.

# Conditional Formatting

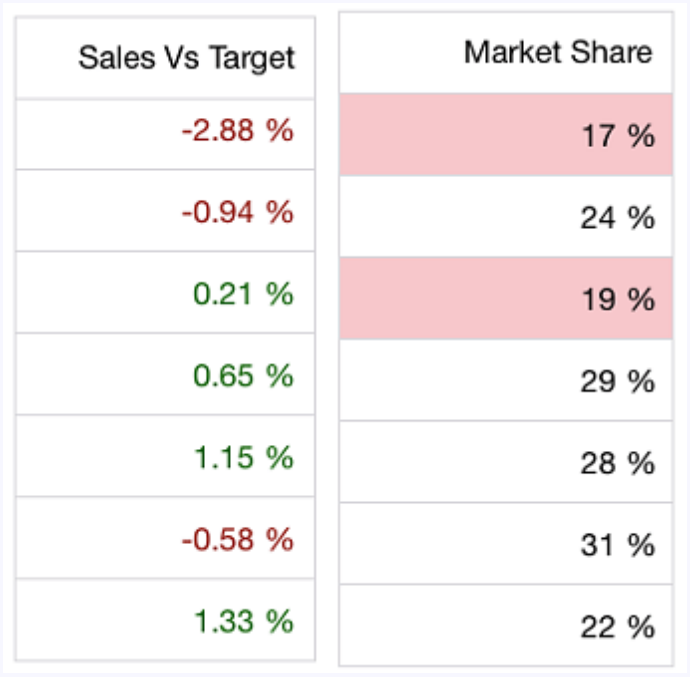
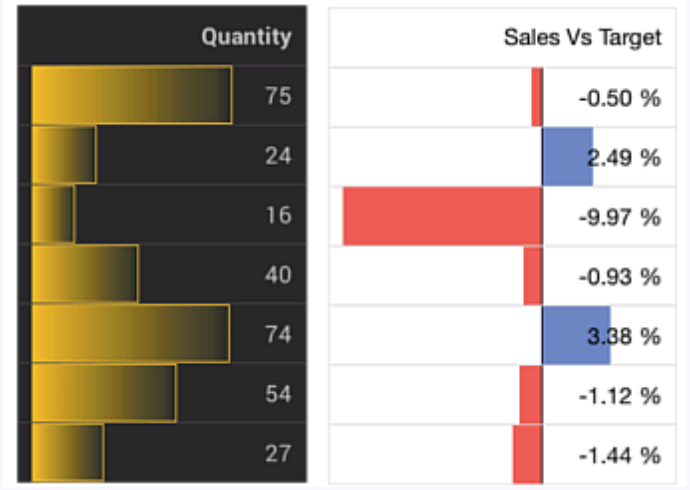
Xamarin Controls > Grid > Conditional Formatting

The GridControl includes a Microsoft Excel-inspired conditional formatting feature, which allows you to change the appearance of individual cells or rows based on specific conditions. This feature helps to highlight critical information, identify trends and exceptions, and compare data. Formatting rules can be created at design time (XAML markup) and runtime (C# code).

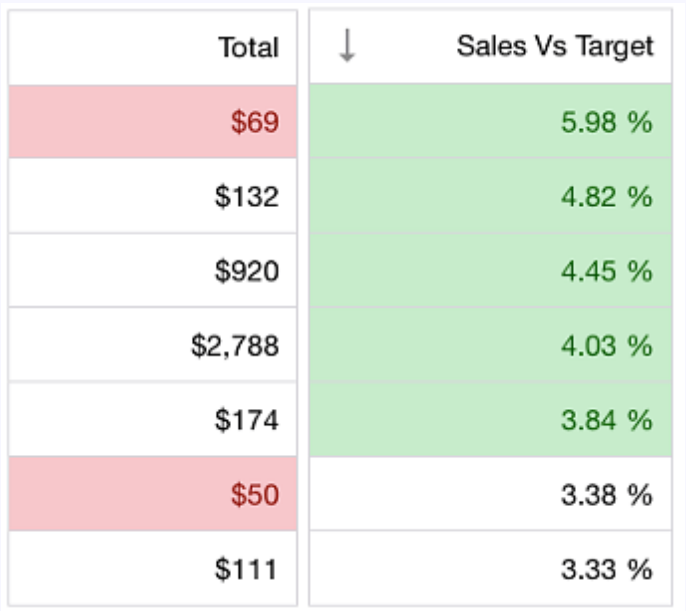
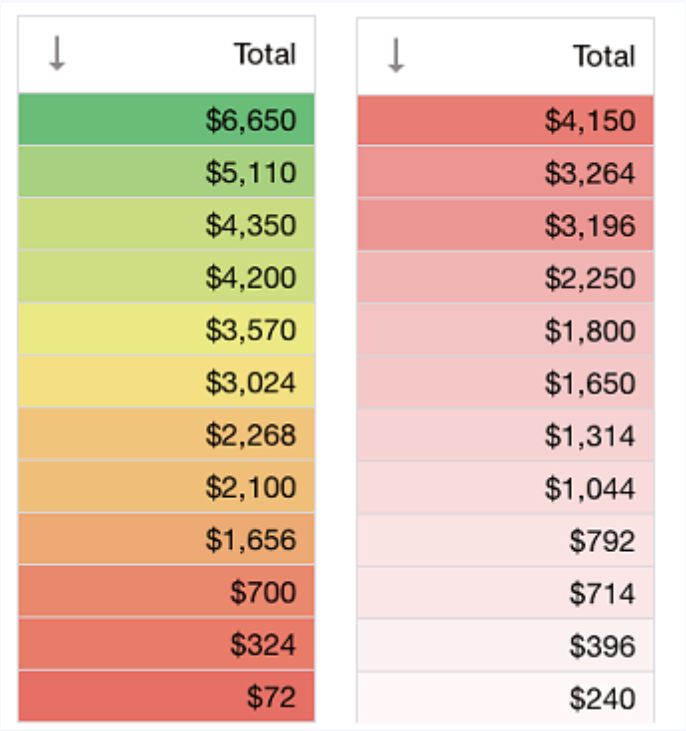
All conditional formatting rules specified in a grid are stored in the FormatConditionCollection collection, accessible from the GridControl.FormatConditions property. The table below lists the available types of conditional formatting rules and classes representing these rules.

| Condition Class | Description | Format Example | Image |
|---|---|---|---|

© 2018 Developer Express Inc.

| | | | |
|---|---|---|---|
| FormatCondition | Applies a format if a column's value meets a specified condition (Equal, Less, Between, etc.), or a specific expression. | How to: Highlight Column Values that Match a Condition<br>How to: Apply a Format Based on an Expression |  |
| DataBarFormatCondition | Formats cells by using data bars. Data bars fill column cells proportionally, based on cell values relative to other cells. | How to: Apply a Data Bar Format to a Column |  |

© 2018 Developer Express Inc.

| | | | |
|---|---|---|---|
| TopBottomRuleFormatCondition | Formats cells whose values are highest or lowest in a column. | How to: Highlight Top Cell Values in a Column |  |
| ColorScaleFormatCondition | Formats column cells by using two or three color scales to display data distribution and variation. | How to: Apply a Color Scale Format to a Column |  |





© 2018 Developer Express Inc.

| | | | |
|---|---|---|---|
| IconSetFormatCondition | Formats cells by using icon sets. | How to: Apply an Icon Set Format to a Column | |



**See Also**
Predefined Format Names

© 2018 Developer Express Inc.

# Predefined Format Names



This document lists all available predefined formats that you can use when specifying conditional formatting rules to be applied in the GridControl. To set one of the predefined formats, assign the corresponding string to the FormatConditionBase.PredefinedFormatName property of an object that represents a conditional formatting rule (for example, DataBarFormatCondition, IconSetFormatCondition, TopBottomRuleFormatCondition, etc.)

### Predefined Standard Formats

"LightRedFillWithDarkRedText"
"YellowFillWithDarkYellowText"
"GreenFillWithDarkGreenText"
"LightRedFill"
"LightGreenFill"
"RedText"
"GreenText"
"BoldText"
"ItalicText"
"StrikethroughText"

### Predefined Color Scale Formats

"GreenYellowRedColorScale"
"RedYellowGreenColorScale"
"GreenWhiteRedColorScale"
"RedWhiteGreenColorScale"
"BlueWhiteRedColorScale"
"RedWhiteBlueColorScale"
"WhiteRedColorScale"
"RedWhiteColorScale"
"GreenWhiteColorScale"
"WhiteGreenColorScale"
"GreenYellowColorScale"
"YellowGreenColorScale"

### Predefined Data Bar Formats

- **Solid**
  "BlueSolidDataBar"
  "GreenSolidDataBar"
  "RedSolidDataBar"
  "OrangeSolidDataBar"
  "LightBlueSolidDataBar"
  "PurpleSolidDataBar"
- **Gradient**
  "BlueGradientDataBar"
  "GreenGradientDataBar"
  "RedGradientDataBar"
  "OrangeGradientDataBar"
  "LightBlueGradientDataBar"
  "PurpleGradientDataBar"

### Predefined Icon Set Formats

"Arrows3ColoredIconSet"
"Arrows3GrayIconSet"
"Arrows4ColoredIconSet"
"Arrows4GrayIconSet"
"Arrows5ColoredIconSet"
"Arrows5GrayIconSet"
"Triangles3IconSet"
"TrafficLights3UnrimmedIconSet"
"TrafficLights3RimmedIconSet"
"Signs3IconSet"
"TrafficLights4IconSet"
"RedToBlackIconSet"
"Symbols3CircledIconSet"

© 2018 Developer Express Inc.

"Symbols3UncircledIconSet"
"Flags3IconSet"
"Stars3IconSet"
"Ratings4IconSet"
"Quarters5IconSet"
"Ratings5IconSet"
"Boxes5IconSet"

**See Also**
Conditional Formatting Examples

© 2018 Developer Express Inc.

# End-User Interaction



Topics in this section:

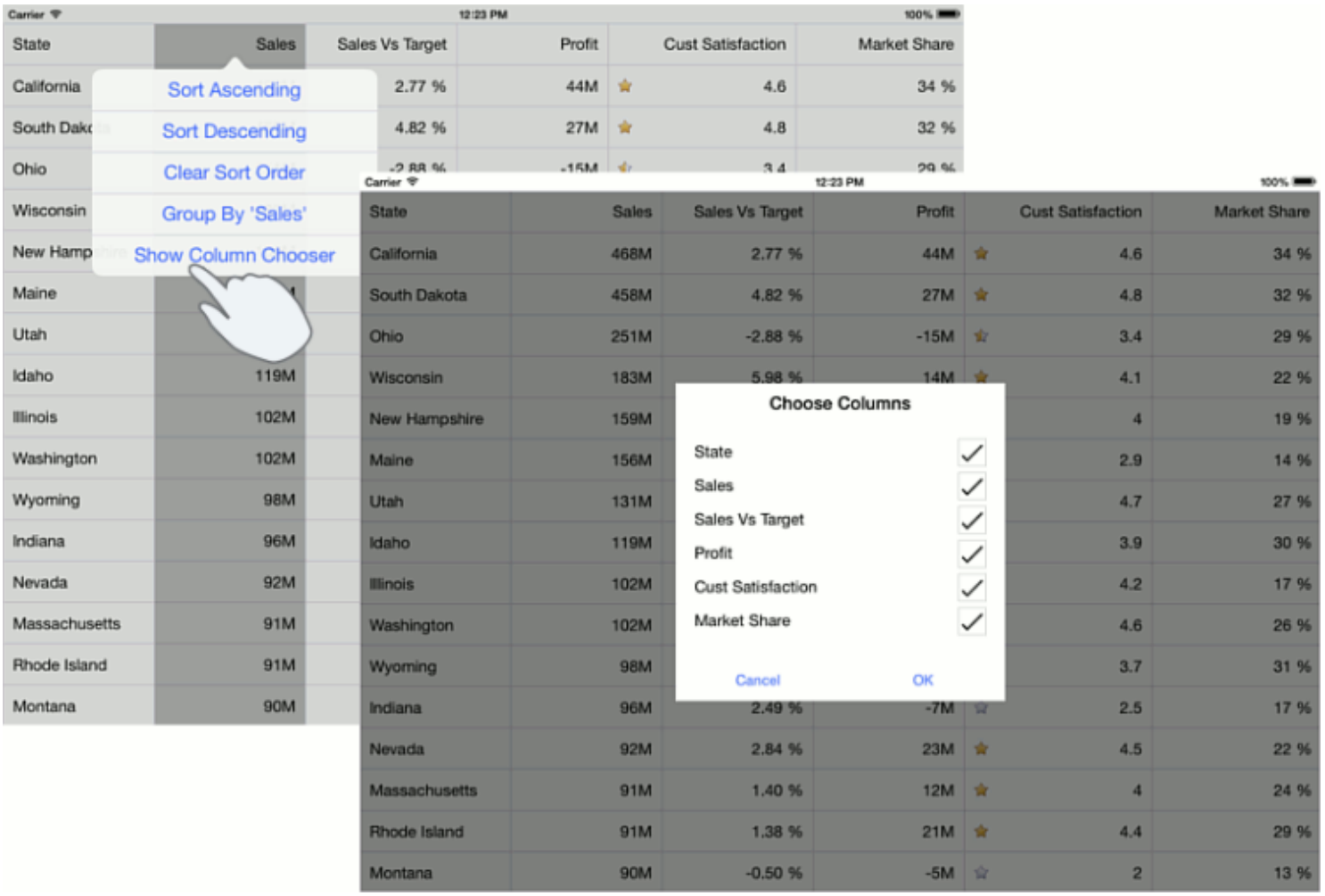

- Column Chooser
- Row Edit Modes
- End-User Restrictions


© 2018 Developer Express Inc.

# Column Chooser

Xamarin Controls > Grid > End-User Interaction > Column Chooser

The **Column Chooser** allows end-users to show and hide a grid's columns at runtime. It lists all columns contained in the GridControl.Columns collection. By default, the Column Chooser is shown when an end-user touches and holds a column header and then taps the **Show Column Chooser** item in the invoked menu.




When an end-user selects or clears a check box next to a column header in the Column Chooser, the GridColumn.IsVisible property of the corresponding column is set to **true** or **false**. You can use this property to hide or show a column in code.

To invoke or close the Column Chooser programmatically, use the GridControl.ShowColumnChooser or GridControl.HideColumnChooser method.

To disable the Column Chooser for end-users, set the GridControl.IsColumnChooserEnabled property to **false**. In this case, the **Show Column Chooser** item disappears from the column header's popup menu.

© 2018 Developer Express Inc.

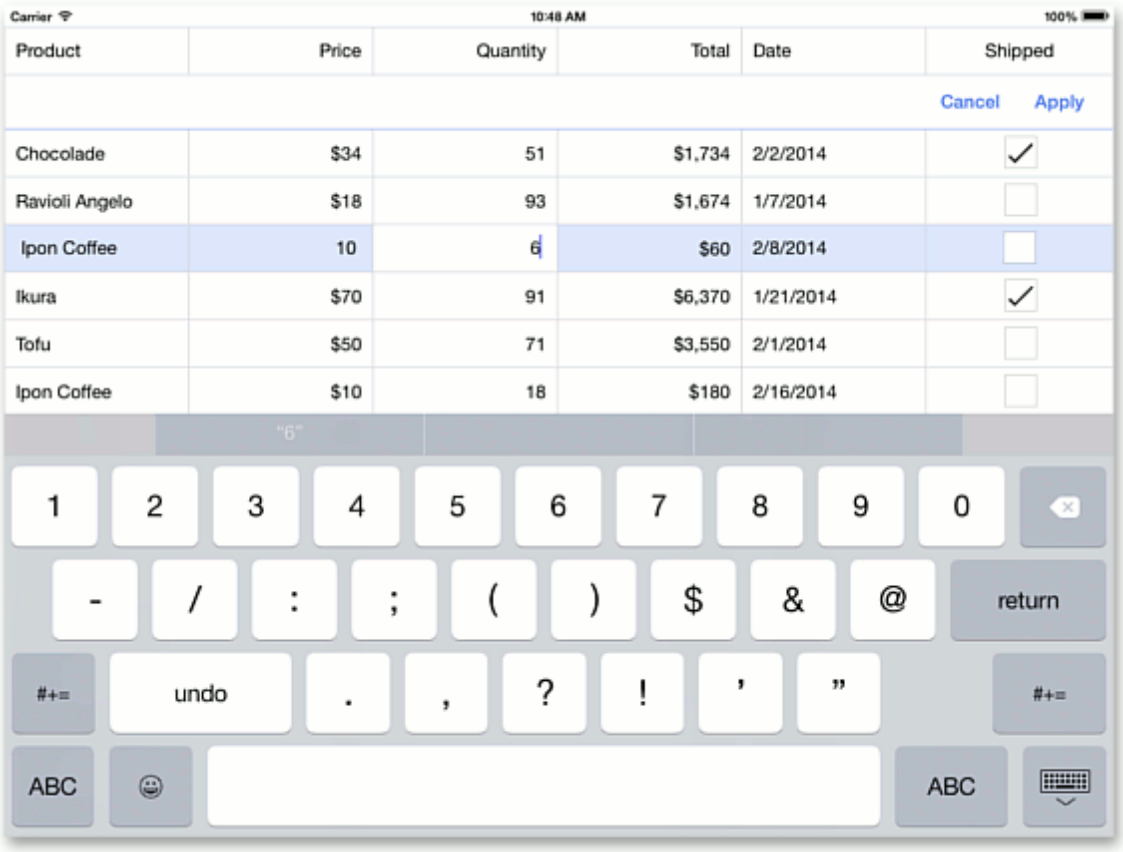
# Row Edit Modes

Xamarin Controls > Grid > End-User Interaction > Row Edit Modes

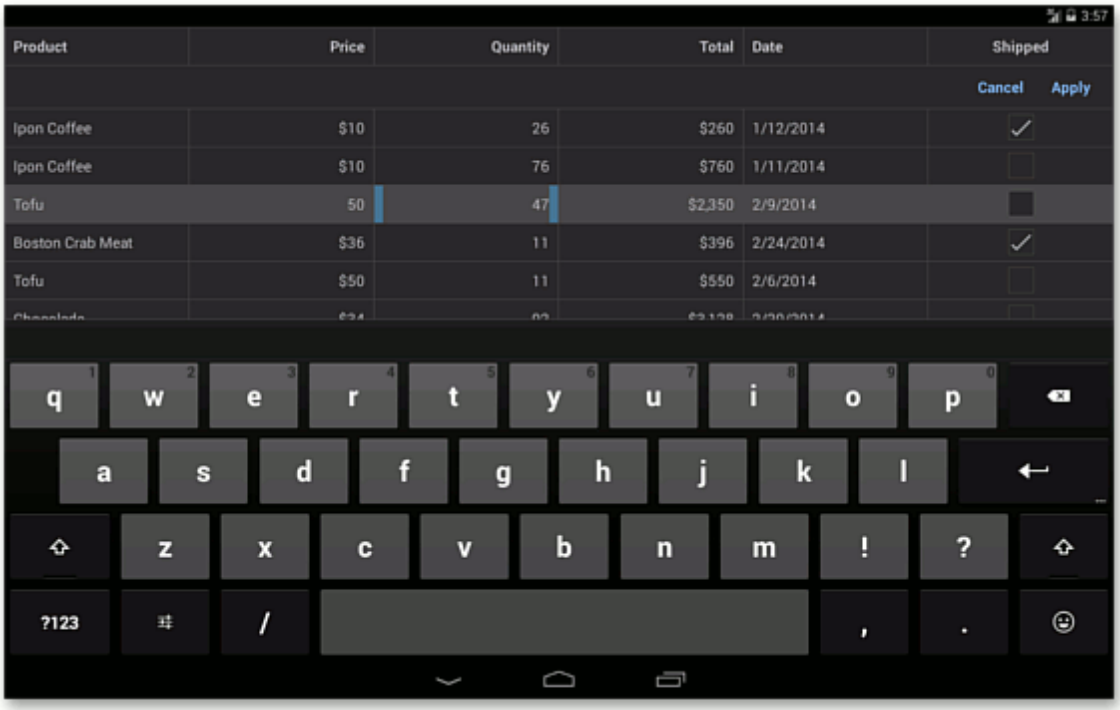
GridControl provides four built-in edit modes that allow end-users to edit grid data. Use the GridControl.RowEditMode property to specify the grid's edit mode. To open the edit form that corresponds to the specified mode, use the GridControl.OpenEditor method. To close the form, use the GridControl.CloseEditor method.

When an end-user starts to edit a row by double-tapping a data cell or adding a new row, the grid displays an in-place row editor or edit form depending on the set row edit mode.

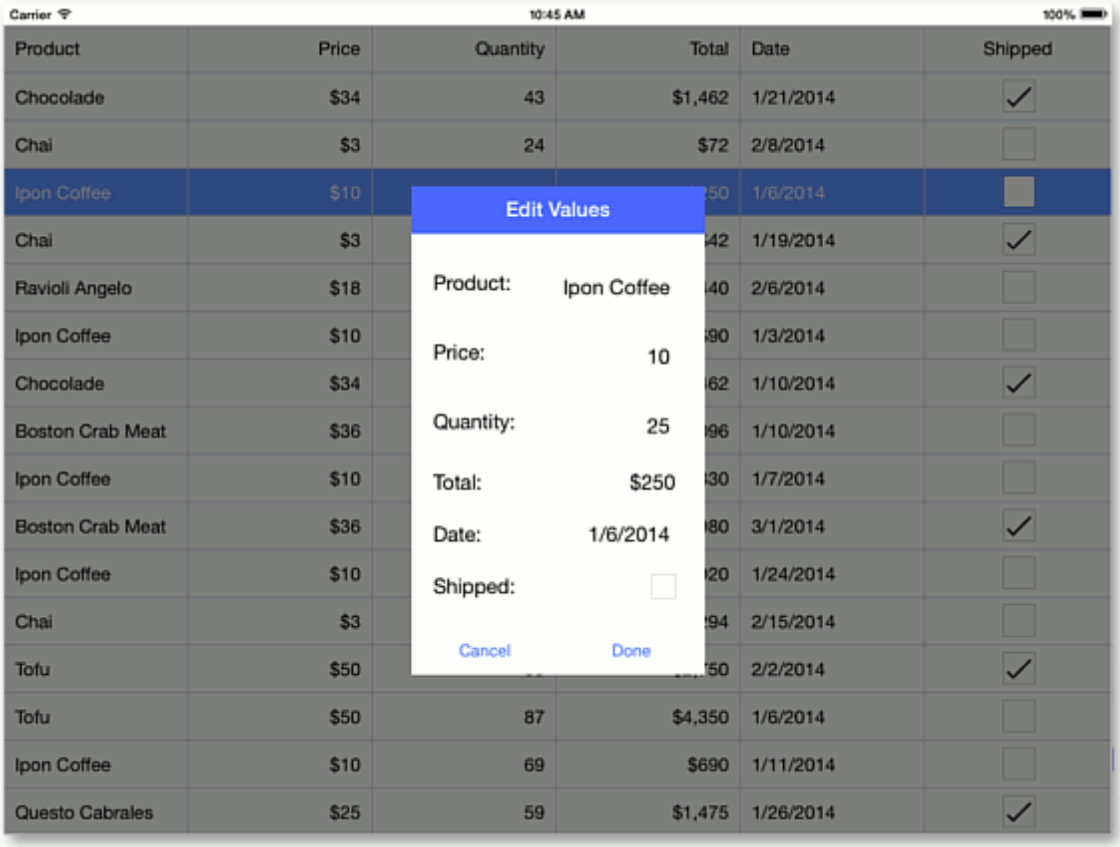
- **Inplace** - an in-place row editor used by default.



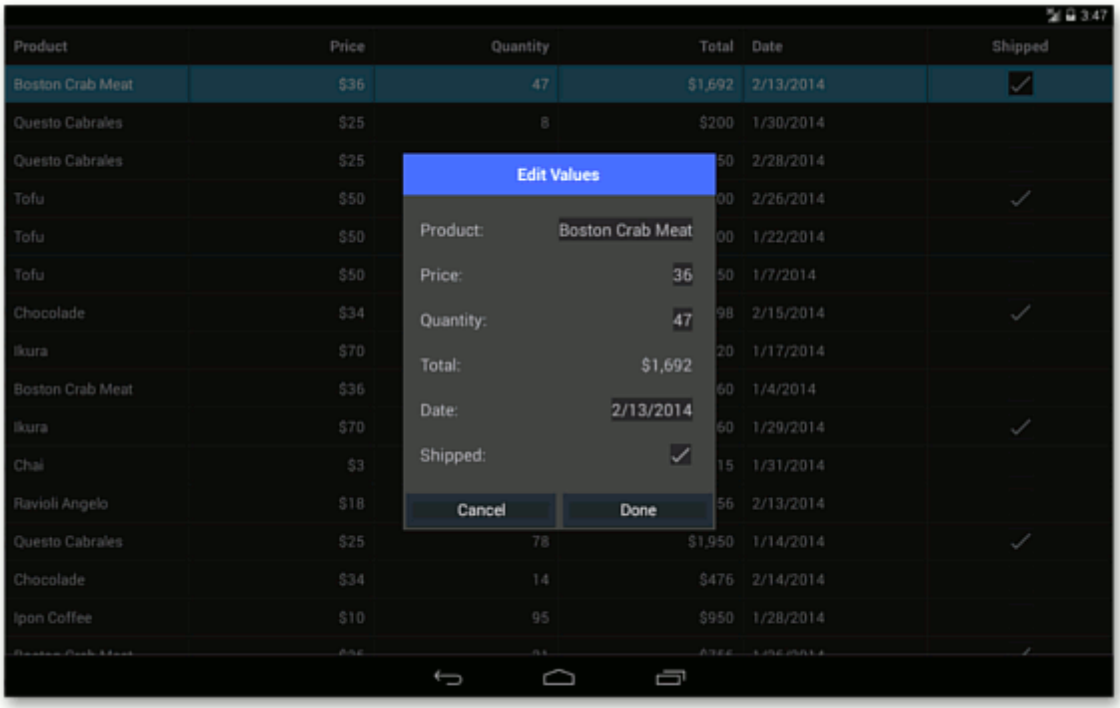

© 2018 Developer Express Inc.



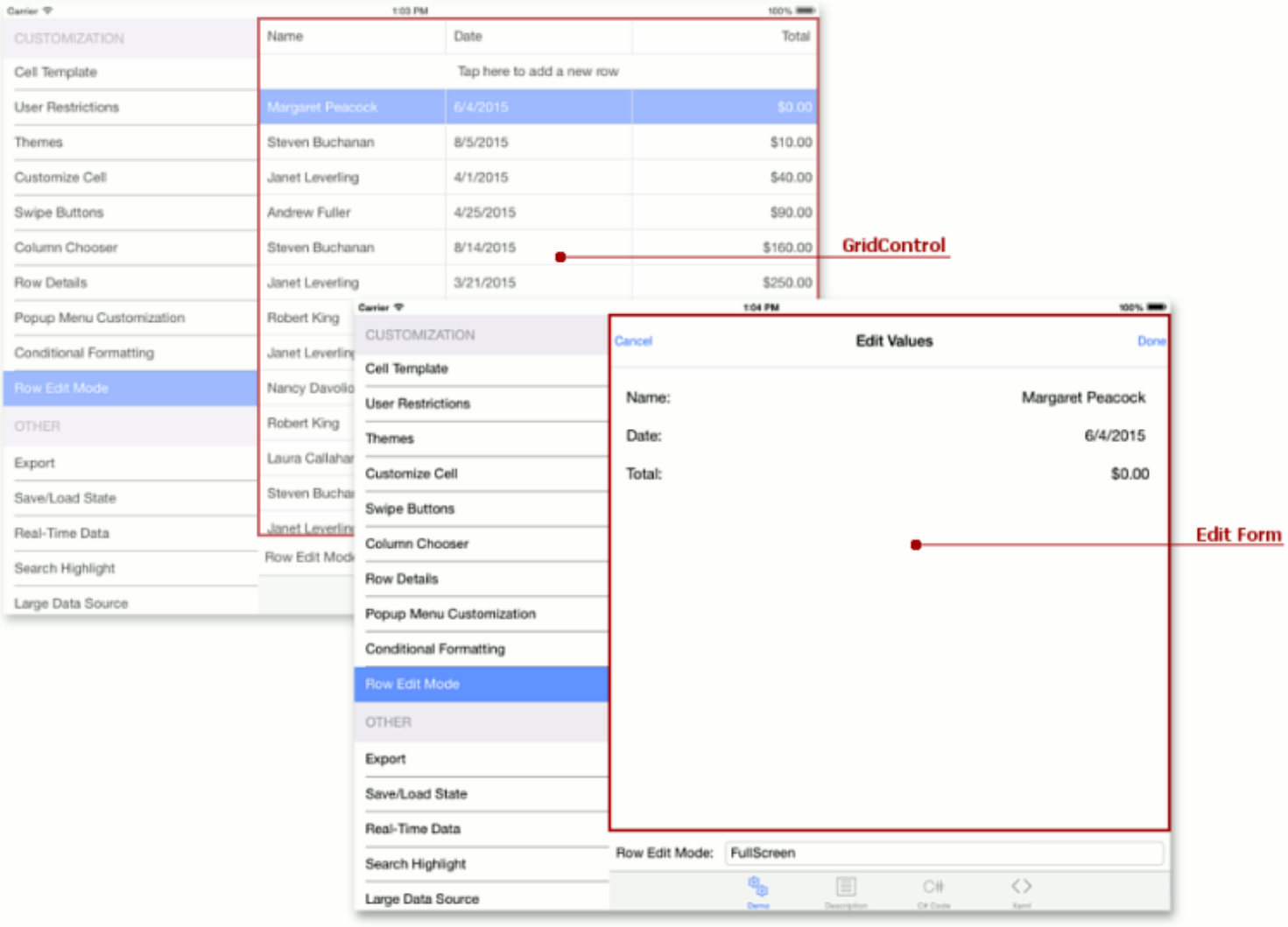

- **Popup Edit Form**



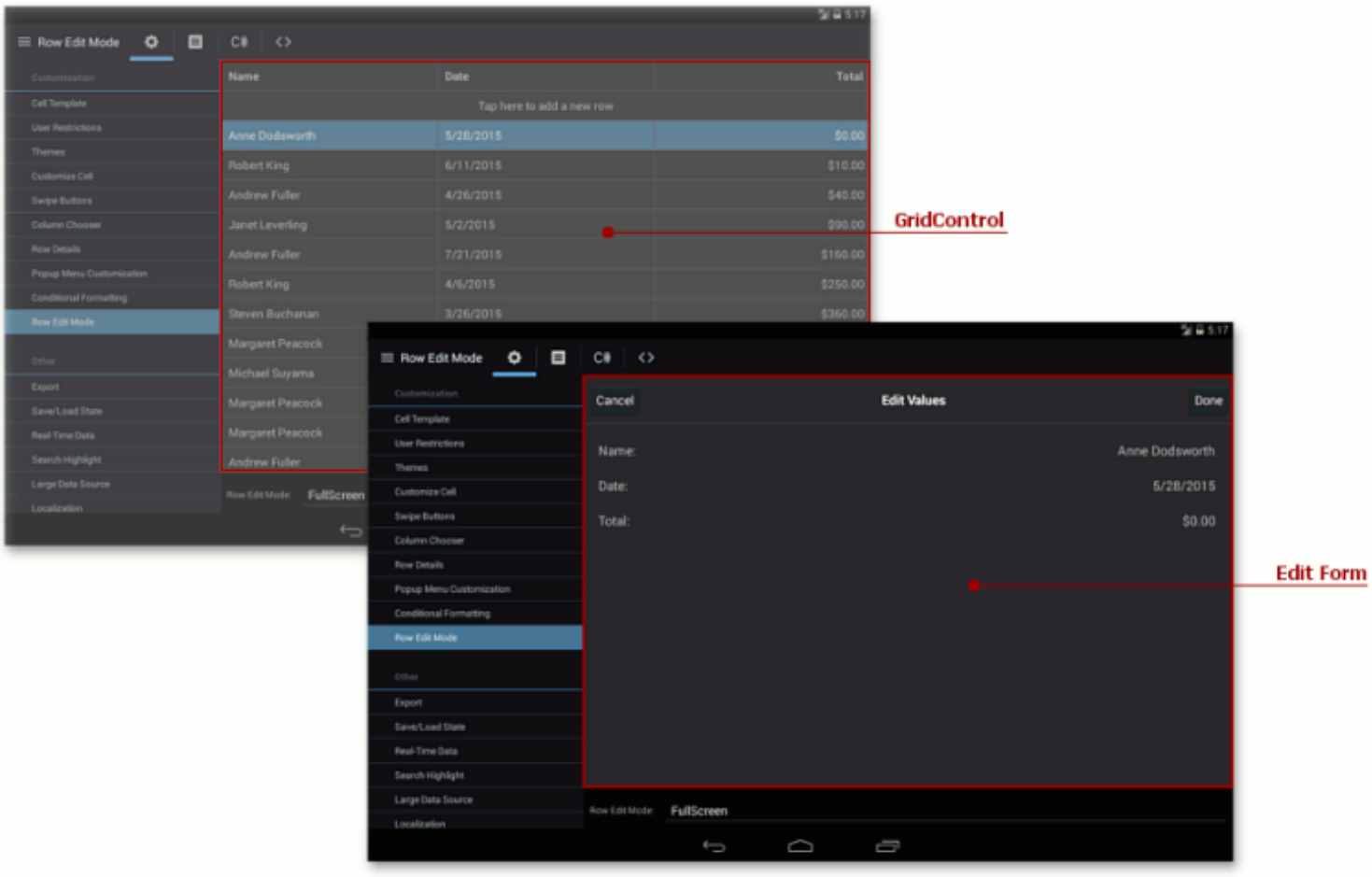

© 2018 Developer Express Inc.




- **Grid Area** - an edit form displayed over the whole grid area.

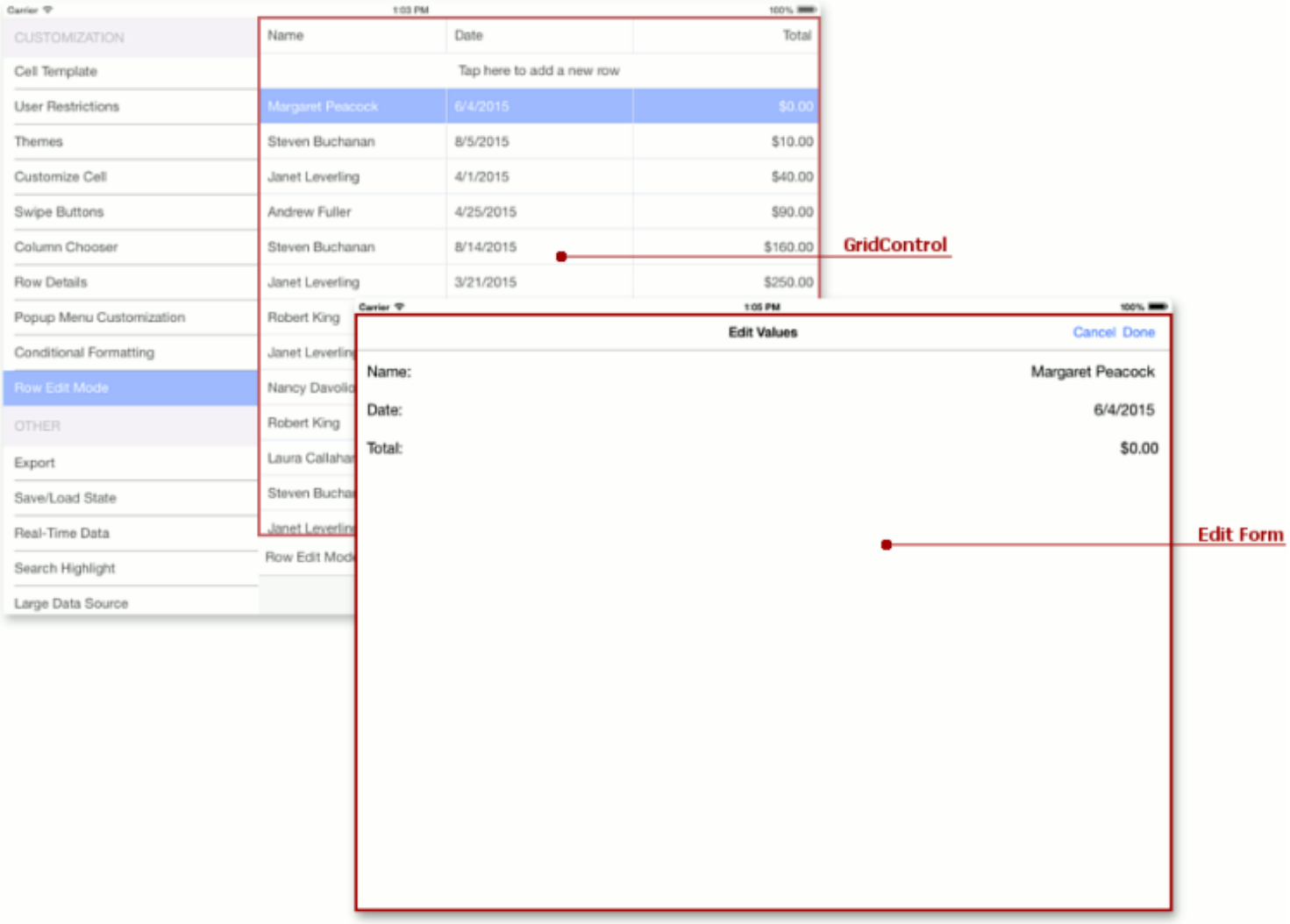


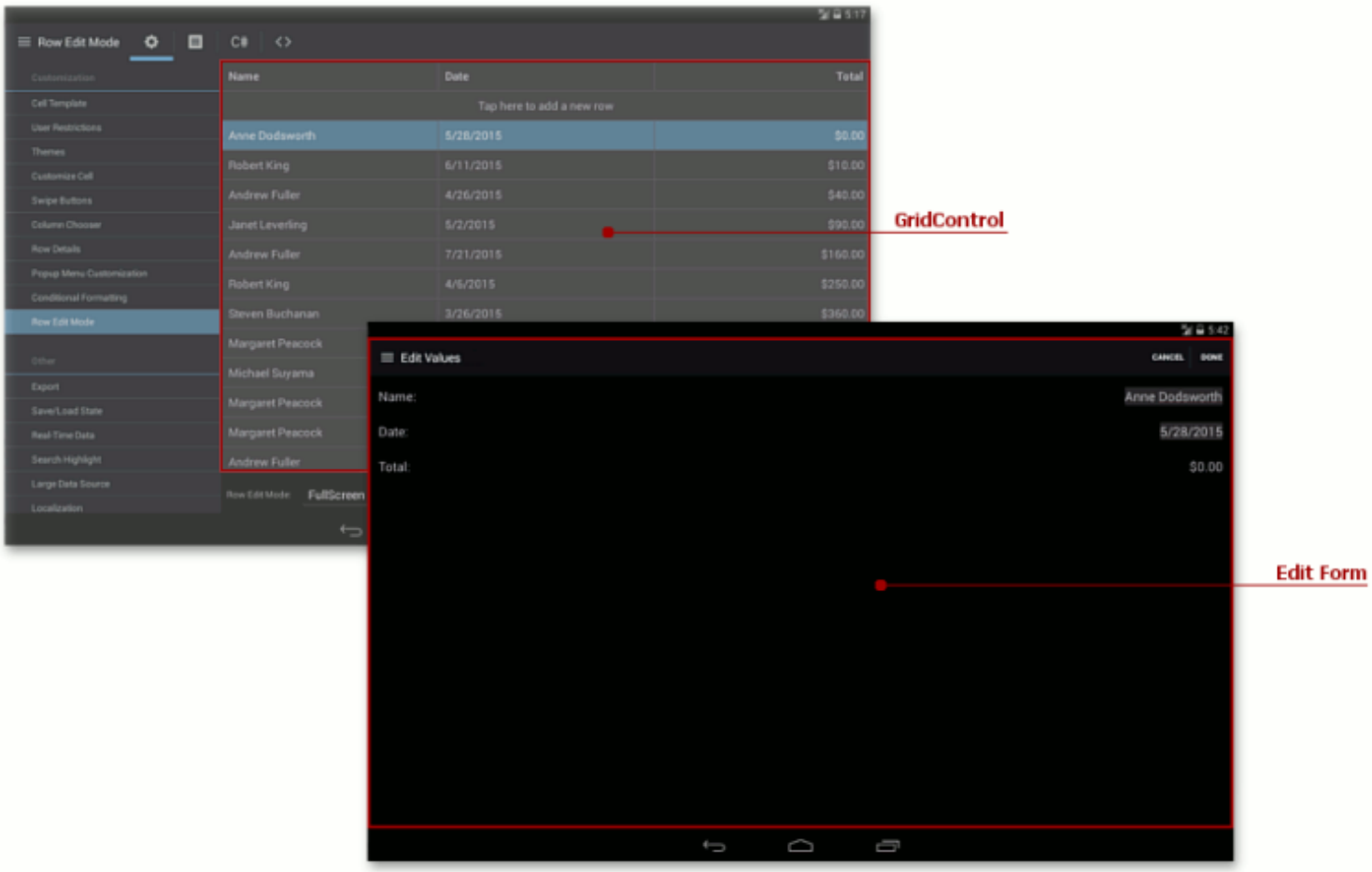

© 2018 Developer Express Inc.




- **Full Screen Area** - an edit form placed on a separate NavigationPage and displayed in the full screen mode.

© 2018 Developer Express Inc.


CUSTOMIZATION
Cell Template
User Restrictions
Themes
Customize Cell
Swipe Buttons
Column Chooser
Row Details
Popup Menu Customization
Conditional Formatting
Row Edit Mode
OTHER
Export
Save/Load State
Real-Time Data
Search Highlight
Large Data Source
Name
Date
Total
Tap here to add a new row
Margaret Peacock 6/4/2015 $0.00
Steven Buchanan 8/5/2015 $10.00
Janet Leverling 4/1/2015 $40.00
Andrew Fuller 4/25/2015 $90.00
Steven Buchanan 8/14/2015 $160.00
Janet Leverling 3/21/2015 $250.00
GridControl
Edit Values
Cancel Done
Name: Margaret Peacock
Date: 6/4/2015
Total: $0.00
Edit Form




If you need to implement a custom edit form, assign your template to the GridControl.EditFormContent property.

To prevent end-users from editing data rows, set the GridControl.AllowEditRows property to **false**.

**See Also**
How to: Implement a Custom Row Edit Form

© 2018 Developer Express Inc.

# End-User Restrictions

Xamarin Controls > Grid > End-User Interaction > End-User Restrictions

In some real-life applications, it might be necessary to prevent end-users from editing data in a grid (modify cell values, insert or delete data rows), or restrict data shaping capabilities (data sorting, grouping and filtering), etc. For example, it may be necessary to show grid data to end-users as read-only.

To implement end-user restrictions in your application, you can use the following properties.

| Restriction | Description |
|---|---|
| GridColumn.IsReadOnly, GridControl.IsReadOnly | Specifies whether end-users are allowed to change cell data values, as well as insert and delete rows in the grid. You can prevent end-users from editing data in specific columns or in the entire grid.<br>In the read-only mode, a row in-place editor (or edit form) is not invoked on double-tapping a data cell, and the **Edit Cell** and **Delete Row** items disappear from the popup menu. |
| GridControl.NewItemRowVisibility | Specifies whether the grid shows the **New Item Row** allowing end-users to add new data rows. |
| GridControl.AllowEditRows | Specifies whether end-users are allowed to modify cell values.<br>If this functionality is disabled, a row in-place editor (or edit form) is not invoked on double-tapping a data cell, and the **Edit Cell** item disappears from the popup menu. |
| GridControl.AllowDeleteRows | Specifies whether end-users are allowed to remove data rows from the grid.<br>If this functionality is disabled, the **Delete Row** item disappears from the popup menu. |
| GridControl.AllowResizeColumns | Specifies whether end-users are allowed to change the column width with the pinch and spread gestures. |
| GridColumn.AllowSort, GridControl.AllowSort | Specifies whether end-users are allowed to sort data in the grid. You can restrict sorting by individual columns, or disable this functionality for the entire grid.<br>Note that when sorting by a column is disabled, grouping by this column becomes unavailable as well.<br>If this functionality is disabled, the **Sort Ascending**, **Sort Descending**, **Clear Sort Order** and **Group By** items disappear from the popup menu, and data sorting in not applied when an end-user taps a column header. |
| GridColumn.AllowGroup, GridControl.AllowGroup | Specifies whether end-users are allowed to group data in the grid. You can restrict grouping by individual columns, or disable this functionality for the entire grid.<br>If this functionality is disabled, the **Group By** item disappears from the popup menu. |
| GridControl.AllowGroupCollapse | Specifies whether end-users are allowed to collapse and expand data groups in the grid by tapping group rows. |
| GridControl.AutoFilterPanelVisibility, GridColumn.AllowAutoFilter | Specifies whether end-users are allowed to filter data by columns using the auto-filter panel shown below column headers. |

© 2018 Developer Express Inc.

| | |
|---|---|
| | You can hide an auto-filter panel in the grid entirely, or disable individual cells of the auto-filter panel to prevent end-users from filtering data by the corresponding columns. |
| GridControl.ColumnHeadersVisibility | Specifies whether column headers are shown in the grid. |
| GridControl.FilterPanelVisibility | Specifies whether end-users are allowed to manage (disable or clear) data filters currently applied in the grid using the filter panel.<br>By default, the filter panel is automatically shown at the bottom of the grid after a filter has been applied. Removing the filter hides this panel. You can make this panel always visible or always hidden by using the **FilterPanelVisibility** property. |
| GridControl.TotalSummaryVisibility | Specifies whether end-users are allowed to calculate total summaries for columns using the total summary panel. |

You can also delete existing items from built-in popup menus using the GridControl.PopupMenuCustomization event. See the How to: Customize Popup Menu example.

© 2018 Developer Express Inc.

# Examples



This section contains the following examples:

- How to: Create Grid Columns for Data of Different Types
- How to: Implement a PickerColumn
- How to: Bind a GridControl to a DataTable
- How to: Change Cell Appearance
- How to: Implement Swipe Buttons for Data Rows
- How to: Implement Pull-to-Refresh
- How to: Implement Load-More
- How to: Customize Popup Menu
- How to: Implement a Custom Row Edit Form
- How to: Format Column Values Using Standard Format Strings
- How to: Implement the Search Functionality
- How to: Use a TemplateColumn for Showing and Editing Time Values
- How to: Check a SwitchColumn by a Single Tap
- Conditional Formatting
- Themes

© 2018 Developer Express Inc.

# Conditional Formatting

Xamarin Controls > Grid > Examples > Conditional Formatting

This section contains the following examples.

- How to: Apply a Color Scale Format to a Column
- How to: Apply a Data Bar Format to a Column
- How to: Apply a Format Based on an Expression
- How to: Apply an Icon Set Format to a Column
- How to: Highlight Column Values that Match a Condition
- How to: Highlight Top Cell Values in a Column
- How to: Set Conditional Formatting Rules in XAML

© 2018 Developer Express Inc.

# How to: Highlight Top Cell Values in a Column

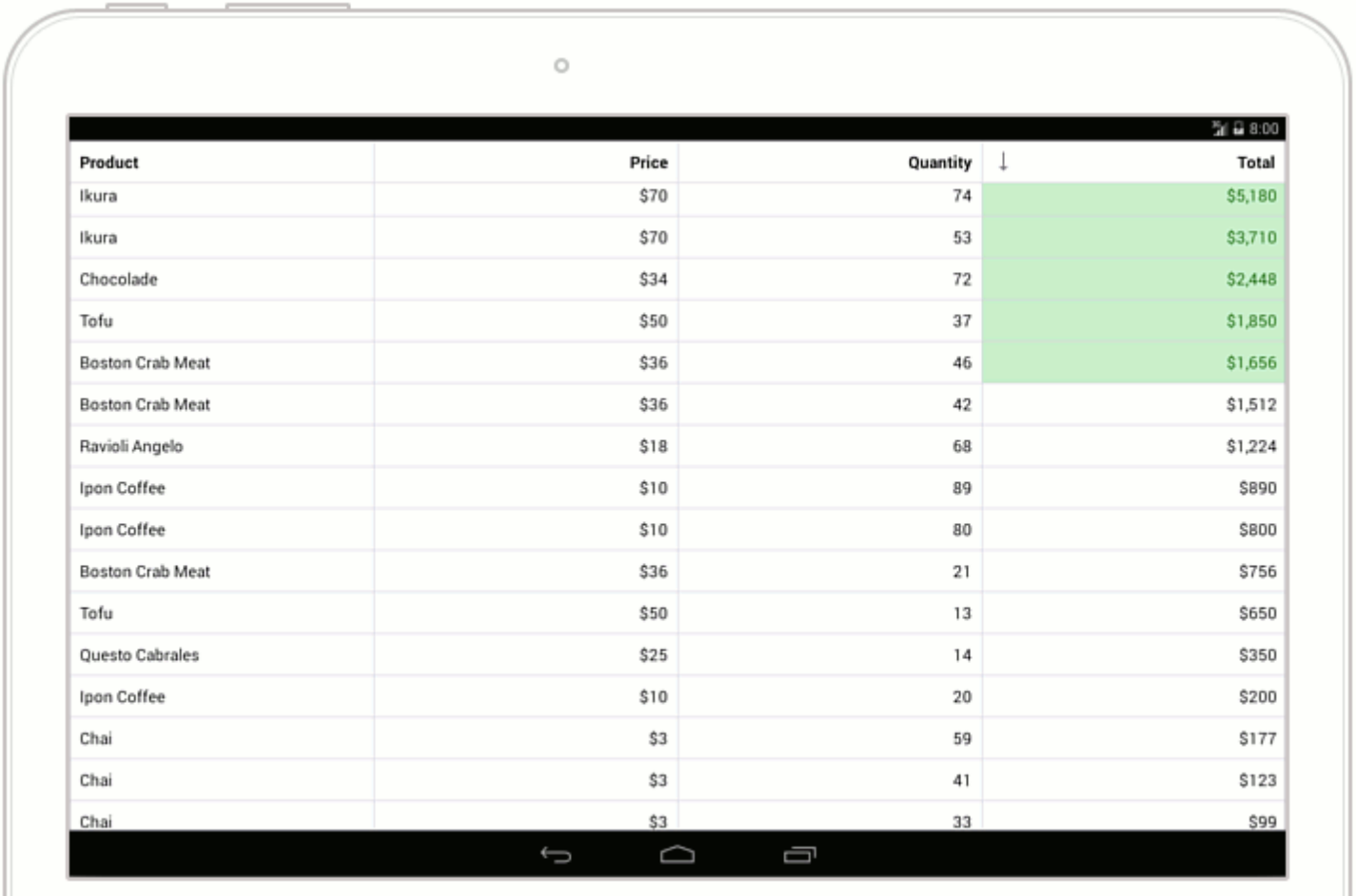
Xamarin Controls > Grid > Examples > Conditional Formatting > How to: Highlight Top Cell Values in a Column

This example illustrates how to apply a top/bottom format to the **Total** column in a GridControl. The top/bottom format allows you to find the highest or lowest cell values. In this tutorial, the five top values in the **Total** column are highlighted.

© 2018 Developer Express Inc.

Carrier 2:56 PM 100%

| Product | Price | Quantity | Total |
|---|---|---|---|
| Ikura | $70 | 88 | $6,160 |
| Tofu | $50 | 96 | $4,800 |
| Tofu | $50 | 92 | $4,600 |
| Ravioli Angelo | $18 | 95 | $1,710 |
| Boston Crab Meat | $36 | 28 | $1,008 |
| Chocolade | $34 | 28 | $952 |
| Ravioli Angelo | $18 | 51 | $918 |
| Chocolade | $34 | 24 | $816 |
| Chocolade | $34 | 21 | $714 |
| Questo Cabrales | $25 | 28 | $700 |
| Boston Crab Meat | $36 | 18 | $648 |
| Ipon Coffee | $10 | 51 | $510 |
| Boston Crab Meat | $36 | 11 | $396 |
| Ravioli Angelo | $18 | 19 | $342 |
| Chocolade | $34 | 10 | $340 |
| Chocolade | $34 | 10 | $340 |



8:00

| Product | Price | Quantity | Total |
|---|---|---|---|
| Ikura | $70 | 74 | $5,180 |
| Ikura | $70 | 53 | $3,710 |
| Chocolade | $34 | 72 | $2,448 |
| Tofu | $50 | 37 | $1,850 |
| Boston Crab Meat | $36 | 46 | $1,656 |
| Boston Crab Meat | $36 | 42 | $1,512 |
| Ravioli Angelo | $18 | 68 | $1,224 |
| Ipon Coffee | $10 | 89 | $890 |
| Ipon Coffee | $10 | 80 | $800 |
| Boston Crab Meat | $36 | 21 | $756 |
| Tofu | $50 | 13 | $650 |
| Questo Cabrales | $25 | 14 | $350 |
| Ipon Coffee | $10 | 20 | $200 |
| Chai | $3 | 59 | $177 |
| Chai | $3 | 41 | $123 |
| Chai | $3 | 33 | $99 |

1. Create a new TopBottomRuleFormatCondition class instance.
2. Set this object's FormatConditionBase.FieldName property to **Total**. This column provides values to test against the formatting rule and the specified format will be applied to this column.
3. Set the TopBottomRuleFormatCondition.Rule property to **TopBottomRule.TopItems** to highlight the highest column values.
4. Use the TopBottomRuleFormatCondition.Threshold property to specify the count of column cells with highest values to be highlighted.
5. Assign one of the predefined formats to the FormatConditionBase.PredefinedFormatName property. All available format names are listed in the Predefined Format Names document.
6. To apply a conditional formatting rule represented by the created **TopBottomRuleFormatCondition** object, add this object to the GridControl.FormatConditions collection.

**C#**

```
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.Core.ConditionalFormatting;
// ...
TopBottomRuleFormatCondition condition = new TopBottomRuleFormatCondition();
condition.FieldName = "Total";
condition.Rule = TopBottomRule.TopItems;
condition.Threshold = 5;
condition.PredefinedFormatName = "GreenFillWithDarkGreenText";
grid.FormatConditions.Add (condition);
```

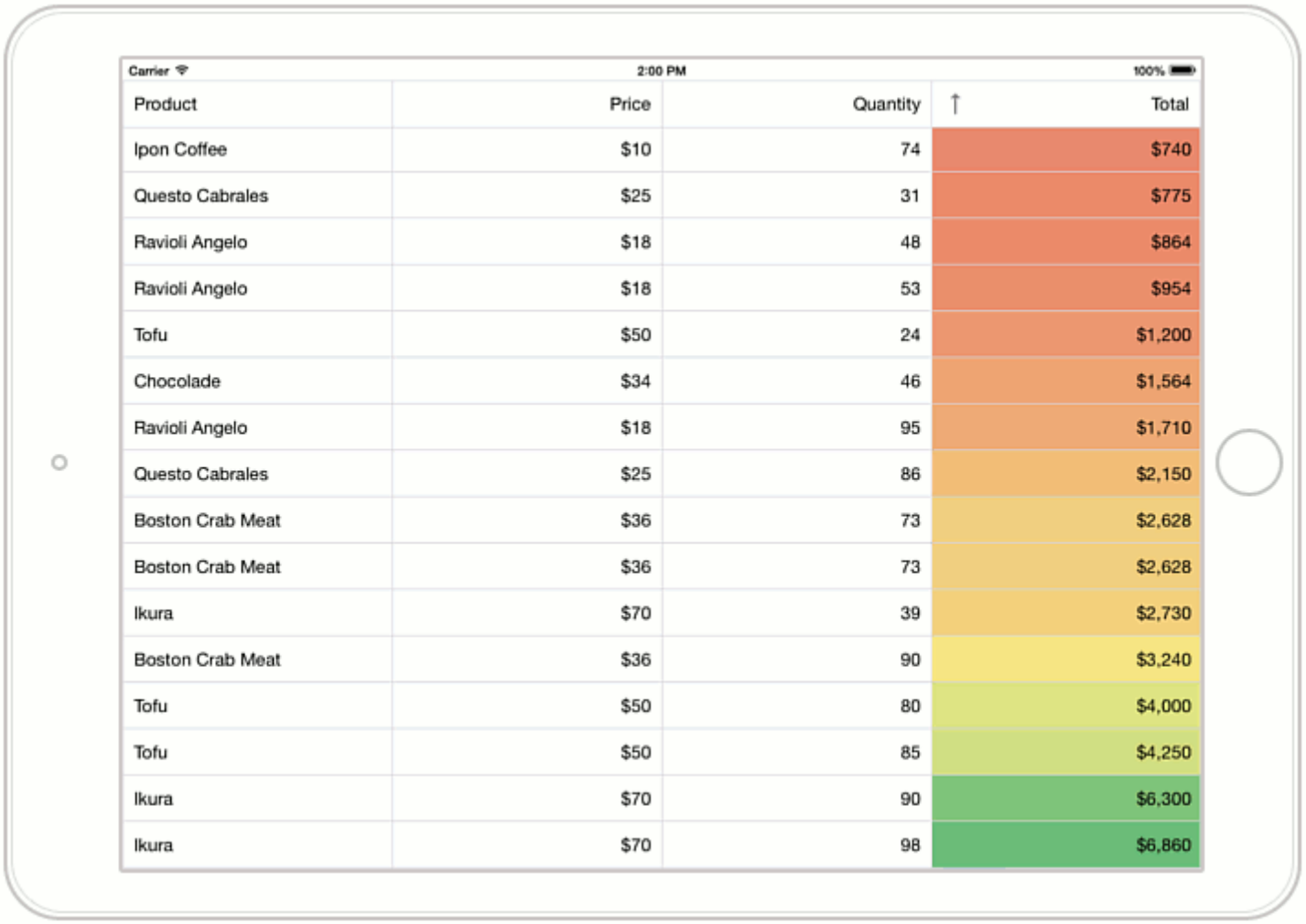
© 2018 Developer Express Inc.

# How to: Apply a Color Scale Format to a Column

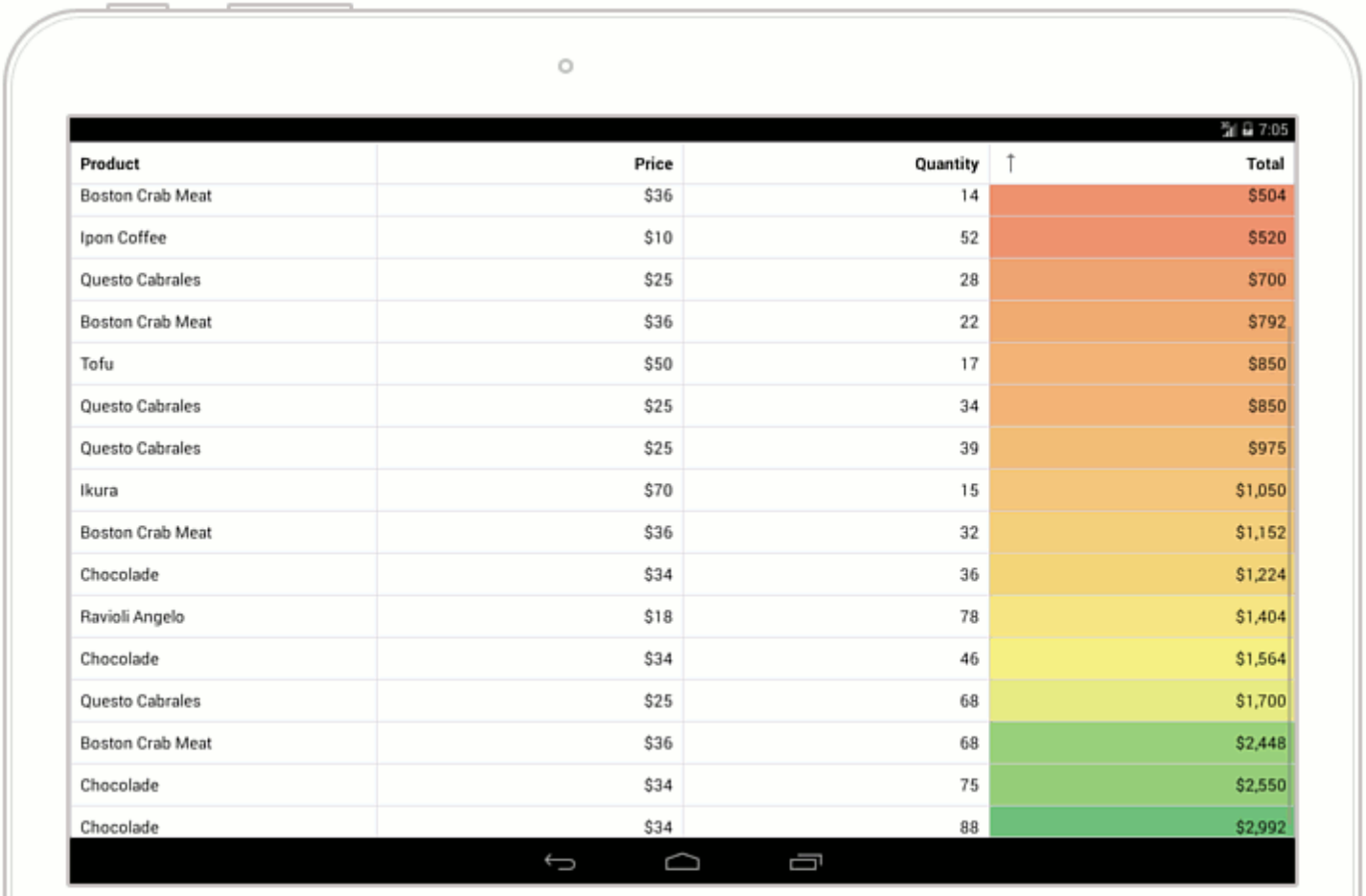


This example demonstrates how to apply a color scale format to the **Total** column in a GridControl. A color scale format reflects data distribution of column cell values using a gradient of two or three colors. In this example, the predefined **Red-Yellow-Green** color scale is used.

© 2018 Developer Express Inc.




| Product | Price | Quantity | Total |
|---|---|---|---|
| Ipon Coffee | $10 | 74 | $740 |
| Questo Cabrales | $25 | 31 | $775 |
| Ravioli Angelo | $18 | 48 | $864 |
| Ravioli Angelo | $18 | 53 | $954 |
| Tofu | $50 | 24 | $1,200 |
| Chocolade | $34 | 46 | $1,564 |
| Ravioli Angelo | $18 | 95 | $1,710 |
| Questo Cabrales | $25 | 86 | $2,150 |
| Boston Crab Meat | $36 | 73 | $2,628 |
| Boston Crab Meat | $36 | 73 | $2,628 |
| Ikura | $70 | 39 | $2,730 |
| Boston Crab Meat | $36 | 90 | $3,240 |
| Tofu | $50 | 80 | $4,000 |
| Tofu | $50 | 85 | $4,250 |
| Ikura | $70 | 90 | $6,300 |
| Ikura | $70 | 98 | $6,860 |




| Product | Price | Quantity | Total |
|---|---|---|---|
| Boston Crab Meat | $36 | 14 | $504 |
| Ipon Coffee | $10 | 52 | $520 |
| Questo Cabrales | $25 | 28 | $700 |
| Boston Crab Meat | $36 | 22 | $792 |
| Tofu | $50 | 17 | $850 |
| Questo Cabrales | $25 | 34 | $850 |
| Questo Cabrales | $25 | 39 | $975 |
| Ikura | $70 | 15 | $1,050 |
| Boston Crab Meat | $36 | 32 | $1,152 |
| Chocolade | $34 | 36 | $1,224 |
| Ravioli Angelo | $18 | 78 | $1,404 |
| Chocolade | $34 | 46 | $1,564 |
| Questo Cabrales | $25 | 68 | $1,700 |
| Boston Crab Meat | $36 | 68 | $2,448 |
| Chocolade | $34 | 75 | $2,550 |
| Chocolade | $34 | 88 | $2,992 |

1. Create a new ColorScaleFormatCondition class instance.
2. Set this object's FormatConditionBase.FieldName property to **Total**. This column provides values to test against the formatting rule and the specified format will be applied to this column.
3. Assign one of the predefined color scales to the FormatConditionBase.PredefinedFormatName property. All available format names are listed in the Predefined Format Names document.
4. To apply a conditional formatting rule represented by the created **ColorScaleFormatCondition** object, add this object to the GridControl.FormatConditions collection.

**C#**

```
using DevExpress.Mobile.DataGrid;
// ...
ColorScaleFormatCondition condition = new ColorScaleFormatCondition();
condition.FieldName = "Total";
condition.PredefinedFormatName = "RedYellowGreenColorScale";
grid.FormatConditions.Add (condition);
```

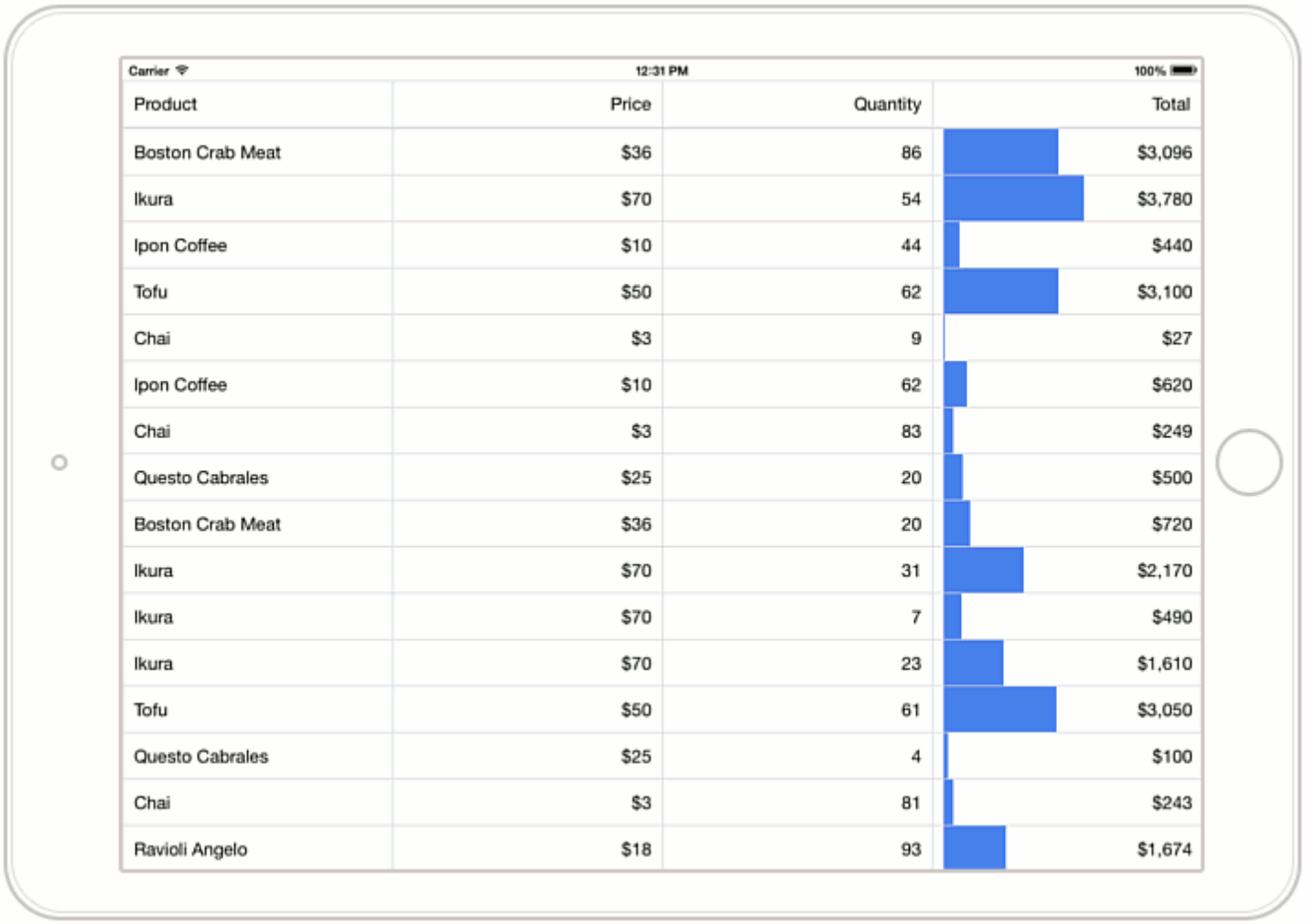
© 2018 Developer Express Inc.

# How to: Apply a Data Bar Format to a Column



This example demonstrates how to apply a data bar format to the **Total** column in a GridControl. Data bars fill column cells proportionally based on cell values relative to other cells. A longer bar corresponds to a higher value, and a shorter bar corresponds to a lower value.

© 2018 Developer Express Inc.




| Product | Price | Quantity | Total |
|---|---|---|---|
| Boston Crab Meat | $36 | 86 | $3,096 |
| Ikura | $70 | 54 | $3,780 |
| Ipon Coffee | $10 | 44 | $440 |
| Tofu | $50 | 62 | $3,100 |
| Chai | $3 | 9 | $27 |
| Ipon Coffee | $10 | 62 | $620 |
| Chai | $3 | 83 | $249 |
| Questo Cabrales | $25 | 20 | $500 |
| Boston Crab Meat | $36 | 20 | $720 |
| Ikura | $70 | 31 | $2,170 |
| Ikura | $70 | 7 | $490 |
| Ikura | $70 | 23 | $1,610 |
| Tofu | $50 | 61 | $3,050 |
| Questo Cabrales | $25 | 4 | $100 |
| Chai | $3 | 81 | $243 |
| Ravioli Angelo | $18 | 93 | $1,674 |

| Product | Price | Quantity | Total |
|---|---|---|---|
| Ravioli Angelo | $18 | 67 | $1,206 |
| Ikura | $70 | 62 | $4,340 |
| Ipon Coffee | $10 | 85 | $850 |
| Chocolade | $34 | 27 | $918 |
| Ipon Coffee | $10 | 3 | $30 |
| Boston Crab Meat | $36 | 90 | $3,240 |
| Ravioli Angelo | $18 | 1 | $18 |
| Tofu | $50 | 33 | $1,650 |
| Ipon Coffee | $10 | 70 | $700 |
| Chocolade | $34 | 74 | $2,516 |
| Tofu | $50 | 97 | $4,850 |
| Chocolade | $34 | 1 | $34 |
| Ipon Coffee | $10 | 71 | $710 |
| Boston Crab Meat | $36 | 2 | $72 |
| Boston Crab Meat | $36 | 73 | $2,628 |
| Ikura | $70 | 72 | $5,040 |

1. Create a new DataBarFormatCondition class instance.
2. Set this object's FormatConditionBase.FieldName property to **Total**. This column provides values to test against the formatting rule and the specified format will be applied to this column.
3. Assign one of the predefined bar styles to the FormatConditionBase.PredefinedFormatName property. All available format names are listed in the Predefined Format Names document.
4. To apply a conditional formatting rule represented by the created **DataBarFormatCondition** object, add this object to the GridControl.FormatConditions collection.

**C#**

```
using DevExpress.Mobile.DataGrid;
// ...
DataBarFormatCondition condition = new DataBarFormatCondition();
condition.FieldName = "Total";
condition.PredefinedFormatName = "LightBlueGradientDataBar";
grid.FormatConditions.Add (condition);
```

© 2018 Developer Express Inc.

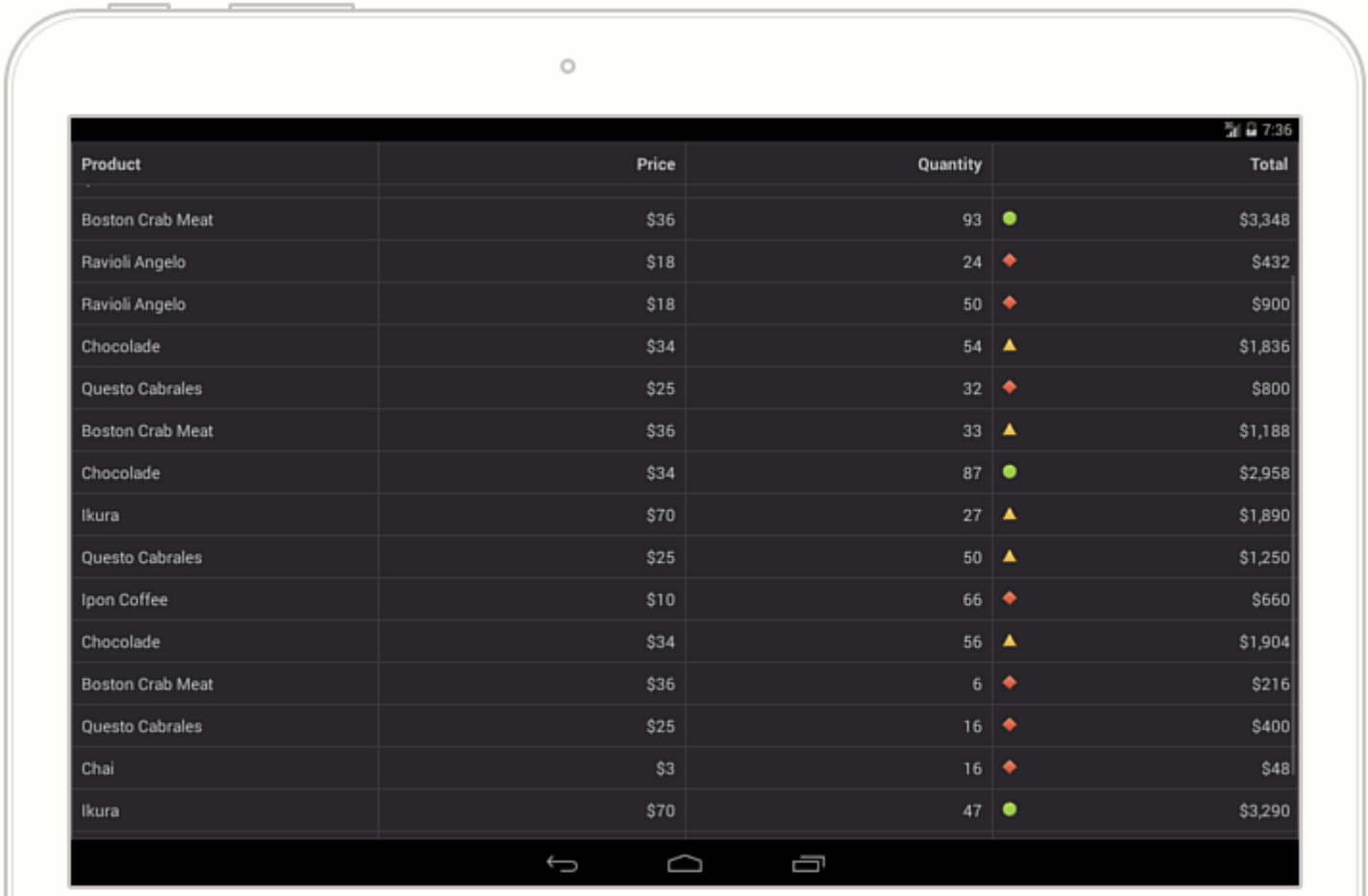
# How to: Apply an Icon Set Format to a Column



This example demonstrates how to apply an icon set conditional formatting rule to the **Total** column in a GridControl.

© 2018 Developer Express Inc.

| Product | Price | Quantity | Total |
|---|---|---|---|
| Ikura | $70 | 9 | $630 |
| Boston Crab Meat | $36 | 81 | $2,916 |
| Questo Cabrales | $25 | 19 | $475 |
| Ravioli Angelo | $18 | 17 | $306 |
| Ravioli Angelo | $18 | 94 | $1,692 |
| Ipon Coffee | $10 | 37 | $370 |
| Ravioli Angelo | $18 | 50 | $900 |
| Boston Crab Meat | $36 | 55 | $1,980 |
| Ikura | $70 | 33 | $2,310 |
| Ravioli Angelo | $18 | 72 | $1,296 |
| Chocolade | $34 | 73 | $2,482 |
| Ikura | $70 | 38 | $2,660 |
| Ikura | $70 | 36 | $2,520 |
| Ikura | $70 | 51 | $3,570 |
| Chocolade | $34 | 80 | $2,720 |
| Chai | $3 | 12 | $36 |




| Product | Price | Quantity | Total |
|---|---|---|---|
| Boston Crab Meat | $36 | 93 | $3,348 |
| Ravioli Angelo | $18 | 24 | $432 |
| Ravioli Angelo | $18 | 50 | $900 |
| Chocolade | $34 | 54 | $1,836 |
| Questo Cabrales | $25 | 32 | $800 |
| Boston Crab Meat | $36 | 33 | $1,188 |
| Chocolade | $34 | 87 | $2,958 |
| Ikura | $70 | 27 | $1,890 |
| Questo Cabrales | $25 | 50 | $1,250 |
| Ipon Coffee | $10 | 66 | $660 |
| Chocolade | $34 | 56 | $1,904 |
| Boston Crab Meat | $36 | 6 | $216 |
| Questo Cabrales | $25 | 16 | $400 |
| Chai | $3 | 16 | $48 |
| Ikura | $70 | 47 | $3,290 |

1. Create a new IconSetFormatCondition class instance.
2. Set this object's FormatConditionBase.FieldName property to **Total**. This column provides values to test against the formatting rule and the specified format will be applied to this column.
3. Assign one of the predefined icon sets to the FormatConditionBase.PredefinedFormatName property. All available format names are listed in the Predefined Format Names document.
4. To apply a conditional formatting rule represented by the created **IconSetFormatCondition** object, add this object to the GridControl.FormatConditions collection.

**C#**

```
using DevExpress.Mobile.DataGrid;
// ...
IconSetFormatCondition condition = new IconSetFormatCondition();
condition.FieldName = "Total";
condition.PredefinedFormatName = "Signs3IconSet";
grid.FormatConditions.Add (condition);
```

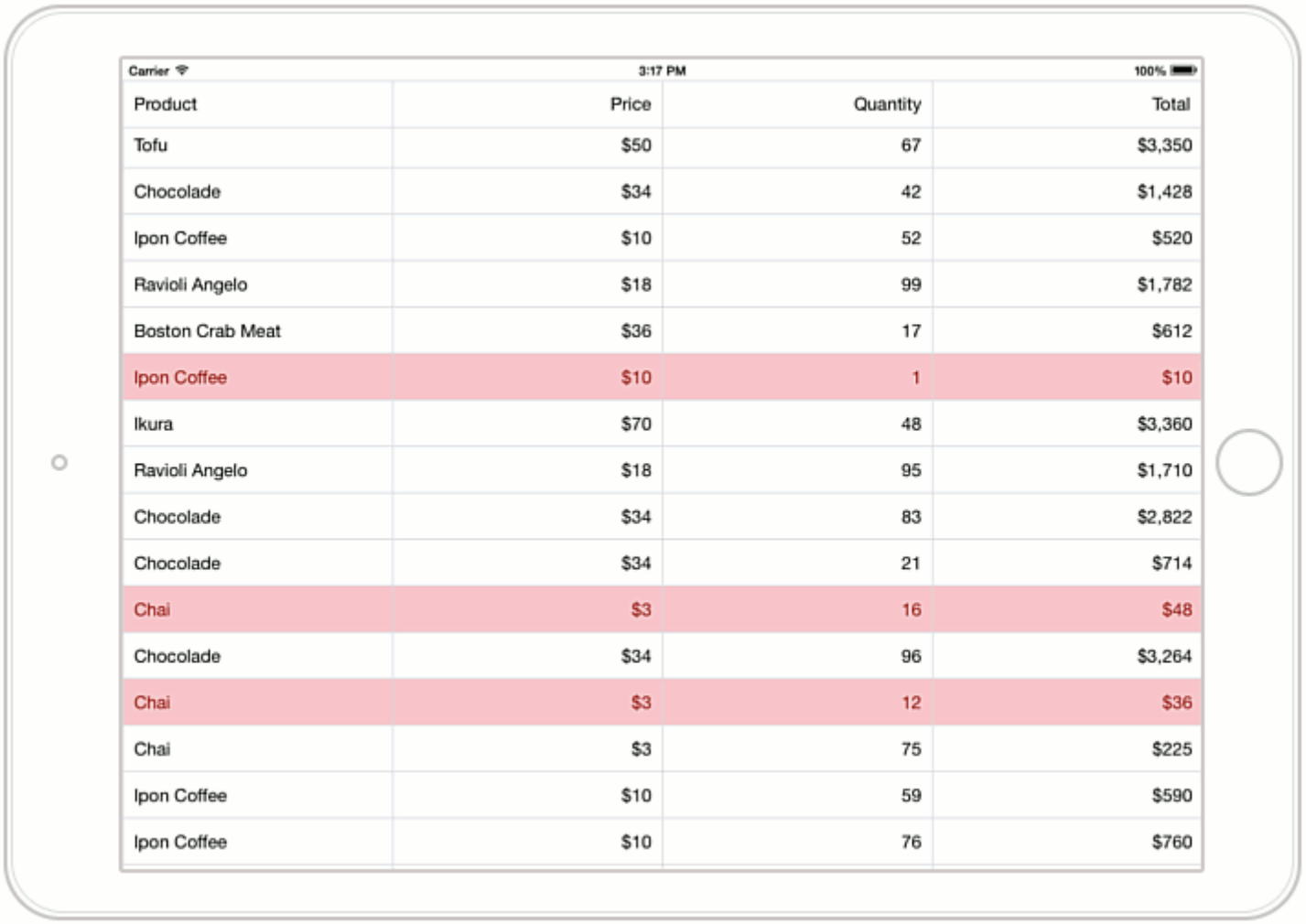
© 2018 Developer Express Inc.

# How to: Apply a Format Based on an Expression

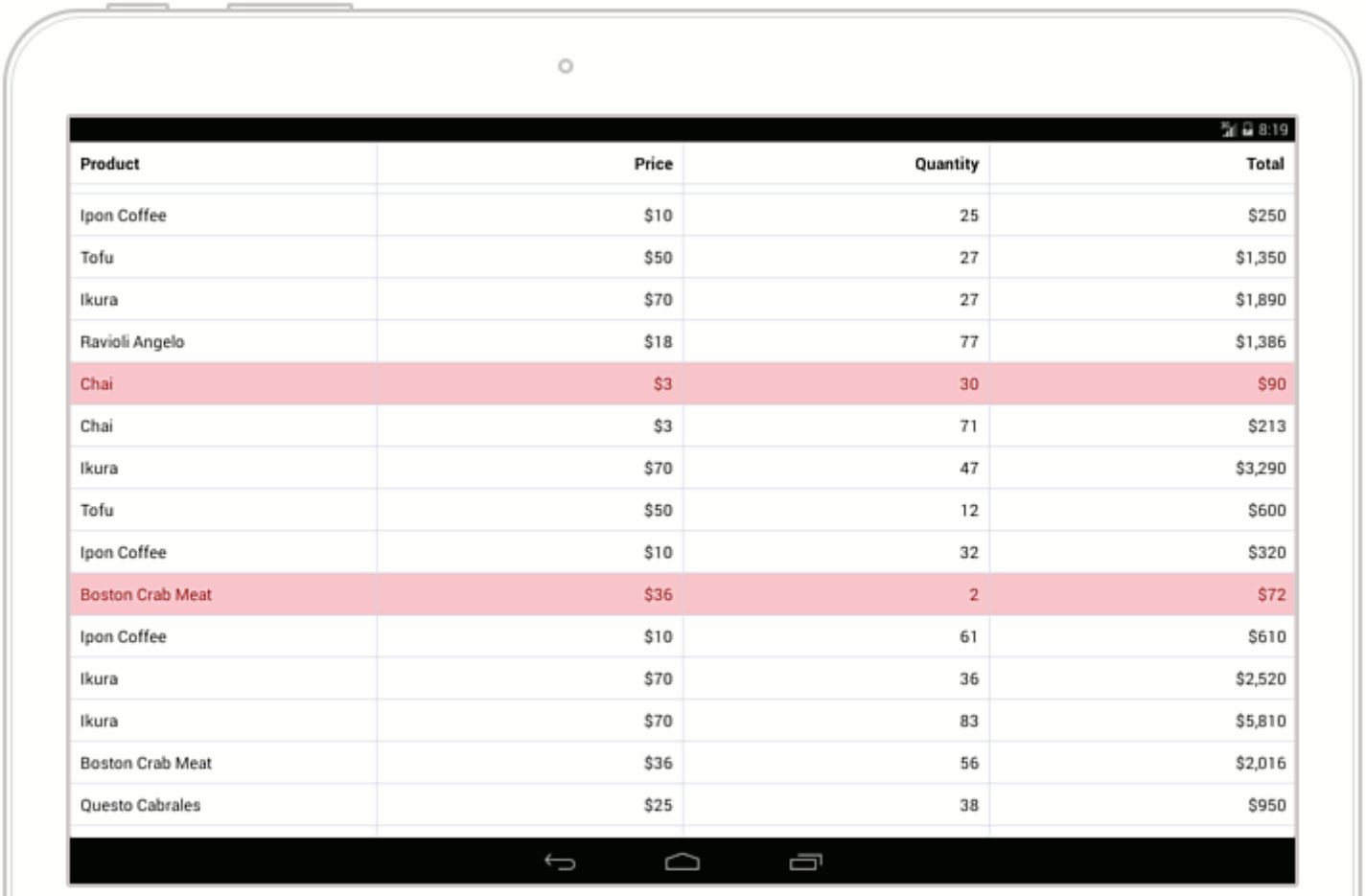


This example illustrates how to apply a format to rows in a GridControl that match a specific expression.

© 2018 Developer Express Inc.




| Product | Price | Quantity | Total |
|---|---|---|---|
| Tofu | $50 | 67 | $3,350 |
| Chocolade | $34 | 42 | $1,428 |
| Ipon Coffee | $10 | 52 | $520 |
| Ravioli Angelo | $18 | 99 | $1,782 |
| Boston Crab Meat | $36 | 17 | $612 |
| Ipon Coffee | $10 | 1 | $10 |
| Ikura | $70 | 48 | $3,360 |
| Ravioli Angelo | $18 | 95 | $1,710 |
| Chocolade | $34 | 83 | $2,822 |
| Chocolade | $34 | 21 | $714 |
| Chai | $3 | 16 | $48 |
| Chocolade | $34 | 96 | $3,264 |
| Chai | $3 | 12 | $36 |
| Chai | $3 | 75 | $225 |
| Ipon Coffee | $10 | 59 | $590 |
| Ipon Coffee | $10 | 76 | $760 |




| Product | Price | Quantity | Total |
|---|---|---|---|
| Ipon Coffee | $10 | 25 | $250 |
| Tofu | $50 | 27 | $1,350 |
| Ikura | $70 | 27 | $1,890 |
| Ravioli Angelo | $18 | 77 | $1,386 |
| Chai | $3 | 30 | $90 |
| Chai | $3 | 71 | $213 |
| Ikura | $70 | 47 | $3,290 |
| Tofu | $50 | 12 | $600 |
| Ipon Coffee | $10 | 32 | $320 |
| Boston Crab Meat | $36 | 2 | $72 |
| Ipon Coffee | $10 | 61 | $610 |
| Ikura | $70 | 36 | $2,520 |
| Ikura | $70 | 83 | $5,810 |
| Boston Crab Meat | $36 | 56 | $2,016 |
| Questo Cabrales | $25 | 38 | $950 |

An expression is a string that, when parsed and processed, evaluates some value. Expressions consist of column names, constants, operators, and functions. In this example, a Boolean expression is used. If the expression evaluates to **true**, the format is applied. The format highlights rows that have values in the **Total** column that are less than **100**.

1. Create a new FormatCondition class instance.
2. Set this object's FormatConditionBase.FieldName property to **Total**. This column provides values to test against the formatting rule.
3. Assign the expression string to the ExpressionConditionBase.Expression property.
4. Assign one of the predefined formats to the FormatConditionBase.PredefinedFormatName property. All available format names are listed in the Predefined Format Names document.
5. Set the ExpressionConditionBase.ApplyToRow property to **true** to apply the format to entire rows instead of individual column cells.
6. To apply a conditional formatting rule represented by the created **FormatCondition** object, add this object to the GridControl.FormatConditions collection.

**C#**

```
using DevExpress.Mobile.DataGrid;
// ...
FormatCondition condition = new FormatCondition();
condition.FieldName = "Total";
condition.Expression = "[Total] < 100";
condition.PredefinedFormatName = "LightRedFillWithDarkRedText";
condition.ApplyToRow = true;
grid.FormatConditions.Add (condition);
```

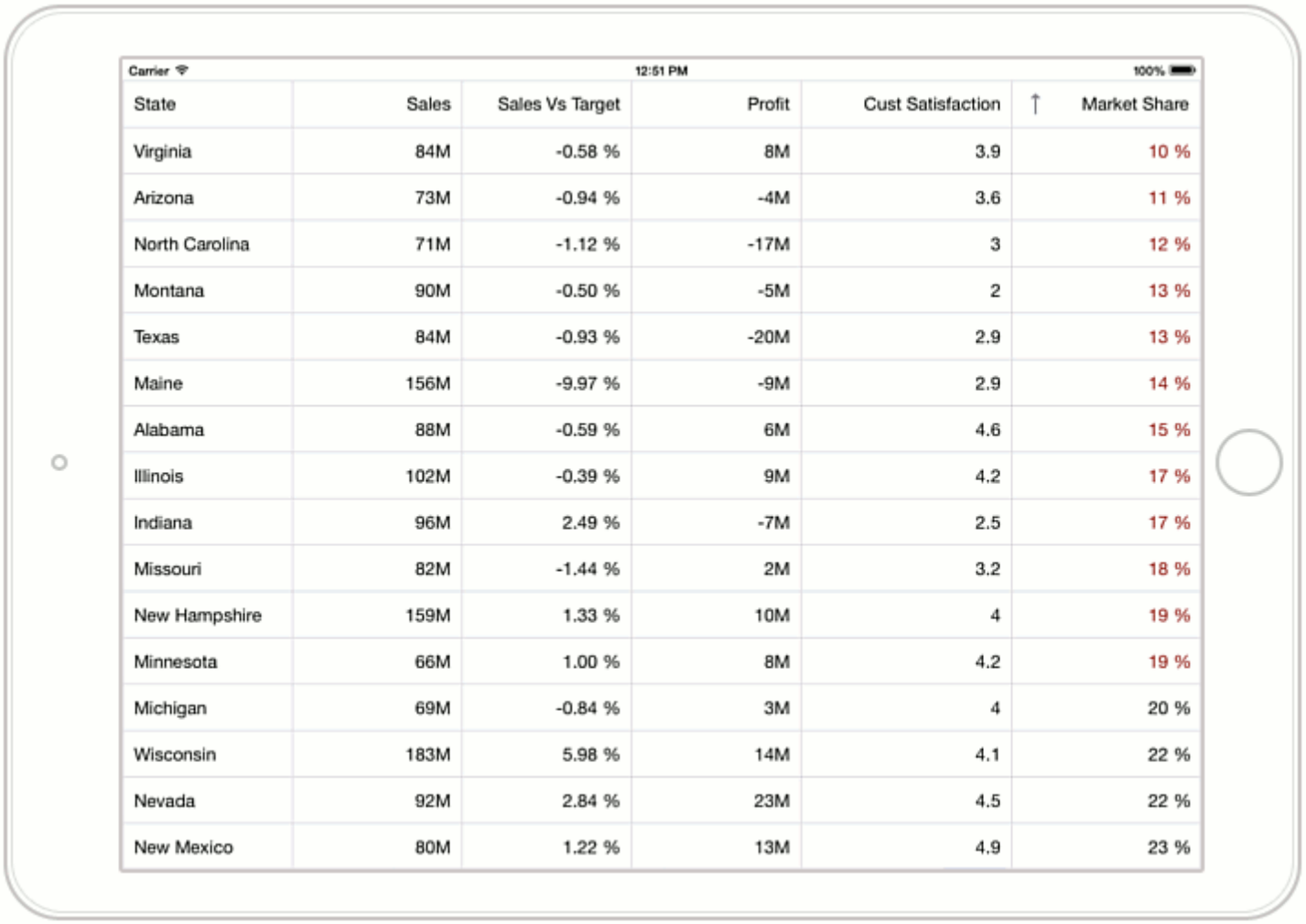
© 2018 Developer Express Inc.

# How to: Highlight Column Values that Match a Condition



This example illustrates how to apply conditional formatting to a **Market Share** column in a GridControl to highlight cells that match a specific condition. Cells containing values that are less than 20% will be highlighted.

© 2018 Developer Express Inc.



Carrier 12:51 PM 100%

| State | Sales | Sales Vs Target | Profit | Cust Satisfaction | ↑ Market Share |
|---|---|---|---|---|---|
| Virginia | 84M | -0.58 % | 8M | 3.9 | 10 % |
| Arizona | 73M | -0.94 % | -4M | 3.6 | 11 % |
| North Carolina | 71M | -1.12 % | -17M | 3 | 12 % |
| Montana | 90M | -0.50 % | -5M | 2 | 13 % |
| Texas | 84M | -0.93 % | -20M | 2.9 | 13 % |
| Maine | 156M | -9.97 % | -9M | 2.9 | 14 % |
| Alabama | 88M | -0.59 % | 6M | 4.6 | 15 % |
| Illinois | 102M | -0.39 % | 9M | 4.2 | 17 % |
| Indiana | 96M | 2.49 % | -7M | 2.5 | 17 % |
| Missouri | 82M | -1.44 % | 2M | 3.2 | 18 % |
| New Hampshire | 159M | 1.33 % | 10M | 4 | 19 % |
| Minnesota | 66M | 1.00 % | 8M | 4.2 | 19 % |
| Michigan | 69M | -0.84 % | 3M | 4 | 20 % |
| Wisconsin | 183M | 5.98 % | 14M | 4.1 | 22 % |
| Nevada | 92M | 2.84 % | 23M | 4.5 | 22 % |
| New Mexico | 80M | 1.22 % | 13M | 4.9 | 23 % |

5:54

| State | Sales | Sales Vs Target | Profit | Cust Satisfaction | ↑ Market Share |
|---|---|---|---|---|---|
| North Carolina | 71M | -1.12 % | -17M | 3 | 12 % |
| Montana | 90M | -0.50 % | -5M | 2 | 13 % |
| Texas | 84M | -0.93 % | -20M | 2.9 | 13 % |
| Maine | 156M | -9.97 % | -9M | 2.9 | 14 % |
| Alabama | 88M | -0.59 % | 6M | 4.6 | 15 % |
| Illinois | 102M | -0.39 % | 9M | 4.2 | 17 % |
| Indiana | 96M | 2.49 % | -7M | 2.5 | 17 % |
| Missouri | 82M | -1.44 % | 2M | 3.2 | 18 % |
| New Hampshire | 159M | 1.33 % | 10M | 4 | 19 % |
| Minnesota | 66M | 1.00 % | 8M | 4.2 | 19 % |
| Michigan | 69M | -0.84 % | 3M | 4 | 20 % |
| Wisconsin | 183M | 5.98 % | 14M | 4.1 | 22 % |
| Nevada | 92M | 2.84 % | 23M | 4.5 | 22 % |
| New Mexico | 80M | 1.22 % | 13M | 4.9 | 23 % |
| Oregon | 71M | 0.42 % | 6M | 4.5 | 23 % |

1. Create a new FormatCondition class instance.
2. Set this object's FormatConditionBase.FieldName property to **MarketShare**. This column provides values to test against the formatting rule and the specified format will be applied to this column.
3. Assign one of the predefined formats to the FormatConditionBase.PredefinedFormatName property. All available format names are listed in the Predefined Format Names document.
4. Specify the comparison operator by setting the FormatCondition.ValueRule property. In this example, the **Less** operator is used for constructing the rule's criteria.
5. Set the FormatCondition.Value1 property to 0.20. This means that the format should be applied to cells with market share less than 20%.
6. To apply a conditional formatting rule represented by the created **FormatCondition** object, add this object to the GridControl.FormatConditions collection.

**C#**

```
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.Core.ConditionalFormatting;
// ...
FormatCondition condition = new FormatCondition ();
condition.FieldName = "MarketShare";
condition.PredefinedFormatName = "RedText";
condition.ValueRule = ConditionRule.Less;
condition.Value1 = 0.20;
grid.FormatConditions.Add (condition);
```

© 2018 Developer Express Inc.

# How to: Set Conditional Formatting Rules in XAML

Xamarin Controls > Grid > Examples > Conditional Formatting > How to: Set Conditional Formatting Rules in XAML

This example shows how to create conditional formatting rules for the GridControl in XAML.

| State | Sales | Sales Vs Target | Profit | Cust Satisfaction | Market Share |
|---|---|---|---|---|---|
| Montana | 90M | -0.50 % | -5M | 2 | 13 % |
| Indiana | 96M | 2.49 % | -7M | 2.5 | 17 % |
| Maine | 156M | -9.97 % | -9M | 2.9 | 14 % |
| Texas | 84M | -0.93 % | -20M | 2.9 | 13 % |
| Connecticut | 73M | 3.38 % | 8M | 2.9 | 24 % |
| North Carolina | 71M | -1.12 % | -17M | 3 | 12 % |
| Missouri | 82M | -1.44 % | 2M | 3.2 | 18 % |
| Ohio | 251M | -2.88 % | -15M | 3.4 | 29 % |
| Arizona | 73M | -0.94 % | -4M | 3.6 | 11 % |
| Wyoming | 98M | 0.21 % | 4M | 3.7 | 31 % |
| Idaho | 119M | 0.65 % | 14M | 3.9 | 30 % |
| New York | 88M | 1.15 % | 15M | 3.9 | 30 % |
| Virginia | 84M | -0.58 % | 8M | 3.9 | 10 % |
| New Hampshire | 159M | 1.33 % | 10M | 4 | 19 % |
| Massachusetts | 91M | 1.40 % | 12M | 4 | 24 % |
| Michigan | 69M | -0.84 % | 3M | 4 | 20 % |

Formatting rules specified by the FormatConditionBase class descendants (DataBarFormatCondition, FormatCondition, TopBottomRuleFormatCondition and IconSetFormatCondition) are added to the GridControl.FormatConditions collection.

**Xaml**

```
<?xml version="1.0" encoding="UTF-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="HelloGrid.MainPage"
             xmlns:dxg="clr-namespace:DevExpress.Mobile.DataGrid;assembly=DevExpress.Mobile.Grid.v15.1">
    <dxg:GridControl x:Name = "grid" ItemsSource="{Binding Items}">
        <!-- ... -->
        <dxg:GridControl.FormatConditions>
           <dxg:DataBarFormatCondition FieldName="Sales" PredefinedFormatName="BlueGradientDataBar"/>
           <dxg:FormatCondition FieldName="SalesVsTarget" Expression="[SalesVsTarget]&lt;0" PredefinedFo
           <dxg:FormatCondition FieldName="SalesVsTarget" Expression="[SalesVsTarget]>=0" PredefinedForma
           <dxg:DataBarFormatCondition FieldName="Profit" PredefinedFormatName="GreenGradientDataBar" />
           <dxg:TopBottomRuleFormatCondition FieldName="CustomersSatisfaction" Rule="BottomPercent" Thres
           <dxg:IconSetFormatCondition FieldName="MarketShare" PredefinedFormatName="Arrows3ColoredIconSe
        </dxg:GridControl.FormatConditions>
    </dxg:GridControl>
</ContentPage>
```

© 2018 Developer Express Inc.

# Themes

Xamarin Controls > Grid > Examples > Themes

This section contains the following examples.

- How to: Apply Themes
- How to: Customize Themes
- How to: Customize Refresh Indicators

© 2018 Developer Express Inc.

# How to: Apply Themes

Xamarin Controls > Grid > Examples > Themes > How to: Apply Themes

DevExpress provides the **Light** and **Dark** built-in themes that you can apply to a GridControl to change its appearance.

The code below is an example of how to apply the Dark theme using the ThemeManager class.

**C#**

```
using Xamarin.Forms;
using DevExpress.Mobile.DataGrid.Theme;
            // ...
            ThemeManager.ThemeName = Themes.Dark;
            // ...
```

After you apply desired theme to the **GridControl**, you can customize the theme's appearance via its customizers (if needed). For more information, see the How to: Customize Themes example.

© 2018 Developer Express Inc.

# How to: Customize Themes

Xamarin Controls > Grid > Examples > Themes > How to: Customize Themes

| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T297553. | |

This example demonstrates how to customize a theme used to paint a GridControl.

To apply a theme to a **GridControl**, assign its name to the ThemeManager.ThemeName property. The Themes class stores available theme names.

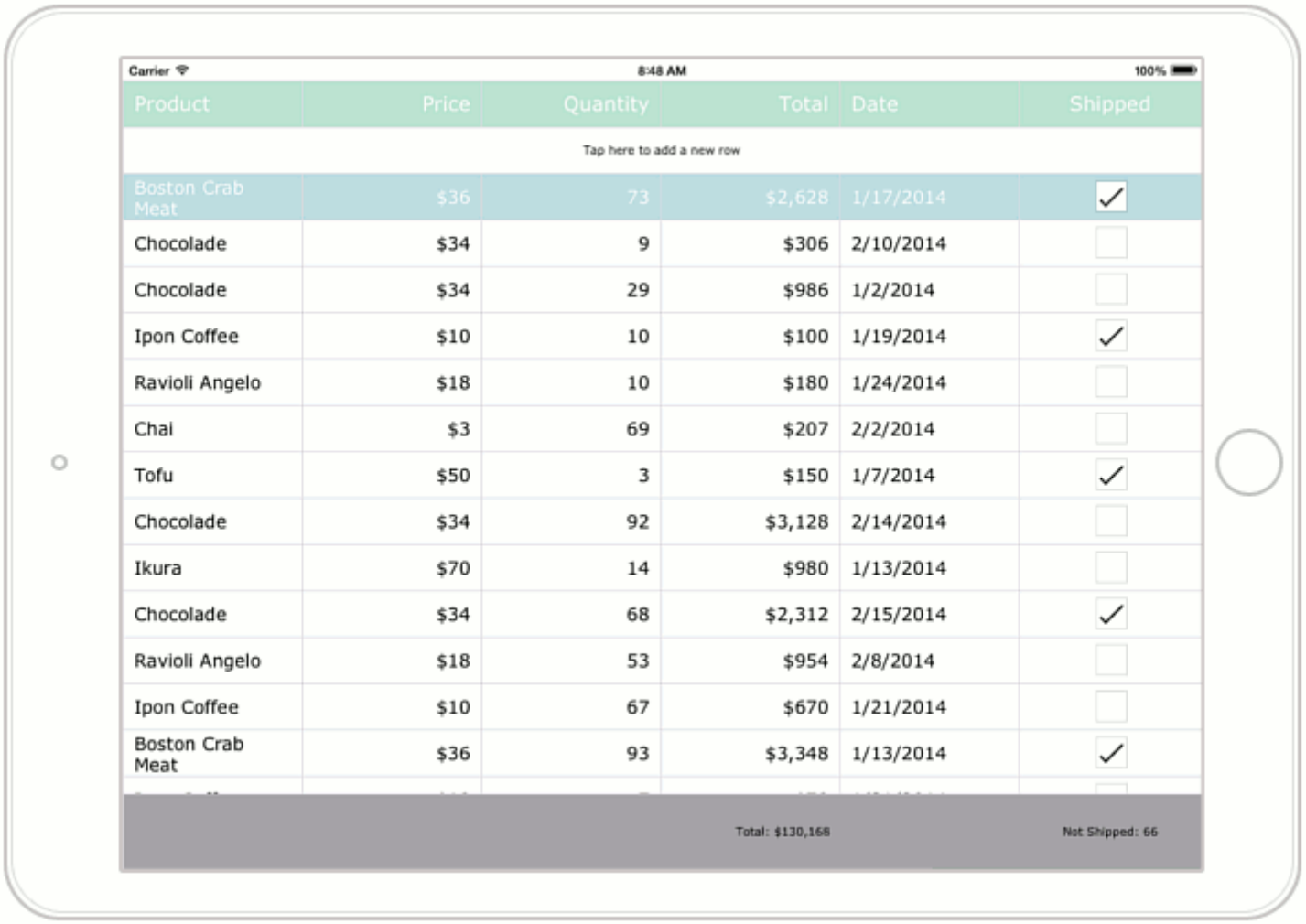
The ThemeBase class's properties provide access to different customizers, which you can adjust to change appearance settings (for example, font attributes, border and background colors, etc.) of the grid's corresponding visual elements (such as data cells, group rows, filter panel, etc.). In this example, the following customizers are used.

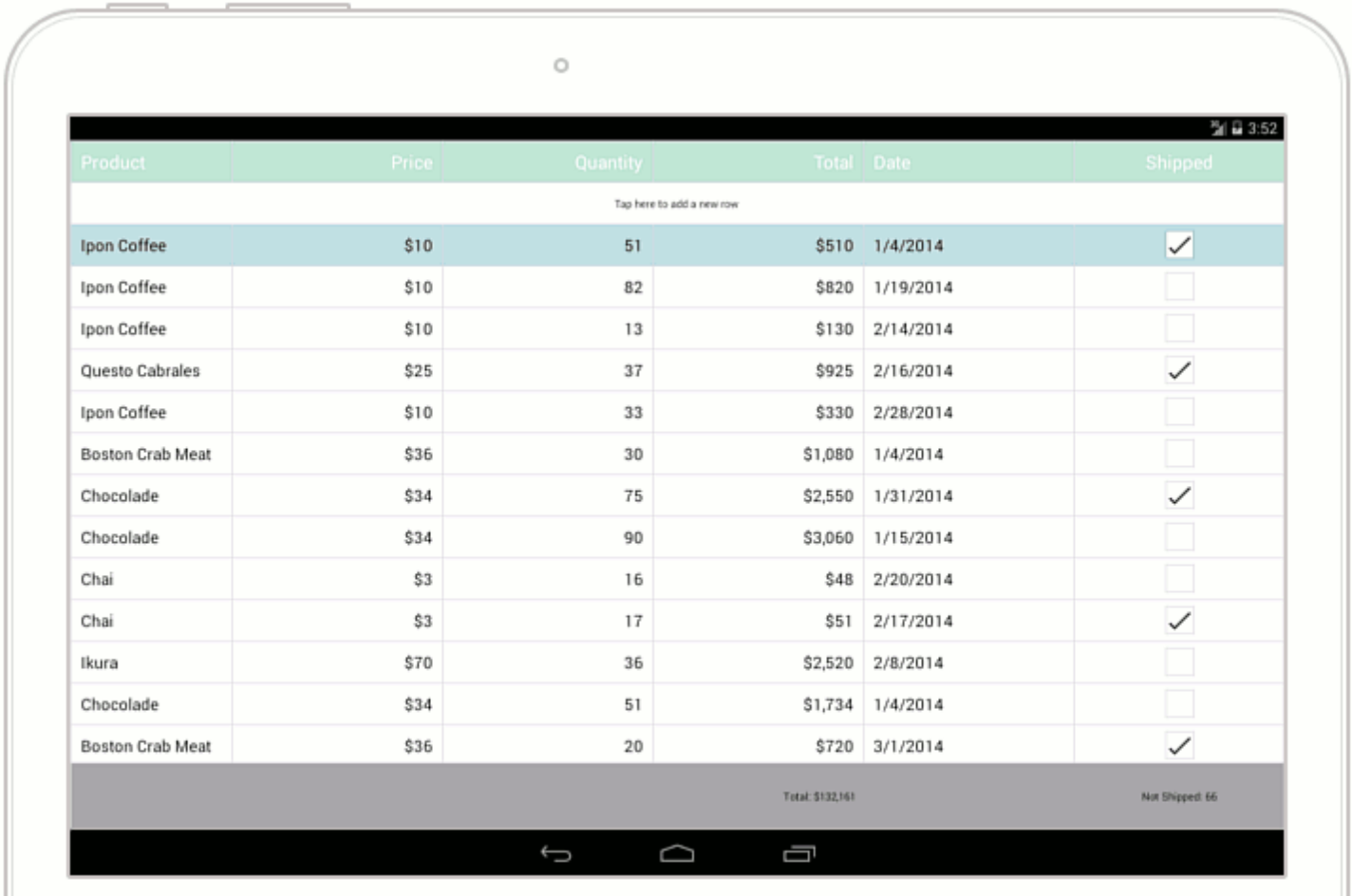
- ThemeBase.HeaderCustomizer
- ThemeBase.CellCustomizer
- ThemeBase.TotalSummaryCustomizer
- ThemeBase.NewItemRowCustomizer

| Important | |
|---|---|
| To apply the customization, call the ThemeManager.RefreshTheme method. | |

This example customizes the Light theme as it is shown in the image below.

© 2018 Developer Express Inc.


Carrier
8:48 AM
100%
Product
Price
Quantity
Total
Date
Shipped
Tap here to add a new row
Boston Crab Meat $36 73 $2,628 1/17/2014
Chocolade $34 9 $306 2/10/2014
Chocolade $34 29 $986 1/2/2014
Ipon Coffee $10 10 $100 1/19/2014
Ravioli Angelo $18 10 $180 1/24/2014
Chai $3 69 $207 2/2/2014
Tofu $50 3 $150 1/7/2014
Chocolade $34 92 $3,128 2/14/2014
Ikura $70 14 $980 1/13/2014
Chocolade $34 68 $2,312 2/15/2014
Ravioli Angelo $18 53 $954 2/8/2014
Ipon Coffee $10 67 $670 1/21/2014
Boston Crab Meat $36 93 $3,348 1/13/2014
Total: $130,168
Not Shipped: 66


3:52
Product
Price
Quantity
Total
Date
Shipped
Tap here to add a new row
Ipon Coffee $10 51 $510 1/4/2014
Ipon Coffee $10 82 $820 1/19/2014
Ipon Coffee $10 13 $130 2/14/2014
Questo Cabrales $25 37 $925 2/16/2014
Ipon Coffee $10 33 $330 2/28/2014
Boston Crab Meat $36 30 $1,080 1/4/2014
Chocolade $34 75 $2,550 1/31/2014
Chocolade $34 90 $3,060 1/15/2014
Chai $3 16 $48 2/20/2014
Chai $3 17 $51 2/17/2014
Ikura $70 36 $2,520 2/8/2014
Chocolade $34 51 $1,734 1/4/2014
Boston Crab Meat $36 20 $720 3/1/2014
Total: $132,161
Not Shipped: 66

**C#**

```
 (MainPage.xaml.cs)
using System;
using System.Collections.Generic;
using Xamarin.Forms;
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.DataGrid.Theme;
namespace ThemeCustomizing {
    public partial class MainPage : ContentPage {
        int count = 0;
        public MainPage() {
            InitializeComponent();
            TestOrdersRepository model = new TestOrdersRepository();
            BindingContext = model;
            ThemeManager.ThemeName = Themes.Light;
            // Header customization.
            ThemeManager.Theme.HeaderCustomizer.BackgroundColor = Color.FromRgb(187, 228, 208);
            ThemeFontAttributes myFont = new ThemeFontAttributes("Verdana",
                                         ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Large),
                                         FontAttributes.None, Color.White);
            ThemeManager.Theme.HeaderCustomizer.Font = myFont;
            // Cell customization.
            ThemeManager.Theme.CellCustomizer.SelectionColor = Color.FromRgb(186, 220, 225);
            ThemeFontAttributes myFont1 = new ThemeFontAttributes("Verdana",
                                         ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Medium),
                                         FontAttributes.None, Color.Black);
            ThemeManager.Theme.CellCustomizer.Font = myFont1;
            // Various customization.
            ThemeManager.Theme.TotalSumaryCustomizer.BackgroundColor = Color.FromRgb(163, 162, 168);
            ThemeFontAttributes myFont2 = new ThemeFontAttributes("Verdana",
                                 ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Default),
                                 FontAttributes.None, Color.Black);
            ThemeManager.Theme.TotalSummaryCustomizer.Font = myFont2;
            ThemeManager.Theme.NewItemRowCustomizer.Font = myFont2;
            ThemeManager.RefreshTheme();
        }
        void OnCalculateCustomSummary(object sender, CustomSummaryEventArgs e) {
            if (e.FieldName.ToString() == "Shipped")
            if (e.IsTotalSummary) {
                if (e.SummaryProcess == CustomSummaryProcess.Calculate) {
                    if (!(bool)e.FieldValue)
                        count++;
                    e.TotalValue = count;
                }
            }
        }
    }
}
```

© 2018 Developer Express Inc.

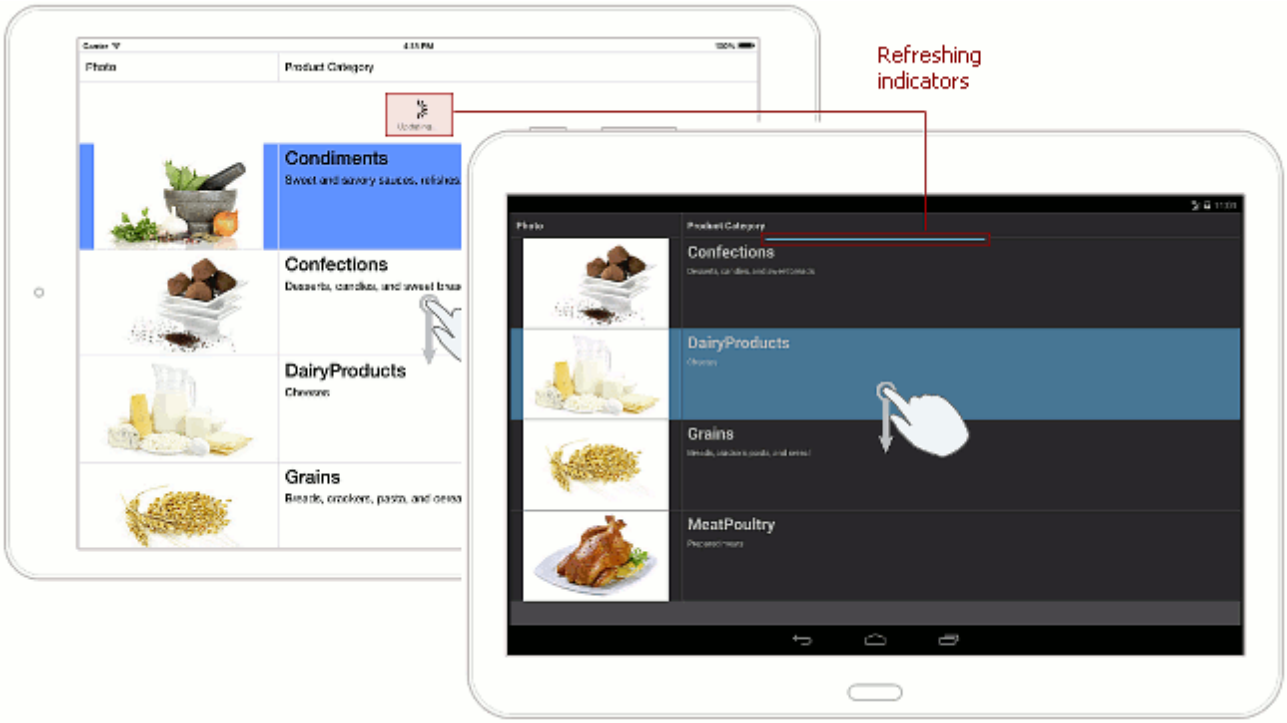
# How to: Customize Refresh Indicators

Xamarin Controls > Grid > Examples > Themes > How to: Customize Refresh Indicators

**Show Me**

A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T326911.

This example demonstrates how to customize a color of an indicator (icon or bar) shown when GridControl refreshes cell data.




To change the color of a refresh indicator, perform the steps below.

1. Activate a grid's **pull-to-refresh** functionality. For this purpose, assign **true** to the GridControl.IsPullToRefreshEnabled property.
2. Use the ThemeBase class's properties to access customizers used to change an indicator's color on different mobile platforms. In this example, the following customizers are used.
   - ThemeBase.AndroidRefreshCustomizer
   - ThemeBase.IOSRefreshCustomizer
3. Apply the customization by calling the ThemeManager.RefreshTheme method.

**C#**

© 2018 Developer Express Inc.

(MainPage.xaml.cs)

```
using System;
using System.Collections.Generic;
using Xamarin.Forms;
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.DataGrid.Theme;
namespace RefreshCustomizers {
    public partial class MainPage : ContentPage {
        public MainPage() {
            InitializeComponent();
            TestOrdersRepository model = new TestOrdersRepository();
            BindingContext = model;
            ThemeManager.Theme.AndroidRefreshCustomizer.BarColor = Color.Blue;
            ThemeManager.Theme.IOSRefreshCustomizer.RefreshColor = Color.Blue;
            ThemeManager.RefreshTheme();
        }
    }
}
```

## Xaml

(MainPage.xaml)

```
<?xml version="1.0" encoding="UTF-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="RefreshCustomizers.MainPage"
             xmlns:dxGrid="clr-namespace:DevExpress.Mobile.DataGrid;assemb
      <ContentPage.Padding>
        <OnPlatform x:TypeArguments="Thickness" iOS="0, 20, 0, 0" />
    </ContentPage.Padding>
    <dxGrid:GridControl x:Name="grid" ItemsSource="{Binding Orders}"
                         NewItemRowVisibility = "false"
                        SortMode = "Multiple" AutoFilterPanelVisibility="t
                        IsPullToRefreshEnabled = "true" >
         <dxGrid:GridControl.Columns>
             <dxGrid:TextColumn FieldName="Product.Name" Caption = "Produc
                                SortOrder = "Descending" SortIndex = "0"/>
             <dxGrid:NumberColumn FieldName="Product.UnitPrice" Caption =
             <dxGrid:NumberColumn FieldName="Quantity"
                                  SortOrder = "Ascending" SortIndex = "1"/
             <dxGrid:NumberColumn FieldName="Total"
                                  UnboundType="Integer" UnboundExpression=
                                  DisplayFormat="C0" IsReadOnly="True"/>
             <dxGrid:DateColumn FieldName="Date" DisplayFormat="d" />
             <dxGrid:SwitchColumn FieldName="Shipped"/>
         </dxGrid:GridControl.Columns>
         <dxGrid:GridControl.GroupSummaries>
            <dxGrid:GridColumnSummary FieldName="Total" Type="Max"/>
        </dxGrid:GridControl.GroupSummaries>
        <dxGrid:GridControl.TotalSummaries>
            <dxGrid:GridColumnSummary FieldName="Total" Type="Sum"
                                      DisplayFormat= "Total: {0:C0}"/>
        </dxGrid:GridControl.TotalSummaries>
    </dxGrid:GridControl>
</ContentPage>
```

© 2018 Developer Express Inc.

# How to: Create Grid Columns for Data of Different Types

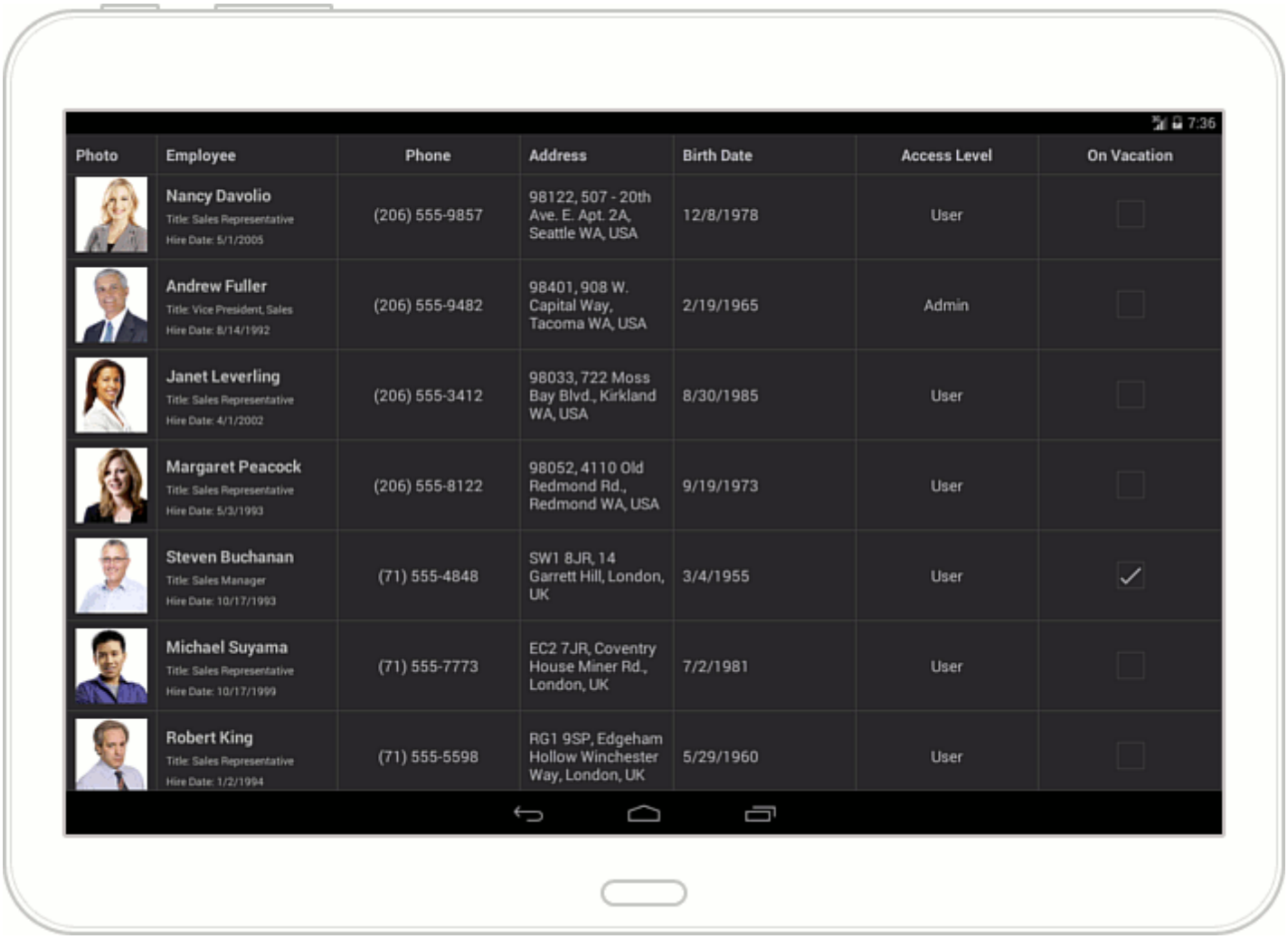


**Show Me**

A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T208848.

This example shows how to add columns to GridControl to display and edit data of different types. The grid is bound to a collection of **Employee** objects. Each **Employee** object contains an employee's photo (image), name, position, phone, address (strings), hire and birth dates (DateTime values), a value that specifies an employee's access level, and a Boolean value indicating whether an employee is on vacation.

© 2018 Developer Express Inc.




Grid columns are stored in the GridControl.Columns collection. An individual column is specified by a GridColumn descendant, which is appropriate to the type of data contained in this column. In this example, to display information on employees and allow end-users to edit it, the following columns are created in the grid.

| Grid Column | Description |
|---|---|
| Photo<br>ImageColumn | This column displays employee photos which are images added to a project as embedded resources. |
| Employee<br>TemplateColumn | A cell in this column displays three employee properties: Name, Position and HireDate. A cell appearance is defined via a template. Each cell contains three Xamarin.Forms.Label elements within a Xamarin.Forms.Grid. Each Label element is bound to a property of the **Employee** class.<br><br>A cell template data context is specified by the CellData object. Its CellData.Value property provides access to a value of a data field assigned to the column's GridColumn.FieldName property. In this example, a column cell displays not only this field value but also values of two more fields. So, it is required to specify the whole data row as a binding source for the template. To do this, set the BindingContext property of the Grid to CellData.Source. Note that BindingContext specified for the Grid is inherited by all its children. This means that all Label elements have the same BindingContext as the Grid, and you can specify |

© 2018 Developer Express Inc.

| | |
|---|---|
| | their simple bindings to properties of that object (Employee).<br><br>Use the GridColumn.AllowSort, GridColumn.AllowGroup and GridColumn.IsReadOnly properties to prevent end-users from sorting and grouping data in the grid by this column and disable data editing. |
| Phone<br>NumberColumn | The employee's **Phone** property is of the string type. The keyboard allowing text input is automatically displayed when an end-user activates this column cell to edit an employee's phone. |
| Address<br>TextColumn | This column is bound to the employee's **Address** string property. The keyboard allowing text input is automatically displayed when an end-user activates this column cell to edit an employee's address. |
| Birth Date<br>DateColumn | This column displays and allows editing employee birth days. The GridColumn.DisplayFormat property specifies the format of displaying dates. |
| Access Level<br>PickerColumn | The employee's **Access** property is of the **AccessLevel** type which is an enumeration with values **Admin** and **User**. To allow editing an employee's access level by selecting one of two enumeration values, this property is bound to the grid's picker column. |
| On Vacation<br>SwitchColumn | The employee's **OnVacation** property is of the Boolean type, so it is bound to the grid's switch column which allows editing a cell value by selecting between two states. |

**C#**

```
 (Employee.cs)
public enum AccessLevel {
    Admin,
    User
}
public class Employee : ModelObject {
    string name;
    public string Name {
        get { return name; }
        set {
            name = value;
            if (Photo == null)
                Photo = ImageSource.FromResource (value.Replace (" ", String.Empty));
        }
    }
    public Employee(string name) {
        this.Name = name;
    }
    public ImageSource Photo { get; set; }
    public DateTime BirthDate { get; set; }
    public DateTime HireDate { get; set; }
    public string Position { get; set; }
    public string Address { get; set; }
    public string Phone { get; set; }
    public AccessLevel Access { get; set; }
    public bool OnVacation { get; set; }
}
```

**C#**

```
 (EmployeesRepository.cs)
public abstract class EmployeesRepository {
    readonly ObservableCollection<Employee> employees;
    public EmployeesRepository() {
        this.employees = new ObservableCollection<Employee>();
    }
    public ObservableCollection<Employee> Employees {
        get { return employees; }
    }
}
public class TestEmployeesRepository : EmployeesRepository {
    public TestEmployeesRepository() : base() {
        GenerateCustomers();
    }
    void GenerateCustomers() {
        Employees.Add(
            new Employee("Nancy Davolio") {
                BirthDate = new DateTime(1978, 12, 8),
                HireDate = new DateTime(2005, 5, 1),
                Position = "Sales Representative",
                Address = "98122, 507 - 20th Ave. E. Apt. 2A, Seattle WA,
                Phone = "(206) 555-9857",
                Access = AccessLevel.User,
                OnVacation = false
            }
        );
        Employees.Add(
            new Employee("Andrew Fuller") {
                BirthDate = new DateTime(1965, 2, 19),
                HireDate = new DateTime(1992, 8, 14),
                Position = "Vice President, Sales",
                Address = "98401, 908 W. Capital Way, Tacoma WA, USA",
                Phone = "(206) 555-9482",
                Access = AccessLevel.Admin,
                OnVacation = false
            }
        );
        Employees.Add(
            new Employee("Janet Leverling") {
                BirthDate = new DateTime(1985, 8, 30),
                HireDate = new DateTime(2002, 4, 1),
                Position = "Sales Representative",
                Address = "98033, 722 Moss Bay Blvd., Kirkland WA, USA",
                Phone = "(206) 555-3412",
                Access = AccessLevel.User,
                OnVacation = false
            }
        );
        Employees.Add(
            new Employee("Margaret Peacock") {
                BirthDate = new DateTime(1973, 9, 19),
```

© 2018 Developer Express Inc.

```
            HireDate = new DateTime(1993, 5, 3),
            Position = "Sales Representative",
            Address = "98052, 4110 Old Redmond Rd., Redmond WA, USA",
            Phone = "(206) 555-8122",
            Access = AccessLevel.User,
            OnVacation = false
        }
    );
    Employees.Add(
        new Employee("Steven Buchanan") {
            BirthDate = new DateTime(1955, 3, 4),
            HireDate = new DateTime(1993, 10, 17),
            Position = "Sales Manager",
            Address = "SW1 8JR, 14 Garrett Hill, London, UK",
            Phone = "(71) 555-4848",
            Access = AccessLevel.User,
            OnVacation = true
        }
    );
    Employees.Add(
        new Employee("Michael Suyama") {
            BirthDate = new DateTime(1981, 7, 2),
            HireDate = new DateTime(1999, 10, 17),
            Position = "Sales Representative",
            Address = "EC2 7JR, Coventry House Miner Rd., London, UK",
            Phone = "(71) 555-7773",
            Access = AccessLevel.User,
            OnVacation = false
        }
    );
    Employees.Add(
        new Employee("Robert King") {
            BirthDate = new DateTime(1960, 5, 29),
            HireDate = new DateTime(1994, 1, 2),
            Position = "Sales Representative",
            Address = "RG1 9SP, Edgeham Hollow Winchester Way, London,
            Phone = "(71) 555-5598",
            Access = AccessLevel.User,
            OnVacation = false
        }
    );
    Employees.Add(
        new Employee("Laura Callahan") {
            BirthDate = new DateTime(1985, 1, 9),
            HireDate = new DateTime(2004, 3, 5),
            Position = "Inside Sales Coordinator",
            Address = "98105, 4726 - 11th Ave. N.E., Seattle WA, USA",
            Phone = "(206) 555-1189",
            Access = AccessLevel.User,
            OnVacation = true
        }
```

© 2018 Developer Express Inc.

```
        );
        Employees.Add(
            new Employee("Anne Dodsworth") {
                BirthDate = new DateTime(1980, 1, 27),
                HireDate = new DateTime(2004, 11, 15),
                Position = "Sales Representative",
                Address = "WG2 7LT, 7 Houndstooth Rd., London, UK",
                Phone = "(71) 555-4444",
                Access = AccessLevel.User,
                OnVacation = false
            }
        );
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl ItemsSource="{Binding Employees}" RowHeight="100">
    <dxGrid:GridControl.Columns>
        <dxGrid:ImageColumn FieldName="Photo" Width="100" />
        <dxGrid:TemplateColumn FieldName="Name" Caption="Employee" Width="
                               IsReadOnly="true" AllowGroup="False" AllowS
          <dxGrid:TemplateColumn.DisplayTemplate>
            <DataTemplate>
              <Grid BindingContext="{Binding Source}" Padding="0,15,0,15">
                <Grid.RowDefinitions>
                  <RowDefinition Height="Auto" />
                  <RowDefinition Height="Auto" />
                  <RowDefinition Height="Auto" />
                </Grid.RowDefinitions>
                <Label Text="{Binding Name}" Font="Bold, 18" Grid.Row="0"
                <Label Text="{Binding Position, StringFormat = 'Title: {0}
                <Label Text="{Binding HireDate, StringFormat = 'Hire Date:
              </Grid>
            </DataTemplate>
          </dxGrid:TemplateColumn.DisplayTemplate>
        </dxGrid:TemplateColumn>
        <dxGrid:NumberColumn FieldName="Phone" ContentAlignment="Center"/>
        <dxGrid:TextColumn FieldName="Address" Width="170"/>
        <dxGrid:DateColumn FieldName="BirthDate" DisplayFormat="d"/>
        <dxGrid:PickerColumn FieldName="Access" Caption="Access Level" Con
        <dxGrid:SwitchColumn FieldName="OnVacation"/>
    </dxGrid:GridControl.Columns>
</dxGrid:GridControl>
```

© 2018 Developer Express Inc.

# How to: Implement a PickerColumn



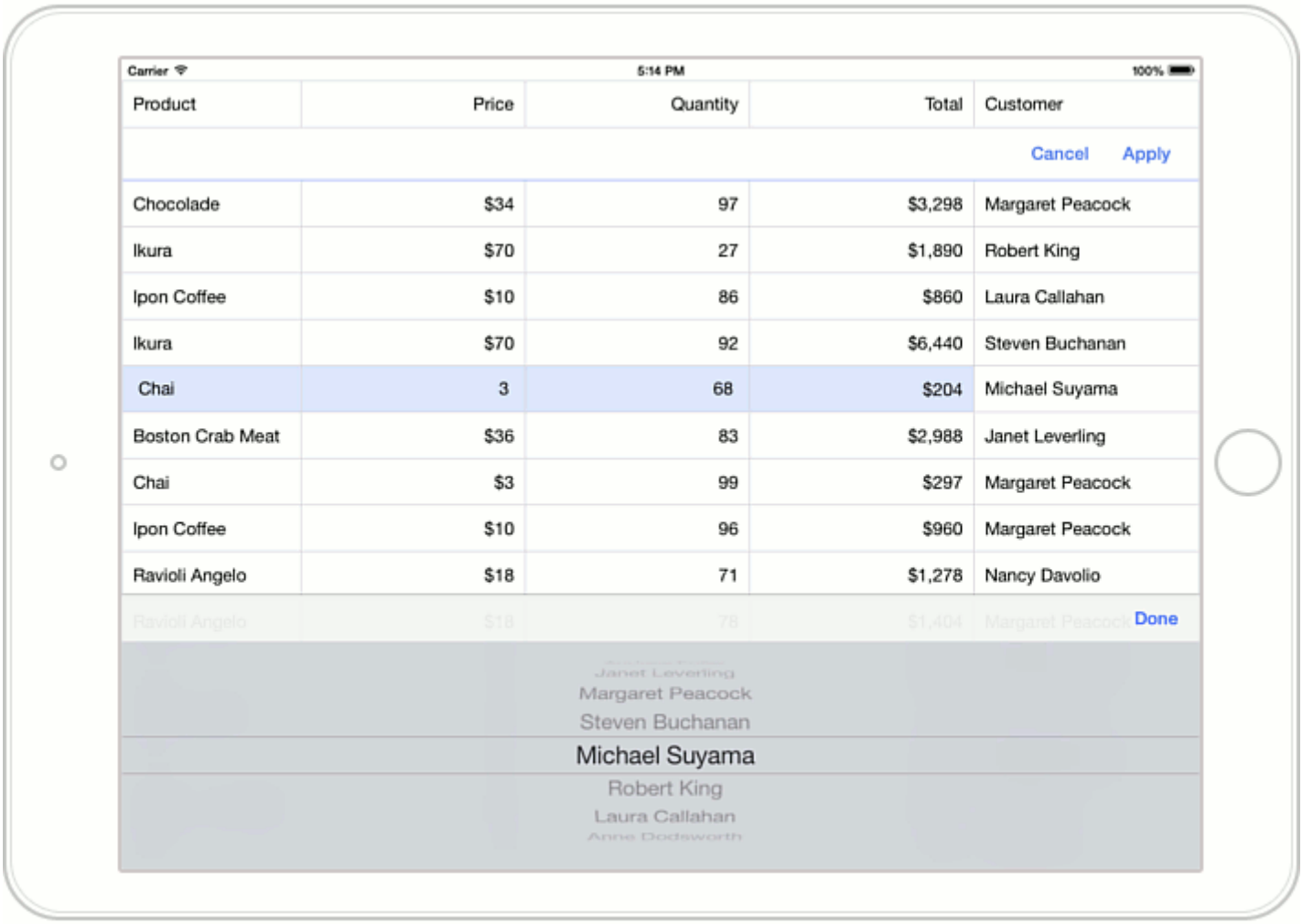
| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T257726. | |

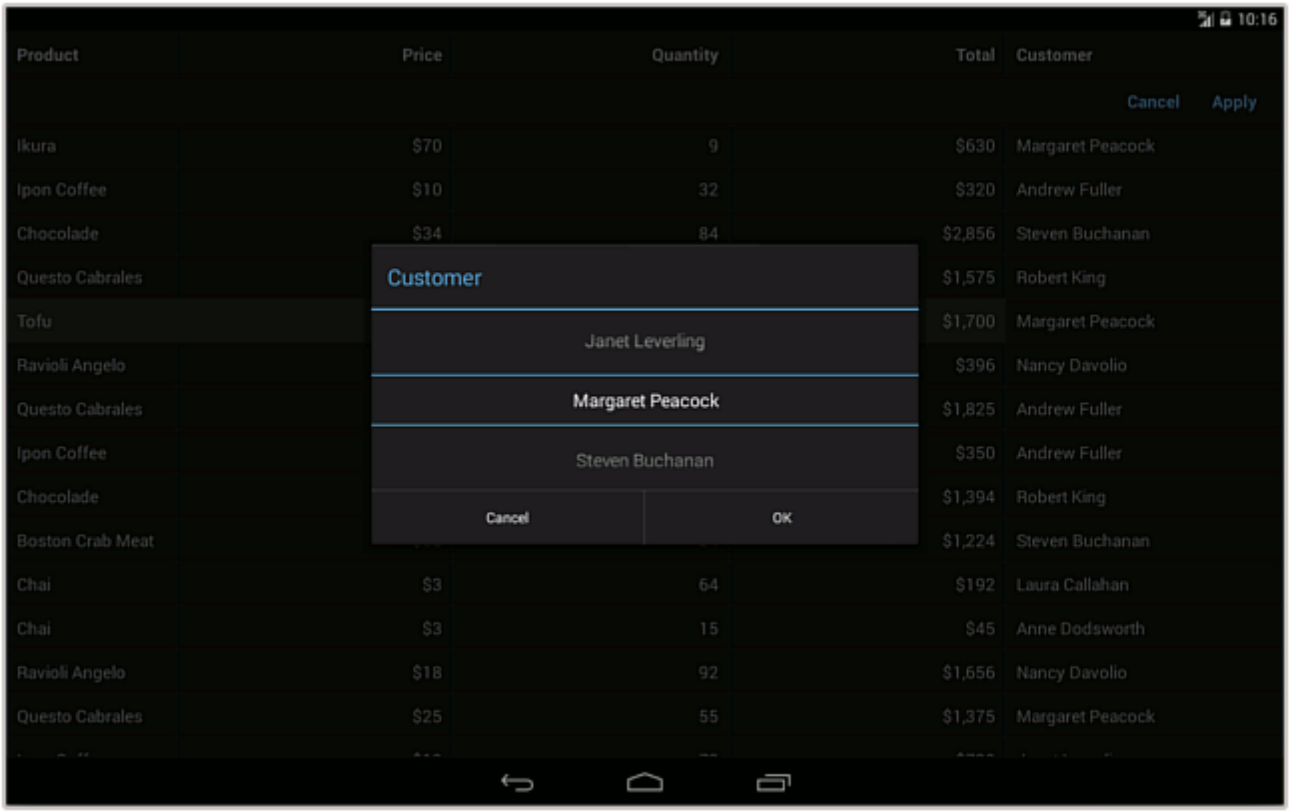
This example demonstrates how to implement the grid's PickerColumn whose items are coming from the grid's underlying data source.

In this example, a grid is bound to a collection of the **Order** objects. Each order contains information about a customer (the **CustomerId** property). A customer is represented by a **Customer** object that has the **Id** and **Name** properties. **Customer** objects are stored in an IList collection. The following properties of the PickerColumn object are set to show customer names for orders in this column within the grid (see the *MainPage.xaml* file):

- GridColumn.FieldName — an order's property to be bound to a column (**CustomerId**).
- PickerColumn.ItemsSource — a collection of customers.
- PickerColumn.ValueMember — a customer's property used to associate a customer with an order (**Id**).
- PickerColumn.DisplayMember — a customer's property to be displayed in a column (**Name**).

© 2018 Developer Express Inc.


Carrier
5:14 PM
100%
Product
Price
Quantity
Total
Customer
Cancel
Apply
Chocolade $34 97 $3,298 Margaret Peacock
Ikura $70 27 $1,890 Robert King
Ipon Coffee $10 86 $860 Laura Callahan
Ikura $70 92 $6,440 Steven Buchanan
Chai 3 68 $204 Michael Suyama
Boston Crab Meat $36 83 $2,988 Janet Leverling
Chai $3 99 $297 Margaret Peacock
Ipon Coffee $10 96 $960 Margaret Peacock
Ravioli Angelo $18 71 $1,278 Nancy Davolio
Done
Janet Leverling
Margaret Peacock
Steven Buchanan
Michael Suyama
Robert King
Laura Callahan


10:16
Product
Price
Quantity
Total
Customer
Cancel
Apply
Ikura $70 9 $630 Margaret Peacock
Ipon Coffee $10 32 $320 Andrew Fuller
Chocolade $34 84 $2,856 Steven Buchanan
Questo Cabrales $1,575 Robert King
Tofu $1,700 Margaret Peacock
Ravioli Angelo $396 Nancy Davolio
Questo Cabrales $1,825 Andrew Fuller
Ipon Coffee $350 Andrew Fuller
Chocolade $1,394 Robert King
Boston Crab Meat $1,224 Steven Buchanan
Chai $3 64 $192 Laura Callahan
Chai $3 15 $45 Anne Dodsworth
Ravioli Angelo $18 92 $1,656 Nancy Davolio
Questo Cabrales $25 55 $1,375 Margaret Peacock
Customer
Janet Leverling
Margaret Peacock
Steven Buchanan
Cancel
OK

C#

```
 (Customer.cs)
using System;
namespace PickerColumnExample
{
    public class Customer : ModelObject {
        string name;
        public Customer(string name, int id) {
            this.name = name;
            this.Id = id;
        }
        public int Id { get; set; }
        public string Name {
            get { return name; }
            set {name = value;}
        }
    }
}
```

C#

```
 (MainPage.xaml.cs)
using System;
using System.Collections.Generic;
using Xamarin.Forms;
namespace PickerColumnExample {
    public partial class MainPage : ContentPage {
        public MainPage () {
            InitializeComponent ();
            TestOrdersRepository model = new TestOrdersRepository ();
            BindingContext = model;
        }
    }
}
```

C#

```
 (ModelObject.cs)
using System;
using System.ComponentModel;
namespace PickerColumnExample
{
    public class ModelObject : INotifyPropertyChanged {
        public event PropertyChangedEventHandler PropertyChanged;
        protected void RaisePropertyChanged(string name) {
            if (PropertyChanged != null)
                PropertyChanged(this, new PropertyChangedEventArgs(name));
        }
    }
}
```

C#

© 2018 Developer Express Inc.

```
 (Order.cs)
using System;
namespace PickerColumnExample
{
    public class Order : ModelObject {
        Product product;
        int quantity;
        int customerId;
        public Order() {
            this.product = new Product ("", 0);
            this.quantity = 0;
        }
        public Order(Product product, int amount, int customerId) {
            this.product = product;
            this.quantity = amount;
            this.customerId = customerId;
        }
        public Product Product {
            get { return product; }
            set { if (product != value) {
                    product = value;
                    RaisePropertyChanged ("Product");}}
        }
        public int Quantity {
            get { return quantity; }
            set { if (quantity != value) {
                    quantity = value;
                    RaisePropertyChanged ("Quantity");}}
        }
        public int CustomerId {
            get { return customerId; }
            set {
                if (customerId != value) {
                    customerId = value;
                    RaisePropertyChanged("CustomerId");
                }
            }
        }
    }
}
```

**C#**

© 2018 Developer Express Inc.

```
 (OrdersRepository.cs)
using System;
using System.Collections.Generic;
using System.Collections.ObjectModel;
namespace PickerColumnExample
{
    public abstract class OrdersRepository {
        readonly ObservableCollection<Order> orders;
        readonly ObservableCollection<Customer> customers;
        public OrdersRepository() {
            this.orders = new ObservableCollection<Order>();
            this.customers = new ObservableCollection<Customer>();
        }
        public ObservableCollection<Order> Orders {
            get { return orders; }
        }
        public IList<Customer> Customers{
            get { return customers; }
        }
    }
    public class TestOrdersRepository : OrdersRepository {
        const int orderCount = 100;
        readonly List<Product> products;
        readonly Random random;
        public TestOrdersRepository() : base() {
            this.random = new Random((int)DateTime.Now.Ticks);
            this.products = new List<Product>();
            GenerateProducts();
            GenerateCustomers();
            for (int i = 0; i < orderCount; i++)
                Orders.Add(GenerateOrder(i));
        }
        Order GenerateOrder(int number) {
            Order order = new Order(RandomItem<Product>(products), random.
            return order;
        }
        T RandomItem<T>(IList<T> list) {
            int index = (int)(random.NextDouble() * 0.99 * (list.Count));
            return list[index];
        }
        void GenerateProducts() {
            products.Add(new Product("Tofu", 50));
            products.Add(new Product("Chocolade", 34));
            products.Add(new Product("Ikura", 70));
            products.Add(new Product("Chai", 3));
            products.Add(new Product("Boston Crab Meat", 36));
            products.Add(new Product("Ravioli Angelo", 18));
            products.Add(new Product("Ipon Coffee", 10));
            products.Add(new Product("Questo Cabrales", 25));
        }
        void GenerateCustomers() {
```

© 2018 Developer Express Inc.

```
            Customers.Add(new Customer("Nancy Davolio", 0));
            Customers.Add(new Customer("Andrew Fuller", 1));
            Customers.Add(new Customer("Janet Leverling", 2));
            Customers.Add(new Customer("Margaret Peacock", 3));
            Customers.Add(new Customer("Steven Buchanan", 4));
            Customers.Add(new Customer("Michael Suyama", 5));
            Customers.Add(new Customer("Robert King", 6));
            Customers.Add(new Customer("Laura Callahan", 7));
            Customers.Add(new Customer("Anne Dodsworth", 8));
        }
    }
}
```

**C#**

```
 (Product.cs)
using System;
namespace PickerColumnExample
{
    public class Product : ModelObject {
        string name;
        int unitPrice;
        public Product(string name, int unitPrice) {
            this.name = name;
            this.unitPrice = unitPrice;
        }
        public string Name {
            get { return name; }
            set { name = value; }
        }
        public int UnitPrice {
            get { return unitPrice; }
            set { unitPrice = value; }
        }
    }
}
```

**Xaml**

© 2018 Developer Express Inc.

```
 (MainPage.xaml)
<?xml version="1.0" encoding="UTF-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="PickerColumnExample.MainPage"
             xmlns:dxg="clr-namespace:DevExpress.Mobile.DataGrid;assembly=
    <ContentPage.Padding>
        <OnPlatform x:TypeArguments="Thickness" iOS="0, 20, 0, 0" />
    </ContentPage.Padding>
    <dxg:GridControl x:Name="grid" ItemsSource="{Binding Orders}">
         <dxg:GridControl.Columns>
             <dxg:TextColumn FieldName="Product.Name" Caption="Product" Wi
             <dxg:NumberColumn FieldName="Product.UnitPrice" Caption="Pric
             <dxg:NumberColumn FieldName="Quantity"/>
             <dxg:NumberColumn FieldName="Total"
                               UnboundType="Integer" UnboundExpression=
                               DisplayFormat="C0" IsReadOnly="True"/>
             <dxg:PickerColumn FieldName="CustomerId" Caption="Customer"
                               ItemsSource="{Binding Customers}" ValueMemb
         </dxg:GridControl.Columns>
    </dxg:GridControl>
</ContentPage>
```

© 2018 Developer Express Inc.

# How to: Bind a GridControl to a DataTable



**Show Me**

A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T243160.

This example demonstrates how to use a DataTable from an iOS and Android projects as a data source for GridControl. The table below lists projects of the solution, files contained in these projects and added assembly references.

| Project | Files | References |
|---|---|---|
| PCL | *MainPage.xaml.cs*, *MainPage.xaml* - The **MainPage** content page with a grid will be the start page of the application. Its **Data** property is used to set a data source for the grid via GridControl.ItemsSource.<br>*DataTableSample.cs* - The **App** class specifies **MainPage** as the start page of the application. The **MainPage.Data** property is bound to the **App.Data** property.<br>*SampleData.cs* - A DataTable property of the SampleData class returns a DataTable instance populated with data. This file is located in the PCL project folder but it is not included in the PCL project. A link to this file is added to the iOS and Android projects. | DevExpress.Mobile.Grid.v18.1.dll<br>DevExpress.Mobile.Core.v18.1.dll |
| iOS | *AppDelegate.cs* - Initializes platform libraries of the GridControl. Creates an App class instance and sets its Data property to a DataTable instance. Passes the created App class instance to the LoadApplication method which starts your application.<br>A link to the *SampleData.cs* file - Specifies DataTable to be a data source for the grid. | DevExpress.Mobile.Core.iOS.v18.1.dll<br>DevExpress.Mobile.Grid.iOS.v18.1.dll<br>System.Data.dll |
| Android | *MainActivity.cs* - Initializes platform libraries of the GridControl. Creates an App class instance and sets its Data property to a DataTable instance. Passes the created App class instance to the LoadApplication method which starts your application.<br>A link to the *SampleData.cs* file - Specifies DataTable to be a data source for the grid. | DevExpress.Mobile.Core.Android.v18.1.dll<br>DevExpress.Mobile.Grid.Android.v18.1.dll<br>System.Data.dll |

**C#**

© 2018 Developer Express Inc.

```
 (AppDelegate.cs)
using System;
using System.Collections.Generic;
using System.Linq;
using Foundation;
using UIKit;
namespace DataTableSample.iOS
{
    [Register ("AppDelegate")]
    public partial class AppDelegate : global::Xamarin.Forms.Platform.iOS.FormsApplicationDelegate
    {
        public override bool FinishedLaunching (UIApplication app, NSDictionary options)
        {
            DevExpress.Mobile.Forms.Init();
            global::Xamarin.Forms.Forms.Init();
            App application = new App();
            application.Data = new SampleData().DataTable;
            LoadApplication(application);
            return base.FinishedLaunching (app, options);
        }
    }
}
```

**C#**

```
 (DataTableSample.cs)
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using Xamarin.Forms;
namespace DataTableSample
{
    public class App : Application {
        public static readonly BindableProperty DataProperty = BindablePro
        public App () {
            // The root page of your application
            MainPage = new MainPage();
            Binding binding = new Binding();
            binding.Source = this;
            binding.Path = "Data";
            binding.Mode = BindingMode.Default;
            MainPage.SetBinding(DataTableSample.MainPage.DataProperty, bin
        }
        public object Data { get { return GetValue(DataProperty); } set {
    }
}
```

**C#**

```
 (MainActivity.cs)
using System;
using Android.App;
using Android.Content;
using Android.Content.PM;
using Android.Runtime;
using Android.Views;
using Android.Widget;
using Android.OS;
namespace DataTableSample.Droid
{
    [Activity (Label = "DataTableSample.Droid", Icon = "@drawable/icon", M
    public class MainActivity : global::Xamarin.Forms.Platform.Android.For
    {
        protected override void OnCreate (Bundle bundle)
        {
            base.OnCreate (bundle);
            DevExpress.Mobile.Forms.Init();
            global::Xamarin.Forms.Forms.Init(this, bundle);
            App application = new App();
            application.Data = new SampleData().DataTable;
            LoadApplication(application);
        }
    }
}
```

**C#**

```
 (MainPage.xaml.cs)
using System;
using System.Collections.Generic;
using Xamarin.Forms;
namespace DataTableSample
{
    public partial class MainPage : ContentPage
    {
        public static readonly BindableProperty DataProperty = BindablePro
        public MainPage() {
            InitializeComponent();
            BindingContext = this;
        }
        public object Data { get { return GetValue(DataProperty); } set {
    }
}
```

**C#**

© 2018 Developer Express Inc.

```
 (SampleData.cs)
using System;
using System.Data;
namespace DataTableSample {
    public class SampleData {
        readonly DataTable dataTable;
        DataColumn colName;
        DataColumn colAge;
        DataColumn colIsMarried;
        public SampleData() {
            dataTable = new DataTable();
            PopulateData();
        }
        public DataTable DataTable { get { return dataTable; } }
        void PopulateData() {
            this.colName = new DataColumn("Name", typeof(string));
            this.colAge = new DataColumn("Age", typeof(int));
            this.colIsMarried = new DataColumn("IsMarried", typeof(bool));
            dataTable.Columns.Add(colName);
            dataTable.Columns.Add(colAge);
            dataTable.Columns.Add(colIsMarried);
            AddRow("Andrew Fuller", 53, true);
            AddRow("Nancy Davolio", 36, true);
            AddRow("Janet Leverling", 30, false);
            AddRow("Margaret Peacock", 41, true);
            AddRow("Steven Buchanan", 48, true);
            AddRow("Michael Suyama", 33, false);
            AddRow("Robert King", 55, true);
            AddRow("Laura Callahan", 30, true);
            AddRow("Anne Dodsworth", 35, false);
        }
        void AddRow(string name, int age, bool isMarried) {
            DataRow row = dataTable.NewRow();
            row[colName] = name;
            row[colAge] = age;
            row[colIsMarried] = isMarried;
            dataTable.Rows.Add(row);
        }
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<?xml version="1.0" encoding="UTF-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="DataTableSample.MainPage"
             xmlns:dxg="clr-namespace:DevExpress.Mobile.DataGrid;assembly=
  <ContentPage.Content>
    <dxg:GridControl x:Name="grid" ItemsSource="{Binding Data}" />
  </ContentPage.Content>
</ContentPage>
```

© 2018 Developer Express Inc.

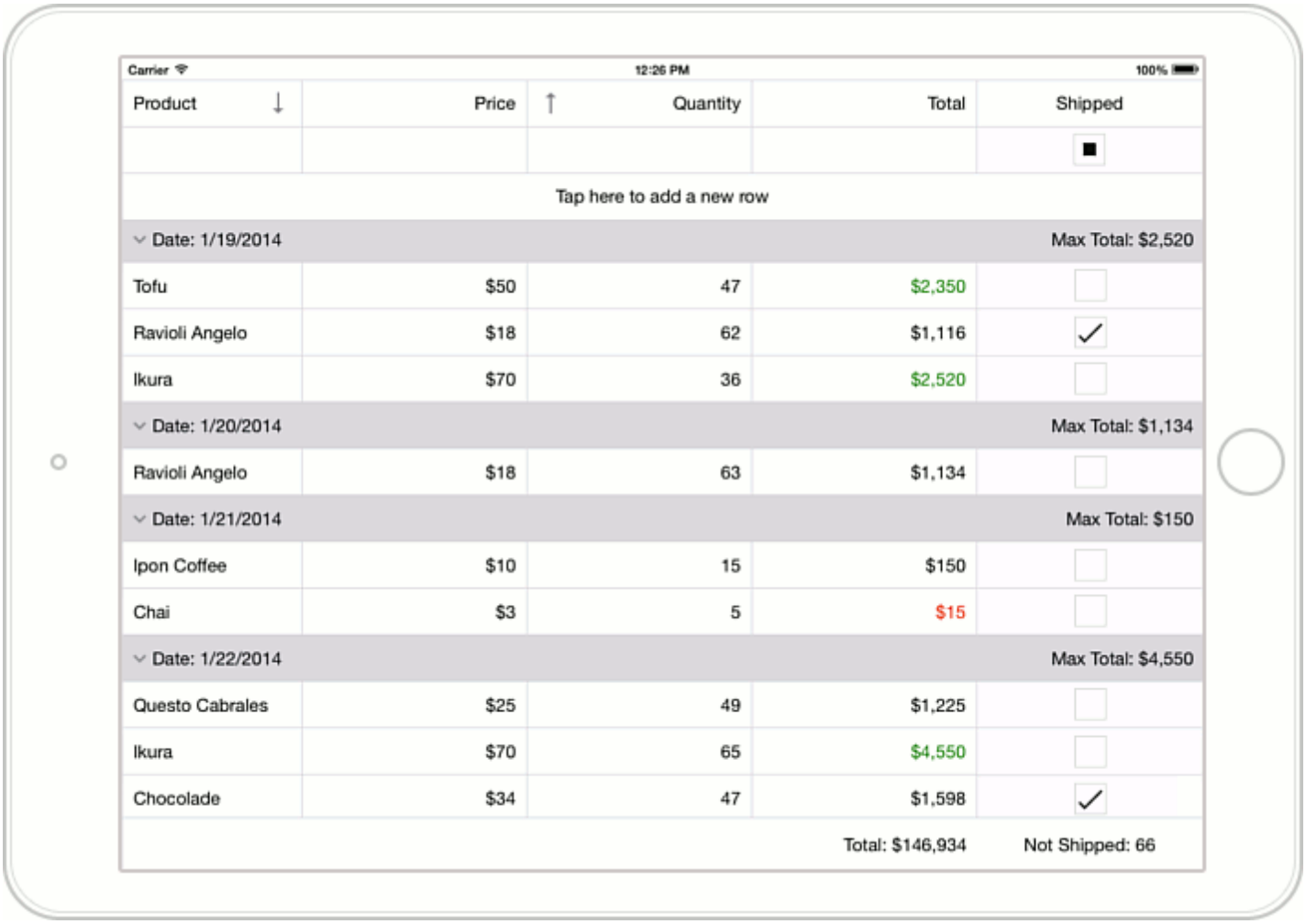
# How to: Change Cell Appearance

Xamarin Controls > Grid > Examples > How to: Change Cell Appearance

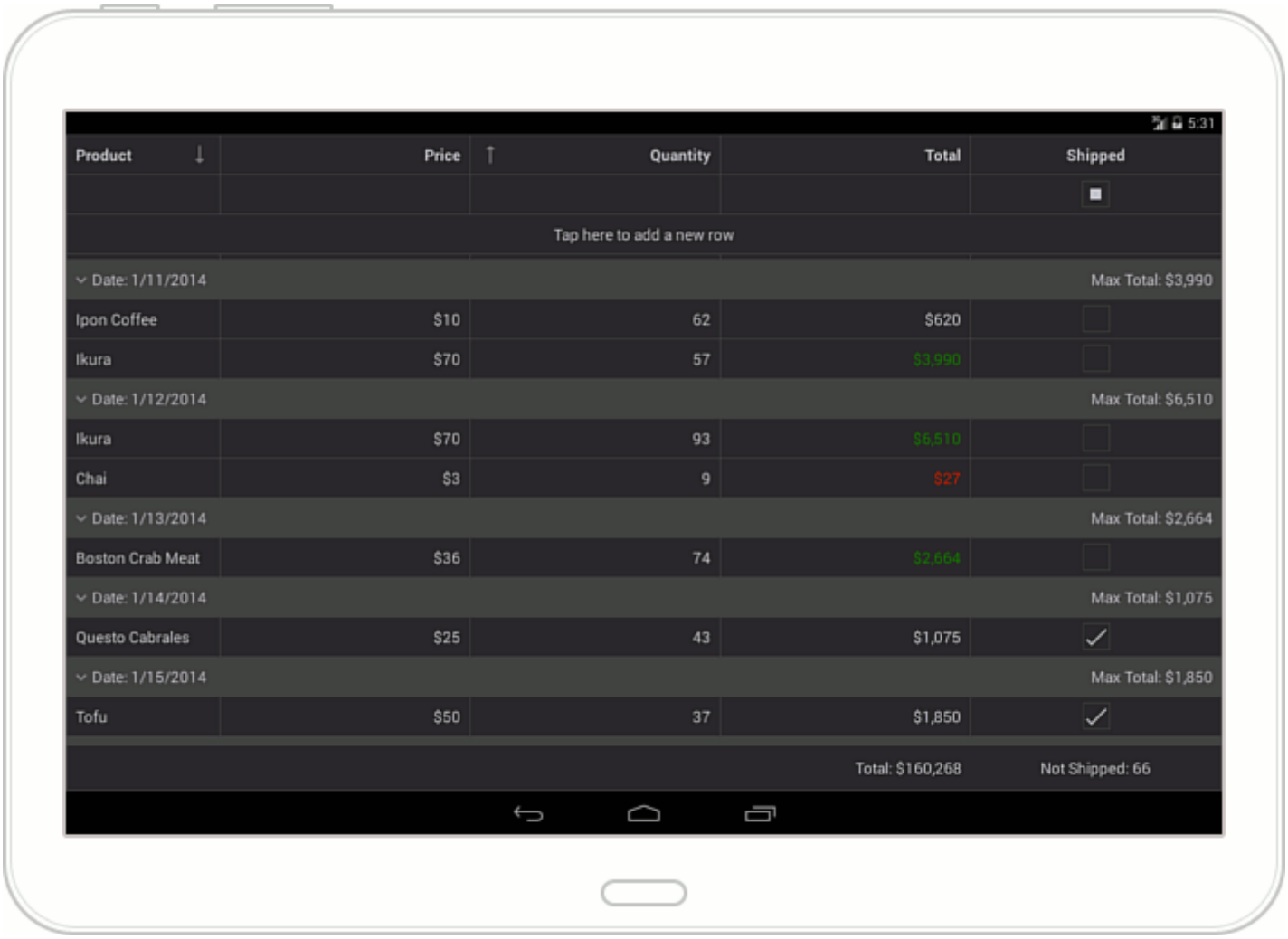
| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T178764. | |

This example demonstrates how to customize the appearance of cells in the specified grid column depending on cell values by handling the GridControl.CustomizeCell event.

The image below illustrates the result.




© 2018 Developer Express Inc.




**C#**

```
 (MainPage.xaml.cs)
void OnCustomizeCell(CustomizeCellEventArgs e)
{
    if (e.FieldName == "Total" && !e.IsSelected)
    {
        int total = Convert.ToInt32(e.Value);
        if (total < 50)
            e.ForeColor = Color.Red;
        else if (total > 2000)
            e.ForeColor = Color.Green;
        e.Handled = true;
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl x:Name="grid" ItemsSource="{Binding Orders}"
                    NewItemRowVisibility="true"
                    CalculateCustomSummary="OnCalculateCustomSummary"
                    SortMode="Multiple" AutoFilterPanelVisibility="true"
                    CustomizeCell="OnCustomizeCell"
                    SwipeButtonShowing="OnSwipeButtonShowing"
                    SwipeButtonClick="OnSwipeButtonClick">
```

© 2018 Developer Express Inc.

# How to: Implement Swipe Buttons for Data Rows



**Show Me**

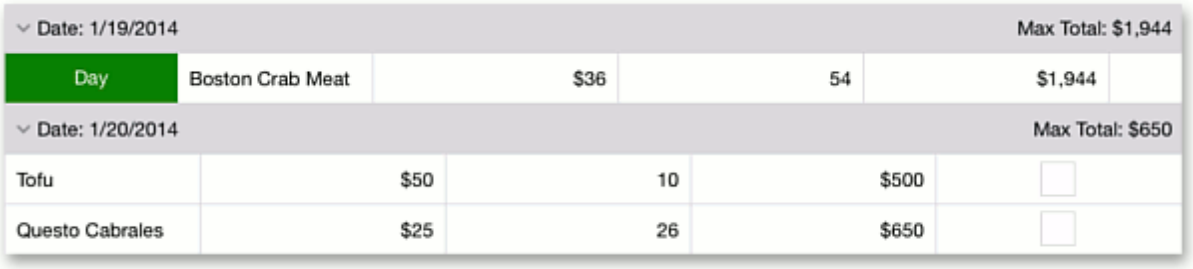
A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T178764 .

This example shows how to extend the grid's UI with additional buttons to be shown when an end-user swipes left to right, or right to left, over a data row. Swipe Buttons. To create a swipe button, add a SwipeButtonInfo object to the GridControl.LeftSwipeButtons or GridControl.RightSwipeButtons collection. Use this object's SwipeButtonInfo.Caption, SwipeButtonInfo.Width, SwipeButtonInfo.BackgroundColor and SwipeButtonInfo.TextColor properties to specify button appearance.

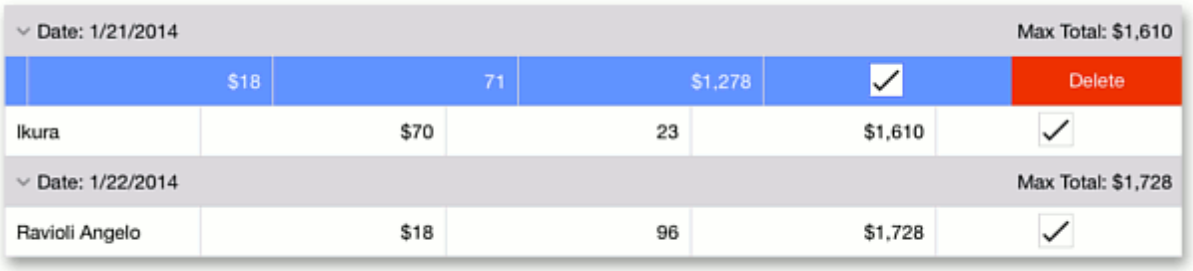
To assign a custom action to a button, handle the GridControl.SwipeButtonClick event. The GridControl.SwipeButtonShowing event allows you to specify when a swipe button should be visible.

In the example, two swipe buttons are created.

- When an end-user swipes any data row left to right, the **Day** button is shown on the left side of the row. Tapping this button displays the day of the week when the corresponding order was made.




- When an end-user swipes right to left, a data row which corresponds to an order that is already shipped (the value in the **Shipped** column is **true**), the **Delete** button is shown on the right side of the row. Tapping this button removes the corresponding data row.




**C#**

© 2018 Developer Express Inc.

```
 (MainPage.xaml.cs)
void OnSwipeButtonShowing(object sender, SwipeButtonShowingEventArgs e)
{
    if ((!(Boolean)grid.GetCellValue(e.RowHandle, "Shipped"))
        && (e.ButtonInfo.ButtonName == "RightButton"))
    {
        e.IsVisible = false;
    }
}
void OnSwipeButtonClick(object sender, SwipeButtonEventArgs e)
{
    if (e.ButtonInfo.ButtonName == "LeftButton")
    {
        DateTime orderDate = (DateTime)grid.GetCellValue(e.RowHandle, "Date");
        string orderDateDay = orderDate.ToString("dddd");
        DisplayAlert("Alert from " + e.ButtonInfo.ButtonName, "Day: " + orderDateDay, "OK");
    }
    if (e.ButtonInfo.ButtonName == "RightButton")
    {
        grid.DeleteRow(e.RowHandle);
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl.LeftSwipeButtons>
  <dxGrid:SwipeButtonInfo ButtonName="LeftButton" Caption="Day" Background
</dxGrid:GridControl.LeftSwipeButtons>
<dxGrid:GridControl.RightSwipeButtons>
  <dxGrid:SwipeButtonInfo ButtonName="RightButton" Caption="Delete" Backgr
</dxGrid:GridControl.RightSwipeButtons>
```

© 2018 Developer Express Inc.

# How to: Implement Pull-to-Refresh

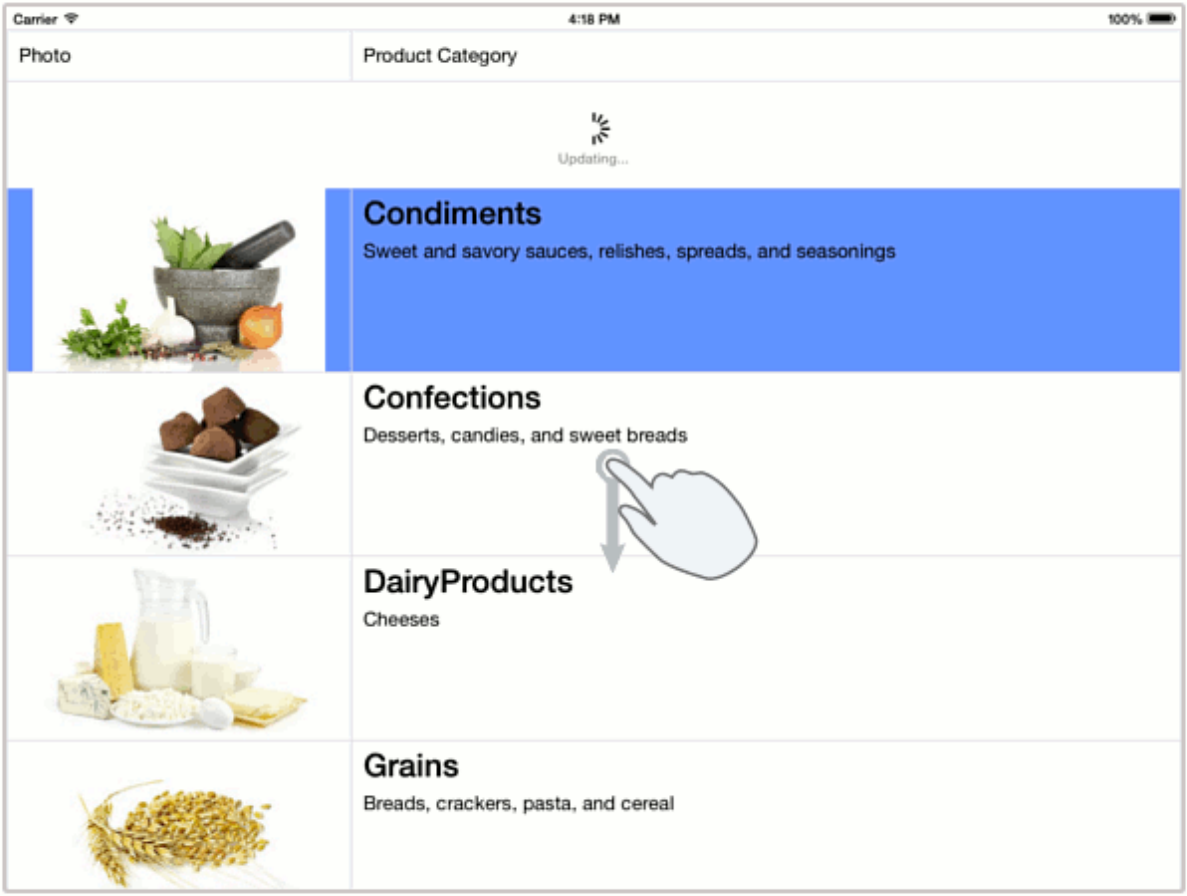


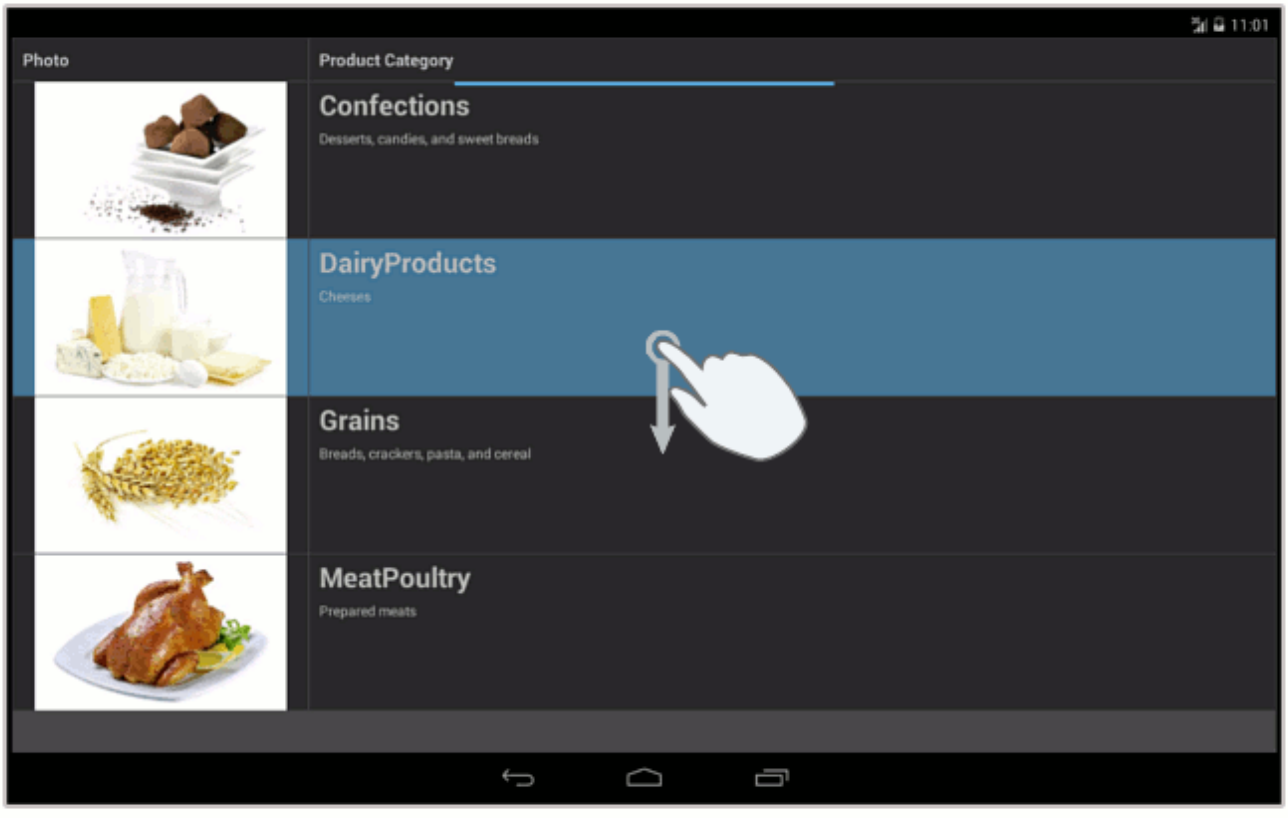
| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T221404 . | |

This example shows how to add the pull-to-refresh functionality to a grid to allow your end-users update grid content with the pull-down gesture. To do this, follow the steps below.

1. Set the GridControl.IsPullToRefreshEnabled property to **true** to enable the grid's pull-to-refresh feature.
2. Create a command to be executed when an end-user pulls the grid down.
3. Bind the created command to the grid using the GridControl.PullToRefreshCommand property.

© 2018 Developer Express Inc.


Carrier
4:18 PM
100%
Photo
Product Category
Updating...
Condiments
Sweet and savory sauces, relishes, spreads, and seasonings
Confections
Desserts, candies, and sweet breads
DairyProducts
Cheeses
Grains
Breads, crackers, pasta, and cereal


11:01
Photo
Product Category
Confections
Desserts, candies, and sweet breads
DairyProducts
Cheeses
Grains
Breads, crackers, pasta, and cereal
MeatPoultry
Prepared meats

©

**C#**

```
 (MainPage.xaml.cs)
public partial class MainPage : ContentPage
{
    ProductsRepository repository;
    public static readonly BindableProperty PullToRefreshProperty = BindableProperty.Create<MainPage, ICor
    public static readonly BindableProperty ProductsProperty = BindableProperty.Create<MainPage, Observabl
    public Command PullToRefreshCommand {
        get { return (Command)GetValue(PullToRefreshProperty); }
        set { SetValue(PullToRefreshProperty, value); }
    }
    public ObservableCollection<Product> Products {
        get { return (ObservableCollection<Product>)GetValue(ProductsProperty); }
        set { SetValue(ProductsProperty, value); }
    }
    public MainPage () {
        InitializeComponent ();
        BindingContext = this;
        this.repository = new TestProductsRepository();
        this.Products = repository.Products;
        PullToRefreshCommand = new Command(ExecutePullToRefreshCommand);
    }
    void ExecutePullToRefreshCommand() {
        repository.RefreshProducts();
        Products = repository.Products;
    }
}
```

**C#**

```
 (Product.cs)
public class Product : ModelObject {
    string name;
    public string Name {
        get { return name; }
        set {
            name = value;
            if (Photo == null)
                Photo = ImageSource.FromResource (value);
        }
    }
    public Product(string name) {
        this.Name = name;
    }
    public ImageSource Photo { get; set; }
    public string Description { get; set; }
}
```

**C#**

© 2018 Developer Express Inc.

```
 (ProductsRepository.cs)
public abstract class ProductsRepository {
    public ObservableCollection<Product> Products { get; set; }
    public ProductsRepository() {
        this.Products = new ObservableCollection<Product>();
    }
    protected abstract ObservableCollection<Product> GenerateAvailableProd
    internal void RefreshProducts() {
        int index = new Random().Next (0, 5);
        Products = GenerateAvailableProducts (index);
    }
}
public class TestProductsRepository : ProductsRepository {
    readonly List<Product> products;
    public TestProductsRepository() : base() {
        this.products = new List<Product>();
        GenerateAllProducts();
        int index = new Random().Next (0, 5);
        Products = GenerateAvailableProducts (index);
    }
    protected override ObservableCollection<Product> GenerateAvailableProd
        ObservableCollection<Product> availableProducts = new ObservableCo
        for (int i = 0; i < 4; i++)
            availableProducts.Add(products[number+i]);
        return availableProducts;
    }
    void GenerateAllProducts() {
        products.Add(new Product("Beverages") {Description = "Soft drinks,
        products.Add(new Product("Condiments"){Description = "Sweet and sa
        products.Add(new Product("Confections"){Description = "Desserts, c
        products.Add(new Product("DairyProducts"){Description = "Cheeses"}
        products.Add(new Product("Grains"){Description = "Breads, crackers
        products.Add(new Product("MeatPoultry"){Description = "Prepared me
        products.Add(new Product("Produce"){Description = "Dried fruit and
        products.Add(new Product("Seafood"){Description = "Seaweed and fis
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl ItemsSource="{Binding Products}" RowHeight="160"
                    IsPullToRefreshEnabled="true" PullToRefreshCommand="{B
```

Another way to configure the pull-to-refresh functionality is to handle the GridControl.PullToRefresh event.

© 2018 Developer Express Inc.

# How to: Implement Load-More



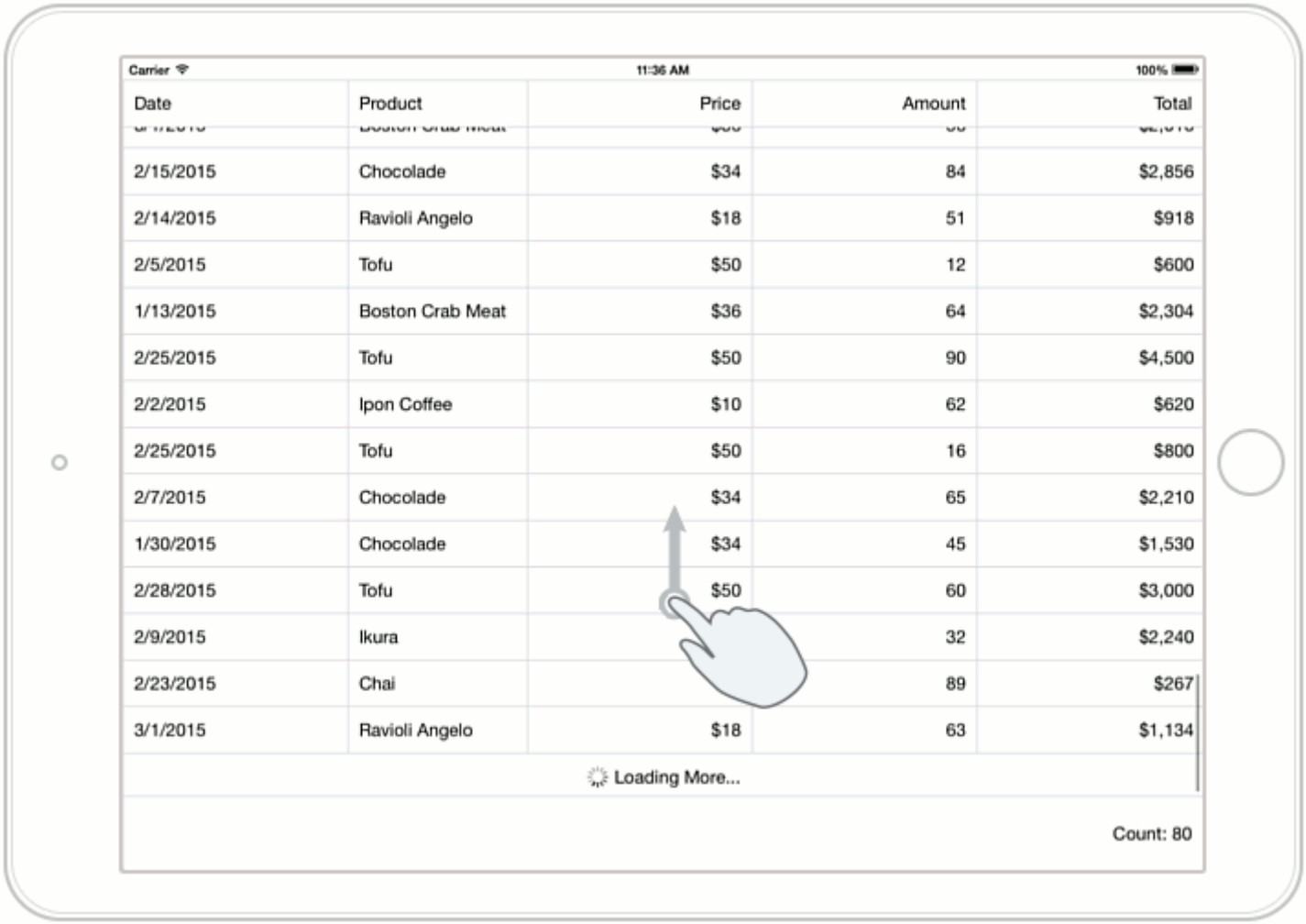
| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T223734. | |

This example shows how to implement the grid's load-more feature - when an end-user scrolls to the bottom of the grid, a set of new data items is added to the end of the grid. To add this functionality, follow the steps below.

1. Set the GridControl.IsLoadMoreEnabled property to **true** to enable the grid's load-more feature.
2. Create a command to be executed when an end-user scrolls to the bottom of the grid.
3. Bind the created command to the grid using the GridControl.LoadMoreCommand property.

In this example, data items for each next load (ten new orders) are generated randomly in code. The maximum number of loads an end-user is allowed to perform is **10**. The count of data items currently loaded to the grid is displayed by the total summary that is automatically updated after each load.

© 2018 Developer Express Inc.


Carrier
11:36 AM
100%
Date
Product
Price
Amount
Total
2/15/2015
Chocolade
$34
84
$2,856
2/14/2015
Ravioli Angelo
$18
51
$918
2/5/2015
Tofu
$50
12
$600
1/13/2015
Boston Crab Meat
$36
64
$2,304
2/25/2015
Tofu
$50
90
$4,500
2/2/2015
Ipon Coffee
$10
62
$620
2/25/2015
Tofu
$50
16
$800
2/7/2015
Chocolade
$34
65
$2,210
1/30/2015
Chocolade
$34
45
$1,530
2/28/2015
Tofu
$50
60
$3,000
2/9/2015
Ikura
32
$2,240
2/23/2015
Chai
89
$267
3/1/2015
Ravioli Angelo
$18
63
$1,134
Loading More...
Count: 80

4:42
Date
Product
Price
Amount
Total
2/7/2015
Chocolade
$34
62
$2,108
2/8/2015
Questo Cabrales
$25
87
$2,175
1/2/2015
Chai
$3
49
$147
1/9/2015
Tofu
$50
11
$550
2/2/2015
Questo Cabrales
$25
33
$825
1/17/2015
Ikura
$70
66
$4,620
2/1/2015
Tofu
$50
19
$950
2/26/2015
Ravioli Angelo
$18
67
$1,206
1/13/2015
Chocolade
$34
35
$1,190
2/17/2015
Tofu
$50
56
$2,800
2/27/2015
Ravioli Angelo
1
$18
2/24/2015
Tofu
19
$950
2/16/2015
Questo Cabrales
$25
55
$1,375
Loading More...
Count: 50

**C#**

```
 (MainPage.xaml.cs)
public partial class MainPage : ContentPage
{
    OrdersRepository repository;
    public static readonly BindableProperty LoadMoreCommandProperty = BindableProperty.Create<MainPage, Lo
    public static readonly BindableProperty OrdersProperty = BindableProperty.Create<MainPage, ObservableC
    public LoadMoreDataCommand LoadMoreCommand {
        get { return (LoadMoreDataCommand)GetValue(LoadMoreCommandProperty); }
        set { SetValue(LoadMoreCommandProperty, value); }
    }
    public ObservableCollection<Order> Orders {
        get { return (ObservableCollection<Order>)GetValue(OrdersProperty); }
        set { SetValue(OrdersProperty, value); }
    }
    public MainPage () {
        InitializeComponent ();
        BindingContext = this;
        this.repository = new TestOrdersRepository();
        this.Orders = repository.Orders;
        LoadMoreCommand = new LoadMoreDataCommand(ExecuteLoadMoreCommand);
    }
    void ExecuteLoadMoreCommand() {
        repository.LoadMoreOrders();
        Orders = repository.Orders;
    }
}
public class LoadMoreDataCommand : ICommand {
    readonly Action execute;
    int numOfLoadMore;
    public event EventHandler CanExecuteChanged;
    public LoadMoreDataCommand(Action execute) {
        this.execute = execute;
    }
    public bool CanExecute(object parameter) {
        if (numOfLoadMore < 10) {
            return true;
        }
        return false;
    }
    public void Execute(object parameter) {
        numOfLoadMore++;
        this.execute();
    }
}
```

**C#**

```
 (OrdersRepository.cs)
public abstract class OrdersRepository {
    public ObservableCollection<Order> Orders { get; private set; }
    public OrdersRepository() {
        this.Orders = new ObservableCollection<Order>();
    }
    protected abstract Order GenerateOrder(int number);
    internal void LoadMoreOrders() {
        ObservableCollection<Order> newOrders = new ObservableCollection<O
        foreach (var order in Orders)
            newOrders.Add(order);
        for (int i = 0; i < 10; i++)
            newOrders.Add(GenerateOrder(i));
        Orders = newOrders;
    }
}
public class TestOrdersRepository : OrdersRepository {
    const int orderCount = 50;
    readonly List<Product> products;
    readonly Random random;
    public TestOrdersRepository() : base() {
        this.random = new Random((int)DateTime.Now.Ticks);
        this.products = new List<Product>();
        GenerateProducts();
        for (int i = 0; i < orderCount; i++)
            Orders.Add(GenerateOrder(i));
    }
    protected override Order GenerateOrder(int number) {
        Order order = new Order(new DateTime(2015, 1, 1).AddDays(random.Ne
            number % 3 == 0, RandomItem<Product>(products), random.Next(1,
        return order;
    }
    T RandomItem<T>(IList<T> list) {
        int index = (int)(random.NextDouble() * 0.99 * (list.Count));
        return list[index];
    }
    void GenerateProducts() {
        products.Add(new Product("Tofu", 50));
        products.Add(new Product("Chocolade", 34));
        products.Add(new Product("Ikura", 70));
        products.Add(new Product("Chai", 3));
        products.Add(new Product("Boston Crab Meat", 36));
        products.Add(new Product("Ravioli Angelo", 18));
        products.Add(new Product("Ipon Coffee", 10));
        products.Add(new Product("Questo Cabrales", 25));
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl ItemsSource="{Binding Orders}"
                    LoadMoreCommand="{Binding LoadMoreCommand}" IsLoadMore
```

© 2018 Developer Express Inc.

Another way to configure the load-more functionality is to handle the GridControl.LoadMore event.

© 2018 Developer Express Inc.

# How to: Customize Popup Menu

Xamarin Controls > Grid > Examples > How to: Customize Popup Menu

The following sample code handles the GridControl.PopupMenuCustomization event to customize the data row popup menu before it is displayed. The code clears default menu items and adds new ones. One of these custom items applies a specific filter to the grid data and another one clears this filter.

| Xaml | |
|---|---|

```
<?xml version="1.0" encoding="UTF-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="HelloGrid.MainPage"
             xmlns:dxg="clr-namespace:DevExpress.Mobile.DataGrid;assembly=DevExpress.Mobile.Grid.v15.1">
    <ContentPage.Padding>
        <OnPlatform x:TypeArguments="Thickness" iOS="0, 20, 0, 0" />
    </ContentPage.Padding>
    <ContentPage.Content>
        <dxg:GridControl x:Name="grid" ItemsSource="{Binding Orders}"
                         PopupMenuCustomization="OnPopupMenuCustomization">
           <dxg:GridControl.Columns>
               <dxg:TextColumn FieldName="Product.Name" Caption="Product"/>
               <dxg:NumberColumn FieldName="Product.UnitPrice" Caption="Price" DisplayFormat="C0"/>
               <dxg:NumberColumn FieldName="Quantity" />
               <dxg:NumberColumn FieldName="Total"
                                 UnboundType="Decimal" UnboundExpression="[Quantity] * [Product.UnitPrice]
                                 IsReadOnly="True" DisplayFormat="C0"/>
           </dxg:GridControl.Columns>
    </dxg:GridControl>
    </ContentPage.Content>
</ContentPage>
```

**C#**

```
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.Core;
// ...
void OnPopupMenuCustomization(object sender, PopupMenuEventArgs e) {
    if (e.MenuType == GridPopupMenuType.DataRow) {
        e.Menu.Items.Clear();
        PopupMenuItem itemFilter = new PopupMenuItem();
        itemFilter.Caption = "Filter by Product";
        itemFilter.Click += ItemFilterClick;
        e.Menu.Items.Insert(0, itemFilter);
        PopupMenuItem itemClearFilter = new PopupMenuItem();
        itemClearFilter.Caption = "Clear Filter";
        itemClearFilter.Click += ItemClearFilterClick;
        e.Menu.Items.Insert(1, itemClearFilter);
    }
}
void ItemFilterClick(object sender, EventArgs e) {
    IRowData rowData = grid.GetRow(grid.SelectedRowHandle);
    Order selectedOrder = rowData.DataObject as Order;
    string productName = selectedOrder.Product.Name;
    grid.Columns ["Product.Name"].AutoFilterValue = productName;
}
void ItemClearFilterClick(object sender, EventArgs e) {
    grid.ClearFilter ();
}
```

© 2018 Developer Express Inc.

# How to: Implement a Custom Row Edit Form



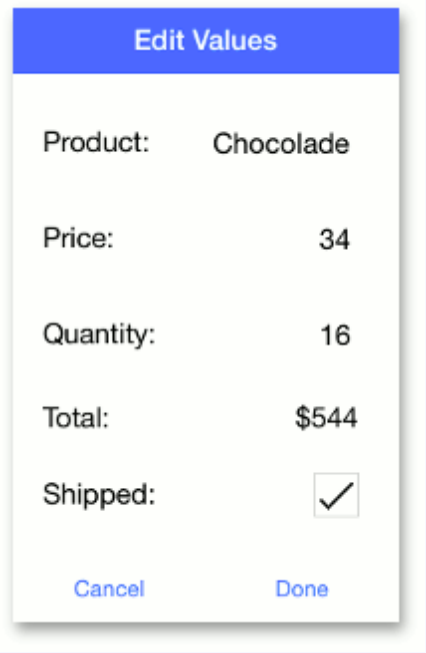
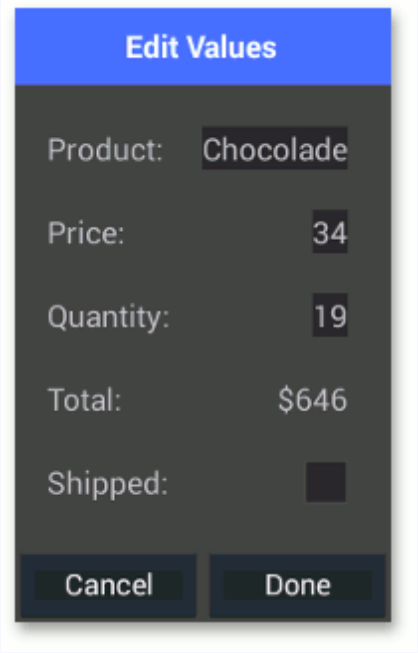
| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T279098. | |

GridControl allows you to substitute the default row edit form with a custom form. This example explains how to do this.

In this application, the grid is bound to a collection of orders. Each data row in the grid represents an individual order and displays the following information:

- Ordered product name (the **Product** column)
- Product unit price (the **Price** column)
- Number of ordered product units (the **Quantity** column)
- Order total value (the **Total** column)
- A value indicating whether an order is shipped (the **Shipped** column)

The default row edit form in this grid looks like the following.

| iOS | Android |
|---|---|
|  |  |





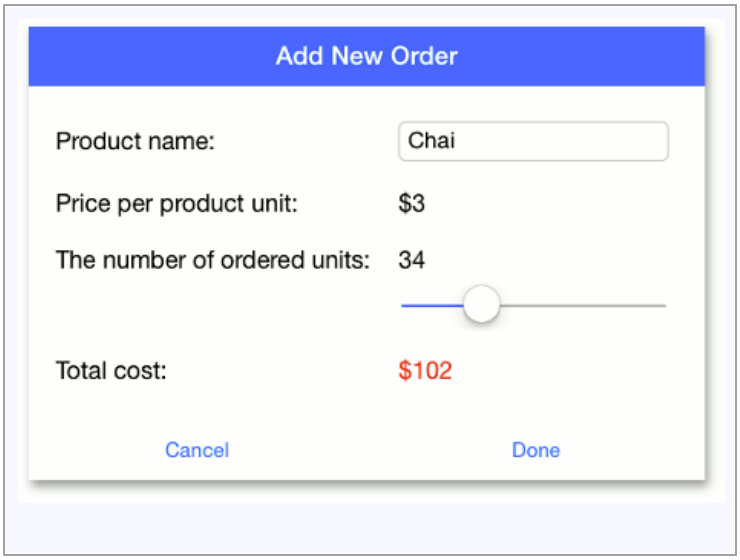
This example demonstrates how to create a custom form to add new data rows and another form to edit existing rows.

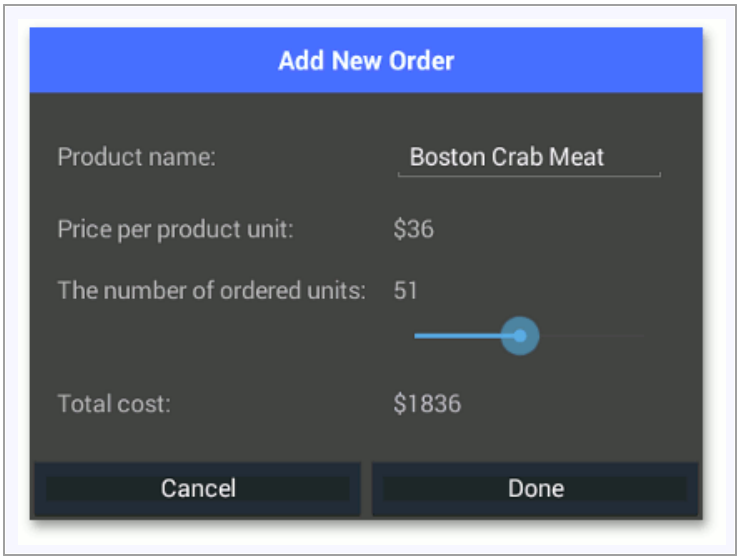
- **Add New Order Form**

This form is invoked on tapping New Item Row in the grid and allows an end user to add a new data row (create a new order). On this form, an end user can select a product name via a Picker control, and set the number of ordered product units via a Slider control. A unit price is automatically shown for the selected product, and the order total value is automatically calculated.

| iOS | Android |
|---|---|

© 2018 Developer Express Inc.

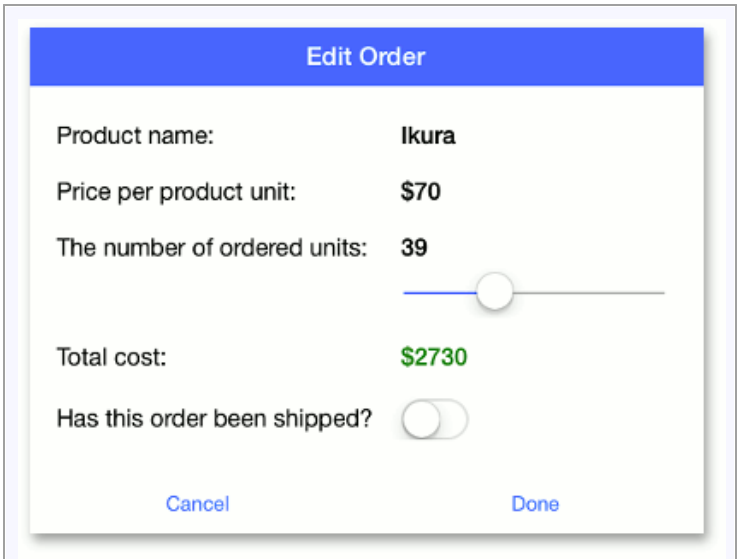


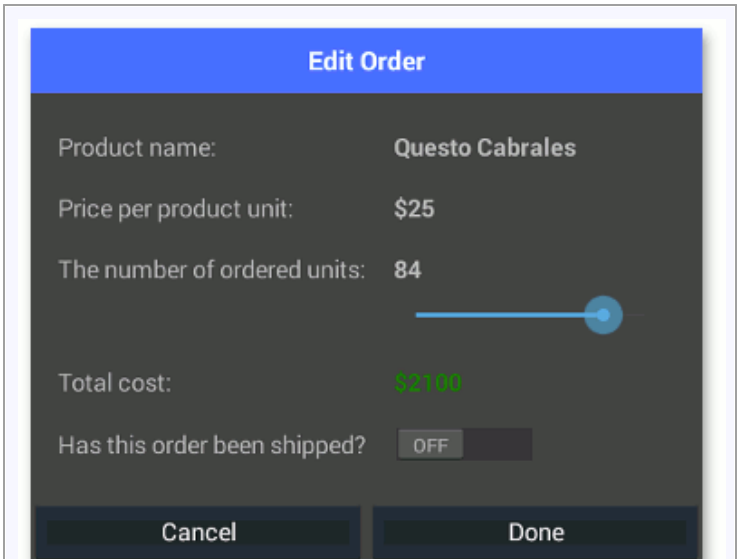




- **Edit Order Form**

This form is invoked on double-tapping a cell or selecting **Edit Cell** from the popup menu. It allows an end user to modify existing data rows. On this form, product name and unit price are shown in Lable controls and cannot be changed. An end user can change the number of ordered product units via a Slider control and use the Switch control to set whether the order is shipped.

| iOS | Android |
| --- | --- |




To create a custom row edit form and show it instead of the default form, execute the steps below.

1. Specify a custom form view. In this example, it is a ContentView descendant with a set of labels and editors inside (see the *CustomFormEditorContent.cs* file). This view's BindingContext is an EditValuesContainer object. Labels and editors are bound to data fields via the EditValuesContainer.GetBindingPath method with data field names passed as parameters.

The EditValuesContainer.IsNewRow property identifies whether a from is invoked for a new or existing row.

Custom form captions are implemented by using the mechanism of localization.

2. Create a DataTemplate instance for your custom form view and assign this template to the GridControl.EditFormContent property (see the *MainPage.xaml.cs* file).
3. Set the GridControl.RowEditMode property to RowEditMode.Popup to show a form instead of an in-place row editor.

| C# | |
| --- | --- |

```
 (Converters.cs)
using System;
using Xamarin.Forms;
using System.Globalization;
namespace CustomEditFormDemo {
    class Int32ToStringConverter : IValueConverter {
        object IValueConverter.Convert(object value, Type targetType, object parameter, CultureInfo cultu
            return value.ToString();
        }
        object IValueConverter.ConvertBack(object value, Type targetType, object parameter, CultureInfo cu
            int result = 0;
            try {
                result = Int32.Parse(value as String);
            } catch {
                result = 0;
            }
            return result;
        }
    }
    class Int32ToPriceStringConverter : IValueConverter {
        object IValueConverter.Convert(object value, Type targetType, object parameter, CultureInfo cultu
            return "$" + value.ToString();
        }
        object IValueConverter.ConvertBack(object value, Type targetType, object parameter, CultureInfo cu
            int result = 0;
            try {
                result = Int32.Parse(value as String);
            } catch {
                result = 0;
            }
            return result;
        }
    }
    class Int32ToDoubleConverter : IValueConverter {
        object IValueConverter.Convert(object value, Type targetType, object parameter, CultureInfo cultu
            return Convert.ToDouble(value);
        }
        object IValueConverter.ConvertBack(object value, Type targetType, object parameter, CultureInfo cu
            return Convert.ToInt32(value);
        }
    }
    class DoubleToInt32Converter : IValueConverter {
        object IValueConverter.Convert(object value, Type targetType, object parameter, CultureInfo cultu
            return Convert.ToInt32(value);
        }
        object IValueConverter.ConvertBack(object value, Type targetType, object parameter, CultureInfo cu
            return Convert.ToDouble(value);
        }
    }
}
```

| C# | |
|---|---|

© 2018 Developer Express Inc.

```
 (CustomFormEditorContent.cs)
using System;
using System.Collections.Generic;
using Xamarin.Forms;
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.DataGrid.Theme;
using System.Globalization;
using DevExpress.Mobile.DataGrid.Localization;
namespace CustomEditFormDemo {
    class EditFormLocalizer : GridLocalizer {
        public void SetEditItemFormCaption() {
            AddString(GridStringId.EditingForm_LabelCaption, "Edit Order")
        }
        public void SetNewItemFormCaption() {
            AddString(GridStringId.EditingForm_LabelCaption, "Add New Orde
        }
    }
    public class CustomEditFormContent : ContentView {
        public TestOrdersRepository ViewModel { get; set; }
        public CustomEditFormContent() {
            HorizontalOptions = LayoutOptions.FillAndExpand;
            VerticalOptions = LayoutOptions.FillAndExpand;
            GridLocalizer.Active = new EditFormLocalizer();
        }
        protected override void OnBindingContextChanged () {
            base.OnBindingContextChanged ();
            EditFormLocalizer localizer = GridLocalizer.Active as EditForm
            if ((BindingContext as EditValuesContainer).IsNewRow) {
                if (localizer != null) {
                    localizer.SetNewItemFormCaption();
                }
                Content = CreateNewItemFormContent();
            }
            else {
                if (localizer != null) {
                    localizer.SetEditItemFormCaption();
                }
                Content = CreateEditItemFormContent ();
            }
        }
        View CreateNewItemFormContent() {
            Grid layoutGrid = CreateGrid(4);
            FillFormGrid(layoutGrid, true);
            return layoutGrid;
        }
        View CreateEditItemFormContent() {
            Grid layoutGrid = CreateGrid(5);
            FillFormGrid(layoutGrid, false);
            return layoutGrid;
        }
        Grid CreateGrid(int rowsCount) {
```

© 2018 Developer Express Inc.

```
            Grid layoutGrid = new Grid();
            layoutGrid.ColumnDefinitions.Add(new ColumnDefinition() { Widt
            layoutGrid.ColumnDefinitions.Add(new ColumnDefinition() { Widt
            for (int i = 0; i < rowsCount; i++) {
                layoutGrid.RowDefinitions.Add(new RowDefinition() { Height
            }
            layoutGrid.RowSpacing = 20;
            layoutGrid.ColumnSpacing = 20;
            return layoutGrid;
        }
        void FillFormGrid(Grid layoutGrid, bool isNewItemForm) {
            FillLabels(layoutGrid, isNewItemForm);
            FillEditors(layoutGrid, isNewItemForm);
        }
        void FillLabels(Grid layoutGrid, bool isNewItemForm) {
            Label productName = new Label() { Text = "Product name:",
                                              FontSize = 18, VerticalOptic
            Label productUnitPrice = new Label() { Text = "Price per produ
                                                   FontSize = 18, Vertical
            Label quantity = new Label() { Text = "The number of ordered u
                                           FontSize = 18, VerticalOptions
            Label total = new Label() { Text = "Total cost:",
                                        FontSize = 18, VerticalOptions = L
            Label isShipped = null;
            if (!isNewItemForm) {
                isShipped = new Label() { Text = "Has this order been ship
                                          FontSize = 18, VerticalOptions =
            }
            layoutGrid.Children.Add(productName, 0, 0);
            layoutGrid.Children.Add(productUnitPrice, 0, 1);
            layoutGrid.Children.Add(quantity, 0, 2);
            layoutGrid.Children.Add(total, 0, 3);
            if (!isNewItemForm) {
                layoutGrid.Children.Add(isShipped, 0, 4);
            }
        }
        void FillEditors(Grid layoutGrid, bool isNewItemForm) {
            View productName;
            // Display a product name in a label in the Edit Item form
            // or change a product name by using a picker in the New Item
            if (isNewItemForm) {
                productName = new Picker() { HorizontalOptions = LayoutOpt
                                             VerticalOptions = LayoutOptic
                InitProductPickerValues(productName as Picker);
            }
            else {
                productName = new Label() { HorizontalOptions = LayoutOpti
                                            VerticalOptions = LayoutOption
                                            FontAttributes = FontAttribute
                productName.SetBinding(Label.TextProperty,
                                       new Binding(EditValuesContainer.Ge
```

© 2018 Developer Express Inc.

```
            }
            // Display a product unit price in a form.
            Int32ToPriceStringConverter priceConverter = new Int32ToPriceS
            Label priceLabel = new Label () { HorizontalOptions = LayoutOp
                                              VerticalOptions = LayoutOpti
            priceLabel.SetBinding (Label.TextProperty, EditValuesContainer
                                   BindingMode.Default, priceConverter);
            // Modify the number of ordered units via a slider.
            StackLayout quantityStack = new StackLayout () { Orientation =
                                                             VerticalOptic
            Label quantityLabel = new Label () { XAlign = TextAlignment.St
                                                 VerticalOptions = LayoutO
            Slider quantitySlider = new Slider () { Maximum = 99, Minimum
            quantitySlider.SetBinding (Slider.ValueProperty, EditValuesCon
                                       BindingMode.TwoWay, new Int32ToDoub
            quantityLabel.BindingContext = quantitySlider;
            quantityLabel.SetBinding(Label.TextProperty, "Value", BindingM
            quantityStack.Children.Add(quantityLabel);
            quantityStack.Children.Add(quantitySlider);
            // Display the order total value calculated automatically.
            Label totalLabel = new Label () { XAlign = TextAlignment.Start
                                              VerticalOptions = LayoutOpti
            totalLabel.PropertyChanged += OnTotalPropertyChanged;
            totalLabel.SetBinding(Label.TextProperty, EditValuesContainer.
                                  BindingMode.OneWay, priceConverter);
            if (!isNewItemForm) {
                priceLabel.FontAttributes = FontAttributes.Bold;
                quantityLabel.FontAttributes = FontAttributes.Bold;
                totalLabel.FontAttributes = FontAttributes.Bold;
            }
            // Add created elements to a form.
            layoutGrid.Children.Add(productName, 1, 0);
            layoutGrid.Children.Add(priceLabel, 1, 1);
            layoutGrid.Children.Add(quantityStack, 1, 2);
            layoutGrid.Children.Add(totalLabel, 1, 3);
            // Show a switch in the Edit Item form allowing end-users to s
            Switch isShipped = new Switch() { VerticalOptions = LayoutOpti
                                              HorizontalOptions = LayoutOp
            isShipped.SetBinding(Switch.IsToggledProperty, EditValuesConta
            if (!isNewItemForm) {
                layoutGrid.Children.Add(isShipped, 1, 4);
            }
        }
        void InitProductPickerValues(Picker picker) {
            EditValuesContainer valuesContainer = BindingContext as EditVa
            String productName = valuesContainer.Values["Product.Name"].To
            int selectedItemIndex = -1;
            foreach (Product product in ViewModel.Products) {
                if (productName == product.Name) {
                    selectedItemIndex = picker.Items.Count;
                }
```

© 2018 Developer Express Inc.

```
                picker.Items.Add(product.Name);
            }
            picker.SelectedIndexChanged += OnProductNameSelectedIndexChang
            if (selectedItemIndex >= 0) {
                picker.SelectedIndex = selectedItemIndex;
            }
        }
        void OnProductNameSelectedIndexChanged(object sender, EventArgs e)
            EditValuesContainer valuesContainer = BindingContext as EditVa
            Picker picker = sender as Picker;
            if (picker.SelectedIndex < 0) {
                return;
            }
            String productName = picker.Items[picker.SelectedIndex];
            valuesContainer.Values["Product.Name"] = productName;
            Product product = ViewModel.Products.Find(p => p.Name == produ
            valuesContainer.Values["Product.UnitPrice"] = product.UnitPric
        }
        void OnTotalPropertyChanged (object sender, System.ComponentModel.
        {
            Label label = (sender as Label);
            if (e.PropertyName == Label.TextProperty.PropertyName && !Stri
                Int32 value = (Int32)((label.BindingContext as EditValuesC
                if (value > 2000) {
                    label.TextColor = Color.Green; }
                else {
                    if (value < 500) {
                        label.TextColor = Color.Red; }
                    else {
                        label.TextColor = ThemeManager.Theme.CellCustomize
                }
            }
        }
    }
}
```

**C#**

© 2018 Developer Express Inc.

```
 (MainPage.xaml.cs)
using System;
using System.Collections.Generic;
using Xamarin.Forms;
using DevExpress.Mobile.DataGrid;
namespace CustomEditFormDemo
{
    public partial class MainPage : ContentPage
    {
        TestOrdersRepository model;
        public MainPage ()
        {
            InitializeComponent ();
            model = new TestOrdersRepository ();
            BindingContext = model;
            DataTemplate editFormContentTemplate = new DataTemplate(GetTem
            grid.EditFormContent = editFormContentTemplate;
            grid.RowEditMode = RowEditMode.Popup;
        }
        object GetTemplate() {
            return new CustomEditFormContent() { ViewModel = model };
        }
        void OnInitNewItemRow(object sender, InitNewRowEventArgs e) {
            e.EditableRowData.SetFieldValue("Quantity", model.MinimumQuant
            e.EditableRowData.SetFieldValue("Product.Name", model.Products
        }
    }
}
```

**Xaml**

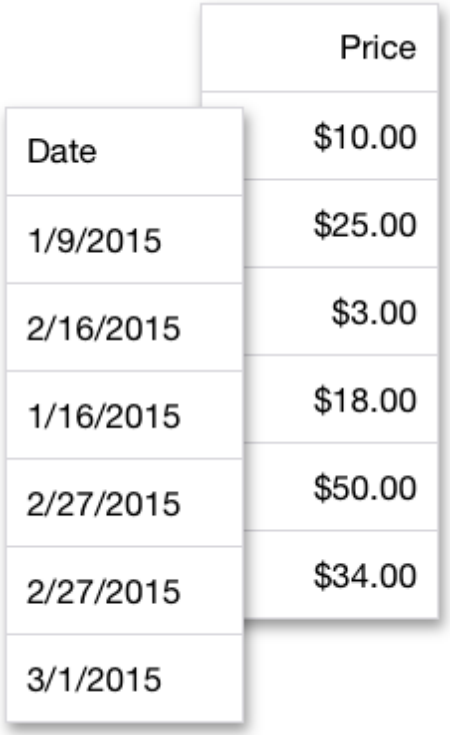
```
 (MainPage.xaml)
<?xml version="1.0" encoding="UTF-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="CustomEditFormDemo.MainPage"
             xmlns:dxGrid="clr-namespace:DevExpress.Mobile.DataGrid;assemb
      <ContentPage.Padding>
        <OnPlatform x:TypeArguments="Thickness" iOS="0, 20, 0, 0" />
    </ContentPage.Padding>
    <dxGrid:GridControl x:Name="grid" ItemsSource="{Binding Orders}"
                        NewItemRowVisibility="true" InitNewRow="OnInitNew
            <dxGrid:GridControl.Columns>
             <dxGrid:TextColumn FieldName="Product.Name" Caption="Product"
             <dxGrid:NumberColumn FieldName="Product.UnitPrice" Caption="P
             <dxGrid:NumberColumn FieldName="Quantity"/>
             <dxGrid:NumberColumn FieldName="Total"
                                  UnboundType="Integer" UnboundExpression=
                                  DisplayFormat="C0" IsReadOnly="True"/>
             <dxGrid:SwitchColumn FieldName="Shipped"/>
          </dxGrid:GridControl.Columns>
    </dxGrid:GridControl>
</ContentPage>
```

© 2018 Developer Express Inc.

# How to: Format Column Values Using Standard Format Strings



This example shows how to format column values using the standard format strings. Values in the *Price* column are formatted as a currency amount, values in the *Date* column are formatted using the Short Date Pattern.




**Xaml**

```
<dxGrid:GridControl.Columns>
    <!-- ... -->
    <dxGrid:NumberColumn FieldName="Price" DisplayFormat="c2"/>
    <dxGrid:DateColumn FieldName="Date" DisplayFormat="d"/>
    <!-- ... -->
</dxGrid:GridControl.Columns>
```

© 2018 Developer Express Inc.

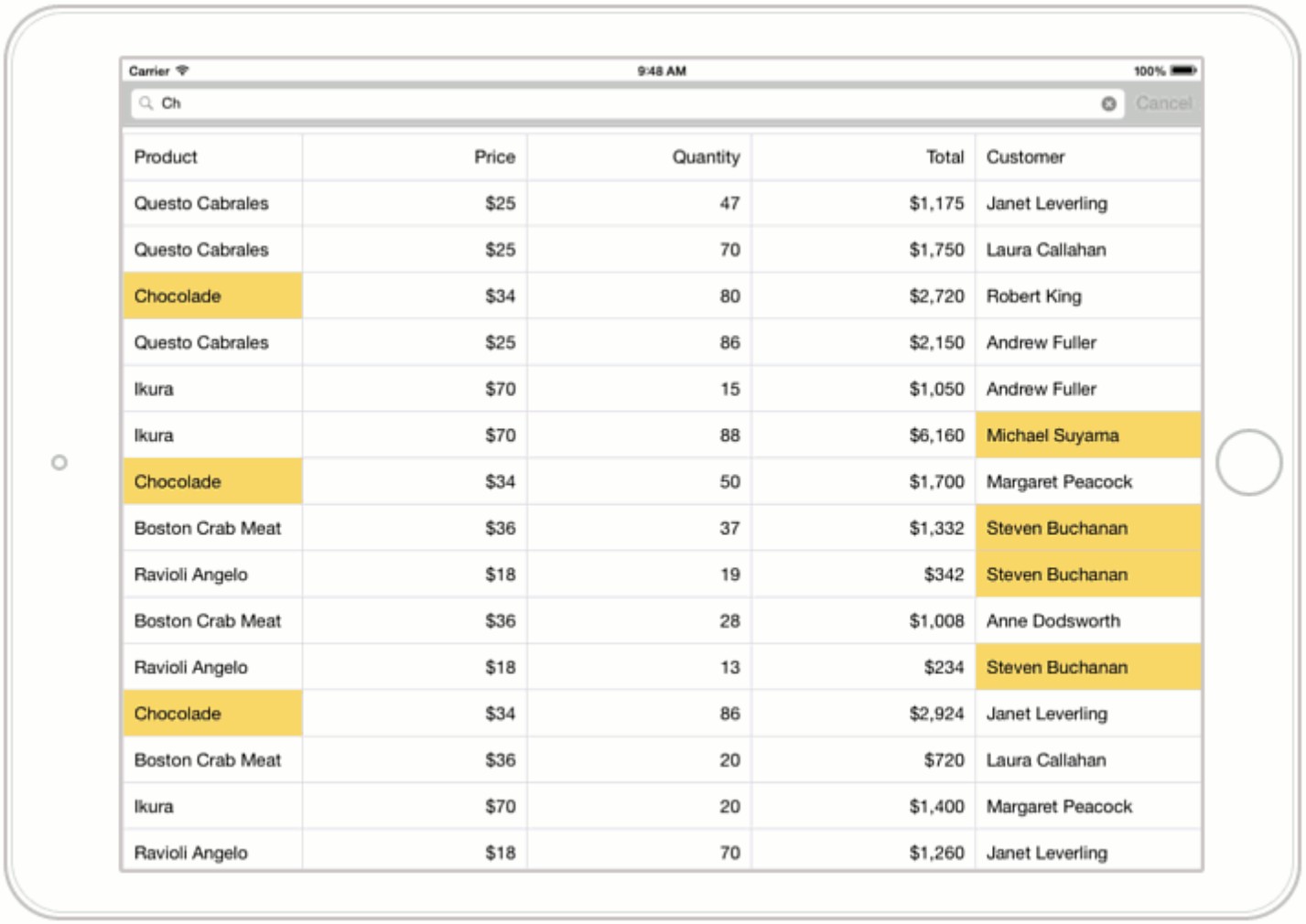
# How to: Implement the Search Functionality



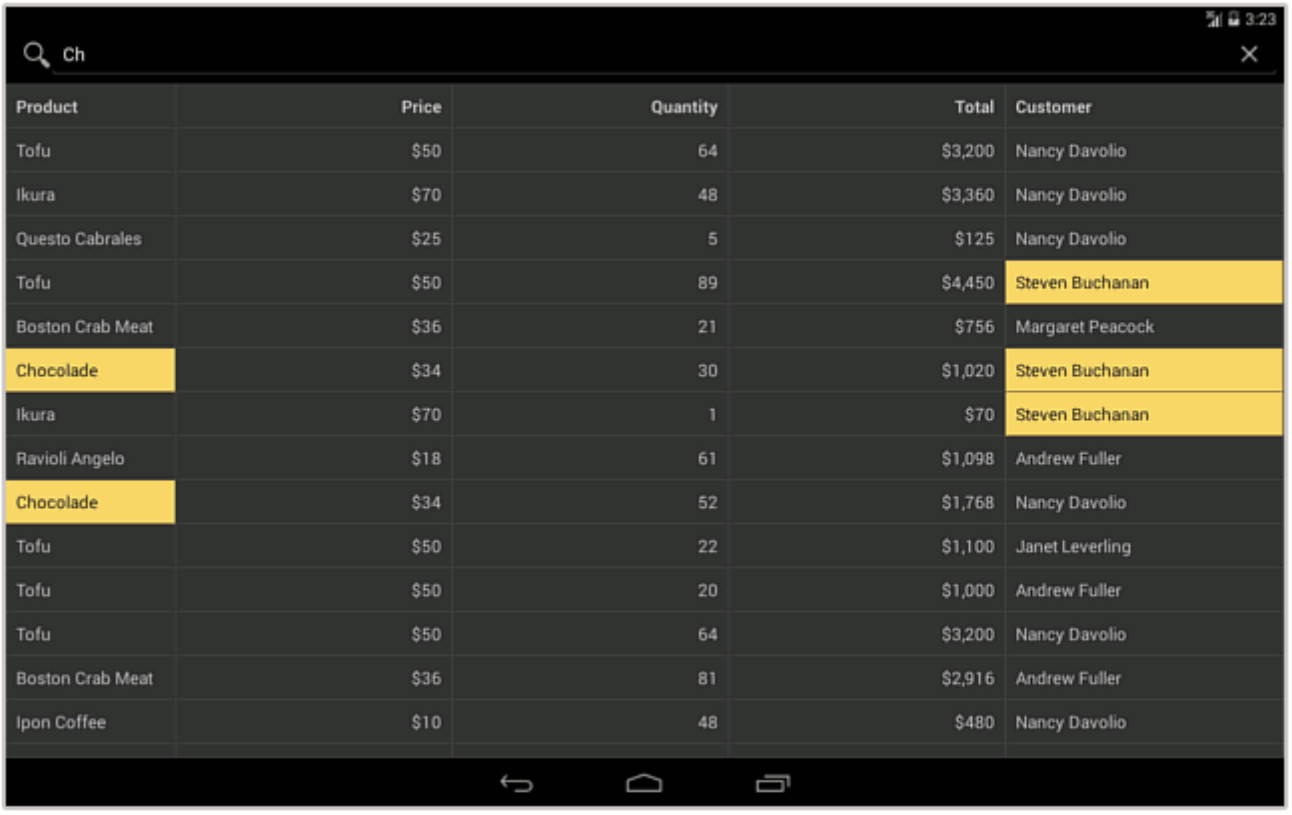
| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T294259. | |

This example demonstrates how to implement searching against grid columns - an end-user types in text in a search bar and all cells containing the entered string are highlighted on the fly.

© 2018 Developer Express Inc.




| Product | Price | Quantity | Total | Customer |
|---|---|---|---|---|
| Questo Cabrales | $25 | 47 | $1,175 | Janet Leverling |
| Questo Cabrales | $25 | 70 | $1,750 | Laura Callahan |
| Chocolade | $34 | 80 | $2,720 | Robert King |
| Questo Cabrales | $25 | 86 | $2,150 | Andrew Fuller |
| Ikura | $70 | 15 | $1,050 | Andrew Fuller |
| Ikura | $70 | 88 | $6,160 | Michael Suyama |
| Chocolade | $34 | 50 | $1,700 | Margaret Peacock |
| Boston Crab Meat | $36 | 37 | $1,332 | Steven Buchanan |
| Ravioli Angelo | $18 | 19 | $342 | Steven Buchanan |
| Boston Crab Meat | $36 | 28 | $1,008 | Anne Dodsworth |
| Ravioli Angelo | $18 | 13 | $234 | Steven Buchanan |
| Chocolade | $34 | 86 | $2,924 | Janet Leverling |
| Boston Crab Meat | $36 | 20 | $720 | Laura Callahan |
| Ikura | $70 | 20 | $1,400 | Margaret Peacock |
| Ravioli Angelo | $18 | 70 | $1,260 | Janet Leverling |




| Product | Price | Quantity | Total | Customer |
|---|---|---|---|---|
| Tofu | $50 | 64 | $3,200 | Nancy Davolio |
| Ikura | $70 | 48 | $3,360 | Nancy Davolio |
| Questo Cabrales | $25 | 5 | $125 | Nancy Davolio |
| Tofu | $50 | 89 | $4,450 | Steven Buchanan |
| Boston Crab Meat | $36 | 21 | $756 | Margaret Peacock |
| Chocolade | $34 | 30 | $1,020 | Steven Buchanan |
| Ikura | $70 | 1 | $70 | Steven Buchanan |
| Ravioli Angelo | $18 | 61 | $1,098 | Andrew Fuller |
| Chocolade | $34 | 52 | $1,768 | Nancy Davolio |
| Tofu | $50 | 22 | $1,100 | Janet Leverling |
| Tofu | $50 | 20 | $1,000 | Andrew Fuller |
| Tofu | $50 | 64 | $3,200 | Nancy Davolio |
| Boston Crab Meat | $36 | 81 | $2,916 | Andrew Fuller |
| Ipon Coffee | $10 | 48 | $480 | Nancy Davolio |

To implement this functionality, follow the steps below.

1. Place the **SearchBar** Xamarin.Forms standard view above the GridControl on the page.
2. Subscribe to the **SearchBar.TextChanged** event that is raised each time the text in the search bar is changed. In this event handler, get the currently entered search string and call the GridControl.Redraw method to recalculate the graphical information of the grid.
3. Before grid cells are repainted, the GridControl.CustomizeCell event is raised. Handle this event to apply the ICellCustomizer.HighlightColor color to cells containing the string currently entered in the search bar.

**C#**

```
 (MainPage.xaml.cs)
using System;
using System.Collections.Generic;
using Xamarin.Forms;
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.DataGrid.Theme;
namespace GridSearchTextSample {
    public partial class MainPage : ContentPage {
        string searchText;
        public MainPage () {
            InitializeComponent ();
            TestOrdersRepository model = new TestOrdersRepository ();
            BindingContext = model;
            searchText = string.Empty;
        }
        void OnCustomizeCell(CustomizeCellEventArgs e) {
            if (string.IsNullOrEmpty(searchText))
                return;
            string cellText = e.DisplayText.ToUpper();
            if (cellText.Contains(searchText.ToUpper()) && !e.IsSelected) {
                e.BackgroundColor = ThemeManager.Theme.CellCustomizer.HighlightColor;
                e.ForeColor = Color.Black;
                e.Handled = true;
            }
        }
        void OnSearchTextChanged(object sender, EventArgs args) {
            SearchBar searchBar = (SearchBar)sender;
            searchText = searchBar.Text;
            grid.Redraw(false);
        }
    }
}
```

**Xaml**

(MainPage.xaml)

```
<?xml version="1.0" encoding="UTF-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="GridSearchTextSample.MainPage"
             xmlns:dxg="clr-namespace:DevExpress.Mobile.DataGrid;assembly=
    <ContentPage.Padding>
        <OnPlatform x:TypeArguments="Thickness" iOS="0, 20, 0, 0" />
    </ContentPage.Padding>
    <ContentPage.Content>
        <Grid>
            <Grid.RowDefinitions>
                <RowDefinition Height="auto"/>
                <RowDefinition Height="*"/>
            </Grid.RowDefinitions>
            <Grid.ColumnDefinitions>
                <ColumnDefinition Width="*"/>
            </Grid.ColumnDefinitions>
            <SearchBar Grid.Row="0" Grid.Column="0" x:Name="searchBar"
                       Placeholder="Enter a value for search"
                       TextChanged="OnSearchTextChanged"/>
            <dxg:GridControl Grid.Row="1" Grid.Column="0" x:Name="grid"
                             ItemsSource="{Binding Orders}"
                             CustomizeCell="OnCustomizeCell">
                <dxg:GridControl.Columns>
                    <dxg:TextColumn FieldName="Product.Name" Caption="Pro
                    <dxg:NumberColumn FieldName="Product.UnitPrice" Capti
                    <dxg:NumberColumn FieldName="Quantity"/>
                    <dxg:NumberColumn FieldName="Total"
                                      UnboundType="Integer" UnboundExpres
                                      DisplayFormat="C0" IsReadOnly="True
                    <dxg:TextColumn FieldName="Customer.Name" Caption="Cu
                </dxg:GridControl.Columns>
            </dxg:GridControl>
        </Grid>
    </ContentPage.Content>
</ContentPage>
```

© 2018 Developer Express Inc.

# How to: Use a TemplateColumn for Showing and Editing Time Values



| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T506284. | |

This example illustrates how to create a TemplateColumn for showing and editing time values. To implement this functionality, follow the steps below.

1. Create a data object (*TestData* in this example) and populate it with the required time values to be shown in the TemplateColumn. Assign the generated time list to the GridControl.ItemsSource property.
2. Add a new TemplateColumn to the GridControl.Columns collection. In this example the following column parameters are used:
   - GridColumn.UnboundType = UnboundColumnType.Object
   - TemplateColumn.DisplayTemplate = new DataTemplate(typeof(DisplayControl))

The *DisplayControl* class represents a label whose *Text* property is bound to the CellData.Value property using the *IValueConverter* object.

   - TemplateColumn.EditTemplate = new DataTemplate(typeof(EditControl))

The *EditControl* class represents the TimePicker control whose *Time* property is two-way bound to the CellData.Value property.

3. Subscribe to the GridControl.CustomUnboundColumnData event and use its handler to set and get time data.

| C# | |
|---|---|

© 2018 Developer Express Inc.

```
 (DXSamplePage.xaml.cs)
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Globalization;
using DevExpress.Data;
using DevExpress.Mobile.Core.Containers;
using DevExpress.Mobile.DataGrid;
using Xamarin.Forms;
namespace DXSample
{
    public partial class DXSamplePage : ContentPage
    {
        List<TestData> List = new List<TestData>();
        public DXSamplePage()
        {
            InitializeComponent();
            grid.Columns.Add(new DateColumn() { FieldName = "Date" });
            grid.Columns.Add(new TemplateColumn()
            {
                FieldName = "UnboundTime",
                UnboundType = UnboundColumnType.Object,
                DisplayTemplate = new DataTemplate(typeof(DisplayControl)),
                EditTemplate = new DataTemplate(typeof(EditControl))
            });
            for (int i = 0; i < 10; i++)
            {
                TestData data = new TestData();
                data.Date = DateTime.Now;
                data.Time = new TimeSpan(i, i * 5, 0);
                List.Add(data);
            }
            grid.CustomUnboundColumnData += (sender, e) =>
            {
                if (e.IsGetData)
                {
                    e.Value = (e.RowData.DataObject as TestData).Time;
                }
                if (e.IsSetData)
                {
                    (e.RowData.DataObject as TestData).Time = (TimeSpan)e.Value;
                }
            };
            grid.ItemsSource = List;
        }
    }
    public class TestData
    {
        public DateTime Date { get; set; }
        public TimeSpan Time { get; set; }
    }
    public class TimeConverter : IValueConverter
    {
        public object Convert(object value, Type targetType, object parameter, CultureInfo culture)
        {
            TimeSpan ts = (TimeSpan)value;
            return ts.ToString();
        }
        public object ConvertBack(object value, Type targetType, object parameter, CultureInfo culture)
        {
            throw new NotImplementedException();
        }
    }
    public class DisplayControl : Label
    {
```

© 2018 Developer Express Inc.

```
        protected override void OnBindingContextChanged()
        {
            base.OnBindingContextChanged();
            CellData cellData = BindingContext as CellData;
            if (cellData != null)
                this.SetBinding(Label.TextProperty, "Value", converter: new TimeConverter());
        }
    }
    public class EditControl : TimePicker
    {
        protected override void OnBindingContextChanged()
        {
            base.OnBindingContextChanged();
            CellData cellData = BindingContext as CellData;
            if (cellData != null)
            {
                this.SetBinding(TimePicker.TimeProperty, "Value", BindingMode.TwoWay);
            }
        }
    }
}
```

**Xaml**

```
 (DXSamplePage.xaml)
<?xml version="1.0" encoding="utf-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
        xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
        xmlns:local="clr-namespace:DXSample"
        x:Class="DXSample.DXSamplePage"
        xmlns:dxg="clr-namespace:DevExpress.Mobile.DataGrid;assembly=DevEx
    <dxg:GridControl x:Name="grid">
    </dxg:GridControl>
</ContentPage>
```

© 2018 Developer Express Inc.

# How to: Check a SwitchColumn by a Single Tap



**Show Me**

A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T499167.

This example illustrates how to implement the SwitchColumn value changing using a single tap (not a usual approach with opening an edit row). To add this functionality, follow the steps below.

1. Set the GridControl.AllowEditRows property to **false** to disable the grid's editing row feature.
2. Create an object (*Data* in this example) that implements the *INotifyPropertyChanged* interface and has a Boolean property (*IsSelected* in this example) that notifies a GridControl whether or not a corresponding cell is selected.
3. Assign the *DevExpress.Mobile.Core.Containers.BindingList<T>* (*T* is the *Data* type in this example) collection to the GridControl.ItemsSource property.
4. Subscribe to the GridControl.RowTap event and use its handler to change the Boolean property's (from step 2) value.

**C#**

```
 (DXSamplePage.xaml.cs)
using System.ComponentModel;
using DevExpress.Mobile.Core.Containers;
using Xamarin.Forms;
namespace DXSample
{
    public partial class DXSamplePage : ContentPage
    {
        BindingList<Data> List = new BindingList<Data>();
        public DXSamplePage()
        {
            InitializeComponent();
            for (int i = 0; i < 10; i++)
            {
                Data data = new Data();
                data.Text = "Row " + i;
                data.IsSelected = (i % 2) == 0;
                List.Add(data);
            }
            grid.ItemsSource = List;
            grid.RowTap += (sender, e) => {
                if (e.FieldName == "IsSelected") {
                    int sourceRowIndex = grid.GetSourceRowIndex(e.RowHandle);
                    List[sourceRowIndex].IsSelected = !List[sourceRowIndex].IsSelected;
                }
            };
        }
    }
    public class Data : INotifyPropertyChanged {
        bool isSelected = false;
        public bool IsSelected {
            get { return isSelected; }
            set {
                isSelected = value;
                if (PropertyChanged != null)
                    PropertyChanged(this, new PropertyChangedEventArgs("IsSelected"));
            }
        }
        public string Text { get; set; }
        public event PropertyChangedEventHandler PropertyChanged;
    }
}
```

**Xaml**

© 2018 Developer Express Inc.

```
 (DXSamplePage.xaml)
<?xml version="1.0" encoding="utf-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
        xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
        xmlns:local="clr-namespace:DXSample"
        x:Class="DXSample.DXSamplePage"
        xmlns:dxg="clr-namespace:DevExpress.Mobile.DataGrid;assembly=DevEx
    <dxg:GridControl AllowEditRows="false" x:Name="grid">
        <dxg:GridControl.Columns>
            <dxg:SwitchColumn FieldName="IsSelected" />
            <dxg:TextColumn FieldName="Text"/>
        </dxg:GridControl.Columns>
    </dxg:GridControl>
</ContentPage>
```

© 2018 Developer Express Inc.

# Tips and Tricks



This section contains information that addresses the development of applications with the DevExpress Grid for Xamarin.Forms.

- Build Android Applications for Release
- Density-Independent Pixels

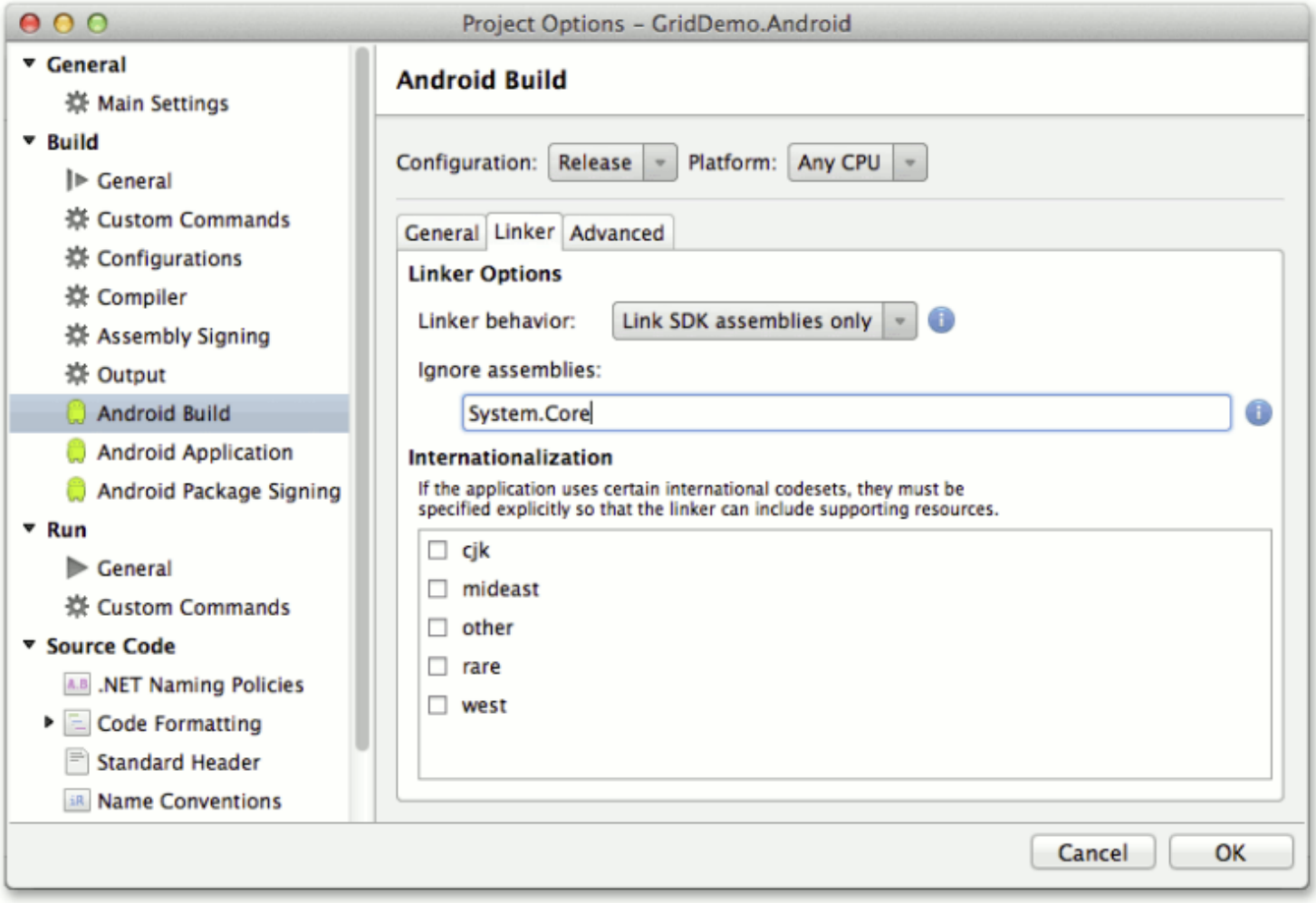
© 2018 Developer Express Inc.

# Build Android Applications for Release



To build an Android application with a GridControl instance for release, exclude the **System.Core.dll** assembly from the linking process. To achieve this, follow the steps below.

1. Right-click your Android project and select **Options**.
2. In the invoked **Project Options** dialog, select **Android Build**.
3. Switch to the **Linker** tab.
4. Select the **Release** configuration and enter the assembly name (without the file extension) into the **Ignore assemblies** box.



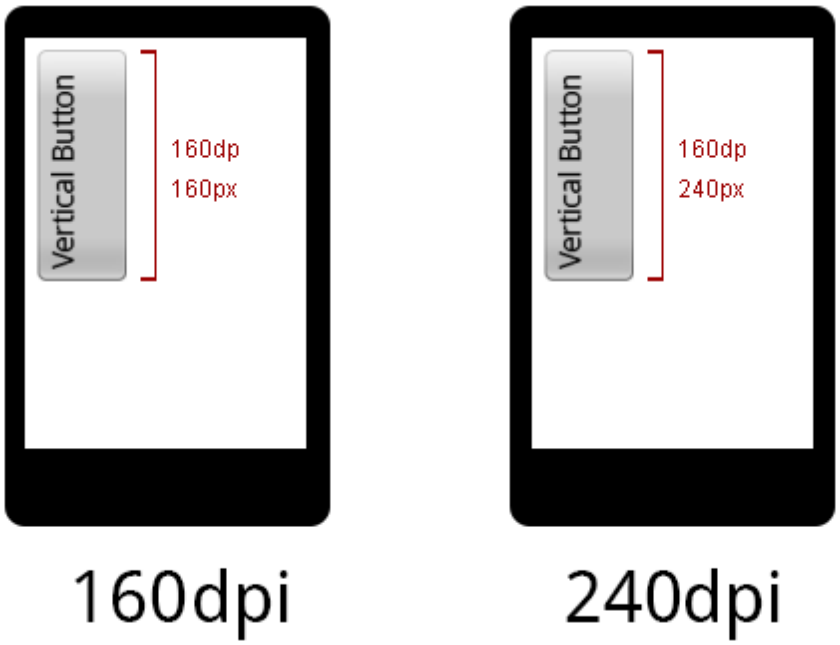

© 2018 Developer Express Inc.

# Density-Independent Pixels



The DevExpress GridControl specifies all dimensions in density-independent pixels (**dp**). Examples of properties accepting these units include: GridControl.RowHeight, GridColumn.MinWidth and SwipeButtonInfo.Width.

**Density-independent pixels** are relative units. On **160 DPI** displays, one **1dp** is equivalent to **1** physical pixel (**px**). On displays with higher DPI settings, the number of pixels in **1dp** will increase proportionally, thus producing a sharper image while preserving element size and layout.




To determine the number of pixels in **1dp** on a particular screen, divide the screen's DPI by **160**. The following table lists the results for a few common DPI settings.

| Display Density | 1 dp equals to |
|---|---|
| 120 dpi (ldpi) | 0.75 px |
| 160 dpi (mdpi) | 1 px |
| 240 dpi (hdpi) | 1.5 px |
| 320 dpi (xhdpi) | 2 px |
| 480 dpi (xxhdpi) | 3 px |
| 640 dpi (xxxhdpi) | 4 px |

© 2018 Developer Express Inc.

# API Reference

# DevExpress.Data

Contains enumerations implementing the data management functionality.

**Enumerations**

| | Enumeration | Description |
|---|---|---|
| | AutoFilterCondition | Lists the comparison operator types for the filter conditions created for specific columns via the auto-filter panel. |
| | ColumnFilterMode | Lists values that specify how column values are filtered via the auto-filter panel. |
| | ColumnGroupInterval | Contains values that specify how data rows are combined into groups when grid data is grouped by a specific column. |
| | ColumnSortMode | Lists values that specify how column values are sorted and grouped. |
| | ColumnSortOrder | Lists values specifying the sort order. |
| | UnboundColumnType | Contains values that specify the data type and binding mode of columns. |

© 2018 Developer Express Inc.

# AutoFilterCondition Enumeration

Lists the comparison operator types for the filter conditions created for specific columns via the auto-filter panel.

**Namespace:** DevExpress.Data
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public enum AutoFilterCondition
```

### Members

| Name | Description |
|---|---|
| Contains | The **Contains** operator selects records whose values in the corresponding column contain the entered string. |
| Default | For text and number columns as well as for date columns being filtered by their display text (the GridColumn.ColumnFilterMode property is set to ColumnFilterMode.DisplayText), the **Like** comparison operator is used as the default to create filter conditions. For other columns, the **Equals** comparison operator is used. |
| Equals | The **Equals** comparison operator selects records whose values in the corresponding column match the entered value. |
| Like | The **Like** comparison operator selects records whose values in the corresponding column start with the entered string. |

### Remarks

The auto-filter panel allows end-users to filter data on the fly. Typing text within this panel's cell automatically creates a filter condition against the corresponding column. The comparison operator used in this condition is determined by the column's GridColumn.AutoFilterCondition property.

### See Also

DevExpress.Data Namespace

© 2018 Developer Express Inc.

# ColumnFilterMode Enumeration

Lists values that specify how column values are filtered via the auto-filter panel.

**Namespace:** DevExpress.Data
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public enum ColumnFilterMode
```

### Members

| Name | Description |
|---|---|
| DisplayText | Column values are filtered by the display text (by formatted values). |
| Value | Column values are filtered by edit values. |

### Remarks

The **ColumnFilterMode** enumeration values are used by the GridColumn.ColumnFilterMode property.

### See Also

DevExpress.Data Namespace

© 2018 Developer Express Inc.

# ColumnGroupInterval Enumeration

Contains values that specify how data rows are combined into groups when grid data is grouped by a specific column.

**Namespace:** DevExpress.Data
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public enum ColumnGroupInterval
```

### Members

| Name | Description |
|---|---|
| Alphabetical | Rows are grouped by the first character of their value. |
| Date | This option is in effect only for columns that store date/time values.<br><br>Rows are grouped by the date part of their values. The time portion is ignored in this grouping mode. |
| DateMonth | This option is in effect only for columns that store date/time values.<br><br>Rows are grouped by the month and year. Examples of such groups: "January, 2014" , "February, 2014", "March, 2015"... |
| DateQuarter | This option is in effect only for columns that store date/time values.<br><br>Rows are grouped by the quarter and year. Examples of groups: "Q3, 2014", "Q4, 2014", "Q1, 2015" ... |
| DateRange | This option is in effect only for columns that store date/time values.<br><br>Rows are grouped into the following groups according to their date value as compared with the current system date: "Today", "Tomorrow", "Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday", "Next Week", "Two Weeks Away", "Three Weeks Away", "Next Month", "Beyond Next Month", "Yesterday", "Last Week", "Two Weeks Ago", "Three Weeks Ago", "Last Month", "Older". |
| DateYear | This option is in effect only for columns that store date/time values.<br><br>Rows are grouped by the year. Examples of such groups: "2013", "2014", "2016"... |
| Default | For columns which store date/time values, this option is the same as the DateRange option. For non date/time columns, this option is the same as the Value option. |
| DisplayText | Rows are grouped by the cell's display values.<br><br>In specific instances, cells can have different edit values, but the same display value. To combine these rows into the same group, use the DisplayText group mode. |
| Value | Rows are grouped by their values. The number of groups matches the number of unique values within the grouping column. |

© 2018 Developer Express Inc.

| | |
|---|---|
| | Note that System.DateTime objects which have the same date portion but different time portions are treated as unique values (each has its own group). |

**Remarks**

The **ColumnGroupInterval** enumeration values are used by the GridColumn.GroupInterval property.

**See Also**
DevExpress.Data Namespace

© 2018 Developer Express Inc.

# ColumnSortMode Enumeration

Lists values that specify how column values are sorted and grouped.

**Namespace:** DevExpress.Data
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public enum ColumnSortMode
```

### Members

| Name | Description |
|---|---|
| DisplayText | Column values are sorted and grouped by the display text (by formatted values). |
| Value | Column values are sorted and grouped by edit values. |

### Remarks

The ColumnSortMode enumeration values are used by the GridColumn.SortMode property.

### See Also

DevExpress.Data Namespace

© 2018 Developer Express Inc.

# ColumnSortOrder Enumeration

Lists values specifying the sort order.

**Namespace:** DevExpress.Data
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public enum ColumnSortOrder
```

### Members

| Name | Description |
|---|---|
| Ascending | Sorts the column in ascending order. |
| Descending | Sorts the columns in descending order. |
| None | No sorting is applied to a column. |

### Remarks

The **ColumnSortOrder** enumeration values are used by the GridColumn.SortOrder property.

### See Also

DevExpress.Data Namespace

© 2018 Developer Express Inc.

# UnboundColumnType Enumeration

Contains values that specify the data type and binding mode of columns.

**Namespace:** DevExpress.Data
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public enum UnboundColumnType
```

### Members

| Name | Description |
|---|---|
| Boolean | Indicates that the column is unbound and it contains Boolean values (the System.Boolean type). |
| Bound | Indicates that the column is bound to a field of the control's underlying data source. The type of data this column contains is determined by the bound field. |
| DateTime | Indicates that the column is unbound and it contains date/time values (the System.DateTime type). |
| Decimal | Indicates that the column is unbound and it contains decimal values (the System.Decimal type). |
| Integer | Indicates that the column is unbound and it contains integer values (the System.Int32 type). |
| Object | Indicates that the column is unbound and it contains values of any type. |
| String | Indicates that the column is unbound and it contains string values (the System.String type). |

### Remarks

The GridControl supports bound and unbound columns. Bound columns obtain their data automatically from the grid's underlying data source. The data for unbound columns should be provided manually by handling the GridControl.CustomUnboundColumnData event or by setting the GridColumn.UnboundExpression property.

Use the GridColumn.UnboundType property to get or set an **UnboundColumnType** value that specifies the column's data type and binding mode.

### See Also

DevExpress.Data Namespace

© 2018 Developer Express Inc.

# DevExpress.Export

Contains settings controlling data export from controls.

**Classes**

| | Class | Description |
|---|---|---|
| | CustomizeCellEventArgs | Provides data for the DataAwareExportOptionsBase.CustomizeCell and the CsvExportOptions.CustomizeCell events. |
| | CustomizeDataAreaEventArgs | Provides data for the DataAwareExportOptionsBase.CustomizeDataArea and the CsvExportOptions.CustomizeDataArea events. |
| | CustomizeFooterEventArgs | Provides data for the DataAwareExportOptionsBase.CustomizeFooter and the CsvExportOptions.CustomizeFooter events. |
| | CustomizeHeaderEventArgs | Provides data for the DataAwareExportOptionsBase.CustomizeHeader and the CsvExportOptions.CustomizeHeader events. |
| | CustomizeSheetEventArgs | Provides data for the DataAwareExportOptionsBase.CustomizeSheet and the CsvExportOptions.CustomizeSheet events. |

**Delegates**

| | Delegate | Description |
|---|---|---|
| | CustomizeCellEventHandler | Represents a method that will handle the DataAwareExportOptionsBase.CustomizeCell and the CsvExportOptions.CustomizeCell events. |
| | CustomizeDataAreaEventHandler | Represents a method that will handle the DataAwareExportOptionsBase.CustomizeDataArea and the CsvExportOptions.CustomizeDataArea events. |
| | CustomizeFooterEventHandler | Represents a method that will handle the DataAwareExportOptionsBase.CustomizeFooter and the CsvExportOptions.CustomizeFooter events. |
| | CustomizeHeaderEventHandler | Represents a method that will handle the DataAwareExportOptionsBase.CustomizeHeader and the CsvExportOptions.CustomizeHeader events. |
| | CustomizeSheetEventHandler | Represents a method that will handle the DataAwareExportOptionsBase.CustomizeSheet and the CsvExportOptions.CustomizeSheet events. |

**Enumerations**

© 2018 Developer Express Inc.

| | Enumeration | Description |
|---|---|---|
| | ExportTarget | Lists values specifying available formats for export. |
| | GroupState | Lists values specifying whether data groups are collapsed or expanded in the output document when exporting data from the grid. |

© 2018 Developer Express Inc.

# CustomizeCellEventArgs Class

Provides data for the DataAwareExportOptionsBase.CustomizeCell and the CsvExportOptions.CustomizeCell events.

**Namespace:** DevExpress.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public class CustomizeCellEventArgs : IDataAwareEventArgsBase
```

### Remarks

The **CustomizeCellEventArgs** objects are automatically created, initialized and passed to the DataAwareExportOptionsBase.CustomizeCell and the CsvExportOptions.CustomizeCell event handlers.

### Inheritance Hierarchy

System.Object
  DevExpress.Export.IDataAwareEventArgsBase
    **CustomizeCellEventArgs**

### See Also

CustomizeCellEventArgs Members
DevExpress.Export Namespace

© 2018 Developer Express Inc.

# CustomizeCellEventArgs Members

Provides data for the DataAwareExportOptionsBase.CustomizeCell and the CsvExportOptions.CustomizeCell events.

The following tables list the members exposed by the CustomizeCellEventArgs type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | CustomizeCellEventArgs | Initializes a new instance of the CustomizeCellEventArgs class. |

### Public Properties

| | Name | Description |
|---|---|---|
| | ColumnFieldName | Gets or sets the data source field name to which the current cell corresponds. |
| | DataSourceOwner | (Inherited from **DevExpress.Export.IDataAwareEventArgsBase**) |
| | DataSourceRowIndex | (Inherited from **DevExpress.Export.IDataAwareEventArgsBase**) |
| | DocumentRow | (Inherited from **DevExpress.Export.IDataAwareEventArgsBase**) |
| | Formatting | Gets or sets the object that defines the cell's formatting settings (font, alignment, background color, format string, etc.) |
| | Handled | (Inherited from **DevExpress.Export.IDataAwareEventArgsBase**) |
| | Hyperlink | Specifies the hyperlink, which belongs to the currently processed cell. |
| | RowHandle | (Inherited from **DevExpress.Export.IDataAwareEventArgsBase**) |
| | Value | Specifies the value of the cell currently being processed. |

### Public Methods

| | Name | Description |
|---|---|---|
| | Equals | (Inherited from **System.Object**) |
| S | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

**See Also**
CustomizeCellEventArgs Members
DevExpress.Export Namespace

© 2018 Developer Express Inc.

# CustomizeCellEventArgs Constructor

Initializes a new instance of the CustomizeCellEventArgs class.

**Namespace:** DevExpress.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public CustomizeCellEventArgs()
```

### Remarks

Instances of the CustomizeCellEventArgs class are automatically created, initialized and passed to the corresponding event handlers.

### See Also

CustomizeCellEventArgs Class
CustomizeCellEventArgs Members
DevExpress.Export Namespace

© 2018 Developer Express Inc.

# CustomizeCellEventArgs Properties

Provides data for the DataAwareExportOptionsBase.CustomizeCell and the CsvExportOptions.CustomizeCell events.

The following tables list the members exposed by the CustomizeCellEventArgs type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | CustomizeCellEventArgs | Initializes a new instance of the CustomizeCellEventArgs class. |

### Public Properties

| | Name | Description |
|---|---|---|
| | ColumnFieldName | Gets or sets the data source field name to which the current cell corresponds. |
| | DataSourceOwner | (Inherited from **DevExpress.Export.IDataAwareEventArgsBase**) |
| | DataSourceRowIndex | (Inherited from **DevExpress.Export.IDataAwareEventArgsBase**) |
| | DocumentRow | (Inherited from **DevExpress.Export.IDataAwareEventArgsBase**) |
| | Formatting | Gets or sets the object that defines the cell's formatting settings (font, alignment, background color, format string, etc.) |
| | Handled | (Inherited from **DevExpress.Export.IDataAwareEventArgsBase**) |
| | Hyperlink | Specifies the hyperlink, which belongs to the currently processed cell. |
| | RowHandle | (Inherited from **DevExpress.Export.IDataAwareEventArgsBase**) |
| | Value | Specifies the value of the cell currently being processed. |

### Public Methods

| | Name | Description |
|---|---|---|
| | Equals | (Inherited from **System.Object**) |
| S | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

**See Also**
CustomizeCellEventArgs Members
DevExpress.Export Namespace

© 2018 Developer Express Inc.

# ColumnFieldName Property

Gets or sets the data source field name to which the current cell corresponds.

**Namespace:** DevExpress.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public string ColumnFieldName { get; set; }
```

### Property Value

The data source field name.

### Remarks

Columns in data-bound controls are bound to certain fields in an underlying data source. When exporting these controls to XLS(X) format, use the **ColumnFieldName** property to identify the field name to which cells in the export document correspond.

### See Also

CustomizeCellEventArgs Class
CustomizeCellEventArgs Members
DevExpress.Export Namespace

© 2018 Developer Express Inc.

# Formatting Property

Gets or sets the object that defines the cell's formatting settings (font, alignment, background color, format string, etc.)

**Namespace:** DevExpress.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public XlFormattingObject Formatting { get; set; }
```

### Property Value

The object that defines the cell's formatting settings.

### See Also

CustomizeCellEventArgs Class
CustomizeCellEventArgs Members
DevExpress.Export Namespace

© 2018 Developer Express Inc.

# Hyperlink Property

Specifies the hyperlink, which belongs to the currently processed cell.

**Namespace:** DevExpress.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public string Hyperlink { get; set; }
```

### Property Value

A string value, that specifies the processed cell's hyperlink.

### See Also

CustomizeCellEventArgs Class
CustomizeCellEventArgs Members
DevExpress.Export Namespace

© 2018 Developer Express Inc.

# Value Property

Specifies the value of the cell currently being processed.

**Namespace:** DevExpress.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public object Value { get; set; }
```

### Property Value

An object which represents the processed cell's value.

### See Also

CustomizeCellEventArgs Class
CustomizeCellEventArgs Members
DevExpress.Export Namespace

© 2018 Developer Express Inc.

# CustomizeCellEventHandler Delegate

Represents a method that will handle the CsvExportOptions.CustomizeCell and the DataAwareExportOptionsBase.CustomizeCell events.

**Namespace:** DevExpress.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public delegate void CustomizeCellEventHandler(
   CustomizeCellEventArgs e
)
```

### Parameters

The declaration of your event handler must have the same parameters as the **CustomizeCellEventHandler** delegate declaration.

*e*
A CustomizeCellEventArgs class instance that contains event data.

### See Also

DevExpress.Export Namespace

© 2018 Developer Express Inc.

# CustomizeDataAreaEventArgs Class

Provides data for the DataAwareExportOptionsBase.CustomizeDataArea and the CsvExportOptions.CustomizeDataArea events.

**Namespace:** DevExpress.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public class CustomizeDataAreaEventArgs : IDataAwareEventArgsBase
```

### Remarks

The **CustomizeDataAreaEventArgs** objects are automatically created, initialized and passed to the DataAwareExportOptionsBase.CustomizeDataArea and the CsvExportOptions.CustomizeDataArea event handlers.

### Inheritance Hierarchy

System.Object
  DevExpress.Export.IDataAwareEventArgsBase
    **CustomizeDataAreaEventArgs**

### See Also

CustomizeDataAreaEventArgs Members
DevExpress.Export Namespace

© 2018 Developer Express Inc.

# CustomizeDataAreaEventArgs Members

Provides data for the DataAwareExportOptionsBase.CustomizeDataArea and the CsvExportOptions.CustomizeDataArea events.

The following tables list the members exposed by the CustomizeDataAreaEventArgs type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | CustomizeDataAreaEventArgs | Initializes a new instance of the CustomizeDataAreaEventArgs class. |

### Public Properties

| | Name | Description |
|---|---|---|
| | Context | Specifies the context in which a data area is customized. |
| | DataSourceOwner | (Inherited from **DevExpress.Export.IDataAwareEventArgsBase**) |
| | DataSourceRowIndex | (Inherited from **DevExpress.Export.IDataAwareEventArgsBase**) |
| | DocumentRow | (Inherited from **DevExpress.Export.IDataAwareEventArgsBase**) |
| | Handled | (Inherited from **DevExpress.Export.IDataAwareEventArgsBase**) |
| | RowHandle | (Inherited from **DevExpress.Export.IDataAwareEventArgsBase**) |

### Public Methods

| | Name | Description |
|---|---|---|
| | Equals | (Inherited from **System.Object**) |
| S | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

### See Also

CustomizeDataAreaEventArgs Members
DevExpress.Export Namespace

© 2018 Developer Express Inc.

# CustomizeDataAreaEventArgs Constructor

Initializes a new instance of the CustomizeDataAreaEventArgs class.

**Namespace:** DevExpress.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public CustomizeDataAreaEventArgs()
```

### Remarks

Instances of the CustomizeDataAreaEventArgs class are automatically created, initialized and passed to the corresponding event handlers.

### See Also

CustomizeDataAreaEventArgs Class
CustomizeDataAreaEventArgs Members
DevExpress.Export Namespace

© 2018 Developer Express Inc.

# CustomizeDataAreaEventArgs Properties

Provides data for the DataAwareExportOptionsBase.CustomizeDataArea and the CsvExportOptions.CustomizeDataArea events.

The following tables list the members exposed by the CustomizeDataAreaEventArgs type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | CustomizeDataAreaEventArgs | Initializes a new instance of the CustomizeDataAreaEventArgs class. |

### Public Properties

| | Name | Description |
|---|---|---|
| | Context | Specifies the context in which a data area is customized. |
| | DataSourceOwner | (Inherited from **DevExpress.Export.IDataAwareEventArgsBase**) |
| | DataSourceRowIndex | (Inherited from **DevExpress.Export.IDataAwareEventArgsBase**) |
| | DocumentRow | (Inherited from **DevExpress.Export.IDataAwareEventArgsBase**) |
| | Handled | (Inherited from **DevExpress.Export.IDataAwareEventArgsBase**) |
| | RowHandle | (Inherited from **DevExpress.Export.IDataAwareEventArgsBase**) |

### Public Methods

| | Name | Description |
|---|---|---|
| | Equals | (Inherited from **System.Object**) |
| S | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

### See Also

CustomizeDataAreaEventArgs Members
DevExpress.Export Namespace

© 2018 Developer Express Inc.

# Context Property

Specifies the context in which a data area is customized.

**Namespace:** DevExpress.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public IExportContext Context { get; set; }
```

### Property Value

An object that represents the context in which the data area is customized.

### See Also

CustomizeDataAreaEventArgs Class
CustomizeDataAreaEventArgs Members
DevExpress.Export Namespace

© 2018 Developer Express Inc.

# CustomizeDataAreaEventHandler Delegate

Represents a method that will handle the CsvExportOptions.CustomizeDataArea and the DataAwareExportOptionsBase.CustomizeDataArea events.

**Namespace:** DevExpress.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public delegate void CustomizeDataAreaEventHandler(
   CustomizeDataAreaEventArgs e
)
```

### Parameters

The declaration of your event handler must have the same parameters as the **CustomizeDataAreaEventHandler** delegate declaration.

*e*
A CustomizeDataAreaEventArgs class instance that contains event data.

### See Also

DevExpress.Export Namespace

© 2018 Developer Express Inc.

# CustomizeFooterEventArgs Class

Provides data for the DataAwareExportOptionsBase.CustomizeFooter and the CsvExportOptions.CustomizeFooter events.

**Namespace:** DevExpress.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public class CustomizeFooterEventArgs : object
```

### Remarks

The **CustomizeFooterEventArgs** objects are automatically created, initialized and passed to the DataAwareExportOptionsBase.CustomizeFooter and the CsvExportOptions.CustomizeFooter event handlers.

### Inheritance Hierarchy

System.Object
  **CustomizeFooterEventArgs**

### See Also

CustomizeFooterEventArgs Members
DevExpress.Export Namespace

© 2018 Developer Express Inc.

# CustomizeFooterEventArgs Members

Provides data for the DataAwareExportOptionsBase.CustomizeFooter and the CsvExportOptions.CustomizeFooter events.

The following tables list the members exposed by the CustomizeFooterEventArgs type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | CustomizeFooterEventArgs | Initializes a new instance of the CustomizeFooterEventArgs class. |

### Public Properties

| | Name | Description |
|---|---|---|
| | Context | Specifies the context in which a footer is customized. |

### Public Methods

| | Name | Description |
|---|---|---|
| | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

**See Also**
CustomizeFooterEventArgs Members
DevExpress.Export Namespace

© 2018 Developer Express Inc.

# CustomizeFooterEventArgs Constructor

Initializes a new instance of the CustomizeFooterEventArgs class.

**Namespace:** DevExpress.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public CustomizeFooterEventArgs()
```

### Remarks

Instances of the CustomizeFooterEventArgs class are automatically created, initialized and passed to the corresponding event handlers.

### See Also

CustomizeFooterEventArgs Class
CustomizeFooterEventArgs Members
DevExpress.Export Namespace

© 2018 Developer Express Inc.

# CustomizeFooterEventArgs Properties

Provides data for the DataAwareExportOptionsBase.CustomizeFooter and the CsvExportOptions.CustomizeFooter events.

The following tables list the members exposed by the CustomizeFooterEventArgs type.

**Public Constructors**

| | Name | Description |
|---|---|---|
| | CustomizeFooterEventArgs | Initializes a new instance of the CustomizeFooterEventArgs class. |

**Public Properties**

| | Name | Description |
|---|---|---|
| | Context | Specifies the context in which a footer is customized. |

**Public Methods**

| | Name | Description |
|---|---|---|
| | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

**See Also**
CustomizeFooterEventArgs Members
DevExpress.Export Namespace

© 2018 Developer Express Inc.

# Context Property

Specifies the context in which a footer is customized.

**Namespace:** DevExpress.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public IFooterExportContext Context { get; set; }
```

### Property Value

An object that represents the context in which the footer is customized.

### See Also

CustomizeFooterEventArgs Class
CustomizeFooterEventArgs Members
DevExpress.Export Namespace

© 2018 Developer Express Inc.

# CustomizeFooterEventHandler Delegate

Represents a method that will handle the CsvExportOptions.CustomizeFooter and the DataAwareExportOptionsBase.CustomizeFooter events.

**Namespace:** DevExpress.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public delegate void CustomizeFooterEventHandler(
   CustomizeFooterEventArgs e
)
```

### Parameters

The declaration of your event handler must have the same parameters as the **CustomizeFooterEventHandler** delegate declaration.

*e*
A CustomizeFooterEventArgs class instance that contains event data.

### See Also

DevExpress.Export Namespace

© 2018 Developer Express Inc.

# CustomizeHeaderEventArgs Class

Provides data for the DataAwareExportOptionsBase.CustomizeHeader and the CsvExportOptions.CustomizeHeader events.

**Namespace:** DevExpress.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public class CustomizeHeaderEventArgs : IDataAwareEventArgsBase
```

### Remarks

The **CustomizeHeaderEventArgs** objects are automatically created, initialized and passed to the DataAwareExportOptionsBase.CustomizeHeader and the CsvExportOptions.CustomizeHeader event handlers.

### Inheritance Hierarchy

System.Object
  DevExpress.Export.IDataAwareEventArgsBase
    **CustomizeHeaderEventArgs**

### See Also

CustomizeHeaderEventArgs Members
DevExpress.Export Namespace

© 2018 Developer Express Inc.

# CustomizeHeaderEventArgs Members

Provides data for the DataAwareExportOptionsBase.CustomizeHeader and the CsvExportOptions.CustomizeHeader events.

The following tables list the members exposed by the CustomizeHeaderEventArgs type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | CustomizeHeaderEventArgs | Initializes a new instance of the CustomizeHeaderEventArgs class. |

### Public Properties

| | Name | Description |
|---|---|---|
| | Context | Specifies the context in which a header is customized. |
| | DataSourceOwner | (Inherited from **DevExpress.Export.IDataAwareEventArgsBase**) |
| | DataSourceRowIndex | (Inherited from **DevExpress.Export.IDataAwareEventArgsBase**) |
| | DocumentRow | (Inherited from **DevExpress.Export.IDataAwareEventArgsBase**) |
| | Handled | (Inherited from **DevExpress.Export.IDataAwareEventArgsBase**) |
| | RowHandle | (Inherited from **DevExpress.Export.IDataAwareEventArgsBase**) |

### Public Methods

| | Name | Description |
|---|---|---|
| | Equals | (Inherited from **System.Object**) |
| S | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

**See Also**
CustomizeHeaderEventArgs Members
DevExpress.Export Namespace

© 2018 Developer Express Inc.

# CustomizeHeaderEventArgs Constructor

Initializes a new instance of the CustomizeHeaderEventArgs class.

**Namespace:** DevExpress.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public CustomizeHeaderEventArgs()
```

### Remarks

Instances of the CustomizeHeaderEventArgs class are automatically created, initialized and passed to the corresponding event handlers.

### See Also

CustomizeHeaderEventArgs Class
CustomizeHeaderEventArgs Members
DevExpress.Export Namespace

© 2018 Developer Express Inc.

# CustomizeHeaderEventArgs Properties

Provides data for the DataAwareExportOptionsBase.CustomizeHeader and the CsvExportOptions.CustomizeHeader events.

The following tables list the members exposed by the CustomizeHeaderEventArgs type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | CustomizeHeaderEventArgs | Initializes a new instance of the CustomizeHeaderEventArgs class. |

### Public Properties

| | Name | Description |
|---|---|---|
| | Context | Specifies the context in which a header is customized. |
| | DataSourceOwner | (Inherited from **DevExpress.Export.IDataAwareEventArgsBase**) |
| | DataSourceRowIndex | (Inherited from **DevExpress.Export.IDataAwareEventArgsBase**) |
| | DocumentRow | (Inherited from **DevExpress.Export.IDataAwareEventArgsBase**) |
| | Handled | (Inherited from **DevExpress.Export.IDataAwareEventArgsBase**) |
| | RowHandle | (Inherited from **DevExpress.Export.IDataAwareEventArgsBase**) |

### Public Methods

| | Name | Description |
|---|---|---|
| | Equals | (Inherited from **System.Object**) |
| S | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

### See Also

CustomizeHeaderEventArgs Members
DevExpress.Export Namespace

© 2018 Developer Express Inc.

# Context Property

Specifies the context in which a header is customized.

**Namespace:** DevExpress.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public IHeaderExportContext Context { get; set; }
```

### Property Value

An object that represents the context in which the header is customized.

### See Also

CustomizeHeaderEventArgs Class
CustomizeHeaderEventArgs Members
DevExpress.Export Namespace

© 2018 Developer Express Inc.

# CustomizeHeaderEventHandler Delegate

Represents a method that will handle the CsvExportOptions.CustomizeHeader and the DataAwareExportOptionsBase.CustomizeHeader events.

**Namespace:** DevExpress.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public delegate void CustomizeHeaderEventHandler(
   CustomizeHeaderEventArgs e
)
```

### Parameters

The declaration of your event handler must have the same parameters as the **CustomizeHeaderEventHandler** delegate declaration.

*e*
A CustomizeHeaderEventArgs class instance that contains event data.

### See Also

DevExpress.Export Namespace

© 2018 Developer Express Inc.

# CustomizeSheetEventArgs Class

Provides data for the DataAwareExportOptionsBase.CustomizeSheet and the CsvExportOptions.CustomizeSheet events.

**Namespace:** DevExpress.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public class CustomizeSheetEventArgs : object
```

### Remarks

The **CustomizeSheetEventArgs** objects are automatically created, initialized and passed to the DataAwareExportOptionsBase.CustomizeSheet and the CsvExportOptions.CustomizeSheet event handlers.

### Inheritance Hierarchy

System.Object
  **CustomizeSheetEventArgs**

### See Also

CustomizeSheetEventArgs Members
DevExpress.Export Namespace

© 2018 Developer Express Inc.

# CustomizeSheetEventArgs Members

Provides data for the DataAwareExportOptionsBase.CustomizeSheet and the CsvExportOptions.CustomizeSheet events.

The following tables list the members exposed by the CustomizeSheetEventArgs type.

**Public Constructors**

| | Name | Description |
|---|---|---|
| | CustomizeSheetEventArgs | Initializes a new instance of the CustomizeSheetEventArgs class. |

**Public Properties**

| | Name | Description |
|---|---|---|
| | Context | Specifies the context in which an output document is customized. |

**Public Methods**

| | Name | Description |
|---|---|---|
| | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

**See Also**
CustomizeSheetEventArgs Members
DevExpress.Export Namespace

© 2018 Developer Express Inc.

# CustomizeSheetEventArgs Constructor

Initializes a new instance of the CustomizeSheetEventArgs class.

**Namespace:** DevExpress.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public CustomizeSheetEventArgs()
```

### Remarks

Instances of the CustomizeSheetEventArgs class are automatically created, initialized and passed to the corresponding event handlers.

### See Also

CustomizeSheetEventArgs Class
CustomizeSheetEventArgs Members
DevExpress.Export Namespace

© 2018 Developer Express Inc.

# CustomizeSheetEventArgs Properties

Provides data for the DataAwareExportOptionsBase.CustomizeSheet and the CsvExportOptions.CustomizeSheet events.

The following tables list the members exposed by the CustomizeSheetEventArgs type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | CustomizeSheetEventArgs | Initializes a new instance of the CustomizeSheetEventArgs class. |

### Public Properties

| | Name | Description |
|---|---|---|
| | Context | Specifies the context in which an output document is customized. |

### Public Methods

| | Name | Description |
|---|---|---|
| | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

**See Also**
CustomizeSheetEventArgs Members
DevExpress.Export Namespace

© 2018 Developer Express Inc.

# Context Property

Specifies the context in which an output document is customized.

**Namespace:** DevExpress.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public ISheetCustomizationContext Context { get; set; }
```

### Property Value

An object that represents the context in which the output document is customized.

### See Also

CustomizeSheetEventArgs Class
CustomizeSheetEventArgs Members
DevExpress.Export Namespace

© 2018 Developer Express Inc.

# CustomizeSheetEventHandler Delegate

Represents a method that will handle the CsvExportOptions.CustomizeSheet and the DataAwareExportOptionsBase.CustomizeSheet events.

**Namespace:** DevExpress.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public delegate void CustomizeSheetEventHandler(
   CustomizeSheetEventArgs e
)
```

### Parameters

The declaration of your event handler must have the same parameters as the **CustomizeSheetEventHandler** delegate declaration.

*e*
A CustomizeSheetEventArgs class instance that contains event data.

### See Also

DevExpress.Export Namespace

© 2018 Developer Express Inc.

# ExportTarget Enumeration

Lists values specifying available formats for export.

**Namespace:** DevExpress.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public enum ExportTarget
```

### Members

| Name | Description |
|---|---|
| Csv | The CSV export format. |
| Xls | The XLS export format. |
| Xlsx | The XLSX export format. |

### Remarks

An **ExportTarget** enumeration member is passed as a parameter to the GridControl.ExportToExcel method.

### See Also

DevExpress.Export Namespace

© 2018 Developer Express Inc.

# GroupState Enumeration

Lists values specifying whether data groups are collapsed or expanded in the output document when exporting data from the grid.

**Namespace:** DevExpress.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public enum GroupState
```

### Members

| Name | Description |
|---|---|
| CollapseAll | All data groups are collapsed in the exported document. |
| Default | Data groups are collapsed and expanded in the exported document as they are in the source grid. |
| ExpandAll | All data groups are expanded in the exported document. |

### Remarks

The **GroupState** enumeration values are used by the DataAwareExportOptionsBase.GroupState property of the XlsExportOptions or XlsxExportOptions object.

### See Also

DevExpress.Export Namespace

© 2018 Developer Express Inc.

# DevExpress.Mobile.Core

Contains common utility classes used by Xamarin.Forms controls from DevExpress.

**Enumerations**

| | Enumeration | Description |
|---|---|---|
| | VisibilityState | Lists values which specify the visibility of an element. |

© 2018 Developer Express Inc.

# VisibilityState Enumeration

Lists values which specify the visibility of an element.

**Namespace:** DevExpress.Mobile.Core
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public enum VisibilityState
```

### Members

| Name | Description |
|---|---|
| Always | An element is always shown. |
| Default | An element is automatically shown when required. |
| Never | An element is never shown. |

### Remarks

For example, the **VisibilityState** enumeration values are used by the GridControl.FilterPanelVisibility and GridControl.TotalSummaryVisibility properties.

### See Also

DevExpress.Mobile.Core Namespace

© 2018 Developer Express Inc.

# DevExpress.Mobile.Core.ConditionalFormatting

Contains enumerations implementing the conditional formatting functionality.

**Enumerations**

| | Enumeration | Description |
|---|---|---|
| | ConditionRule | Lists the values that specify the conditional formatting rule. |
| | TopBottomRule | Lists the values that specify the rule for the top/bottom conditional formatting. |

© 2018 Developer Express Inc.

# ConditionRule Enumeration

Lists the values that specify the conditional formatting rule.

**Namespace:** DevExpress.Mobile.Core.ConditionalFormatting
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public enum ConditionRule
```

### Members

| Name | Description |
|---|---|
| Between | The format is applied to cells whose values fall into the range specified by the FormatCondition.Value1 and FormatCondition.Value2 properties. |
| Equal | The format is applied to cells whose values match the FormatCondition.Value1 property value. |
| Expression | The format is applied to cells for which the ExpressionConditionBase.Expression evaluates to **true**. |
| Greater | The format is applied to cells whose values are greater than the FormatCondition.Value1 property value. |
| GreaterOrEqual | The format is applied to cells whose values are greater or equal to the FormatCondition.Value1 property value. |
| Less | The format is applied to cells whose values are less than the FormatCondition.Value1 property value. |
| LessOrEqual | The format is applied to cells whose values are less or equal to the FormatCondition.Value1 property value. |
| None | The format is applied to all cells of the target column. |
| NotBetween | The format is applied to cells whose values fall outside of the range specified by the FormatCondition.Value1 and FormatCondition.Value2 properties. |
| NotEqual | The format is applied to cells whose values do not match the FormatCondition.Value1 property value. |

### Remarks

The values listed by the **ConditionRule** enumeration are used to set the FormatCondition.ValueRule property.

### See Also

DevExpress.Mobile.Core.ConditionalFormatting Namespace

© 2018 Developer Express Inc.

# TopBottomRule Enumeration

Lists the values that specify the rule for the top/bottom conditional formatting.

**Namespace:** DevExpress.Mobile.Core.ConditionalFormatting
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public enum TopBottomRule
```

### Members

| Name | Description |
|---|---|
| AboveAverage | The format is applied to cells whose values are above the average value for the field. |
| BelowAverage | The format is applied to cells whose values are below the average value for the field. |
| BottomItems | The format is applied to cells whose values are the lowest for the field. The number of cells is specified by the TopBottomRuleFormatCondition.Threshold property. |
| BottomPercent | The format is applied to cells whose values are the lowest for the field. The percentage of cells is specified by the TopBottomRuleFormatCondition.Threshold property. |
| TopItems | The format is applied to cells whose values are the largest for the field. The number of cells is specified by the TopBottomRuleFormatCondition.Threshold property. |
| TopPercent | The format is applied to cells whose values are the largest for the field. The percentage of cells is specified by the TopBottomRuleFormatCondition.Threshold property. |

### Remarks

The values listed by the **TopBottomRule** enumeration are used to set the TopBottomRuleFormatCondition.Rule property.

### See Also

DevExpress.Mobile.Core.ConditionalFormatting Namespace

© 2018 Developer Express Inc.

# DevExpress.Mobile.DataGrid

Contains classes which implement the main functionality of the Grid for Xamarin control.

**Classes**

| | Class | Description |
|---|---|---|
| | AutoGeneratingColumnEventArgs | Provides data for the AutoGeneratingColumn event. |
| | CellData | Contains information on a data cell. |
| | ColorScaleFormatCondition | Represents a color scale format condition. |
| | CustomizeCellDisplayTextEventArgs | Provides data for the CustomizeCellDisplayText event. |
| | CustomizeCellEventArgs | Provides data for the CustomizeCell event. |
| | CustomSummaryEventArgs | Provides data for the CalculateCustomSummary event. |
| | DataBarFormatCondition | Represents a data bar format condition. |
| | DateColumn | A grid column to display and edit date values. |
| | EditValuesContainer | Contains information on a data row currently being edited by an end-user via an edit form. |
| | ExpressionConditionBase | Serves as a base for classes that represent expression-based format conditions. |
| | FormatCondition | Represents a format condition based on a specified rule(s) or expression (equal, less, between, etc.). |
| | FormatConditionBase | Serves as a base for classes that represent format conditions. |
| | FormatConditionCollection | A collection of format conditions. |
| | GridColumn | An abstract class that serves as the base class for grid columns. |
| | GridColumnCollection | A grid's column collection. |
| | GridColumnDataEventArgs | Provides data for the CustomUnboundColumnData event. |
| | GridColumnSummary | A data summary item. |
| | GridControl | The grid control. |
| | HorizontalScrollOffsetEventArgs | Provides data for the HorizontalScrollOffsetChanged event. |
| | IconSetFormatCondition | Represents an icon set format condition. |
| | ImageColumn | A column type that allows you to display images. |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | IndicatorFormatConditionBase | Serves as a base for classes that represent indicator-type format conditions. |
| | InitNewRowEventArgs | Provides data for the InitNewRow event. |
| | NumberColumn | A grid column to display and edit numeric values. |
| | PickerColumn | A grid column allowing an end-user to edit a cell value by selecting an item from the predefined set. |
| | PopupMenuEventArgs | Provides data for the PopupMenuCustomization event. |
| | RowAllowEventArgs | Provides data for the GroupRowExpanding and GroupRowCollapsing events. |
| | RowEditingEventArgs | Provides data for the EndRowEdit event. |
| | RowEventArgs | Provides data for the RowTap event. |
| | RowTapEventArgs | Provides data for the RowTap event. |
| | SourceRowEventArgs | Provides data for the EndRowEdit event. |
| | SwipeButtonEventArgs | Provides data for the SwipeButtonClick event. |
| | SwipeButtonInfo | Contains information on a button shown when an end-user swipes left to right, or right to left, over a data row. |
| | SwipeButtonShowingEventArgs | Provides data for the SwipeButtonShowing event. |
| | SwitchColumn | A grid column to display Boolean values and select between two states. |
| | TemplateColumn | A column type that allows you to define a custom template for column cells. |
| | TextColumn | A grid column to display and edit text values. |
| | ThemeFontAttributes | Allows you to specify required font attributes for the grid's themes. |
| | TopBottomRuleFormatCondition | Represents a top/bottom format condition. |

**Interfaces**

| | Interface | Description |
|---|---|---|
| | IEditableRowData | Contains information on a grid's row and allows you to modify field values in this row. |
| | IGroupInfo | Provides properties to obtain information on a data group. |
| | IRowData | Provides access to the data in a grid's row. |

**Structures**

| | Structure | Description |
|---|---|---|

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | CellIndex | Stores the information on a cell location within the grid. |

**Delegates**

| | Delegate | Description |
|---|---|---|
| | AutoGeneratingColumnEventHandler | Represents a method that will handle the AutoGeneratingColumn event. |
| | CustomizeCellDisplayTextEventHandler | Represents a method that will handle the CustomizeCellDisplayText event. |
| | CustomizeCellEventHandler | Represents a method that will handle the CustomizeCell event. |
| | CustomSummaryEventHandler | Represents a method that will handle the CalculateCustomSummary event. |
| | GridColumnDataEventHandler | Represents a method that will handle the CustomUnboundColumnData event. |
| | HorizontalScrollOffsetEventHandler | Represents a method that will handle the HorizontalScrollOffsetChanged event. |
| | InitNewRowEventHandler | Represents a method that will handle the InitNewRow event. |
| | PopupMenuEventHandler | Represents a method that will handle the PopupMenuCustomization event. |
| | RowAllowEventHandler | Represents a method that will handle the GroupRowExpanding and GroupRowCollapsed events. |
| | RowEditingEventHandler | Represents a method that will handle the EndRowEdit event. |
| | RowEventHandler | Represents a method that will handle the RowTap event. |
| | RowTapEventHandler | Represents a method that will handle the RowTap event. |
| | SourceRowEventHandler | Represents a method that will handle the EndRowEdit event. |
| | SwipeButtonEventHandler | Represents a method that will handle the SwipeButtonClick event. |
| | SwipeButtonShowingEventHandler | Represents a method that will handle the SwipeButtonShowing event. |

**Enumerations**

| | Enumeration | Description |
|---|---|---|
| | AutoGenerateColumnsMode | Lists values used to specify how columns are generated in the grid. |
| | ColumnContentAlignment | Specifies the content alignment within the column cells. |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | CustomSummaryProcess | Lists values that specify the current status of custom summary calculation. |
| | EditingRowAction | Lists values specifying whether cell values modified via an in-place editor are saved or discarded. |
| | FixedStyle | Lists values that specify a column's behavior when a grid is horizontally scrolled. |
| | GridPopupMenuType | Lists values specifying GridControl's context menu types. |
| | GridSortMode | Lists values specifying whether the grid allows data sorting by multiple columns or by a single column only. |
| | RowEditMode | Lists values that specify the GridControl's editing mode. |
| | SummaryType | Lists values specifying the aggregate function types. |

© 2018 Developer Express Inc.

# AutoGenerateColumnsMode Enumeration

Lists values used to specify how columns are generated in the grid.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public enum AutoGenerateColumnsMode
```

### Members

| Name | Description |
|---|---|
| Add | Creates columns for all fields of a data source, preserving the columns the grid already contains. |
| Auto | Creates columns for all fields of a data source if the grid does not contain any columns. |
| None | Does not create columns. |
| Replace | Creates columns for all fields of a datasource, removing columns the grid already contains. |

### Remarks

The **AutoGenerateColumnsMode** enumeration values are used by the GridControl.AutoGenerateColumnsMode property.

### See Also

DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# AutoGeneratingColumnEventArgs Class

Provides data for the GridControl.AutoGeneratingColumn event.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public class AutoGeneratingColumnEventArgs : CancelEventArgs
```

### Remarks

The **AutoGeneratingColumnEventArgs** objects are automatically created, initialized and passed to the GridControl.AutoGeneratingColumn event handlers.

### Inheritance Hierarchy

System.Object
  System.EventArgs
    System.ComponentModel.CancelEventArgs
      **AutoGeneratingColumnEventArgs**

### See Also

AutoGeneratingColumnEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# AutoGeneratingColumnEventArgs Members

Provides data for the GridControl.AutoGeneratingColumn event.

The following tables list the members exposed by the AutoGeneratingColumnEventArgs type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | AutoGeneratingColumnEventArgs | Initializes a new instance of the AutoGeneratingColumnEventArgs class. |

### Public Properties

| | Name | Description |
|---|---|---|
| | Cancel | (Inherited from **System.ComponentModel.CancelEventArgs**) |
| | Column | Gets the column being auto-generated. |

### Public Methods

| | Name | Description |
|---|---|---|
| | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

### Public Fields

| | Name | Description |
|---|---|---|
| | Empty | (Inherited from **System.EventArgs**) |

**See Also**
AutoGeneratingColumnEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# AutoGeneratingColumnEventArgs Constructor

Initializes a new instance of the AutoGeneratingColumnEventArgs class.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public AutoGeneratingColumnEventArgs(
   GridColumn column
)
```

### Parameters

*column*
A GridColumn object that specifies the grid column to be auto-generated. This value is assigned to the Column property.

### Remarks

Instances of the AutoGeneratingColumnEventArgs class are automatically created, initialized and passed to the corresponding event handlers.

### See Also

AutoGeneratingColumnEventArgs Class
AutoGeneratingColumnEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# AutoGeneratingColumnEventArgs Properties

Provides data for the GridControl.AutoGeneratingColumn event.

The following tables list the members exposed by the AutoGeneratingColumnEventArgs type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | AutoGeneratingColumnEventArgs | Initializes a new instance of the AutoGeneratingColumnEventArgs class. |

### Public Properties

| | Name | Description |
|---|---|---|
| | Cancel | (Inherited from **System.ComponentModel.CancelEventArgs**) |
| | Column | Gets the column being auto-generated. |

### Public Methods

| | Name | Description |
|---|---|---|
| | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

### Public Fields

| | Name | Description |
|---|---|---|
| | Empty | (Inherited from **System.EventArgs**) |

**See Also**
AutoGeneratingColumnEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# Column Property

Gets the column being auto-generated.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public GridColumn Column { get; }
```

### Property Value

A GridColumn object or descendant specifying the column being auto-generated.

### See Also

AutoGeneratingColumnEventArgs Class
AutoGeneratingColumnEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# AutoGeneratingColumnEventHandler Delegate

Represents a method that will handle the GridControl.AutoGeneratingColumn event.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public delegate void AutoGeneratingColumnEventHandler(
   object sender,
   AutoGeneratingColumnEventArgs e
)
```

### Parameters

The declaration of your event handler must have the same parameters as the **AutoGeneratingColumnEventHandler** delegate declaration.

*sender*
The event source. Identifies the GridControl object that raised the event.

*e*
A AutoGeneratingColumnEventArgs class instance that contains event data.

### See Also

DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# CellData Class

Contains information on a data cell.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public class CellData : NotificationObject
```

### Remarks

A **CellData** object contains information about a data cell and serves as a data context (binding source) for a template that defines the presentation of cells in a grid's TemplateColumn columns.

### Inheritance Hierarchy

System.Object
  DevExpress.Mobile.DataGrid.Internal.NotificationObject
    **CellData**

### See Also

CellData Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# CellData Members

Contains information on a data cell.

The following tables list the members exposed by the CellData type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | CellData | Initializes a new instance of the CellData class. |

### Public Properties

| | Name | Description |
|---|---|---|
| | DisplayText | Gets or sets a string that specifies the formatted cell value as it is displayed. |
| | Index | Gets or sets an object that contains information on a cell position. |
| | IsSelected | Gets whether a cell belongs to a row that is currently selected in the grid. |
| | Source | Gets or sets an object that corresponds to a data row. |
| | Value | Gets or sets an object that specifies a data field value. |

### Public Methods

| | Name | Description |
|---|---|---|
| S | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

### Public Events

| | Name | Description |
|---|---|---|
| | PropertyChanged | (Inherited from **DevExpress.Mobile.DataGrid.Internal.NotificationObject**) |

**See Also**

CellData Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# CellData Constructor

Initializes a new instance of the CellData class.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**
**C#**

```
public CellData()
```

**See Also**
CellData Class
CellData Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# CellData Properties

Contains information on a data cell.

The following tables list the members exposed by the CellData type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | CellData | Initializes a new instance of the CellData class. |

### Public Properties

| | Name | Description |
|---|---|---|
| | DisplayText | Gets or sets a string that specifies the formatted cell value as it is displayed. |
| | Index | Gets or sets an object that contains information on a cell position. |
| | IsSelected | Gets whether a cell belongs to a row that is currently selected in the grid. |
| | Source | Gets or sets an object that corresponds to a data row. |
| | Value | Gets or sets an object that specifies a data field value. |

### Public Methods

| | Name | Description |
|---|---|---|
| | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

### Public Events

| | Name | Description |
|---|---|---|
| | PropertyChanged | (Inherited from **DevExpress.Mobile.DataGrid.Internal.NotificationObject**) |

**See Also**
CellData Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# DisplayText Property

Gets or sets a string that specifies the formatted cell value as it is displayed.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public string DisplayText { get; set; }
```

### Property Value

A string that specifies the cell's display text.

### Remarks

Initially, the **DisplayText** property contains text currently displayed in a cell - the cell's value (Value) to which the column format (GridColumn.DisplayFormat) is applied. To provide a custom text to be displayed in a cell, assign it to the **DisplayText** property.

### See Also

CellData Class
CellData Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# Index Property

Gets or sets an object that contains information on a cell position.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public CellIndex Index { get; set; }
```

### Property Value

A CellIndex object that specifies a cell location within the grid.

### See Also

CellData Class
CellData Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# IsSelected Property

Gets whether a cell belongs to a row that is currently selected in the grid.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public bool IsSelected { get; }
```

### Property Value

**true**, if the row is selected; otherwise, **false**.

### See Also

CellData Class
CellData Members
DevExpress.Mobile.DataGrid Namespace
GridControl.SelectedRowHandle

© 2018 Developer Express Inc.

# Source Property

Gets or sets an object that corresponds to a data row.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public object Source { get; set; }
```

### Property Value

An object in a data source that corresponds to a row.

### Remarks

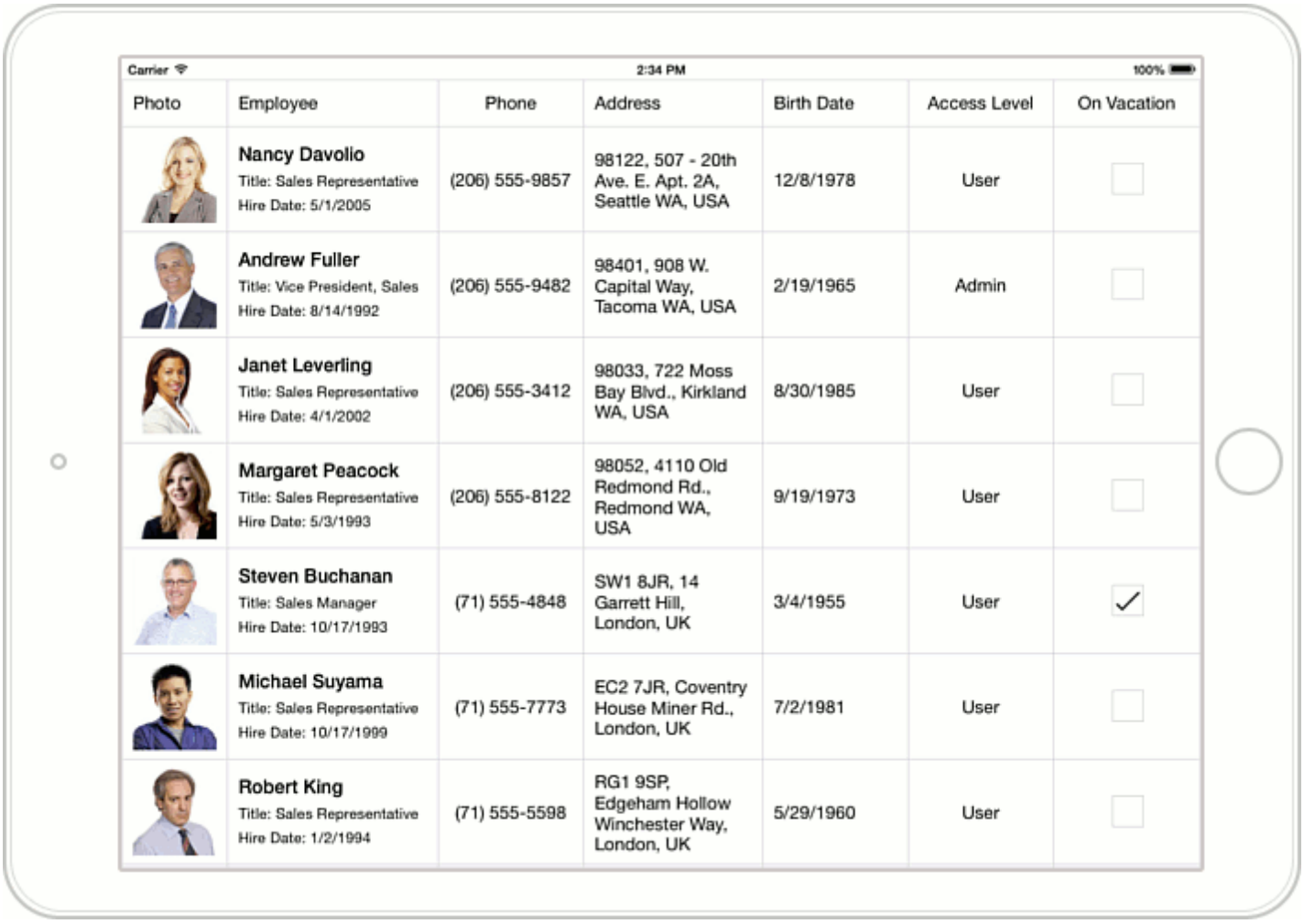
Use the **Source** property to set the entire data row as a binding source for a cell template. For example, this can be useful if a cell for which a template is defined has bindings to multiple properties of a data row object.

### Example

| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T208848. | |

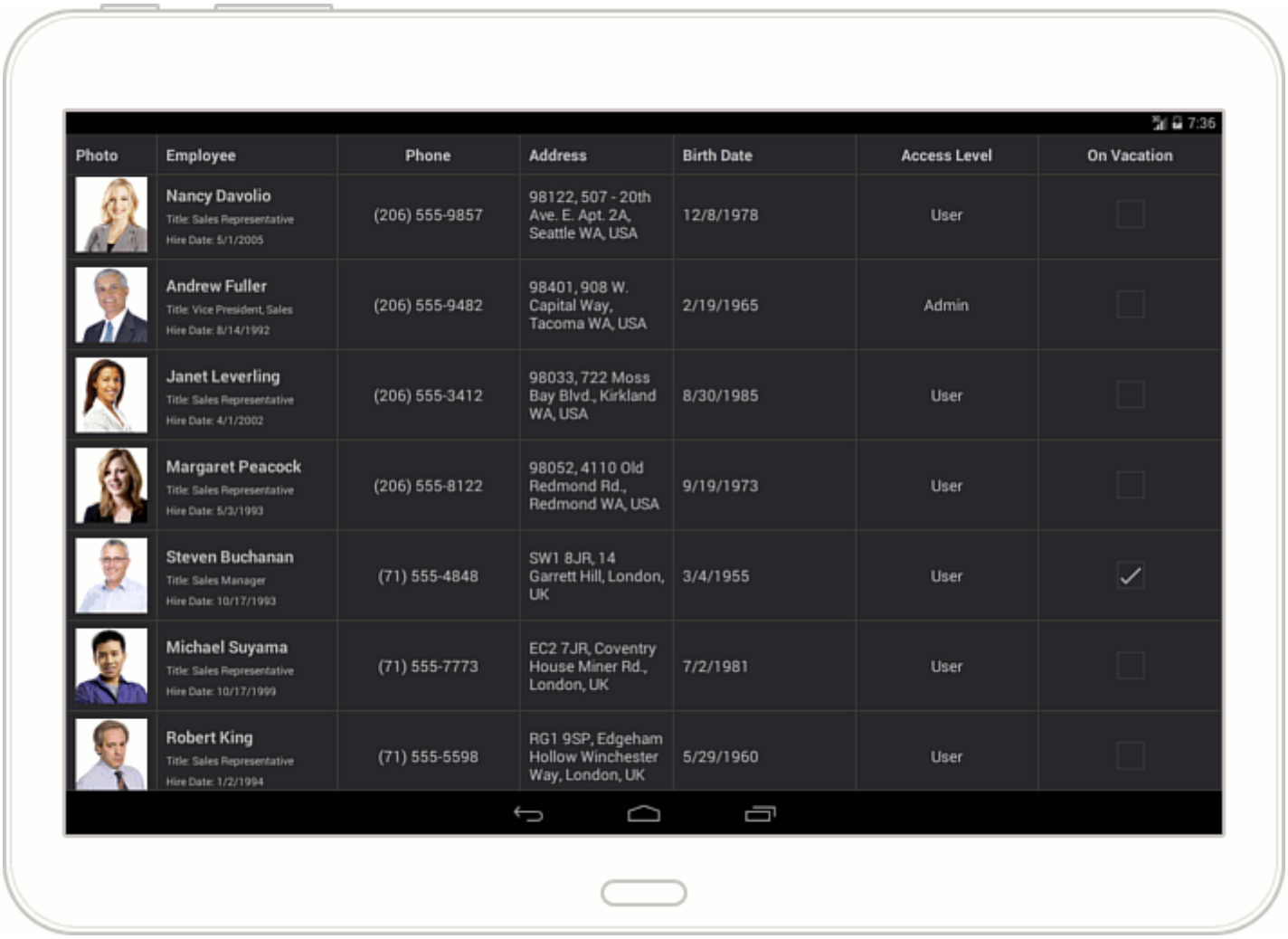
This example shows how to add columns to GridControl to display and edit data of different types. The grid is bound to a collection of **Employee** objects. Each **Employee** object contains an employee's photo (image), name, position, phone, address (strings), hire and birth dates (DateTime values), a value that specifies an employee's access level, and a Boolean value indicating whether an employee is on vacation.

© 2018 Developer Express Inc.




| Photo | Employee | Phone | Address | Birth Date | Access Level | On Vacation |
|---|---|---|---|---|---|---|
| | **Nancy Davolio**<br>Title: Sales Representative<br>Hire Date: 5/1/2005 | (206) 555-9857 | 98122, 507 - 20th Ave. E. Apt. 2A, Seattle WA, USA | 12/8/1978 | User | ☐ |
| | **Andrew Fuller**<br>Title: Vice President, Sales<br>Hire Date: 8/14/1992 | (206) 555-9482 | 98401, 908 W. Capital Way, Tacoma WA, USA | 2/19/1965 | Admin | ☐ |
| | **Janet Leverling**<br>Title: Sales Representative<br>Hire Date: 4/1/2002 | (206) 555-3412 | 98033, 722 Moss Bay Blvd., Kirkland WA, USA | 8/30/1985 | User | ☐ |
| | **Margaret Peacock**<br>Title: Sales Representative<br>Hire Date: 5/3/1993 | (206) 555-8122 | 98052, 4110 Old Redmond Rd., Redmond WA, USA | 9/19/1973 | User | ☐ |
| | **Steven Buchanan**<br>Title: Sales Manager<br>Hire Date: 10/17/1993 | (71) 555-4848 | SW1 8JR, 14 Garrett Hill, London, UK | 3/4/1955 | User | ☑ |
| | **Michael Suyama**<br>Title: Sales Representative<br>Hire Date: 10/17/1999 | (71) 555-7773 | EC2 7JR, Coventry House Miner Rd., London, UK | 7/2/1981 | User | ☐ |
| | **Robert King**<br>Title: Sales Representative<br>Hire Date: 1/2/1994 | (71) 555-5598 | RG1 9SP, Edgeham Hollow Winchester Way, London, UK | 5/29/1960 | User | ☐ |

© 2018 Developer Express Inc.




Grid columns are stored in the GridControl.Columns collection. An individual column is specified by a GridColumn descendant, which is appropriate to the type of data contained in this column. In this example, to display information on employees and allow end-users to edit it, the following columns are created in the grid.

| Grid Column | Description |
|---|---|
| Photo<br>ImageColumn | This column displays employee photos which are images added to a project as embedded resources. |
| Employee<br>TemplateColumn | A cell in this column displays three employee properties: Name, Position and HireDate. A cell appearance is defined via a template. Each cell contains three Xamarin.Forms.Label elements within a Xamarin.Forms.Grid. Each Label element is bound to a property of the **Employee** class.<br><br>A cell template data context is specified by the CellData object. Its Value property provides access to a value of a data field assigned to the column's GridColumn.FieldName property. In this example, a column cell displays not only this field value but also values of two more fields. So, it is required to specify the whole data row as a binding source for the template. To do this, set the BindingContext property of the Grid to Source. Note that BindingContext specified for the Grid is inherited by all its children. This means that all Label elements have the same BindingContext as the Grid, and you can specify their simple bindings to properties of that object (Employee). |

© 2018 Developer Express Inc.

| | |
|---|---|
| | Use the GridColumn.AllowSort, GridColumn.AllowGroup and GridColumn.IsReadOnly properties to prevent end-users from sorting and grouping data in the grid by this column and disable data editing. |
| Phone<br>NumberColumn | The employee's **Phone** property is of the string type. The keyboard allowing text input is automatically displayed when an end-user activates this column cell to edit an employee's phone. |
| Address<br>TextColumn | This column is bound to the employee's **Address** string property. The keyboard allowing text input is automatically displayed when an end-user activates this column cell to edit an employee's address. |
| Birth Date<br>DateColumn | This column displays and allows editing employee birth days. The GridColumn.DisplayFormat property specifies the format of displaying dates. |
| Access Level<br>PickerColumn | The employee's **Access** property is of the **AccessLevel** type which is an enumeration with values **Admin** and **User**. To allow editing an employee's access level by selecting one of two enumeration values, this property is bound to the grid's picker column. |
| On Vacation<br>SwitchColumn | The employee's **OnVacation** property is of the Boolean type, so it is bound to the grid's switch column which allows editing a cell value by selecting between two states. |

**C#**

```
 (Employee.cs)
public enum AccessLevel {
    Admin,
    User
}
public class Employee : ModelObject {
    string name;
    public string Name {
        get { return name; }
        set {
            name = value;
            if (Photo == null)
                Photo = ImageSource.FromResource (value.Replace (" ", String.Empty));
        }
    }
    public Employee(string name) {
        this.Name = name;
    }
    public ImageSource Photo { get; set; }
    public DateTime BirthDate { get; set; }
    public DateTime HireDate { get; set; }
    public string Position { get; set; }
    public string Address { get; set; }
    public string Phone { get; set; }
    public AccessLevel Access { get; set; }
    public bool OnVacation { get; set; }
}
```

**C#**

```
 (EmployeesRepository.cs)
public abstract class EmployeesRepository {
    readonly ObservableCollection<Employee> employees;
    public EmployeesRepository() {
        this.employees = new ObservableCollection<Employee>();
    }
    public ObservableCollection<Employee> Employees {
        get { return employees; }
    }
}
public class TestEmployeesRepository : EmployeesRepository {
    public TestEmployeesRepository() : base() {
        GenerateCustomers();
    }
    void GenerateCustomers() {
        Employees.Add(
            new Employee("Nancy Davolio") {
                BirthDate = new DateTime(1978, 12, 8),
                HireDate = new DateTime(2005, 5, 1),
                Position = "Sales Representative",
                Address = "98122, 507 - 20th Ave. E. Apt. 2A, Seattle WA, USA",
                Phone = "(206) 555-9857",
                Access = AccessLevel.User,
                OnVacation = false
            }
        );
        Employees.Add(
            new Employee("Andrew Fuller") {
                BirthDate = new DateTime(1965, 2, 19),
                HireDate = new DateTime(1992, 8, 14),
                Position = "Vice President, Sales",
                Address = "98401, 908 W. Capital Way, Tacoma WA, USA",
                Phone = "(206) 555-9482",
                Access = AccessLevel.Admin,
                OnVacation = false
            }
        );
        Employees.Add(
            new Employee("Janet Leverling") {
                BirthDate = new DateTime(1985, 8, 30),
                HireDate = new DateTime(2002, 4, 1),
                Position = "Sales Representative",
                Address = "98033, 722 Moss Bay Blvd., Kirkland WA, USA",
                Phone = "(206) 555-3412",
                Access = AccessLevel.User,
                OnVacation = false
            }
        );
        Employees.Add(
            new Employee("Margaret Peacock") {
                BirthDate = new DateTime(1973, 9, 19),
                HireDate = new DateTime(1993, 5, 3),
                Position = "Sales Representative",
                Address = "98052, 4110 Old Redmond Rd., Redmond WA, USA",
                Phone = "(206) 555-8122",
                Access = AccessLevel.User,
                OnVacation = false
            }
        );
        Employees.Add(
            new Employee("Steven Buchanan") {
                BirthDate = new DateTime(1955, 3, 4),
                HireDate = new DateTime(1993, 10, 17),
                Position = "Sales Manager",
                Address = "SW1 8JR, 14 Garrett Hill, London, UK",
```

© 2018 Developer Express Inc.

```
                Phone = "(71) 555-4848",
                Access = AccessLevel.User,
                OnVacation = true
            }
        );
        Employees.Add(
            new Employee("Michael Suyama") {
                BirthDate = new DateTime(1981, 7, 2),
                HireDate = new DateTime(1999, 10, 17),
                Position = "Sales Representative",
                Address = "EC2 7JR, Coventry House Miner Rd., London, UK",
                Phone = "(71) 555-7773",
                Access = AccessLevel.User,
                OnVacation = false
            }
        );
        Employees.Add(
            new Employee("Robert King") {
                BirthDate = new DateTime(1960, 5, 29),
                HireDate = new DateTime(1994, 1, 2),
                Position = "Sales Representative",
                Address = "RG1 9SP, Edgeham Hollow Winchester Way, London, UK",
                Phone = "(71) 555-5598",
                Access = AccessLevel.User,
                OnVacation = false
            }
        );
        Employees.Add(
            new Employee("Laura Callahan") {
                BirthDate = new DateTime(1985, 1, 9),
                HireDate = new DateTime(2004, 3, 5),
                Position = "Inside Sales Coordinator",
                Address = "98105, 4726 - 11th Ave. N.E., Seattle WA, USA",
                Phone = "(206) 555-1189",
                Access = AccessLevel.User,
                OnVacation = true
            }
        );
        Employees.Add(
            new Employee("Anne Dodsworth") {
                BirthDate = new DateTime(1980, 1, 27),
                HireDate = new DateTime(2004, 11, 15),
                Position = "Sales Representative",
                Address = "WG2 7LT, 7 Houndstooth Rd., London, UK",
                Phone = "(71) 555-4444",
                Access = AccessLevel.User,
                OnVacation = false
            }
        );
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl ItemsSource="{Binding Employees}" RowHeight="100">
    <dxGrid:GridControl.Columns>
        <dxGrid:ImageColumn FieldName="Photo" Width="100" />
        <dxGrid:TemplateColumn FieldName="Name" Caption="Employee" Width="
                               IsReadOnly="true" AllowGroup="False" AllowS
          <dxGrid:TemplateColumn.DisplayTemplate>
            <DataTemplate>
              <Grid BindingContext="{Binding Source}" Padding="0,15,0,15">
                <Grid.RowDefinitions>
                  <RowDefinition Height="Auto" />
                  <RowDefinition Height="Auto" />
                  <RowDefinition Height="Auto" />
                </Grid.RowDefinitions>
                <Label Text="{Binding Name}" Font="Bold, 18" Grid.Row="0"
                <Label Text="{Binding Position, StringFormat = 'Title: {0}
                <Label Text="{Binding HireDate, StringFormat = 'Hire Date:
              </Grid>
            </DataTemplate>
          </dxGrid:TemplateColumn.DisplayTemplate>
        </dxGrid:TemplateColumn>
        <dxGrid:NumberColumn FieldName="Phone" ContentAlignment="Center"/>
        <dxGrid:TextColumn FieldName="Address" Width="170"/>
        <dxGrid:DateColumn FieldName="BirthDate" DisplayFormat="d"/>
        <dxGrid:PickerColumn FieldName="Access" Caption="Access Level" Con
        <dxGrid:SwitchColumn FieldName="OnVacation"/>
    </dxGrid:GridControl.Columns>
</dxGrid:GridControl>
```

## See Also

CellData Class
CellData Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# Value Property

Gets or sets an object that specifies a data field value.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public object Value { get; set; }
```

### Property Value

An object in a data source that specifies a data field value.

### Remarks

The CellData object serves as a binding source for a cell template defined for a TemplateColumn column. The **Value** property provides access to an object that specifies the value of a data field assigned to the column's GridColumn.FieldName property. If a cell for which a template is defined has bindings to multiple properties of a data row object, set the entire row as the binding source for a cell template using the Source property.

### See Also

CellData Class
CellData Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# CellIndex Structure

Stores the information on a cell location within the grid.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public struct CellIndex
```

### Remarks

The **CellIndex** structure introduces the RowHandle and FieldName properties that define a row and column containing a cell.

### Inheritance Hierarchy

System.Object
  System.ValueType
    **CellIndex**

### See Also

CellIndex Members
DevExpress.Mobile.DataGrid Namespace
Identify and Access Rows
Identify and Access Columns
Cells

© 2018 Developer Express Inc.

# CellIndex Members

Stores the information on a cell location within the grid.

The following tables list the members exposed by the CellIndex type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | CellIndex | Initializes a new instance of the CellIndex class. |

### Public Properties

| | Name | Description |
|---|---|---|
| | FieldName | Gets the field name of the column that contains the cell. |
| S | InvalidIndex | Gets an invalid cell index. |
| | IsValid | Indicates whether the cell index is valid. |
| | RowHandle | Gets the handle of the row that contains the cell. |

### Public Methods

| | Name | Description |
|---|---|---|
| | Equals | Overloaded. Determines whether the specified object is equal to the current CellIndex instance. |
| S | Equals | (Inherited from **System.Object**) |
| | GetHashCode | Gets the hash code (a number) that corresponds to the value of the current CellIndex object. |
| | GetType | (Inherited from **System.Object**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.ValueType**) |

**See Also**
CellIndex Members
DevExpress.Mobile.DataGrid Namespace
Identify and Access Rows
Identify and Access Columns
Cells

© 2018 Developer Express Inc.

# CellIndex Constructor

Initializes a new instance of the CellIndex class.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public CellIndex(
   int rowHandle,
   string fieldName
)
```

### Parameters

*rowHandle*
An integer value that specifies the handle of the row containing the cell.
*fieldName*
A string value that specifies the field name of a column containing the cell.

### See Also

CellIndex Structure
CellIndex Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# CellIndex Properties

Stores the information on a cell location within the grid.

The following tables list the members exposed by the CellIndex type.

**Public Constructors**

| | Name | Description |
|---|---|---|
| | CellIndex | Initializes a new instance of the CellIndex class. |

**Public Properties**

| | Name | Description |
|---|---|---|
| | FieldName | Gets the field name of the column that contains the cell. |
| S | InvalidIndex | Gets an invalid cell index. |
| | IsValid | Indicates whether the cell index is valid. |
| | RowHandle | Gets the handle of the row that contains the cell. |

**Public Methods**

| | Name | Description |
|---|---|---|
| | Equals | Overloaded. Determines whether the specified object is equal to the current CellIndex instance. |
| S | Equals | (Inherited from **System.Object**) |
| | GetHashCode | Gets the hash code (a number) that corresponds to the value of the current CellIndex object. |
| | GetType | (Inherited from **System.Object**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.ValueType**) |

**See Also**
CellIndex Members
DevExpress.Mobile.DataGrid Namespace
Identify and Access Rows
Identify and Access Columns
Cells

© 2018 Developer Express Inc.

# FieldName Property

Gets the field name of the column that contains the cell.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public string FieldName { get; }
```

### Property Value

A string value that specifies the field name of the column containing the cell.

### See Also

CellIndex Structure
CellIndex Members
DevExpress.Mobile.DataGrid Namespace
Columns
Identify and Access Columns

© 2018 Developer Express Inc.

# InvalidIndex Property

Gets an invalid cell index.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public static CellIndex InvalidIndex { get; }
```

### Property Value

A CellIndex object with the RowHandle property set to the GridControl.InvalidRowHandle value and the FieldName property set to an empty string.

### Remarks

An invalid cell index is returned when an attempt to obtain a particular cell fails.

### See Also

CellIndex Structure
CellIndex Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# IsValid Property

Indicates whether the cell index is valid.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public bool IsValid { get; }
```

### Property Value

**true**, if the current cell index is not equal to InvalidIndex; otherwise, **false**.

### See Also

CellIndex Structure
CellIndex Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# RowHandle Property

Gets the handle of the row that contains the cell.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public int RowHandle { get; }
```

### Property Value

An integer value identifying the row that contains the cell.

### See Also

CellIndex Structure
CellIndex Members
DevExpress.Mobile.DataGrid Namespace
Identify and Access Rows

© 2018 Developer Express Inc.

# CellIndex Methods

Stores the information on a cell location within the grid.

The following tables list the members exposed by the CellIndex type.

**Public Methods**

| | Name | Description |
|---|---|---|
| | Equals | Overloaded. Determines whether the specified object is equal to the current CellIndex instance. |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | Gets the hash code (a number) that corresponds to the value of the current CellIndex object. |
| | GetType | (Inherited from **System.Object**) |
| | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.ValueType**) |

**See Also**
CellIndex Members
DevExpress.Mobile.DataGrid Namespace
Identify and Access Rows
Identify and Access Columns
Cells

© 2018 Developer Express Inc.

# Equals Method

Determines whether the specified object is equal to the current CellIndex instance.

**Overload List**

| Name | Description |
|---|---|
| bool Equals(object obj) | Determines whether the specified object is equal to the current CellIndex instance. |
| static bool Equals(object objA, object objB) | Inherited from **System.Object** |

**See Also**
CellIndex Structure
CellIndex Members
DevExpress.Mobile.DataGrid Namespace
CellIndex.Equals Overload List

Determines whether the specified object is equal to the current CellIndex instance.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**
**C#**
```
public override bool Equals(
   object obj
)
```

**Parameters**
*obj*
The object to compare with the current object.

**Return Value**

**true** if the specified object is equal to the current CellIndex instance; otherwise **false**.

**Remarks**

Two CellIndex objects are considered equal if their RowHandle and FieldName property values are equal.

**See Also**
CellIndex Structure
CellIndex Members
DevExpress.Mobile.DataGrid Namespace
CellIndex.Equals Overload List

© 2018 Developer Express Inc.

# GetHashCode Method

Gets the hash code (a number) that corresponds to the value of the current CellIndex object.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public override int GetHashCode()
```

### Return Value

An integer value representing the hash code for the current object.

### See Also

CellIndex Structure
CellIndex Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# ColorScaleFormatCondition Class

Represents a color scale format condition.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

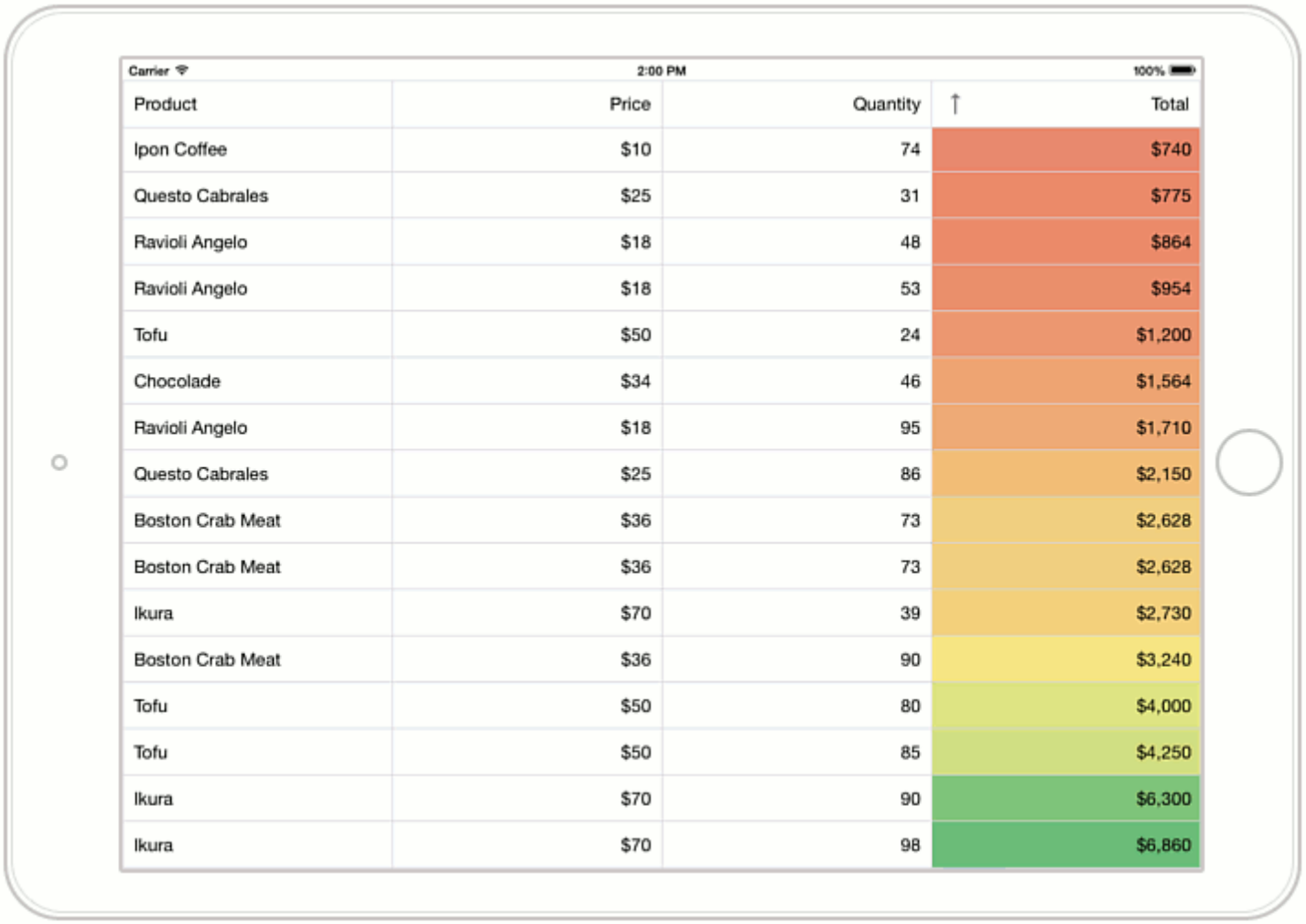
**C#**

```
public class ColorScaleFormatCondition : IndicatorFormatConditionBase
```

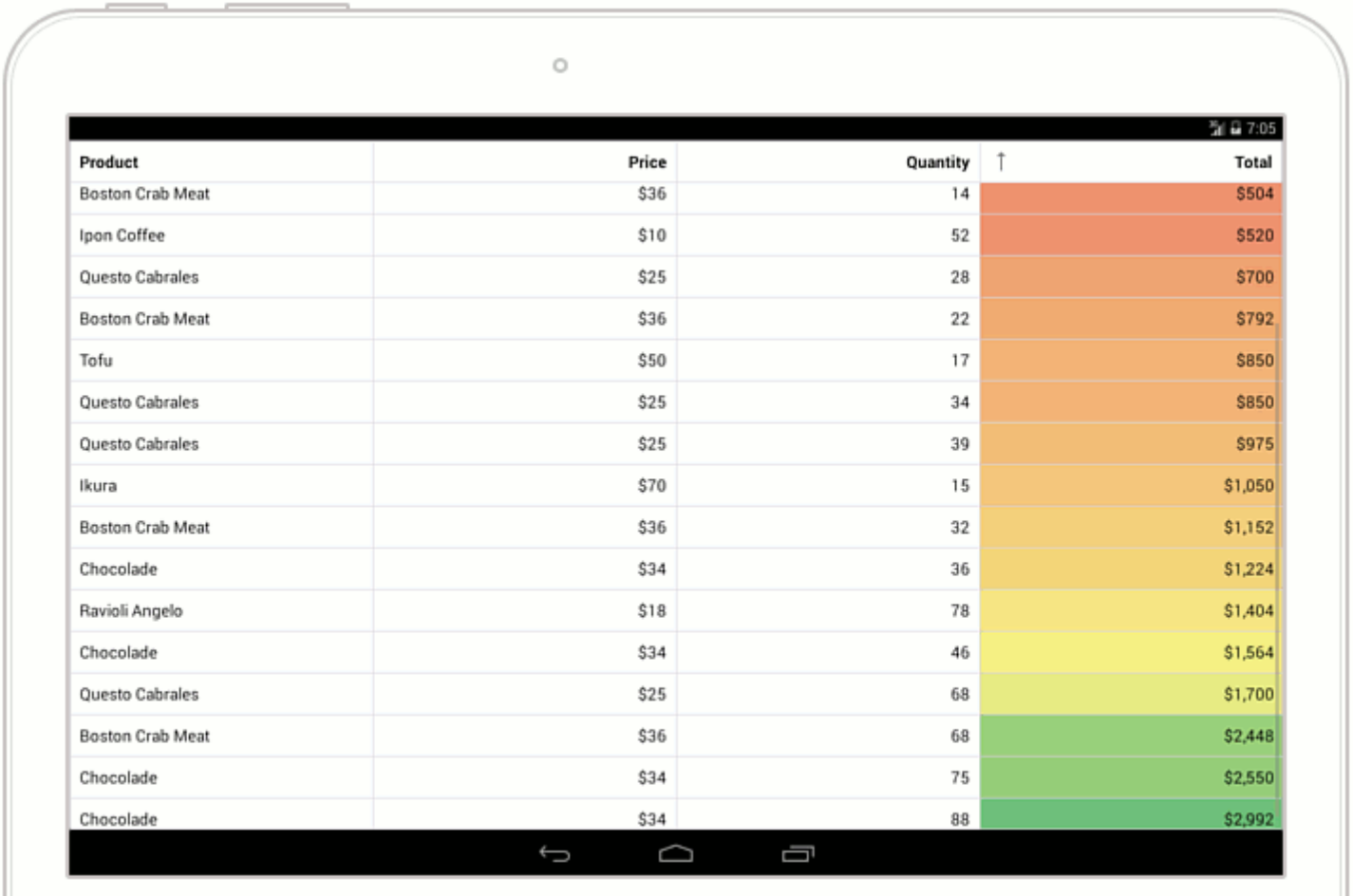
### Example

This example demonstrates how to apply a color scale format to the **Total** column in a GridControl. A color scale format reflects data distribution of column cell values using a gradient of two or three colors. In this example, the predefined **Red-Yellow-Green** color scale is used.

© 2018 Developer Express Inc.




| Product | Price | Quantity | Total |
|---|---|---|---|
| Ipon Coffee | $10 | 74 | $740 |
| Questo Cabrales | $25 | 31 | $775 |
| Ravioli Angelo | $18 | 48 | $864 |
| Ravioli Angelo | $18 | 53 | $954 |
| Tofu | $50 | 24 | $1,200 |
| Chocolade | $34 | 46 | $1,564 |
| Ravioli Angelo | $18 | 95 | $1,710 |
| Questo Cabrales | $25 | 86 | $2,150 |
| Boston Crab Meat | $36 | 73 | $2,628 |
| Boston Crab Meat | $36 | 73 | $2,628 |
| Ikura | $70 | 39 | $2,730 |
| Boston Crab Meat | $36 | 90 | $3,240 |
| Tofu | $50 | 80 | $4,000 |
| Tofu | $50 | 85 | $4,250 |
| Ikura | $70 | 90 | $6,300 |
| Ikura | $70 | 98 | $6,860 |




| Product | Price | Quantity | Total |
|---|---|---|---|
| Boston Crab Meat | $36 | 14 | $504 |
| Ipon Coffee | $10 | 52 | $520 |
| Questo Cabrales | $25 | 28 | $700 |
| Boston Crab Meat | $36 | 22 | $792 |
| Tofu | $50 | 17 | $850 |
| Questo Cabrales | $25 | 34 | $850 |
| Questo Cabrales | $25 | 39 | $975 |
| Ikura | $70 | 15 | $1,050 |
| Boston Crab Meat | $36 | 32 | $1,152 |
| Chocolade | $34 | 36 | $1,224 |
| Ravioli Angelo | $18 | 78 | $1,404 |
| Chocolade | $34 | 46 | $1,564 |
| Questo Cabrales | $25 | 68 | $1,700 |
| Boston Crab Meat | $36 | 68 | $2,448 |
| Chocolade | $34 | 75 | $2,550 |
| Chocolade | $34 | 88 | $2,992 |

1. Create a new ColorScaleFormatCondition class instance.
2. Set this object's FormatConditionBase.FieldName property to **Total**. This column provides values to test against the formatting rule and the specified format will be applied to this column.
3. Assign one of the predefined color scales to the FormatConditionBase.PredefinedFormatName property. All available format names are listed in the Predefined Format Names document.
4. To apply a conditional formatting rule represented by the created **ColorScaleFormatCondition** object, add this object to the GridControl.FormatConditions collection.

**C#**

```
using DevExpress.Mobile.DataGrid;
// ...
ColorScaleFormatCondition condition = new ColorScaleFormatCondition();
condition.FieldName = "Total";
condition.PredefinedFormatName = "RedYellowGreenColorScale";
grid.FormatConditions.Add (condition);
```

### Inheritance Hierarchy

System.Object
  Xamarin.Forms.BindableObject
    FormatConditionBase
      IndicatorFormatConditionBase
        **ColorScaleFormatCondition**

### See Also

ColorScaleFormatCondition Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# ColorScaleFormatCondition Members

Represents a color scale format condition.

The following tables list the members exposed by the ColorScaleFormatCondition type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | ColorScaleFormatCondition | Initializes a new instance of the ColorScaleFormatCondition class with default settings. |

### Public Properties

| | Name | Description |
|---|---|---|
| | BindingContext | (Inherited from **Xamarin.Forms.BindableObject**) |
| | FieldName | Gets or sets the name of the column which provides values to test against the current format rule, and to which the format should be applied. This is a bindable property. (Inherited from **FormatConditionBase**) |
| | Format | Gets or sets the formatting settings applied to cells whose values take part in conditional formatting. This is a bindable property. |
| | MaxValue | Specifies the maximum column value to accept for conditional formatting. This is a bindable property. (Inherited from **IndicatorFormatConditionBase**) |
| | MinValue | Specifies the minimum column value to accept for conditional formatting. This is a bindable property. (Inherited from **IndicatorFormatConditionBase**) |
| | Owner | Gets the format condition's owner. This is a bindable property. (Inherited from **FormatConditionBase**) |
| | PredefinedFormatName | Gets or sets the name of the predefined format to be applied to target cells. This is a bindable property. (Inherited from **FormatConditionBase**) |
| | SelectiveExpression | Specifies a custom rule for applying conditional formatting. This is a bindable property. (Inherited from **IndicatorFormatConditionBase**) |
| | TypeName | Gets or sets the data type name. This property supports the internal infrastructure and is not intended to be used directly from your code. (Inherited from **FormatConditionBase**) |

© 2018 Developer Express Inc.

### Public Methods

| | Name | Description |
|---|---|---|
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ToString | (Inherited from **System.Object**) |

### Public Fields

| | Name | Description |
|---|---|---|
| S | BindingContextProperty | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | FieldNameProperty | Identifies the FormatConditionBase.FieldName bindable property. (Inherited from **FormatConditionBase**) |
| S | FormatProperty | Identifies the Format bindable property. |
| S | MaxValueProperty | Identifies the IndicatorFormatConditionBase.MaxValue bindable property. (Inherited from **IndicatorFormatConditionBase**) |
| S | MinValueProperty | Identifies the IndicatorFormatConditionBase.MinValue bindable property. (Inherited from **IndicatorFormatConditionBase**) |
| S | OwnerProperty | Identifies the FormatConditionBase.Owner bindable property. (Inherited from **FormatConditionBase**) |
| S | PredefinedFormatNameProperty | Identifies the FormatConditionBase.PredefinedFormatName bindable property. (Inherited from **FormatConditionBase**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | SelectiveExpressionProperty | Identifies the IndicatorFormatConditionBase.SelectiveExpression bindable property. (Inherited from **IndicatorFormatConditionBase**) |

**Public Events**

| | Name | Description |
|---|---|---|
| | BindingContextChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanging | (Inherited from **Xamarin.Forms.BindableObject**) |

**See Also**
ColorScaleFormatCondition Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# ColorScaleFormatCondition Constructor

Initializes a new instance of the ColorScaleFormatCondition class with default settings.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**
**C#**
```
public ColorScaleFormatCondition()
```

**See Also**
ColorScaleFormatCondition Class
ColorScaleFormatCondition Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# ColorScaleFormatCondition Properties

Represents a color scale format condition.

The following tables list the members exposed by the ColorScaleFormatCondition type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | ColorScaleFormatCondition | Initializes a new instance of the ColorScaleFormatCondition class with default settings. |

### Public Properties

| | Name | Description |
|---|---|---|
| | BindingContext | (Inherited from **Xamarin.Forms.BindableObject**) |
| | FieldName | Gets or sets the name of the column which provides values to test against the current format rule, and to which the format should be applied. This is a bindable property. (Inherited from **FormatConditionBase**) |
| | Format | Gets or sets the formatting settings applied to cells whose values take part in conditional formatting. This is a bindable property. |
| | MaxValue | Specifies the maximum column value to accept for conditional formatting. This is a bindable property. (Inherited from **IndicatorFormatConditionBase**) |
| | MinValue | Specifies the minimum column value to accept for conditional formatting. This is a bindable property. (Inherited from **IndicatorFormatConditionBase**) |
| | Owner | Gets the format condition's owner. This is a bindable property. (Inherited from **FormatConditionBase**) |
| | PredefinedFormatName | Gets or sets the name of the predefined format to be applied to target cells. This is a bindable property. (Inherited from **FormatConditionBase**) |
| | SelectiveExpression | Specifies a custom rule for applying conditional formatting. This is a bindable property. (Inherited from **IndicatorFormatConditionBase**) |
| | TypeName | Gets or sets the data type name. This property supports the internal infrastructure and is not intended to be used directly from your code. (Inherited from **FormatConditionBase**) |

© 2018 Developer Express Inc.

### Public Methods

| | Name | Description |
|---|---|---|
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ToString | (Inherited from **System.Object**) |

### Public Fields

| | Name | Description |
|---|---|---|
| S | BindingContextProperty | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | FieldNameProperty | Identifies the FormatConditionBase.FieldName bindable property. (Inherited from **FormatConditionBase**) |
| S | FormatProperty | Identifies the Format bindable property. |
| S | MaxValueProperty | Identifies the IndicatorFormatConditionBase.MaxValue bindable property. (Inherited from **IndicatorFormatConditionBase**) |
| S | MinValueProperty | Identifies the IndicatorFormatConditionBase.MinValue bindable property. (Inherited from **IndicatorFormatConditionBase**) |
| S | OwnerProperty | Identifies the FormatConditionBase.Owner bindable property. (Inherited from **FormatConditionBase**) |
| S | PredefinedFormatNameProperty | Identifies the FormatConditionBase.PredefinedFormatName bindable property. (Inherited from **FormatConditionBase**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | SelectiveExpressionProperty | Identifies the IndicatorFormatConditionBase.SelectiveExpression bindable property. (Inherited from **IndicatorFormatConditionBase**) |

**Public Events**

| | Name | Description |
|---|---|---|
| | BindingContextChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanging | (Inherited from **Xamarin.Forms.BindableObject**) |

**See Also**
ColorScaleFormatCondition Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# Format Property

Gets or sets the formatting settings applied to cells whose values take part in conditional formatting. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public ColorScaleFormat Format { get; set; }
```

### Property Value

A **ColorScaleFormat** object that provides formatting settings.

### See Also

ColorScaleFormatCondition Class
ColorScaleFormatCondition Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# ColumnContentAlignment Enumeration

Specifies the content alignment within the column cells.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**
**C#**
```
public enum ColumnContentAlignment
```

**Members**

| Name | Description |
|---|---|
| Center | Indicates that the column cells content is aligned at the center. |
| End | Indicates that the column cells content is aligned at the end. |
| Start | Indicates that the column cells content is aligned at the start. |

**See Also**
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# CustomizeCellDisplayTextEventArgs Class

Provides data for the GridControl.CustomizeCellDisplayText event.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public class CustomizeCellDisplayTextEventArgs : EventArgs
```

### Remarks

The GridControl.CustomizeCellDisplayText event is raised for each cell displayed within a grid. It allows you to display custom text within any cell. The CustomizeCellDisplayTextEventArgs class provides the Index property that identifies the processed cell. The cell's display text can be specified using the DisplayText property.

### Inheritance Hierarchy

System.Object
  System.EventArgs
    **CustomizeCellDisplayTextEventArgs**

### See Also

CustomizeCellDisplayTextEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# CustomizeCellDisplayTextEventArgs Members

Provides data for the GridControl.CustomizeCellDisplayText event.

The following tables list the members exposed by the CustomizeCellDisplayTextEventArgs type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | CustomizeCellDisplayTextEventArgs | Initializes a new instance of the CustomizeCellDisplayTextEventArgs class. |

### Public Properties

| | Name | Description |
|---|---|---|
| | DisplayText | Specifies the display text for the cell currently being processed. |
| | Index | Gets the processed cell's index. |
| | Source | Gets the grid control that raised the event. |
| | Value | Gets the processed cell's value. |

### Public Methods

| | Name | Description |
|---|---|---|
| | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

### Public Fields

| | Name | Description |
|---|---|---|
| | Empty | (Inherited from **System.EventArgs**) |

**See Also**
CustomizeCellDisplayTextEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# CustomizeCellDisplayTextEventArgs Constructor

Initializes a new instance of the CustomizeCellDisplayTextEventArgs class.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public CustomizeCellDisplayTextEventArgs()
```

### Remarks

Instances of the CustomizeCellDisplayTextEventArgs class are automatically created, initialized and passed to the corresponding event handlers.

### See Also

CustomizeCellDisplayTextEventArgs Class
CustomizeCellDisplayTextEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# CustomizeCellDisplayTextEventArgs Properties

Provides data for the GridControl.CustomizeCellDisplayText event.

The following tables list the members exposed by the CustomizeCellDisplayTextEventArgs type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | CustomizeCellDisplayTextEventArgs | Initializes a new instance of the CustomizeCellDisplayTextEventArgs class. |

### Public Properties

| | Name | Description |
|---|---|---|
| | DisplayText | Specifies the display text for the cell currently being processed. |
| | Index | Gets the processed cell's index. |
| | Source | Gets the grid control that raised the event. |
| | Value | Gets the processed cell's value. |

### Public Methods

| | Name | Description |
|---|---|---|
| S | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

### Public Fields

| | Name | Description |
|---|---|---|
| S | Empty | (Inherited from **System.EventArgs**) |

**See Also**
CustomizeCellDisplayTextEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# DisplayText Property

Specifies the display text for the cell currently being processed.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public string DisplayText { get; set; }
```

### Property Value

A string value that specifies the cell's display text.

### Remarks

Initially, when handling the GridControl.CustomizeCellDisplayText event, the **DisplayText** property contains the text currently displayed within the processed cell. To replace the default text with the custom text, assign it to the **DisplayText** property.

### See Also

CustomizeCellDisplayTextEventArgs Class
CustomizeCellDisplayTextEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# Index Property

Gets the processed cell's index.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public CellIndex Index { get; }
```

### Property Value

A CellIndex object that represents the index of the processed cell.

### See Also

CustomizeCellDisplayTextEventArgs Class
CustomizeCellDisplayTextEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# Source Property

Gets the grid control that raised the event.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public GridControl Source { get; }
```

### Property Value

A GridControl object that raised the event.

### See Also

CustomizeCellDisplayTextEventArgs Class
CustomizeCellDisplayTextEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# Value Property

Gets the processed cell's value.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public object Value { get; }
```

### Property Value

An object that represents the value of the processed cell.

### See Also

CustomizeCellDisplayTextEventArgs Class
CustomizeCellDisplayTextEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# CustomizeCellDisplayTextEventHandler Delegate

Represents a method that will handle the GridControl.CustomizeCellDisplayText event.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public delegate void CustomizeCellDisplayTextEventHandler(
   CustomizeCellDisplayTextEventArgs e
)
```

### Parameters

The declaration of your event handler must have the same parameters as the **CustomizeCellDisplayTextEventHandler** delegate declaration.

*e*
A CustomizeCellDisplayTextEventArgs object that contains event data.

### See Also

DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# CustomizeCellEventArgs Class

Provides data for the GridControl.CustomizeCell event.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public class CustomizeCellEventArgs : EventArgs
```

### Remarks

The GridControl.CalculateCustomSummary event allows you to customize the appearance of cells in the grid. When this event is raised, the **CustomizeCellEventArgs** object with appropriate settings is automatically created and passed to the event handler. Use this object's properties to identify a column and row of the cell currently being processed (FieldName, RowHandle) and set cell colors (BackgroundColor, ForeColor).

The Handled property specifies whether default painting is required after the event handler has been executed. If this property is set to **true** in a handler, then the default painting is canceled. Otherwise, any custom painting performed in an event handler will be overridden by the GridControl's standard painting.

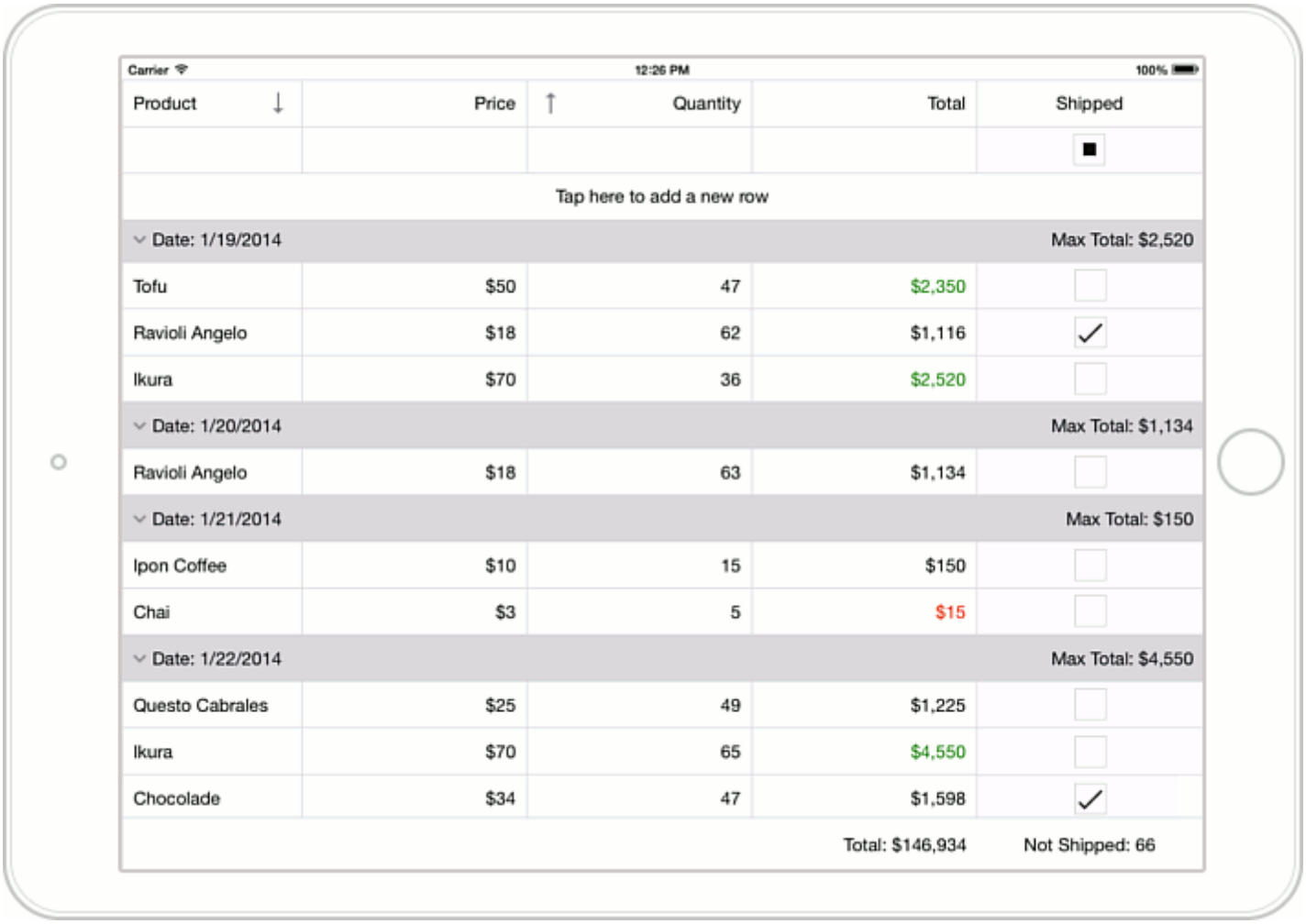
### Example

| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T178764. | |

This example demonstrates how to customize the appearance of cells in the specified grid column depending on cell values by handling the GridControl.CustomizeCell event.

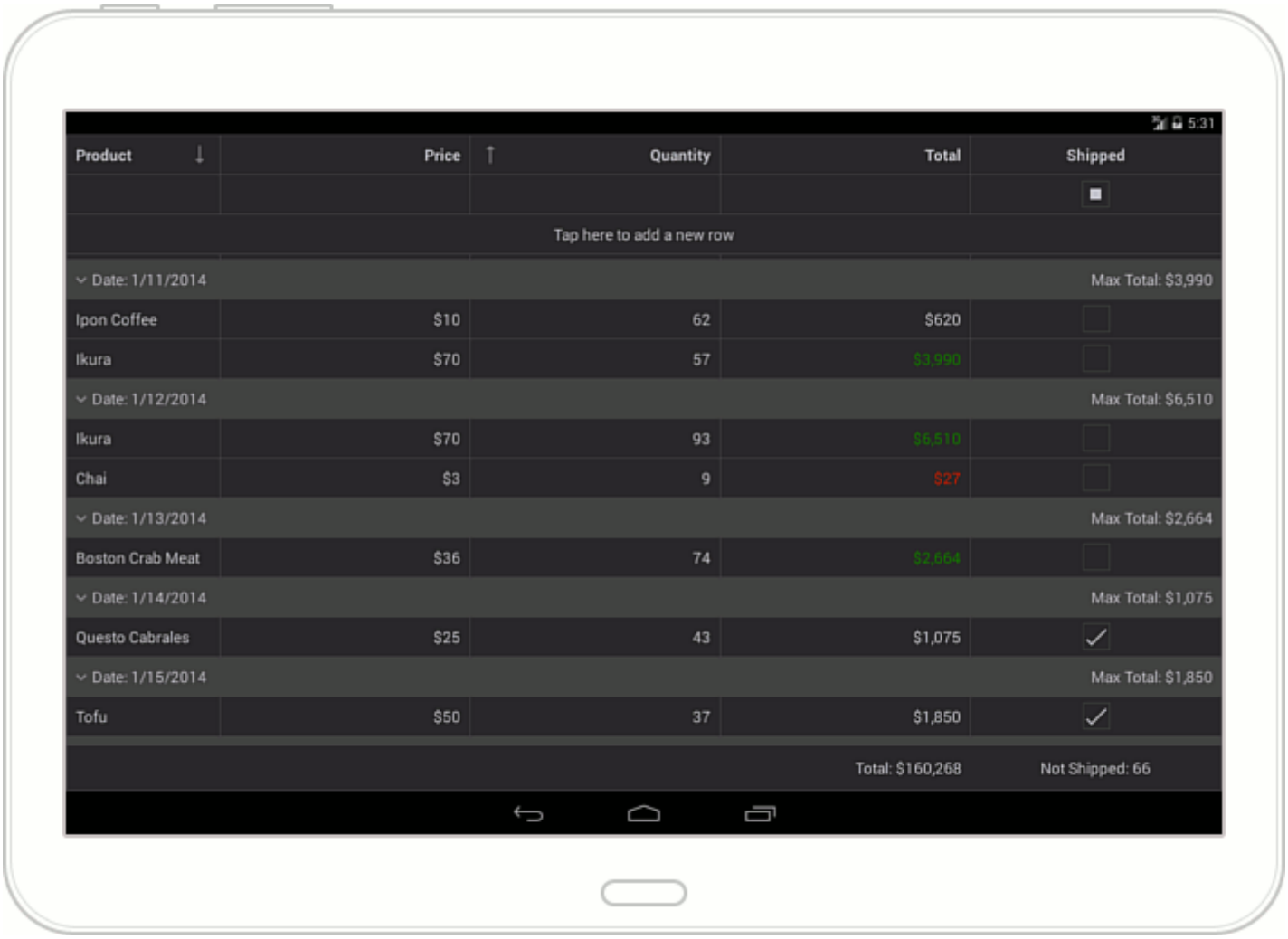
The image below illustrates the result.

© 2018 Developer Express Inc.




| Product | Price | Quantity | Total | Shipped |
|---|---|---|---|---|
| | | | | ■ |
| Tap here to add a new row | | | | |
| Date: 1/19/2014 | | | | Max Total: $2,520 |
| Tofu | $50 | 47 | $2,350 | ☐ |
| Ravioli Angelo | $18 | 62 | $1,116 | ☑ |
| Ikura | $70 | 36 | $2,520 | ☐ |
| Date: 1/20/2014 | | | | Max Total: $1,134 |
| Ravioli Angelo | $18 | 63 | $1,134 | ☐ |
| Date: 1/21/2014 | | | | Max Total: $150 |
| Ipon Coffee | $10 | 15 | $150 | ☐ |
| Chai | $3 | 5 | $15 | ☐ |
| Date: 1/22/2014 | | | | Max Total: $4,550 |
| Questo Cabrales | $25 | 49 | $1,225 | ☐ |
| Ikura | $70 | 65 | $4,550 | ☐ |
| Chocolade | $34 | 47 | $1,598 | ☑ |
| | | | Total: $146,934 | Not Shipped: 66 |

© 2018 Developer Express Inc.




**C#**

```
 (MainPage.xaml.cs)
void OnCustomizeCell(CustomizeCellEventArgs e)
{
    if (e.FieldName == "Total" && !e.IsSelected)
    {
        int total = Convert.ToInt32(e.Value);
        if (total < 50)
            e.ForeColor = Color.Red;
        else if (total > 2000)
            e.ForeColor = Color.Green;
        e.Handled = true;
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl x:Name="grid" ItemsSource="{Binding Orders}"
                    NewItemRowVisibility="true"
                    CalculateCustomSummary="OnCalculateCustomSummary"
                    SortMode="Multiple" AutoFilterPanelVisibility="true"
                    CustomizeCell="OnCustomizeCell"
                    SwipeButtonShowing="OnSwipeButtonShowing"
                    SwipeButtonClick="OnSwipeButtonClick">
```

## Inheritance Hierarchy

© 2018 Developer Express Inc.

System.Object
    System.EventArgs
        **CustomizeCellEventArgs**

## See Also

CustomizeCellEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# CustomizeCellEventArgs Members

Provides data for the GridControl.CustomizeCell event.

The following tables list the members exposed by the CustomizeCellEventArgs type.

### Public Properties

| | Name | Description |
|---|---|---|
| | BackgroundColor | Gets or sets the cell background color. |
| | DisplayText | Gets or sets the text displayed in the cell. |
| | FieldName | Gets the field name of the column that contains the cell. |
| | ForeColor | Gets or sets the color used to paint the cell value. |
| | Handled | Gets or sets whether the GridControl.CustomizeCell event was handled. If it was handled, the default cell painting is not required. |
| | IsSelected | Gets whether the processed cell belongs to a row that is currently selected. |
| | RowHandle | Gets the handle of the row that contains the processed cell. |
| | Value | Gets the processed cell value. |

### Public Methods

| | Name | Description |
|---|---|---|
| S | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

### Public Fields

| | Name | Description |
|---|---|---|
| S | Empty | (Inherited from **System.EventArgs**) |

**See Also**

CustomizeCellEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# CustomizeCellEventArgs Properties

Provides data for the GridControl.CustomizeCell event.

The following tables list the members exposed by the CustomizeCellEventArgs type.

**Public Properties**

| | Name | Description |
|---|---|---|
| | BackgroundColor | Gets or sets the cell background color. |
| | DisplayText | Gets or sets the text displayed in the cell. |
| | FieldName | Gets the field name of the column that contains the cell. |
| | ForeColor | Gets or sets the color used to paint the cell value. |
| | Handled | Gets or sets whether the GridControl.CustomizeCell event was handled. If it was handled, the default cell painting is not required. |
| | IsSelected | Gets whether the processed cell belongs to a row that is currently selected. |
| | RowHandle | Gets the handle of the row that contains the processed cell. |
| | Value | Gets the processed cell value. |

**Public Methods**

| | Name | Description |
|---|---|---|
| S | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

**Public Fields**

| | Name | Description |
|---|---|---|
| S | Empty | (Inherited from **System.EventArgs**) |

**See Also**

CustomizeCellEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# BackgroundColor Property

Gets or sets the cell background color.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public Color BackgroundColor { get; set; }
```

### Property Value

A Xamarin.Forms.Color object specifying the cell background color.

### See Also

CustomizeCellEventArgs Class
CustomizeCellEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# DisplayText Property

Gets or sets the text displayed in the cell.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public string DisplayText { get; }
```

### Property Value

A string that is the cell's current display text.

### Remarks

The cell's edit value can be obtained via the Value property.

### See Also

CustomizeCellEventArgs Class
CustomizeCellEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# FieldName Property

Gets the field name of the column that contains the cell.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public string FieldName { get; }
```

### Property Value

A string value that specifies the field name of the column that contains the cell to be painted.

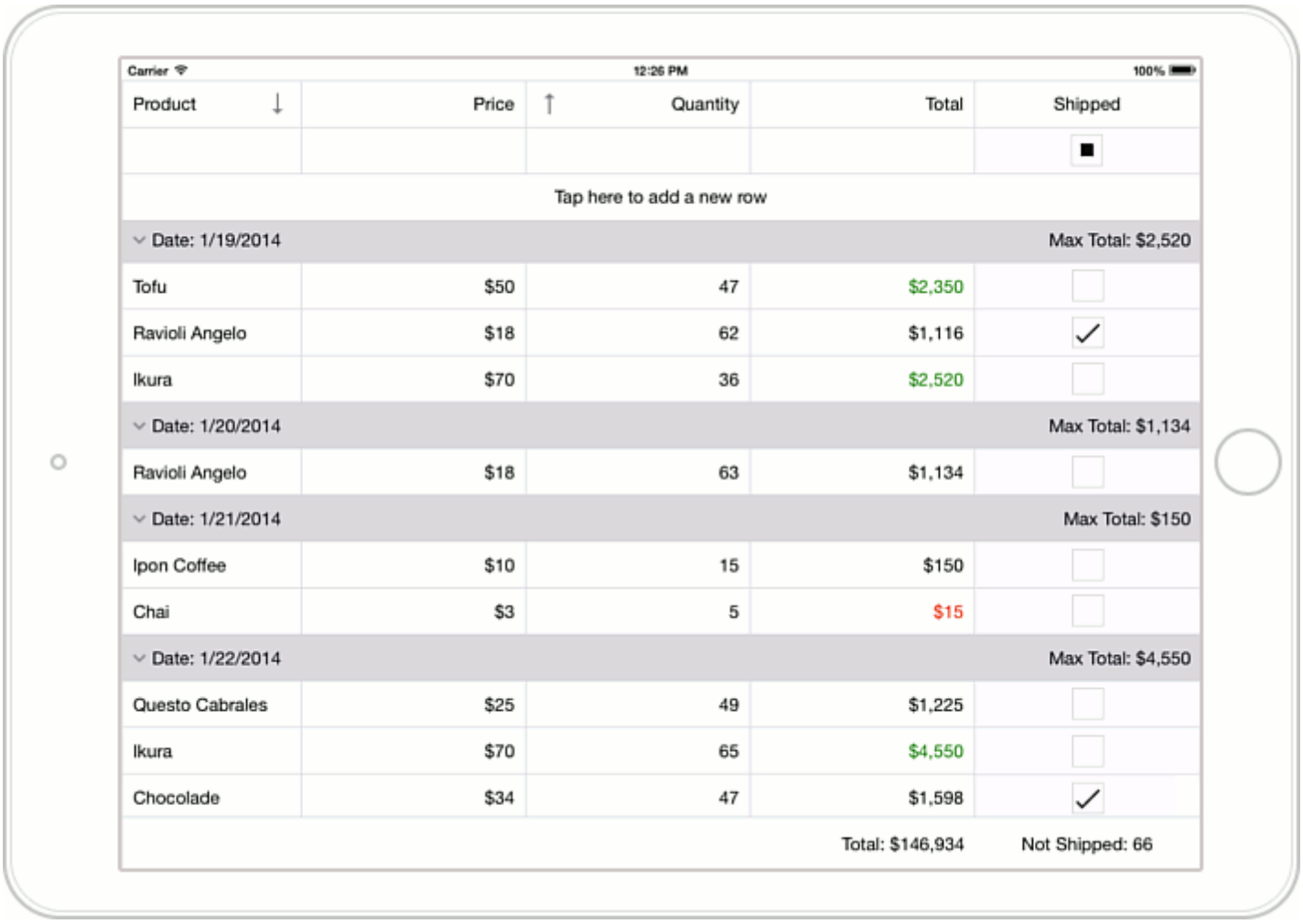
### Example

**Show Me**

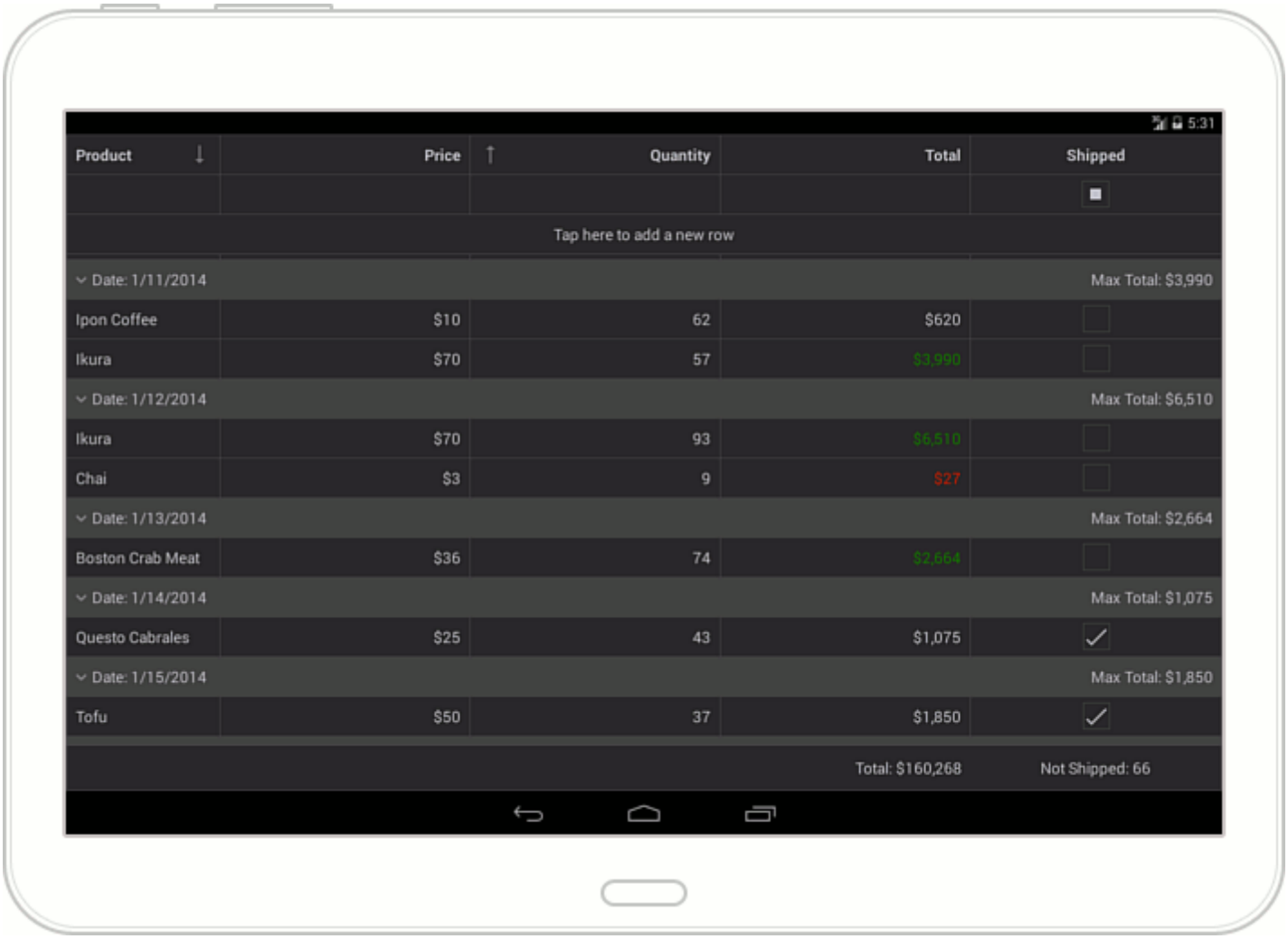
A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T178764.

This example demonstrates how to customize the appearance of cells in the specified grid column depending on cell values by handling the GridControl.CustomizeCell event.

The image below illustrates the result.




© 2018 Developer Express Inc.




**C#**

```
 (MainPage.xaml.cs)
void OnCustomizeCell(CustomizeCellEventArgs e)
{
    if (e.FieldName == "Total" && !e.IsSelected)
    {
        int total = Convert.ToInt32(e.Value);
        if (total < 50)
            e.ForeColor = Color.Red;
        else if (total > 2000)
            e.ForeColor = Color.Green;
        e.Handled = true;
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl x:Name="grid" ItemsSource="{Binding Orders}"
                    NewItemRowVisibility="true"
                    CalculateCustomSummary="OnCalculateCustomSummary"
                    SortMode="Multiple" AutoFilterPanelVisibility="true"
                    CustomizeCell="OnCustomizeCell"
                    SwipeButtonShowing="OnSwipeButtonShowing"
                    SwipeButtonClick="OnSwipeButtonClick">
```

## See Also

© 2018 Developer Express Inc.

CustomizeCellEventArgs Class
CustomizeCellEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# ForeColor Property

Gets or sets the color used to paint the cell value.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public Color ForeColor { get; set; }
```

### Property Value

A Xamarin.Forms.Color object specifying the cell foreground color.

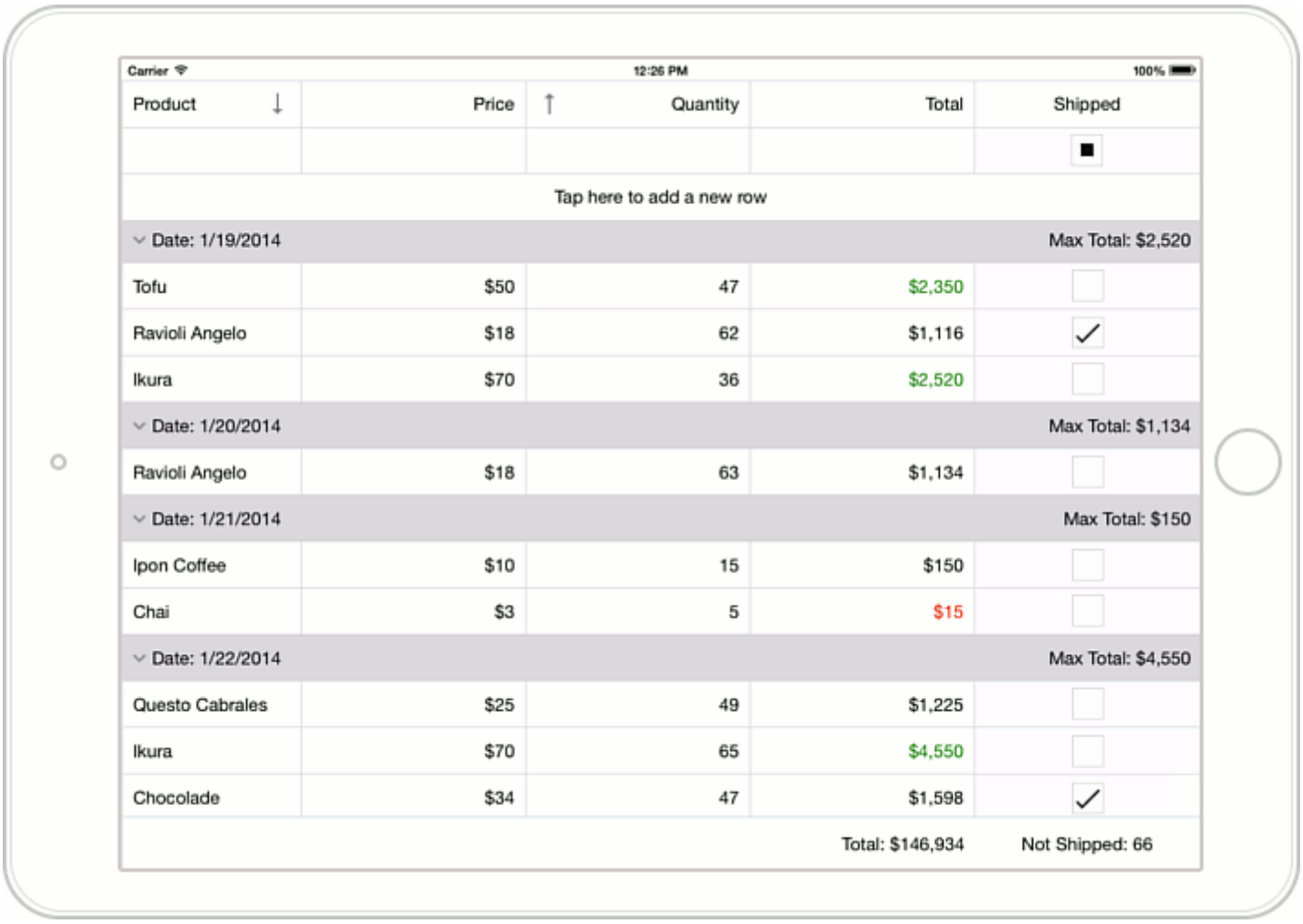
### Example

**Show Me**

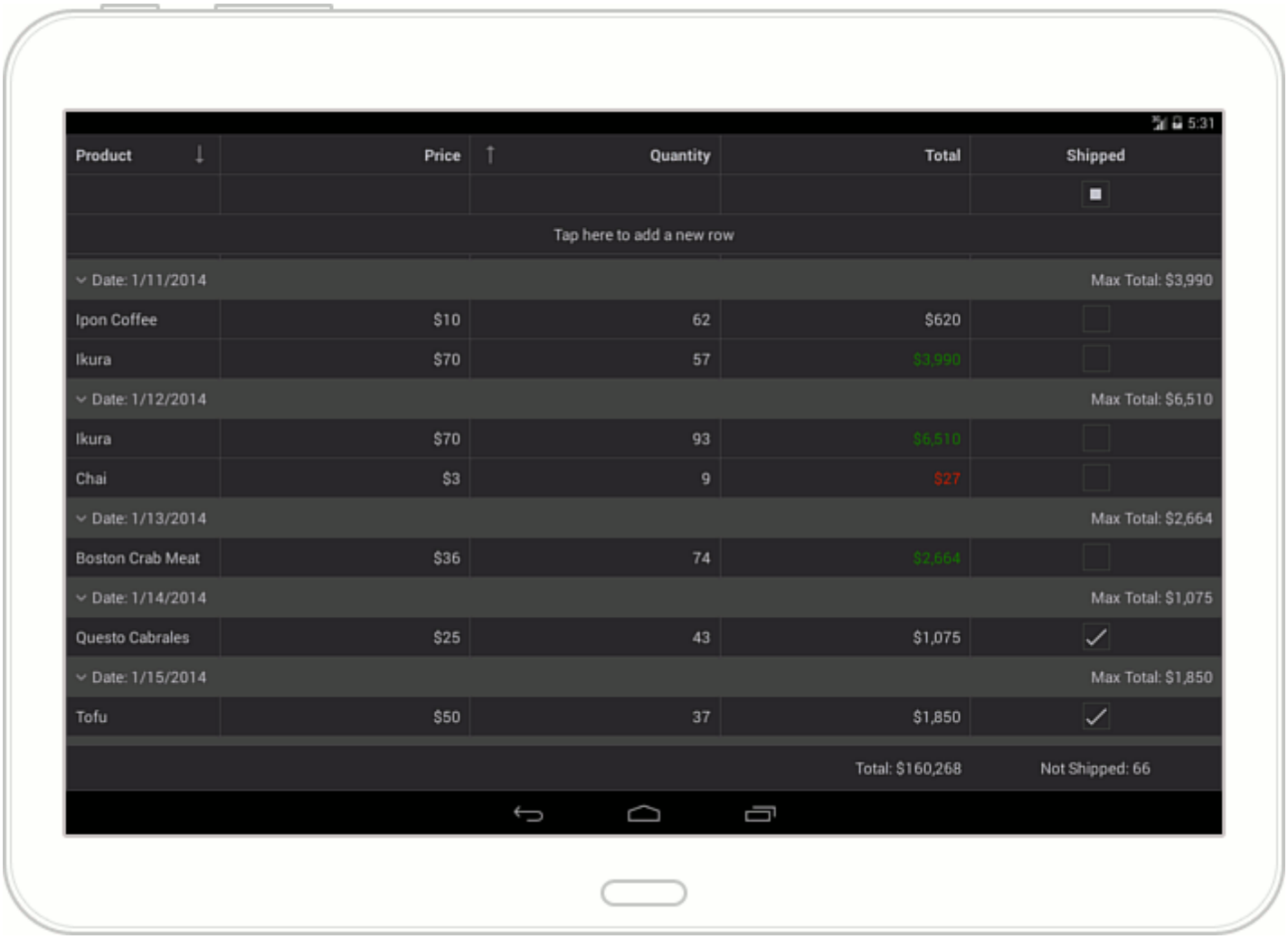
| | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T178764. | |

This example demonstrates how to customize the appearance of cells in the specified grid column depending on cell values by handling the GridControl.CustomizeCell event.

The image below illustrates the result.




© 2018 Developer Express Inc.




**C#**

```
 (MainPage.xaml.cs)
void OnCustomizeCell(CustomizeCellEventArgs e)
{
    if (e.FieldName == "Total" && !e.IsSelected)
    {
        int total = Convert.ToInt32(e.Value);
        if (total < 50)
            e.ForeColor = Color.Red;
        else if (total > 2000)
            e.ForeColor = Color.Green;
        e.Handled = true;
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl x:Name="grid" ItemsSource="{Binding Orders}"
                    NewItemRowVisibility="true"
                    CalculateCustomSummary="OnCalculateCustomSummary"
                    SortMode="Multiple" AutoFilterPanelVisibility="true"
                    CustomizeCell="OnCustomizeCell"
                    SwipeButtonShowing="OnSwipeButtonShowing"
                    SwipeButtonClick="OnSwipeButtonClick">
```

## See Also

© 2018 Developer Express Inc.

CustomizeCellEventArgs Class
CustomizeCellEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# Handled Property

Gets or sets whether the GridControl.CustomizeCell event was handled. If it was handled, the default cell painting is not required.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public bool Handled { get; set; }
```

### Property Value

**true** if default cell painting is not required; otherwise, **false**.

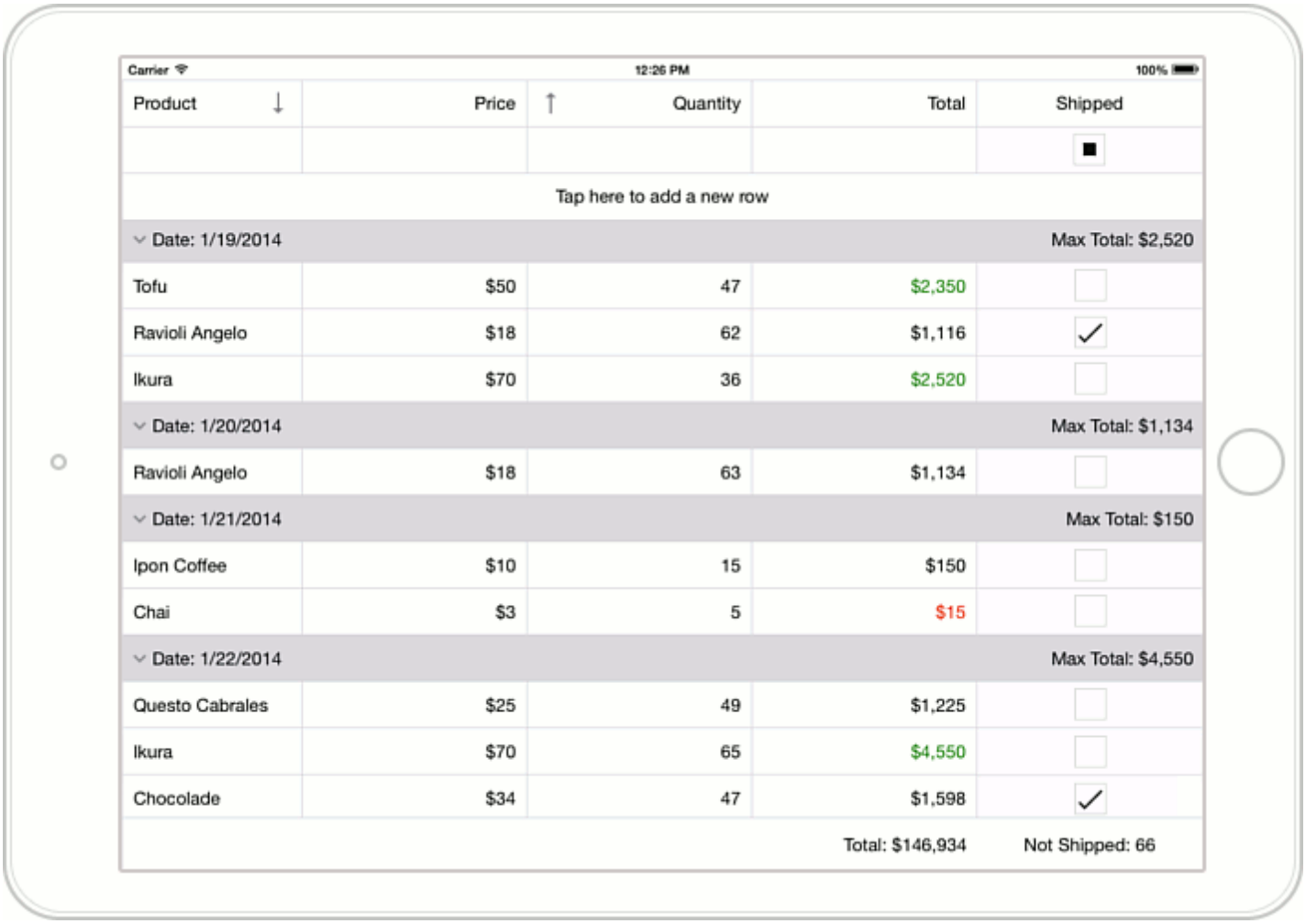
### Example

**Show Me**

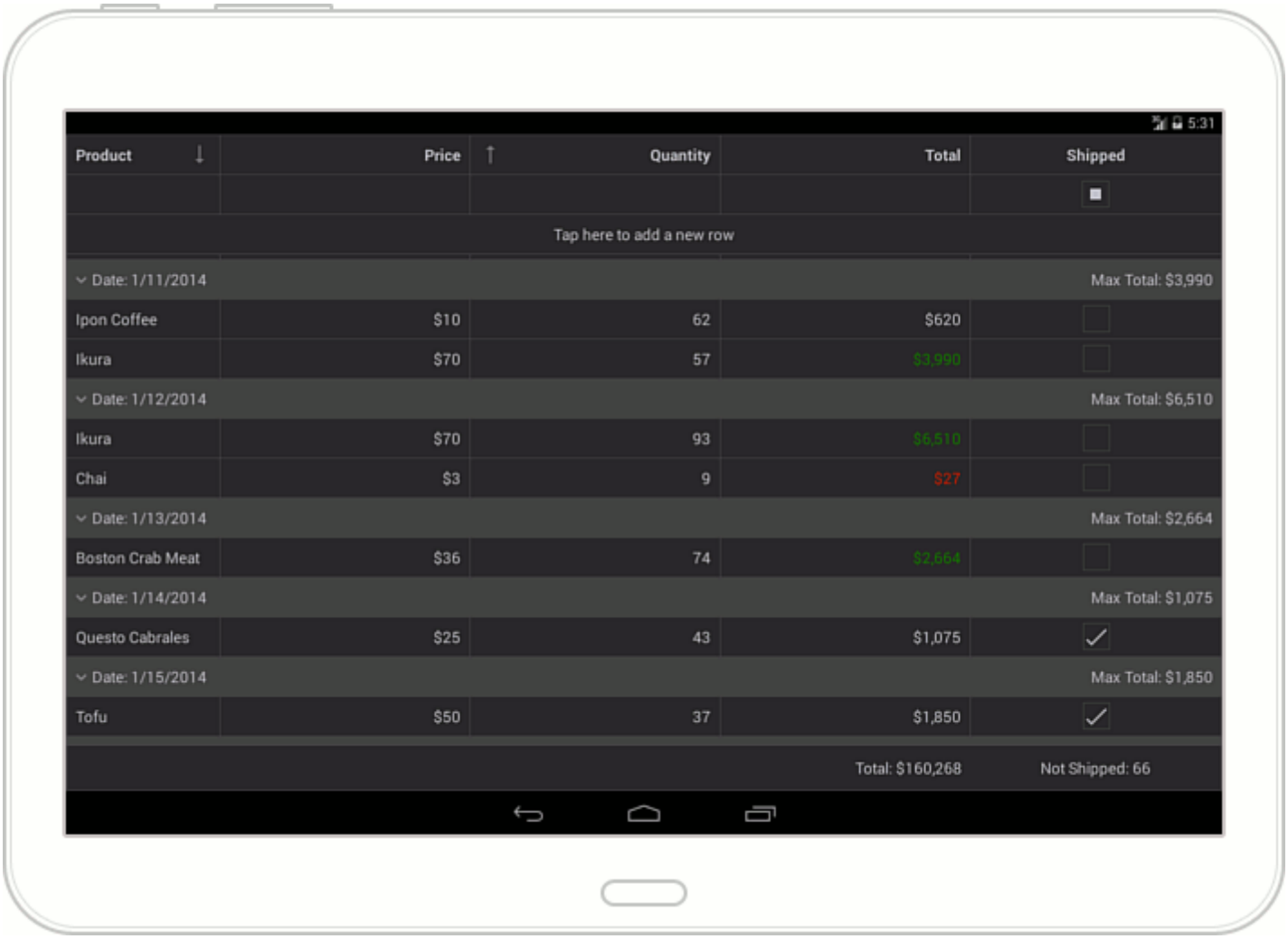
A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T178764.

This example demonstrates how to customize the appearance of cells in the specified grid column depending on cell values by handling the GridControl.CustomizeCell event.

The image below illustrates the result.




© 2018 Developer Express Inc.




**C#**

```
 (MainPage.xaml.cs)
void OnCustomizeCell(CustomizeCellEventArgs e)
{
    if (e.FieldName == "Total" && !e.IsSelected)
    {
        int total = Convert.ToInt32(e.Value);
        if (total < 50)
            e.ForeColor = Color.Red;
        else if (total > 2000)
            e.ForeColor = Color.Green;
        e.Handled = true;
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl x:Name="grid" ItemsSource="{Binding Orders}"
                    NewItemRowVisibility="true"
                    CalculateCustomSummary="OnCalculateCustomSummary"
                    SortMode="Multiple" AutoFilterPanelVisibility="true"
                    CustomizeCell="OnCustomizeCell"
                    SwipeButtonShowing="OnSwipeButtonShowing"
                    SwipeButtonClick="OnSwipeButtonClick">
```

## See Also

© 2018 Developer Express Inc.

CustomizeCellEventArgs Class
CustomizeCellEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# IsSelected Property

Gets whether the processed cell belongs to a row that is currently selected.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public bool IsSelected { get; }
```

### Property Value

**true**, if the cell belongs to the selected row; otherwise, **false**.

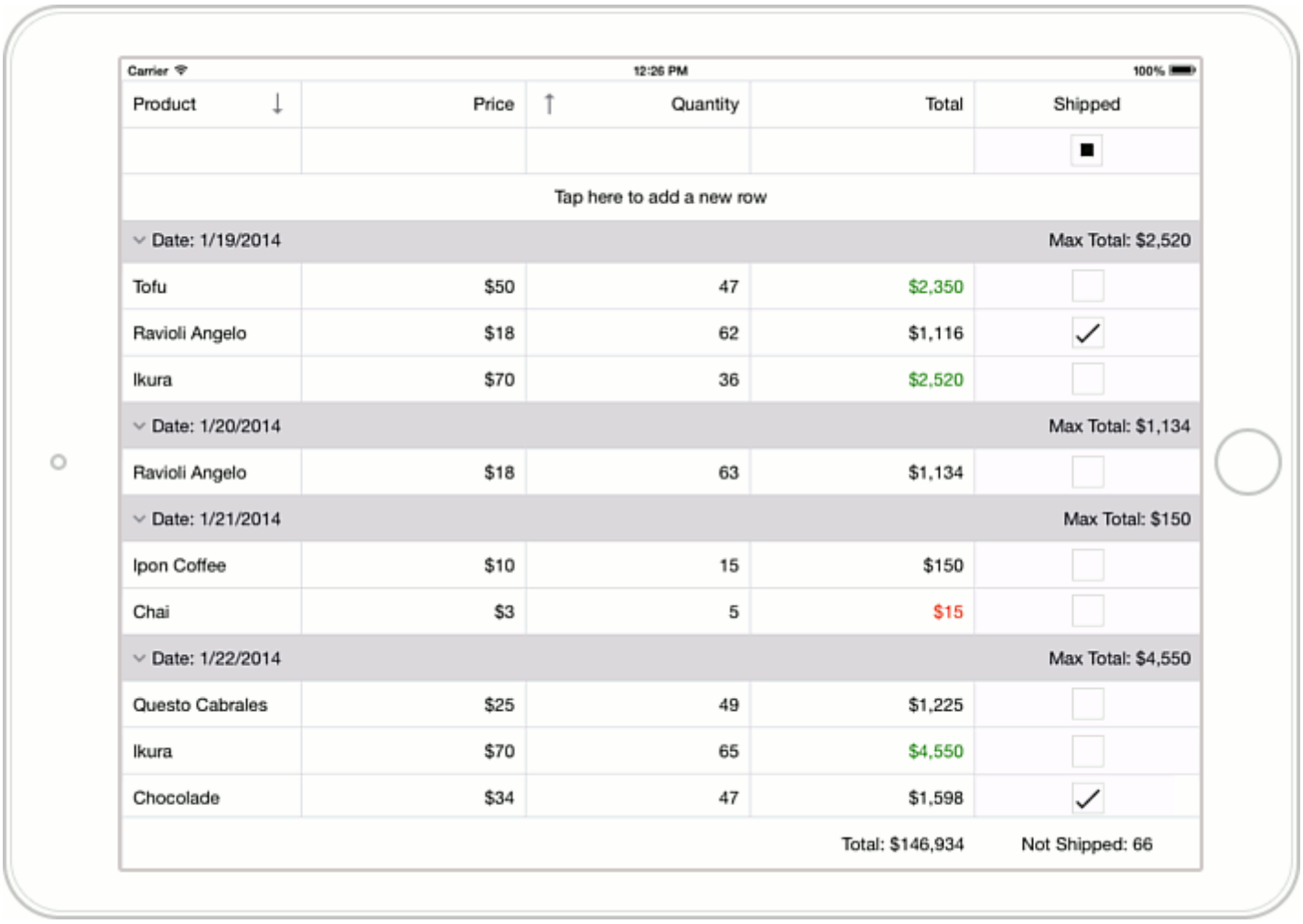
### Example

**Show Me**

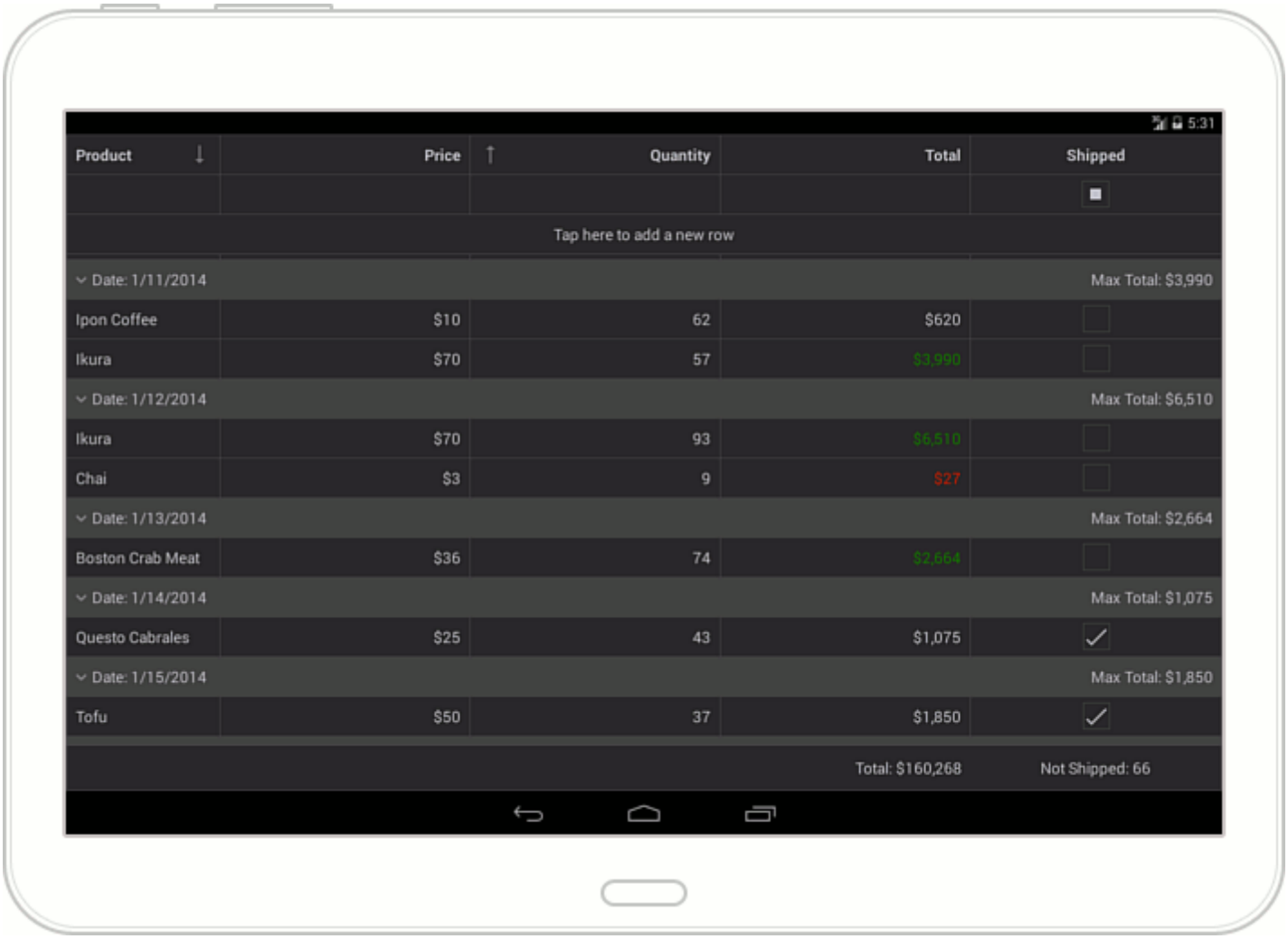
A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T178764.

This example demonstrates how to customize the appearance of cells in the specified grid column depending on cell values by handling the GridControl.CustomizeCell event.

The image below illustrates the result.




© 2018 Developer Express Inc.




**C#**

```
 (MainPage.xaml.cs)
void OnCustomizeCell(CustomizeCellEventArgs e)
{
    if (e.FieldName == "Total" && !e.IsSelected)
    {
        int total = Convert.ToInt32(e.Value);
        if (total < 50)
            e.ForeColor = Color.Red;
        else if (total > 2000)
            e.ForeColor = Color.Green;
        e.Handled = true;
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl x:Name="grid" ItemsSource="{Binding Orders}"
                    NewItemRowVisibility="true"
                    CalculateCustomSummary="OnCalculateCustomSummary"
                    SortMode="Multiple" AutoFilterPanelVisibility="true"
                    CustomizeCell="OnCustomizeCell"
                    SwipeButtonShowing="OnSwipeButtonShowing"
                    SwipeButtonClick="OnSwipeButtonClick">
```

## See Also

© 2018 Developer Express Inc.

CustomizeCellEventArgs Class
CustomizeCellEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# RowHandle Property

Gets the handle of the row that contains the processed cell.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public int RowHandle { get; }
```

### Property Value

An integer value identifying the row that contains the cell to be painted.

### Remarks

For detailed information on how to identify rows by their handles, see Identify and Access Rows.

### See Also

CustomizeCellEventArgs Class
CustomizeCellEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# Value Property

Gets the processed cell value.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public object Value { get; }
```

### Property Value

An object which is the cell value.

### Remarks

Use the **Value** property to obtain the value of a cell to be painted.

### Example

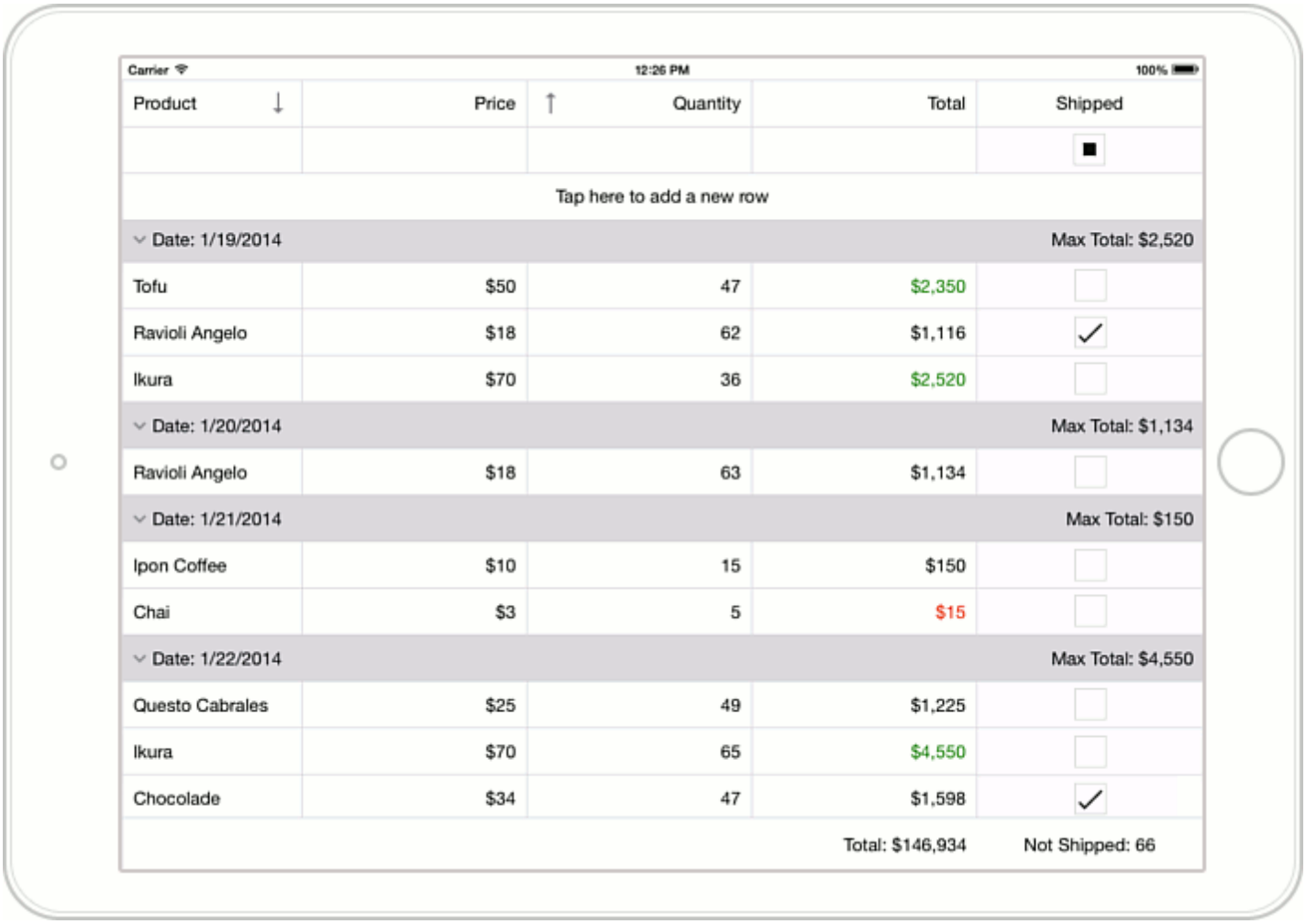
**Show Me**

A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T178764.

This example demonstrates how to customize the appearance of cells in the specified grid column depending on cell values by handling the GridControl.CustomizeCell event.

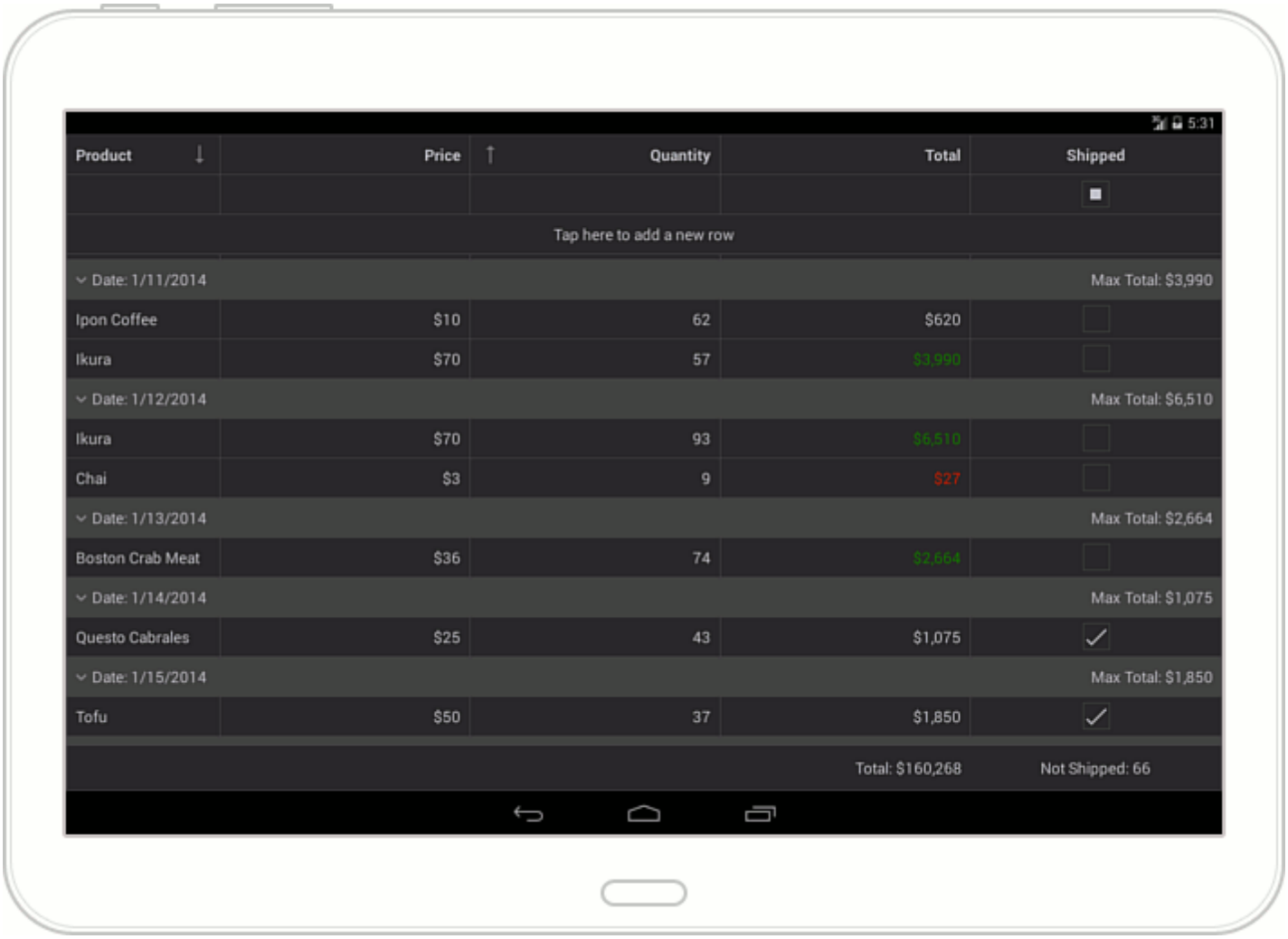
The image below illustrates the result.

© 2018 Developer Express Inc.




| Product | Price | Quantity | Total | Shipped |
|---|---|---|---|---|
| | | | | ■ |
| Tap here to add a new row | | | | |
| Date: 1/19/2014 | | | | Max Total: $2,520 |
| Tofu | $50 | 47 | $2,350 | ☐ |
| Ravioli Angelo | $18 | 62 | $1,116 | ☑ |
| Ikura | $70 | 36 | $2,520 | ☐ |
| Date: 1/20/2014 | | | | Max Total: $1,134 |
| Ravioli Angelo | $18 | 63 | $1,134 | ☐ |
| Date: 1/21/2014 | | | | Max Total: $150 |
| Ipon Coffee | $10 | 15 | $150 | ☐ |
| Chai | $3 | 5 | $15 | ☐ |
| Date: 1/22/2014 | | | | Max Total: $4,550 |
| Questo Cabrales | $25 | 49 | $1,225 | ☐ |
| Ikura | $70 | 65 | $4,550 | ☐ |
| Chocolade | $34 | 47 | $1,598 | ☑ |
| | | | Total: $146,934 | Not Shipped: 66 |

© 2018 Developer Express Inc.




**C#**

```
 (MainPage.xaml.cs)
void OnCustomizeCell(CustomizeCellEventArgs e)
{
    if (e.FieldName == "Total" && !e.IsSelected)
    {
        int total = Convert.ToInt32(e.Value);
        if (total < 50)
            e.ForeColor = Color.Red;
        else if (total > 2000)
            e.ForeColor = Color.Green;
        e.Handled = true;
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl x:Name="grid" ItemsSource="{Binding Orders}"
                    NewItemRowVisibility="true"
                    CalculateCustomSummary="OnCalculateCustomSummary"
                    SortMode="Multiple" AutoFilterPanelVisibility="true"
                    CustomizeCell="OnCustomizeCell"
                    SwipeButtonShowing="OnSwipeButtonShowing"
                    SwipeButtonClick="OnSwipeButtonClick">
```

## See Also

© 2018 Developer Express Inc.

CustomizeCellEventArgs Class
CustomizeCellEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# CustomizeCellEventHandler Delegate

Represents a method that will handle the GridControl.CustomizeCell event.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public delegate void CustomizeCellEventHandler(
   CustomizeCellEventArgs e
)
```

### Parameters

The declaration of your event handler must have the same parameters as the **CustomizeCellEventHandler** delegate declaration.

*e*
A CustomizeCellEventArgs class instance that contains event data.

### See Also

DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# CustomSummaryEventArgs Class

Provides data for the GridControl.CalculateCustomSummary event.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public class CustomSummaryEventArgs : EventArgs
```

### Remarks

The **CustomSummaryEventArgs** objects are automatically created, initialized and passed to the GridControl.CalculateCustomSummary event handlers.

### Example

**Show Me**

A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T178764.

This example shows how to use the GridControl.CalculateCustomSummary event to create a custom summary that counts the total number of data rows whose cells in the **Shipped** column (SwitchColumn) contain **false**.

**C#**

```
 (MainPage.xaml.cs)
void OnCalculateCustomSummary(object sender, CustomSummaryEventArgs e){
    if (e.FieldName.ToString () == "Shipped")
    if (e.IsTotalSummary){
        if (e.SummaryProcess == CustomSummaryProcess.Start) {
            count = 0;
        }
        if (e.SummaryProcess == CustomSummaryProcess.Calculate) {
            if (!(bool)e.FieldValue)
                count++;
            e.TotalValue = count;
        }
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl.TotalSummaries>
    <dxGrid:GridColumnSummary FieldName="Total" Type="Sum"
                              DisplayFormat= "Total: {0:C0}"/>
    <dxGrid:GridColumnSummary FieldName="Shipped" Type="Custom"
                              DisplayFormat= "Not Shipped: {0}"/>
</dxGrid:GridControl.TotalSummaries>
```

## Inheritance Hierarchy

System.Object
    System.EventArgs
        **CustomSummaryEventArgs**

## See Also

CustomSummaryEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# CustomSummaryEventArgs Members

Provides data for the GridControl.CalculateCustomSummary event.

The following tables list the members exposed by the CustomSummaryEventArgs type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | CustomSummaryEventArgs | Initializes a new instance of the CustomSummaryEventArgs class. |

### Public Properties

| | Name | Description |
|---|---|---|
| | FieldName | Gets the processed field name. |
| | FieldValue | Gets the processed field value. |
| | GroupRowHandle | Gets a value identifying the group row whose child data rows are involved in summary calculation. |
| | IsGroupSummary | Gets whether a group summary value is being calculated. |
| | IsTotalSummary | Gets whether a total summary value is being calculated. |
| | Row | Gets an object that contains information on the currently processed row. |
| | SummaryProcess | Gets a value indicating the data summary calculation stage. |
| | TotalValue | Gets or sets the total summary value. |
| | TotalValueReady | Gets or sets whether the Calculation stage of the custom summary calculation process should be skipped. |

### Public Methods

| | Name | Description |
|---|---|---|
| S | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | GetValue | Returns the value in the specified field. |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

### Public Fields

© 2018 Developer Express Inc.

| | Name | Description |
|---|---|---|
| | Empty | (Inherited from **System.EventArgs**) |

**See Also**
CustomSummaryEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# CustomSummaryEventArgs Constructor

Initializes a new instance of the CustomSummaryEventArgs class.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**
**C#**

```
public CustomSummaryEventArgs()
```

**See Also**
CustomSummaryEventArgs Class
CustomSummaryEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# CustomSummaryEventArgs Properties

Provides data for the GridControl.CalculateCustomSummary event.

The following tables list the members exposed by the CustomSummaryEventArgs type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | CustomSummaryEventArgs | Initializes a new instance of the CustomSummaryEventArgs class. |

### Public Properties

| | Name | Description |
|---|---|---|
| | FieldName | Gets the processed field name. |
| | FieldValue | Gets the processed field value. |
| | GroupRowHandle | Gets a value identifying the group row whose child data rows are involved in summary calculation. |
| | IsGroupSummary | Gets whether a group summary value is being calculated. |
| | IsTotalSummary | Gets whether a total summary value is being calculated. |
| | Row | Gets an object that contains information on the currently processed row. |
| | SummaryProcess | Gets a value indicating the data summary calculation stage. |
| | TotalValue | Gets or sets the total summary value. |
| | TotalValueReady | Gets or sets whether the Calculation stage of the custom summary calculation process should be skipped. |

### Public Methods

| | Name | Description |
|---|---|---|
| S | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | GetValue | Returns the value in the specified field. |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

### Public Fields

© 2018 Developer Express Inc.

| | Name | Description |
|---|---|---|
| | Empty | (Inherited from **System.EventArgs**) |

**See Also**
CustomSummaryEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# FieldName Property

Gets the processed field name.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax
**C#**
```
public string FieldName { get; }
```

### Property Value

A string that is the name of the field for which the summary is calculated.

### Example

**Show Me**

A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T178764.

This example shows how to use the GridControl.CalculateCustomSummary event to create a custom summary that counts the total number of data rows whose cells in the **Shipped** column (SwitchColumn) contain **false**.

**C#**

```
 (MainPage.xaml.cs)
void OnCalculateCustomSummary(object sender, CustomSummaryEventArgs e){
    if (e.FieldName.ToString () == "Shipped")
    if (e.IsTotalSummary){
        if (e.SummaryProcess == CustomSummaryProcess.Start) {
            count = 0;
        }
        if (e.SummaryProcess == CustomSummaryProcess.Calculate) {
            if (!(bool)e.FieldValue)
                count++;
            e.TotalValue = count;
        }
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl.TotalSummaries>
    <dxGrid:GridColumnSummary FieldName="Total" Type="Sum"
                              DisplayFormat= "Total: {0:C0}"/>
    <dxGrid:GridColumnSummary FieldName="Shipped" Type="Custom"
                              DisplayFormat= "Not Shipped: {0}"/>
</dxGrid:GridControl.TotalSummaries>
```

## See Also
CustomSummaryEventArgs Class
CustomSummaryEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# FieldValue Property

Gets the processed field value.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public object FieldValue { get; }
```

### Property Value

An object specifying the processed field value.

### Remarks

Default summary calculation algorithms process values that belong to a field specified by a summary item. When calculating a summary value manually, this column's values are passed to the event GridControl.CalculateCustomSummary handler via the **FieldValue** property. Use this property to access the processed value immediately and update the TotalValue property value accordingly.

Note that sometimes, you may need to include several fields in your summary calculations. In this case, you will need to use the Row parameter to identify the row being processed and access its values within the desired fields, or call the GetValue method with the specified field name.

### Example

**Show Me**

A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T178764.

This example shows how to use the GridControl.CalculateCustomSummary event to create a custom summary that counts the total number of data rows whose cells in the **Shipped** column (SwitchColumn) contain **false**.

**C#**

```
 (MainPage.xaml.cs)
void OnCalculateCustomSummary(object sender, CustomSummaryEventArgs e){
    if (e.FieldName.ToString () == "Shipped")
    if (e.IsTotalSummary){
        if (e.SummaryProcess == CustomSummaryProcess.Start) {
            count = 0;
        }
        if (e.SummaryProcess == CustomSummaryProcess.Calculate) {
            if (!(bool)e.FieldValue)
                count++;
            e.TotalValue = count;
        }
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl.TotalSummaries>
    <dxGrid:GridColumnSummary FieldName="Total" Type="Sum"
                              DisplayFormat= "Total: {0:C0}"/>
    <dxGrid:GridColumnSummary FieldName="Shipped" Type="Custom"
                              DisplayFormat= "Not Shipped: {0}"/>
</dxGrid:GridControl.TotalSummaries>
```

## See Also

CustomSummaryEventArgs Class
CustomSummaryEventArgs Members

© 2018 Developer Express Inc.

DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# GroupRowHandle Property

Gets a value identifying the group row whose child data rows are involved in summary calculation.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public int GroupRowHandle { get; }
```

### Property Value

An integer value specifying the handle of the group row containing the processed row as a child.

### See Also

CustomSummaryEventArgs Class
CustomSummaryEventArgs Members
DevExpress.Mobile.DataGrid Namespace
Identify and Access Rows

© 2018 Developer Express Inc.

# IsGroupSummary Property

Gets whether a group summary value is being calculated.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public bool IsGroupSummary { get; }
```

### Property Value

**true**, if a group summary value is being calculated; otherwise, **false**.

### See Also

CustomSummaryEventArgs Class
CustomSummaryEventArgs Members
DevExpress.Mobile.DataGrid Namespace
IsTotalSummary

© 2018 Developer Express Inc.

# IsTotalSummary Property

Gets whether a total summary value is being calculated.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public bool IsTotalSummary { get; }
```

### Property Value

**true**, if a total summary value is being calculated; otherwise, **false**.

### Example

**Show Me**

A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T178764.

This example shows how to use the GridControl.CalculateCustomSummary event to create a custom summary that counts the total number of data rows whose cells in the **Shipped** column (SwitchColumn) contain **false**.

**C#**

```
 (MainPage.xaml.cs)
void OnCalculateCustomSummary(object sender, CustomSummaryEventArgs e){
    if (e.FieldName.ToString () == "Shipped")
    if (e.IsTotalSummary){
        if (e.SummaryProcess == CustomSummaryProcess.Start) {
            count = 0;
        }
        if (e.SummaryProcess == CustomSummaryProcess.Calculate) {
            if (!(bool)e.FieldValue)
                count++;
            e.TotalValue = count;
        }
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl.TotalSummaries>
    <dxGrid:GridColumnSummary FieldName="Total" Type="Sum"
                              DisplayFormat= "Total: {0:C0}"/>
    <dxGrid:GridColumnSummary FieldName="Shipped" Type="Custom"
                              DisplayFormat= "Not Shipped: {0}"/>
</dxGrid:GridControl.TotalSummaries>
```

## See Also

CustomSummaryEventArgs Class
CustomSummaryEventArgs Members
DevExpress.Mobile.DataGrid Namespace
IsGroupSummary

© 2018 Developer Express Inc.

# Row Property

Gets an object that contains information on the currently processed row.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public IRowData Row { get; }
```

### Property Value

An IRowData object containing information on a data row.

### See Also

CustomSummaryEventArgs Class
CustomSummaryEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# SummaryProcess Property

Gets a value indicating the data summary calculation stage.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public CustomSummaryProcess SummaryProcess { get; }
```

### Property Value

A CustomSummaryProcess enumeration value indicating the calculation stage.

### Remarks

By default, the process of custom summary calculation consists of three stages.

- Initialization - the **SummaryProcess** property is set to CustomSummaryProcess.Start, and the GridControl.CalculateCustomSummary event is raised once.
- Calculation - The **SummaryProcess** property is set to CustomSummaryProcess.Calculate, and the GridControl.CalculateCustomSummary event occurs multiple times, once for each data row in the grid or group.
- Finalization - The **SummaryProcess** property is set to CustomSummaryProcess.Finalize, and the GridControl.CalculateCustomSummary event is raised once.

### Example

**Show Me**

A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T178764.

This example shows how to use the GridControl.CalculateCustomSummary event to create a custom summary that counts the total number of data rows whose cells in the **Shipped** column (SwitchColumn) contain **false**.

**C#**

```
 (MainPage.xaml.cs)
void OnCalculateCustomSummary(object sender, CustomSummaryEventArgs e){
    if (e.FieldName.ToString () == "Shipped")
    if (e.IsTotalSummary){
        if (e.SummaryProcess == CustomSummaryProcess.Start) {
            count = 0;
        }
        if (e.SummaryProcess == CustomSummaryProcess.Calculate) {
            if (!(bool)e.FieldValue)
                count++;
            e.TotalValue = count;
        }
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl.TotalSummaries>
    <dxGrid:GridColumnSummary FieldName="Total" Type="Sum"
                              DisplayFormat= "Total: {0:C0}"/>
    <dxGrid:GridColumnSummary FieldName="Shipped" Type="Custom"
                              DisplayFormat= "Not Shipped: {0}"/>
</dxGrid:GridControl.TotalSummaries>
```

## See Also

CustomSummaryEventArgs Class
CustomSummaryEventArgs Members

© 2018 Developer Express Inc.

DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# TotalValue Property

Gets or sets the total summary value.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public object TotalValue { get; set; }
```

### Property Value

An object specifying the total summary value.

### Remarks

**TotalValue** is the total summary value that accumulates the custom calculated value. Note that this property should be initialized before the summary calculation process starts. To identify the status of the custom summary calculation, use the SummaryProcess property. When performing calculations, you need to refresh the **TotalValue** property for it to be updated with the new values.

You can also modify the **TotalValue** property's value when finalizing the calculation. In this case, the SummaryProcess property returns the CustomSummaryProcess.Finalize value.

### Example

**Show Me**

A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T178764.

This example shows how to use the GridControl.CalculateCustomSummary event to create a custom summary that counts the total number of data rows whose cells in the **Shipped** column (SwitchColumn) contain **false**.

**C#**

```
 (MainPage.xaml.cs)
void OnCalculateCustomSummary(object sender, CustomSummaryEventArgs e){
    if (e.FieldName.ToString () == "Shipped")
    if (e.IsTotalSummary){
        if (e.SummaryProcess == CustomSummaryProcess.Start) {
            count = 0;
        }
        if (e.SummaryProcess == CustomSummaryProcess.Calculate) {
            if (!(bool)e.FieldValue)
                count++;
            e.TotalValue = count;
        }
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl.TotalSummaries>
    <dxGrid:GridColumnSummary FieldName="Total" Type="Sum"
                              DisplayFormat= "Total: {0:C0}"/>
    <dxGrid:GridColumnSummary FieldName="Shipped" Type="Custom"
                              DisplayFormat= "Not Shipped: {0}"/>
</dxGrid:GridControl.TotalSummaries>
```

## See Also

CustomSummaryEventArgs Class
CustomSummaryEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# TotalValueReady Property

Gets or sets whether the Calculation stage of the custom summary calculation process should be skipped.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public bool TotalValueReady { get; set; }
```

### Property Value

**true** if the Calculation stage of the custom summary calculation process should be skipped; otherwise, **false**.

### Remarks

By default, the process of custom summary calculation consists of three stages: Start, Calculation and Finalization. During the Start stage the GridControl.CalculateCustomSummary event fires once with the SummaryProcess parameter set to CustomSummaryProcess.Start. During the Calculation stage, the event is called multiple times - once for each record. Finally, during the last stage the event is called once with the SummaryProcess parameter set to CustomSummaryProcess.Finalize.

The **TotalValueReady** property allows you to interrupt the process of summary calculation during the Start stage. You cannot interrupt the process in the other stages. This can be useful if you need to calculate a custom summary during the Start or Finalization stage while skipping the time-consuming Calculation stage.

If the **TotalValueReady** property is set to **true** the Calculation stage is skipped and the Finalization stage starts immediately (the summary calculation event will be called with the SummaryProcess parameter set to CustomSummaryProcess.Finalize).

### See Also

CustomSummaryEventArgs Class
CustomSummaryEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# CustomSummaryEventArgs Methods

Provides data for the GridControl.CalculateCustomSummary event.

The following tables list the members exposed by the CustomSummaryEventArgs type.

**Public Methods**

| | Name | Description |
|---|---|---|
| | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | GetValue | Returns the value in the specified field. |
| | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

**See Also**
CustomSummaryEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# GetValue Method

Returns the value in the specified field.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public object GetValue(
   string fieldName
)
```

### Parameters

*fieldName*
A string that identifies the field whose value must be returned.

### Return Value

An object that represents the value of the specified field.

### See Also

CustomSummaryEventArgs Class
CustomSummaryEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# CustomSummaryEventHandler Delegate

Represents a method that will handle the GridControl.CalculateCustomSummary event.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public delegate void CustomSummaryEventHandler(
   object sender,
   CustomSummaryEventArgs e
)
```

### Parameters

The declaration of your event handler must have the same parameters as the **CustomSummaryEventHandler** delegate declaration.

*sender*
The event source. Identifies the GridControl object that raised the event.

*e*
A CustomSummaryEventArgs object that contains the event data.

### See Also

DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# CustomSummaryProcess Enumeration

Lists values that specify the current status of custom summary calculation.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public enum CustomSummaryProcess
```

### Members

| Name | Description |
|---|---|
| Calculate | Indicates that custom summary calculation is in progress. This means the event is raised for a specific data row containing the field value used to calculate the processed custom summary. |
| Finalize | Indicates that the process of custom summary calculation is finished. |
| Start | Indicates that the process of custom summary calculation is about to start. |

### Remarks

The **CustomSummaryProcess** enumeration contains values indicating the current stage of the custom summary calculation process. These members are used by the CustomSummaryEventArgs.SummaryProcess property.

### See Also

DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# DataBarFormatCondition Class

Represents a data bar format condition.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

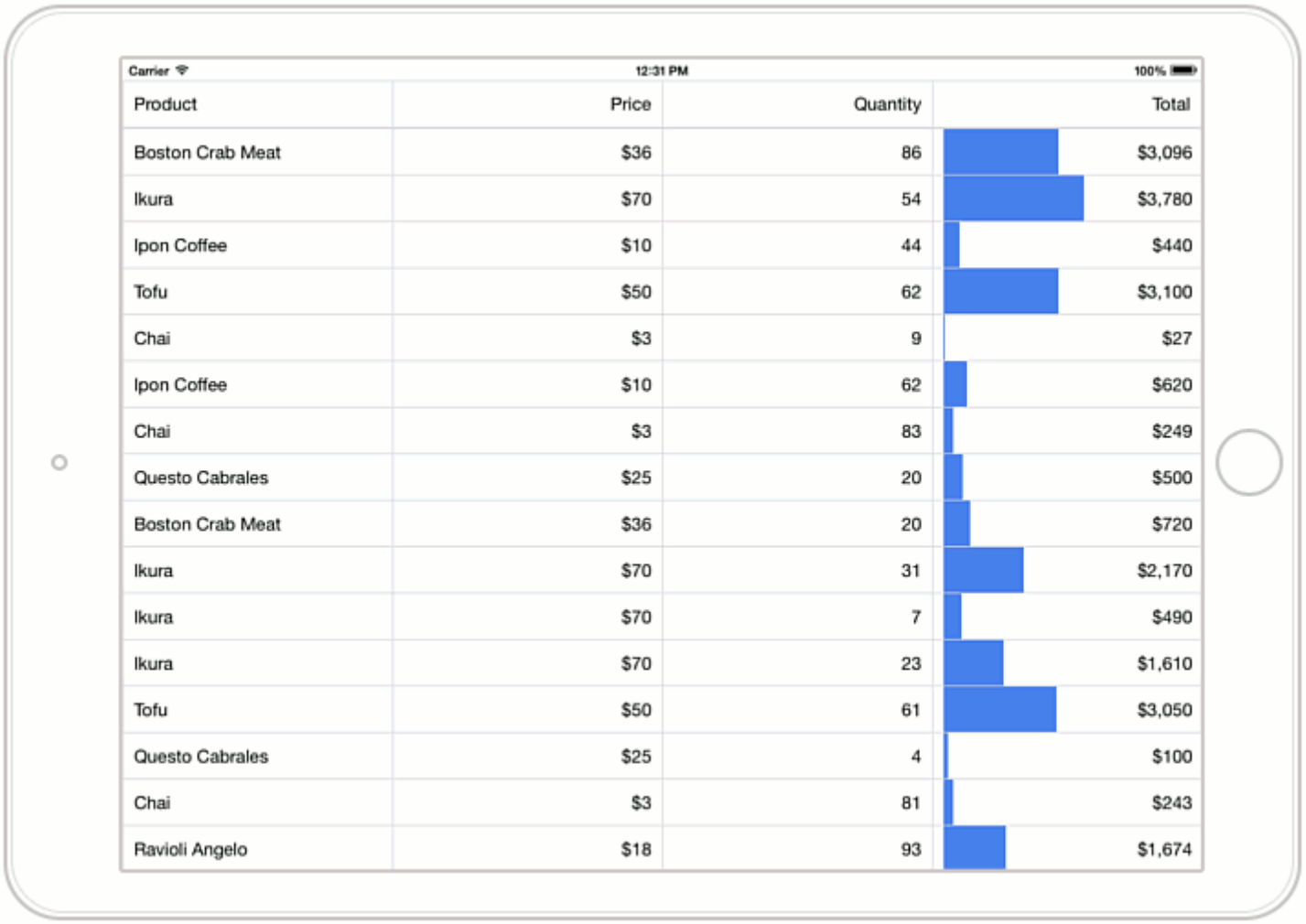
**C#**

```
public class DataBarFormatCondition : IndicatorFormatConditionBase
```

### Example

This example demonstrates how to apply a data bar format to the **Total** column in a GridControl. Data bars fill column cells proportionally based on cell values relative to other cells. A longer bar corresponds to a higher value, and a shorter bar corresponds to a lower value.

© 2018 Developer Express Inc.




| Product | Price | Quantity | Total |
|---|---|---|---|
| Boston Crab Meat | $36 | 86 | $3,096 |
| Ikura | $70 | 54 | $3,780 |
| Ipon Coffee | $10 | 44 | $440 |
| Tofu | $50 | 62 | $3,100 |
| Chai | $3 | 9 | $27 |
| Ipon Coffee | $10 | 62 | $620 |
| Chai | $3 | 83 | $249 |
| Questo Cabrales | $25 | 20 | $500 |
| Boston Crab Meat | $36 | 20 | $720 |
| Ikura | $70 | 31 | $2,170 |
| Ikura | $70 | 7 | $490 |
| Ikura | $70 | 23 | $1,610 |
| Tofu | $50 | 61 | $3,050 |
| Questo Cabrales | $25 | 4 | $100 |
| Chai | $3 | 81 | $243 |
| Ravioli Angelo | $18 | 93 | $1,674 |

| Product | Price | Quantity | Total |
|---|---|---|---|
| Ravioli Angelo | $18 | 67 | $1,206 |
| Ikura | $70 | 62 | $4,340 |
| Ipon Coffee | $10 | 85 | $850 |
| Chocolade | $34 | 27 | $918 |
| Ipon Coffee | $10 | 3 | $30 |
| Boston Crab Meat | $36 | 90 | $3,240 |
| Ravioli Angelo | $18 | 1 | $18 |
| Tofu | $50 | 33 | $1,650 |
| Ipon Coffee | $10 | 70 | $700 |
| Chocolade | $34 | 74 | $2,516 |
| Tofu | $50 | 97 | $4,850 |
| Chocolade | $34 | 1 | $34 |
| Ipon Coffee | $10 | 71 | $710 |
| Boston Crab Meat | $36 | 2 | $72 |
| Boston Crab Meat | $36 | 73 | $2,628 |
| Ikura | $70 | 72 | $5,040 |

1. Create a new DataBarFormatCondition class instance.
2. Set this object's FormatConditionBase.FieldName property to **Total**. This column provides values to test against the formatting rule and the specified format will be applied to this column.
3. Assign one of the predefined bar styles to the FormatConditionBase.PredefinedFormatName property. All available format names are listed in the Predefined Format Names document.
4. To apply a conditional formatting rule represented by the created **DataBarFormatCondition** object, add this object to the GridControl.FormatConditions collection.

**C#**

```
using DevExpress.Mobile.DataGrid;
// ...
DataBarFormatCondition condition = new DataBarFormatCondition();
condition.FieldName = "Total";
condition.PredefinedFormatName = "LightBlueGradientDataBar";
grid.FormatConditions.Add (condition);
```

### Inheritance Hierarchy

System.Object
  Xamarin.Forms.BindableObject
    FormatConditionBase
      IndicatorFormatConditionBase
        **DataBarFormatCondition**

### See Also

DataBarFormatCondition Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# DataBarFormatCondition Members

Represents a data bar format condition.

The following tables list the members exposed by the DataBarFormatCondition type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | DataBarFormatCondition | Initializes a new instance of the DataBarFormatCondition class with default settings. |

### Public Properties

| | Name | Description |
|---|---|---|
| | BindingContext | (Inherited from **Xamarin.Forms.BindableObject**) |
| | FieldName | Gets or sets the name of the column which provides values to test against the current format rule, and to which the format should be applied. This is a bindable property. (Inherited from **FormatConditionBase**) |
| | Format | Gets or sets the formatting settings applied to cells whose values take part in conditional formatting. This is a bindable property. |
| | MaxValue | Specifies the maximum column value to accept for conditional formatting. This is a bindable property. (Inherited from **IndicatorFormatConditionBase**) |
| | MinValue | Specifies the minimum column value to accept for conditional formatting. This is a bindable property. (Inherited from **IndicatorFormatConditionBase**) |
| | Owner | Gets the format condition's owner. This is a bindable property. (Inherited from **FormatConditionBase**) |
| | PredefinedFormatName | Gets or sets the name of the predefined format to be applied to target cells. This is a bindable property. (Inherited from **FormatConditionBase**) |
| | SelectiveExpression | Specifies a custom rule for applying conditional formatting. This is a bindable property. (Inherited from **IndicatorFormatConditionBase**) |
| | TypeName | Gets or sets the data type name. This property supports the internal infrastructure and is not intended to be used directly from your code. (Inherited from **FormatConditionBase**) |

© 2018 Developer Express Inc.

### Public Methods

| | Name | Description |
|---|---|---|
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Equals | (Inherited from **System.Object**) |
| S | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ToString | (Inherited from **System.Object**) |

### Public Fields

| | Name | Description |
|---|---|---|
| S | BindingContextProperty | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | FieldNameProperty | Identifies the FormatConditionBase.FieldName bindable property. (Inherited from **FormatConditionBase**) |
| S | FormatProperty | Identifies the Format bindable property. |
| S | MaxValueProperty | Identifies the IndicatorFormatConditionBase.MaxValue bindable property. (Inherited from **IndicatorFormatConditionBase**) |
| S | MinValueProperty | Identifies the IndicatorFormatConditionBase.MinValue bindable property. (Inherited from **IndicatorFormatConditionBase**) |
| S | OwnerProperty | Identifies the FormatConditionBase.Owner bindable property. (Inherited from **FormatConditionBase**) |
| S | PredefinedFormatNameProperty | Identifies the FormatConditionBase.PredefinedFormatName bindable property. (Inherited from **FormatConditionBase**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | SelectiveExpressionProperty | Identifies the IndicatorFormatConditionBase.SelectiveExpression bindable property. (Inherited from **IndicatorFormatConditionBase**) |

**Public Events**

| | Name | Description |
|---|---|---|
| | BindingContextChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanging | (Inherited from **Xamarin.Forms.BindableObject**) |

**See Also**
DataBarFormatCondition Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# DataBarFormatCondition Constructor

Initializes a new instance of the DataBarFormatCondition class with default settings.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**
**C#**

```
public DataBarFormatCondition()
```

**See Also**
DataBarFormatCondition Class
DataBarFormatCondition Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# DataBarFormatCondition Properties

Represents a data bar format condition.

The following tables list the members exposed by the DataBarFormatCondition type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | DataBarFormatCondition | Initializes a new instance of the DataBarFormatCondition class with default settings. |

### Public Properties

| | Name | Description |
|---|---|---|
| | BindingContext | (Inherited from **Xamarin.Forms.BindableObject**) |
| | FieldName | Gets or sets the name of the column which provides values to test against the current format rule, and to which the format should be applied. This is a bindable property. (Inherited from **FormatConditionBase**) |
| | Format | Gets or sets the formatting settings applied to cells whose values take part in conditional formatting. This is a bindable property. |
| | MaxValue | Specifies the maximum column value to accept for conditional formatting. This is a bindable property. (Inherited from **IndicatorFormatConditionBase**) |
| | MinValue | Specifies the minimum column value to accept for conditional formatting. This is a bindable property. (Inherited from **IndicatorFormatConditionBase**) |
| | Owner | Gets the format condition's owner. This is a bindable property. (Inherited from **FormatConditionBase**) |
| | PredefinedFormatName | Gets or sets the name of the predefined format to be applied to target cells. This is a bindable property. (Inherited from **FormatConditionBase**) |
| | SelectiveExpression | Specifies a custom rule for applying conditional formatting. This is a bindable property. (Inherited from **IndicatorFormatConditionBase**) |
| | TypeName | Gets or sets the data type name. This property supports the internal infrastructure and is not intended to be used directly from your code. (Inherited from **FormatConditionBase**) |

© 2018 Developer Express Inc.

### Public Methods

| | Name | Description |
|---|---|---|
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Equals | (Inherited from **System.Object**) |
| S | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ToString | (Inherited from **System.Object**) |

### Public Fields

| | Name | Description |
|---|---|---|
| S | BindingContextProperty | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | FieldNameProperty | Identifies the FormatConditionBase.FieldName bindable property. (Inherited from **FormatConditionBase**) |
| S | FormatProperty | Identifies the Format bindable property. |
| S | MaxValueProperty | Identifies the IndicatorFormatConditionBase.MaxValue bindable property. (Inherited from **IndicatorFormatConditionBase**) |
| S | MinValueProperty | Identifies the IndicatorFormatConditionBase.MinValue bindable property. (Inherited from **IndicatorFormatConditionBase**) |
| S | OwnerProperty | Identifies the FormatConditionBase.Owner bindable property. (Inherited from **FormatConditionBase**) |
| S | PredefinedFormatNameProperty | Identifies the FormatConditionBase.PredefinedFormatName bindable property. (Inherited from **FormatConditionBase**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | SelectiveExpressionProperty | Identifies the IndicatorFormatConditionBase.SelectiveExpression bindable property. (Inherited from **IndicatorFormatConditionBase**) |

**Public Events**

| | Name | Description |
|---|---|---|
| | BindingContextChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanging | (Inherited from **Xamarin.Forms.BindableObject**) |

**See Also**
DataBarFormatCondition Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# Format Property

Gets or sets the formatting settings applied to cells whose values take part in conditional formatting. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public DataBarFormat Format { get; set; }
```

### Property Value

A **DataBarFormat** object that provides formatting settings.

### See Also

DataBarFormatCondition Class
DataBarFormatCondition Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# DateColumn Class

A grid column to display and edit date values.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

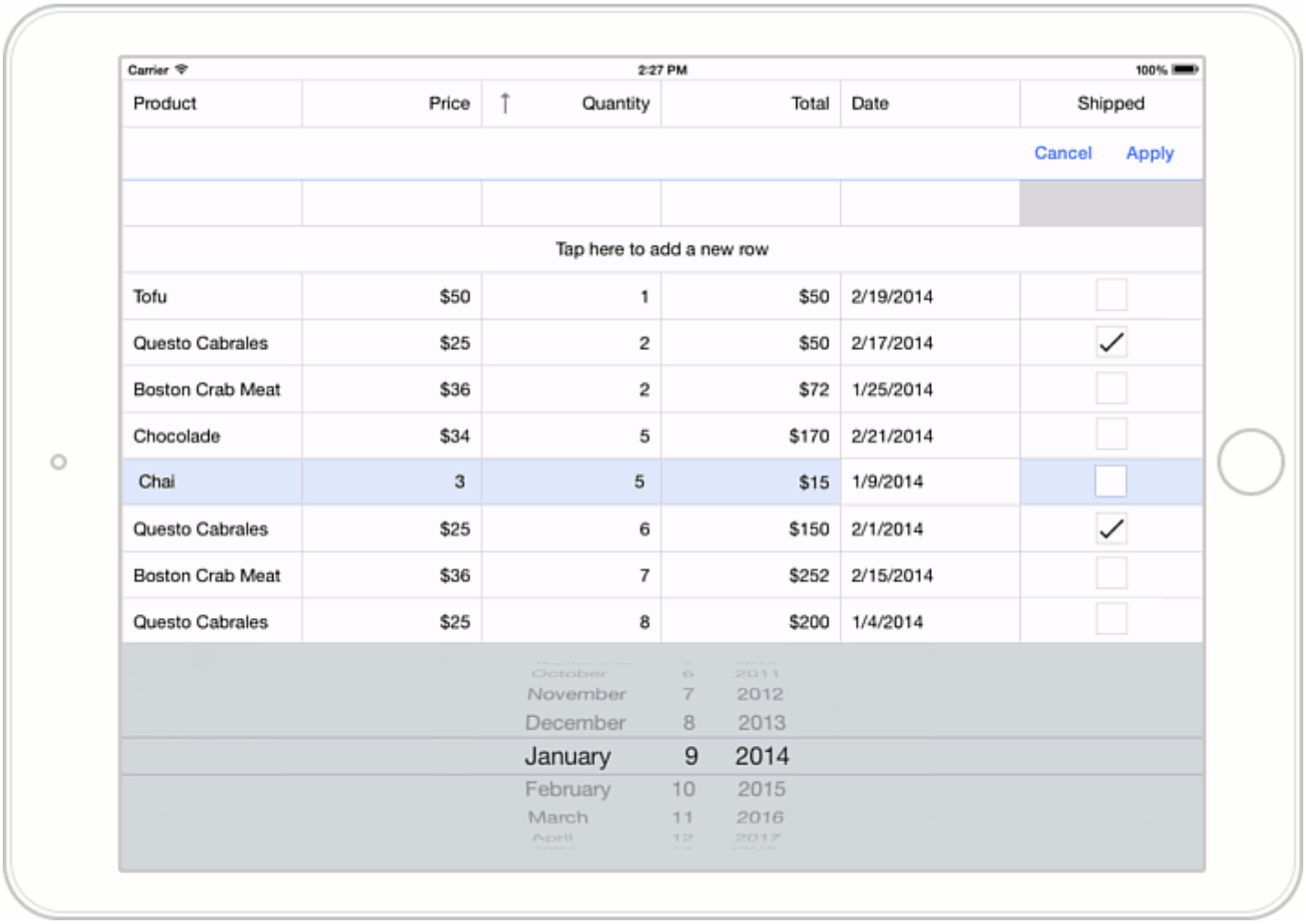
**C#**

```
public class DateColumn : TextColumn
```

### Remarks

In GridControl, data fields are represented as columns and records are displayed as rows. So, to display data, a grid should contain columns bound to fields of the underlying data source.

An individual column is specified by the **DateColumn**, NumberColumn, TextColumn or SwitchColumn object (a GridColumn descendant) depending on the type of data displayed in this column. Use columns of the **DateColumn** type to display dates and allow end-user to edit them.




Columns objects provide a set of properties to adjust column settings (for example, GridColumn.DisplayFormat, GridColumn.Width, GridColumn.Caption, GridColumn.IsVisible, etc.).

GridControl stores its columns in the GridControl.Columns collection, and displays them in the order in which they are added to this collection.

### Example

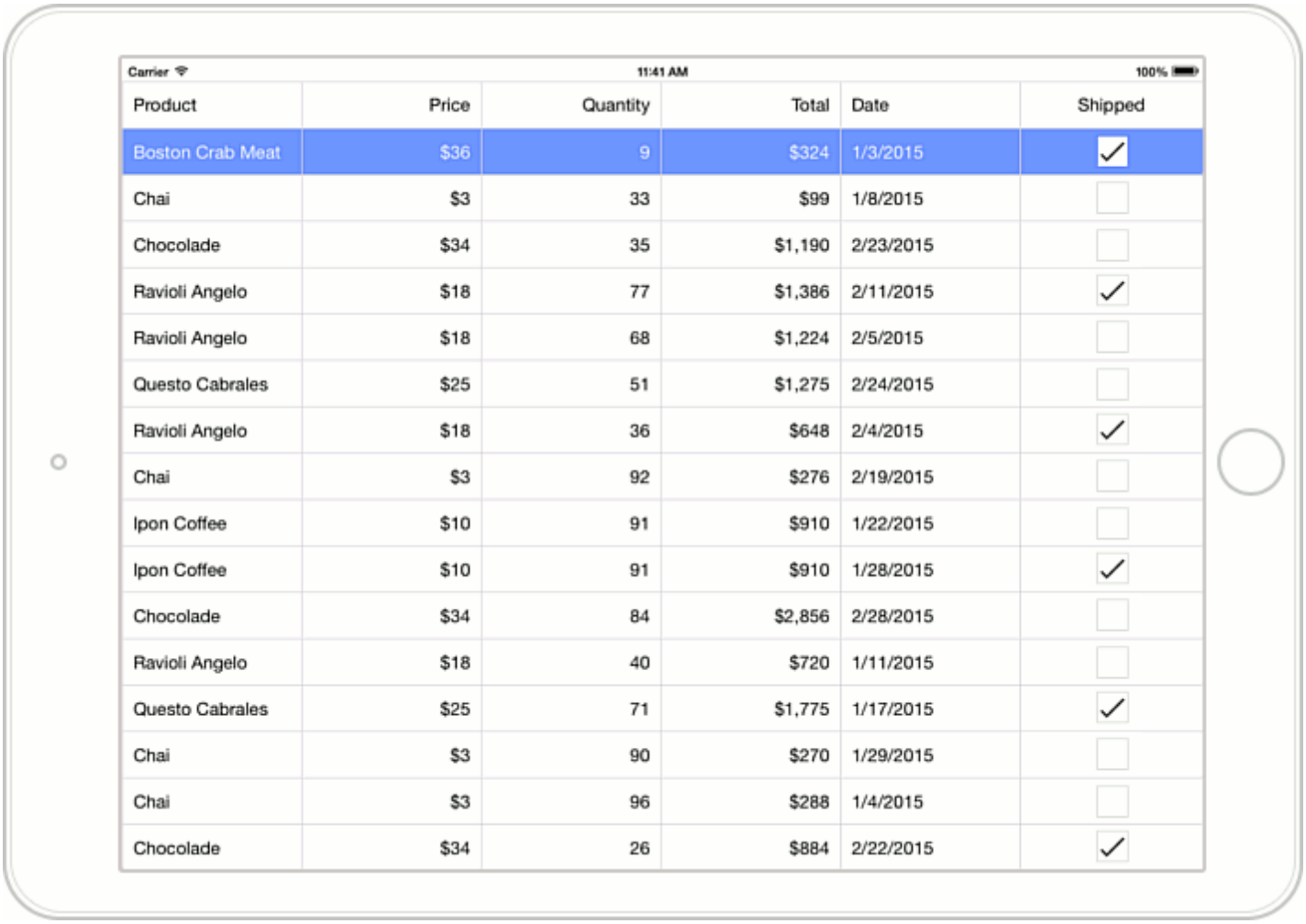
**Show Me**

© 2018 Developer Express Inc.

| | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T178764. | |

This example shows how to create and customize grid columns for displaying and editing data of different types (text, numbers, dates and Boolean values). The specified collection contains columns bound to the data source fields (*Product.Name*, *Product.UnitPrice*, *Quantity*, *Date* and *Shipped*) and one unbound column (*Total*) displaying data values calculated according to a formula based on the values of other columns.

The image below illustrates the result.




© 2018 Developer Express Inc.

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl.Columns>
         <dxGrid:TextColumn FieldName="Product.Name" Caption = "Product" Width = "170"
                            SortOrder = "Descending" SortIndex = "0"/>
         <dxGrid:NumberColumn FieldName="Product.UnitPrice" Caption = "Price" DisplayFormat="C0"/>
         <dxGrid:NumberColumn FieldName="Quantity"
                              SortOrder = "Ascending" SortIndex = "1"/>
         <dxGrid:NumberColumn FieldName="Total"
                              UnboundType="Integer" UnboundExpression="[Quantity] * [Product.UnitPrice]"
                              DisplayFormat="C0" IsReadOnly="True"/>
         <dxGrid:DateColumn FieldName="Date" DisplayFormat="d"
                            IsGrouped = "true" GroupInterval = "Date"/>
         <dxGrid:SwitchColumn FieldName="Shipped" AllowSort = "False" AllowAutoFilter="false"/>
     </dxGrid:GridControl.Columns>
```

### Inheritance Hierarchy

System.Object
  Xamarin.Forms.BindableObject
    GridColumn
      TextColumn
        **DateColumn**

### See Also

DateColumn Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# DateColumn Members

A grid column to display and edit date values.

The following tables list the members exposed by the DateColumn type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | DateColumn | Initializes a new instance of the DateColumn class with default settings. |

### Public Properties

| | Name | Description |
|---|---|---|
| | ActualCaption | Gets the column caption as it is displayed in a column header. (Inherited from **GridColumn**) |
| | ActualWidth | Gets the column width. This is a bindable property. (Inherited from **GridColumn**) |
| | AllowAutoFilter | Gets or sets whether an end-user can filter data by the column's values using the auto-filter panel. This is a bindable property. (Inherited from **GridColumn**) |
| | AllowGroup | Gets or sets whether end-users are allowed to group data by the current column. This is a bindable property. (Inherited from **GridColumn**) |
| | AllowSort | Gets or sets whether end-users are allowed to sort data by the column's values. This is a bindable property. (Inherited from **GridColumn**) |
| | AutoFilterCondition | Gets or sets the comparison operator used to create filter conditions for the current column via the auto-filter panel. This is a bindable property. (Inherited from **GridColumn**) |
| | AutoFilterValue | Gets or sets the value in the auto-filter panel for the column. This is a bindable property. (Inherited from **GridColumn**) |
| | BindingContext | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Caption | Gets or sets the column's display caption. This is a bindable property. (Inherited from **GridColumn**) |
| | ColumnFilterMode | Gets or sets how column values are filtered via the auto-filter panel. This is a bindable property. (Inherited from **GridColumn**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | ContentAlignment | Gets or sets the content alignment within column cells. This is a bindable property. (Inherited from **GridColumn**) |
| | DisplayFormat | Gets or sets formatting for values in the column. This is a bindable property. (Inherited from **GridColumn**) |
| | FieldName | Gets or sets the column's unique field name. For bound columns, this property refers to a field in the underlying data source. For unbound columns, this property serves as a column's identifier.<br><br>This is a bindable property.<br><br>(Inherited from **GridColumn**) |
| | FixedStyle | Gets or sets whether a column is fixed. This is a bindable property. (Inherited from **GridColumn**) |
| | GroupInterval | Gets or sets how data rows are grouped when grouping by the current column is applied. This is a bindable property. (Inherited from **GridColumn**) |
| | HeaderTemplate | Gets or sets a template used to define the column headers visual representation. (Inherited from **GridColumn**) |
| | ImmediateUpdateAutoFilter | Gets or sets whether the column's filter condition is updated as soon as an end-user modifies the text within the auto filter panel's cell. This is a bindable property. (Inherited from **GridColumn**) |
| | IsGrouped | Gets or sets whether the grid's data is grouped by the current column. This is a bindable property. (Inherited from **GridColumn**) |
| | IsNullable | Indicates that the column is bound to the **Nullable<DateTime>** property. |
| | IsReadOnly | Gets or sets whether end-users are allowed to change cell values in the column. This is a bindable property. (Inherited from **GridColumn**) |
| | IsUnbound | Gets whether the column is unbound. (Inherited from **GridColumn**) |
| | IsVisible | Gets or sets whether the column is visible in the grid. This is a bindable property. (Inherited from **GridColumn**) |
| | MinWidth | Gets or sets the minimum column width allowed. This is a bindable property. (Inherited from **GridColumn**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | SerializationTypeName | Gets the string containing the column type name. (Inherited from **GridColumn**) |
| | SortIndex | Gets or sets the column's position among sorted columns. This is a bindable property. (Inherited from **GridColumn**) |
| | SortMode | Specifies how column values are sorted and grouped. This is a bindable property. (Inherited from **GridColumn**) |
| | SortOrder | Gets or sets the column's sort order. This is a bindable property. (Inherited from **GridColumn**) |
| | UnboundExpression | Gets or sets an expression used to evaluate values for the current Unbound Columns. This is a bindable property. (Inherited from **GridColumn**) |
| | UnboundType | Gets or sets the data type and binding mode of the column. This is a bindable property. (Inherited from **GridColumn**) |
| | Width | Gets or sets the column width. This is a bindable property. (Inherited from **GridColumn**) |

**Public Methods**

| | Name | Description |
|---|---|---|
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ToString | (Inherited from **System.Object**) |

**Public Fields**

© 2018 Developer Express Inc.

| | Name | Description |
|---|---|---|
| | ActualWidthProperty | Identifies the GridColumn.ActualWidth bindable property. (Inherited from **GridColumn**) |
| | AllowAutoFilterProperty | Identifies the GridColumn.AllowAutoFilter bindable property. (Inherited from **GridColumn**) |
| | AllowGroupProperty | Identifies the GridColumn.AllowGroup bindable property. (Inherited from **GridColumn**) |
| | AllowSortProperty | Identifies the GridColumn.AllowSort bindable property. (Inherited from **GridColumn**) |
| | AutoFilterConditionProperty | Identifies the GridColumn.AutoFilterCondition bindable property. (Inherited from **GridColumn**) |
| | AutoFilterValueProperty | Identifies the GridColumn.AutoFilterValue bindable property. (Inherited from **GridColumn**) |
| | BindingContextProperty | (Inherited from **Xamarin.Forms.BindableObject**) |
| | CaptionProperty | Identifies the GridColumn.Caption bindable property. (Inherited from **GridColumn**) |
| | ColumnFilterModeProperty | Identifies the GridColumn.ColumnFilterMode bindable property. (Inherited from **GridColumn**) |
| | ContentAlignmentProperty | Identifies the GridColumn.ContentAlignment bindable property. (Inherited from **GridColumn**) |
| | DisplayFormatProperty | Identifies the GridColumn.DisplayFormat bindable property. (Inherited from **GridColumn**) |
| | FieldNameProperty | Identifies the GridColumn.FieldName bindable property. (Inherited from **GridColumn**) |
| | FixedStyleProperty | Identifies the GridColumn.FixedStyle bindable property. (Inherited from **GridColumn**) |
| | GroupIntervalProperty | Identifies the GridColumn.GroupInterval bindable property. (Inherited from **GridColumn**) |
| | HeaderTemplateProperty | Identifies the GridColumn.HeaderTemplate bindable property. (Inherited from **GridColumn**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| S | ImmediateUpdateAutoFilterProperty | Identifies the GridColumn.ImmediateUpdateAutoFilter bindable property. (Inherited from **GridColumn**) |
| S | IsGroupedProperty | Identifies the GridColumn.IsGrouped bindable property. (Inherited from **GridColumn**) |
| S | IsReadOnlyProperty | Identifies the GridColumn.IsReadOnly bindable property. (Inherited from **GridColumn**) |
| S | IsVisibleProperty | Identifies the GridColumn.IsVisible bindable property. (Inherited from **GridColumn**) |
| S | MinWidthProperty | Identifies the GridColumn.MinWidth bindable property. (Inherited from **GridColumn**) |
| S | SortIndexProperty | Identifies the GridColumn.SortIndex bindable property. (Inherited from **GridColumn**) |
| S | SortModeProperty | Identifies the GridColumn.SortMode bindable property. (Inherited from **GridColumn**) |
| S | SortOrderProperty | Identifies the GridColumn.SortOrder bindable property. (Inherited from **GridColumn**) |
| S | UnboundExpressionProperty | Identifies the GridColumn.UnboundExpression bindable property. (Inherited from **GridColumn**) |
| S | UnboundTypeProperty | Identifies the GridColumn.UnboundType bindable property. (Inherited from **GridColumn**) |
| S | WidthProperty | Identifies the GridColumn.Width bindable property. (Inherited from **GridColumn**) |

**Public Events**

| | Name | Description |
|---|---|---|
| | BindingContextChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanging | (Inherited from **Xamarin.Forms.BindableObject**) |

**See Also**
DateColumn Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# DateColumn Constructor

Initializes a new instance of the DateColumn class with default settings.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**
**C#**
```
public DateColumn()
```

**See Also**
DateColumn Class
DateColumn Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# DateColumn Properties

A grid column to display and edit date values.

The following tables list the members exposed by the DateColumn type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | DateColumn | Initializes a new instance of the DateColumn class with default settings. |

### Public Properties

| | Name | Description |
|---|---|---|
| | ActualCaption | Gets the column caption as it is displayed in a column header. (Inherited from **GridColumn**) |
| | ActualWidth | Gets the column width. This is a bindable property. (Inherited from **GridColumn**) |
| | AllowAutoFilter | Gets or sets whether an end-user can filter data by the column's values using the auto-filter panel. This is a bindable property. (Inherited from **GridColumn**) |
| | AllowGroup | Gets or sets whether end-users are allowed to group data by the current column. This is a bindable property. (Inherited from **GridColumn**) |
| | AllowSort | Gets or sets whether end-users are allowed to sort data by the column's values. This is a bindable property. (Inherited from **GridColumn**) |
| | AutoFilterCondition | Gets or sets the comparison operator used to create filter conditions for the current column via the auto-filter panel. This is a bindable property. (Inherited from **GridColumn**) |
| | AutoFilterValue | Gets or sets the value in the auto-filter panel for the column. This is a bindable property. (Inherited from **GridColumn**) |
| | BindingContext | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Caption | Gets or sets the column's display caption. This is a bindable property. (Inherited from **GridColumn**) |
| | ColumnFilterMode | Gets or sets how column values are filtered via the auto-filter panel. This is a bindable property. (Inherited from **GridColumn**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | ContentAlignment | Gets or sets the content alignment within column cells. This is a bindable property. (Inherited from **GridColumn**) |
| | DisplayFormat | Gets or sets formatting for values in the column. This is a bindable property. (Inherited from **GridColumn**) |
| | FieldName | Gets or sets the column's unique field name. For bound columns, this property refers to a field in the underlying data source. For unbound columns, this property serves as a column's identifier.<br><br>This is a bindable property.<br><br>(Inherited from **GridColumn**) |
| | FixedStyle | Gets or sets whether a column is fixed. This is a bindable property. (Inherited from **GridColumn**) |
| | GroupInterval | Gets or sets how data rows are grouped when grouping by the current column is applied. This is a bindable property. (Inherited from **GridColumn**) |
| | HeaderTemplate | Gets or sets a template used to define the column headers visual representation. (Inherited from **GridColumn**) |
| | ImmediateUpdateAutoFilter | Gets or sets whether the column's filter condition is updated as soon as an end-user modifies the text within the auto filter panel's cell. This is a bindable property. (Inherited from **GridColumn**) |
| | IsGrouped | Gets or sets whether the grid's data is grouped by the current column. This is a bindable property. (Inherited from **GridColumn**) |
| | IsNullable | Indicates that the column is bound to the **Nullable<DateTime>** property. |
| | IsReadOnly | Gets or sets whether end-users are allowed to change cell values in the column. This is a bindable property. (Inherited from **GridColumn**) |
| | IsUnbound | Gets whether the column is unbound. (Inherited from **GridColumn**) |
| | IsVisible | Gets or sets whether the column is visible in the grid. This is a bindable property. (Inherited from **GridColumn**) |
| | MinWidth | Gets or sets the minimum column width allowed. This is a bindable property. (Inherited from **GridColumn**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | SerializationTypeName | Gets the string containing the column type name. (Inherited from **GridColumn**) |
| | SortIndex | Gets or sets the column's position among sorted columns. This is a bindable property. (Inherited from **GridColumn**) |
| | SortMode | Specifies how column values are sorted and grouped. This is a bindable property. (Inherited from **GridColumn**) |
| | SortOrder | Gets or sets the column's sort order. This is a bindable property. (Inherited from **GridColumn**) |
| | UnboundExpression | Gets or sets an expression used to evaluate values for the current Unbound Columns. This is a bindable property. (Inherited from **GridColumn**) |
| | UnboundType | Gets or sets the data type and binding mode of the column. This is a bindable property. (Inherited from **GridColumn**) |
| | Width | Gets or sets the column width. This is a bindable property. (Inherited from **GridColumn**) |

**Public Methods**

| | Name | Description |
|---|---|---|
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ToString | (Inherited from **System.Object**) |

**Public Fields**

© 2018 Developer Express Inc.

| | Name | Description |
|---|---|---|
| | ActualWidthProperty | Identifies the GridColumn.ActualWidth bindable property. (Inherited from **GridColumn**) |
| | AllowAutoFilterProperty | Identifies the GridColumn.AllowAutoFilter bindable property. (Inherited from **GridColumn**) |
| | AllowGroupProperty | Identifies the GridColumn.AllowGroup bindable property. (Inherited from **GridColumn**) |
| | AllowSortProperty | Identifies the GridColumn.AllowSort bindable property. (Inherited from **GridColumn**) |
| | AutoFilterConditionProperty | Identifies the GridColumn.AutoFilterCondition bindable property. (Inherited from **GridColumn**) |
| | AutoFilterValueProperty | Identifies the GridColumn.AutoFilterValue bindable property. (Inherited from **GridColumn**) |
| | BindingContextProperty | (Inherited from **Xamarin.Forms.BindableObject**) |
| | CaptionProperty | Identifies the GridColumn.Caption bindable property. (Inherited from **GridColumn**) |
| | ColumnFilterModeProperty | Identifies the GridColumn.ColumnFilterMode bindable property. (Inherited from **GridColumn**) |
| | ContentAlignmentProperty | Identifies the GridColumn.ContentAlignment bindable property. (Inherited from **GridColumn**) |
| | DisplayFormatProperty | Identifies the GridColumn.DisplayFormat bindable property. (Inherited from **GridColumn**) |
| | FieldNameProperty | Identifies the GridColumn.FieldName bindable property. (Inherited from **GridColumn**) |
| | FixedStyleProperty | Identifies the GridColumn.FixedStyle bindable property. (Inherited from **GridColumn**) |
| | GroupIntervalProperty | Identifies the GridColumn.GroupInterval bindable property. (Inherited from **GridColumn**) |
| | HeaderTemplateProperty | Identifies the GridColumn.HeaderTemplate bindable property. (Inherited from **GridColumn**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | ImmediateUpdateAutoFilterProperty | Identifies the GridColumn.ImmediateUpdateAutoFilter bindable property. (Inherited from **GridColumn**) |
| | IsGroupedProperty | Identifies the GridColumn.IsGrouped bindable property. (Inherited from **GridColumn**) |
| | IsReadOnlyProperty | Identifies the GridColumn.IsReadOnly bindable property. (Inherited from **GridColumn**) |
| | IsVisibleProperty | Identifies the GridColumn.IsVisible bindable property. (Inherited from **GridColumn**) |
| | MinWidthProperty | Identifies the GridColumn.MinWidth bindable property. (Inherited from **GridColumn**) |
| | SortIndexProperty | Identifies the GridColumn.SortIndex bindable property. (Inherited from **GridColumn**) |
| | SortModeProperty | Identifies the GridColumn.SortMode bindable property. (Inherited from **GridColumn**) |
| | SortOrderProperty | Identifies the GridColumn.SortOrder bindable property. (Inherited from **GridColumn**) |
| | UnboundExpressionProperty | Identifies the GridColumn.UnboundExpression bindable property. (Inherited from **GridColumn**) |
| | UnboundTypeProperty | Identifies the GridColumn.UnboundType bindable property. (Inherited from **GridColumn**) |
| | WidthProperty | Identifies the GridColumn.Width bindable property. (Inherited from **GridColumn**) |

**Public Events**

| | Name | Description |
|---|---|---|
| | BindingContextChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanging | (Inherited from **Xamarin.Forms.BindableObject**) |

**See Also**
DateColumn Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# IsNullable Property

Indicates that the column is bound to the **Nullable<DateTime>** property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public bool IsNullable { get; set; }
```

### Property Value

**true**, if the column is bound to the **Nullable<DateTime>** property; otherwise, **false**.

### Remarks

You can bind columns of the DateColumn type to both the **DateTime** and the **Nullable<DateTime>** properties. By default, DateColumn expects to be bound to the **DateTime** property. If you need to bind a DateColumn to the **Nullable<DateTime>** property, set the **IsNullable** property to **true**.

### See Also

DateColumn Class
DateColumn Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# EditingRowAction Enumeration

Lists values specifying whether cell values modified via an in-place editor are saved or discarded.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public enum EditingRowAction
```

### Members

| Name | Description |
|---|---|
| Apply | Changed cell values are saved when closing a data row in-place editor. |
| Cancel | Changed cell values are discarded when closing a data row in-place editor. |

### Remarks

Values listed by the **EditingRowAction** enumeration are returned by the RowEditingEventArgs.Action property, which is exposed via the GridControl.EndRowEdit event's parameter.

### See Also

DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# EditValuesContainer Class

Contains information on a data row currently being edited by an end-user via an edit form.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public class EditValuesContainer : object, INotifyPropertyChanged
```

### Remarks

An **EditValuesContainer** object contains information about a data row currently being edited and serves as a data context (binding source) for a template that defines presentation of a row edit form and is assigned to the GridControl.EditFormContent property.

### Example

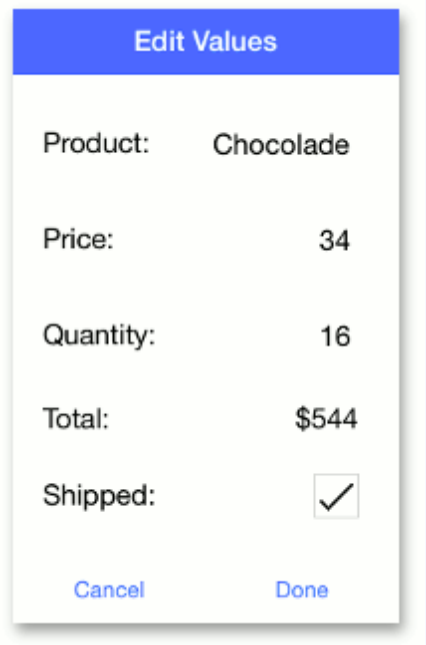
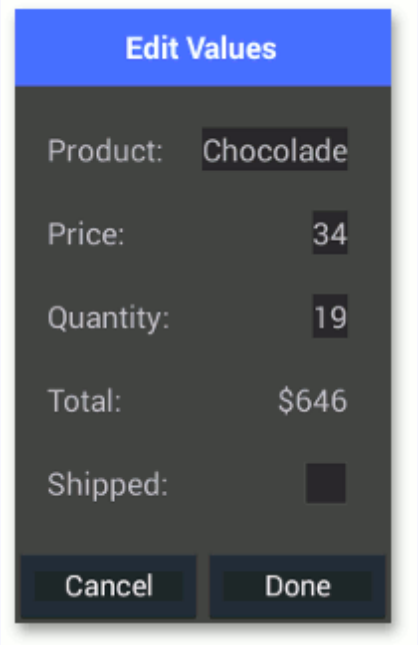
| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T279098. | |

GridControl allows you to substitute the default row edit form with a custom form. This example explains how to do this.

In this application, the grid is bound to a collection of orders. Each data row in the grid represents an individual order and displays the following information:

- Ordered product name (the **Product** column)
- Product unit price (the **Price** column)
- Number of ordered product units (the **Quantity** column)
- Order total value (the **Total** column)
- A value indicating whether an order is shipped (the **Shipped** column)

The default row edit form in this grid looks like the following.

| iOS | Android |
|---|---|
|  |  |



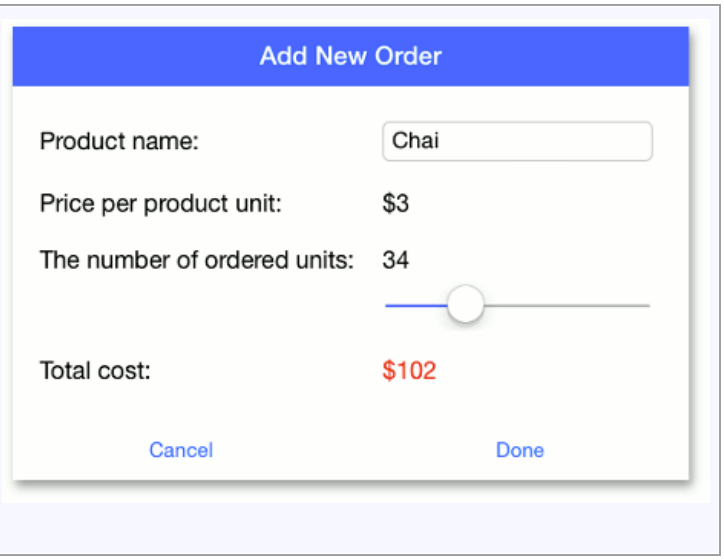
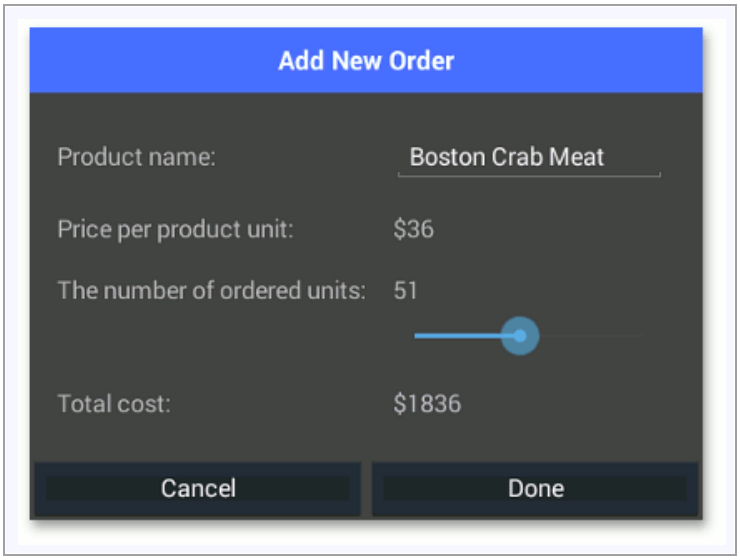


© 2018 Developer Express Inc.

This example demonstrates how to create a custom form to add new data rows and another form to edit existing rows.

- **Add New Order Form**

This form is invoked on tapping New Item Row in the grid and allows an end user to add a new data row (create a new order). On this form, an end user can select a product name via a Picker control, and set the number of ordered product units via a Slider control. A unit price is automatically shown for the selected product, and the order total value is automatically calculated.

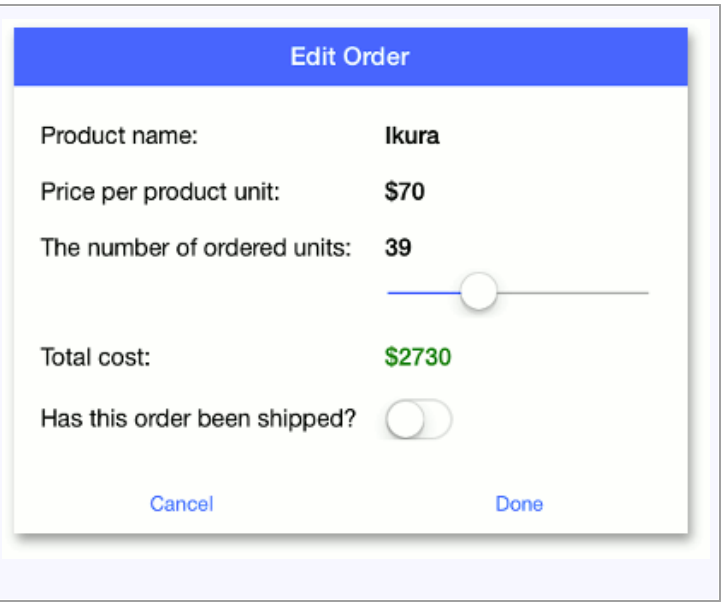
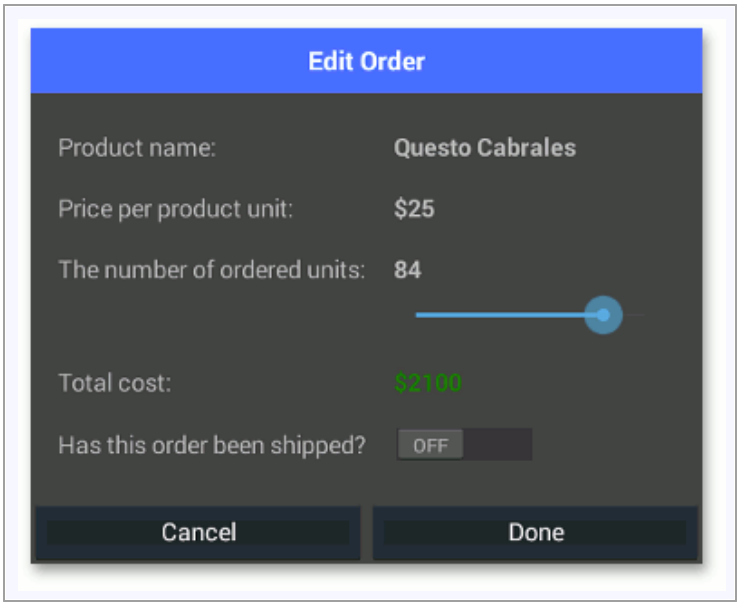
| iOS | Android |
|---|---|
|  |  |





- **Edit Order Form**

This form is invoked on double-tapping a cell or selecting **Edit Cell** from the popup menu. It allows an end user to modify existing data rows. On this form, product name and unit price are shown in Lable controls and cannot be changed. An end user can change the number of ordered product units via a Slider control and use the Switch control to set whether the order is shipped.

| iOS | Android |
|---|---|
|  |  |





To create a custom row edit form and show it instead of the default form, execute the steps below.

1. Specify a custom form view. In this example, it is a ContentView descendant with a set of labels and editors inside (see the *CustomFormEditorContent.cs* file). This view's BindingContext is an EditValuesContainer object. Labels and editors are bound to data fields via the GetBindingPath method with data field names passed as parameters.

The IsNewRow property identifies whether a from is invoked for a new or existing row.

Custom form captions are implemented by using the mechanism of localization.

© 2018 Developer Express Inc.

2. Create a DataTemplate instance for your custom form view and assign this template to the GridControl.EditFormContent property (see the *MainPage.xaml.cs* file).
3. Set the GridControl.RowEditMode property to RowEditMode.Popup to show a form instead of an in-place row editor.

**C#**

```
 (Converters.cs)
using System;
using Xamarin.Forms;
using System.Globalization;
namespace CustomEditFormDemo {
    class Int32ToStringConverter : IValueConverter {
        object IValueConverter.Convert(object value, Type targetType, object parameter, CultureInfo cultu
            return value.ToString();
        }
        object IValueConverter.ConvertBack(object value, Type targetType, object parameter, CultureInfo cu
            int result = 0;
            try {
                result = Int32.Parse(value as String);
            } catch {
                result = 0;
            }
            return result;
        }
    }
    class Int32ToPriceStringConverter : IValueConverter {
        object IValueConverter.Convert(object value, Type targetType, object parameter, CultureInfo cultu
            return "$" + value.ToString();
        }
        object IValueConverter.ConvertBack(object value, Type targetType, object parameter, CultureInfo cu
            int result = 0;
            try {
                result = Int32.Parse(value as String);
            } catch {
                result = 0;
            }
            return result;
        }
    }
    class Int32ToDoubleConverter : IValueConverter {
        object IValueConverter.Convert(object value, Type targetType, object parameter, CultureInfo cultu
            return Convert.ToDouble(value);
        }
        object IValueConverter.ConvertBack(object value, Type targetType, object parameter, CultureInfo cu
            return Convert.ToInt32(value);
        }
    }
    class DoubleToInt32Converter : IValueConverter {
        object IValueConverter.Convert(object value, Type targetType, object parameter, CultureInfo cultu
            return Convert.ToInt32(value);
        }
        object IValueConverter.ConvertBack(object value, Type targetType, object parameter, CultureInfo cu
            return Convert.ToDouble(value);
        }
    }
}
```

**C#**

```
 (CustomFormEditorContent.cs)
using System;
using System.Collections.Generic;
using Xamarin.Forms;
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.DataGrid.Theme;
using System.Globalization;
using DevExpress.Mobile.DataGrid.Localization;
namespace CustomEditFormDemo {
    class EditFormLocalizer : GridLocalizer {
        public void SetEditItemFormCaption() {
            AddString(GridStringId.EditingForm_LabelCaption, "Edit Order");
        }
        public void SetNewItemFormCaption() {
            AddString(GridStringId.EditingForm_LabelCaption, "Add New Order");
        }
    }
    public class CustomEditFormContent : ContentView {
        public TestOrdersRepository ViewModel { get; set; }
        public CustomEditFormContent() {
            HorizontalOptions = LayoutOptions.FillAndExpand;
            VerticalOptions = LayoutOptions.FillAndExpand;
            GridLocalizer.Active = new EditFormLocalizer();
        }
        protected override void OnBindingContextChanged () {
            base.OnBindingContextChanged ();
            EditFormLocalizer localizer = GridLocalizer.Active as EditFormLocalizer;
            if ((BindingContext as EditValuesContainer).IsNewRow) {
                if (localizer != null) {
                    localizer.SetNewItemFormCaption();
                }
                Content = CreateNewItemFormContent();
            }
            else {
                if (localizer != null) {
                    localizer.SetEditItemFormCaption();
                }
                Content = CreateEditItemFormContent ();
            }
        }
        View CreateNewItemFormContent() {
            Grid layoutGrid = CreateGrid(4);
            FillFormGrid(layoutGrid, true);
            return layoutGrid;
        }
        View CreateEditItemFormContent() {
            Grid layoutGrid = CreateGrid(5);
            FillFormGrid(layoutGrid, false);
            return layoutGrid;
        }
        Grid CreateGrid(int rowsCount) {
            Grid layoutGrid = new Grid();
            layoutGrid.ColumnDefinitions.Add(new ColumnDefinition() { Width = GridLength.Auto });
            layoutGrid.ColumnDefinitions.Add(new ColumnDefinition() { Width = new GridLength(200, GridUnit
            for (int i = 0; i < rowsCount; i++) {
                layoutGrid.RowDefinitions.Add(new RowDefinition() { Height = GridLength.Auto });
            }
            layoutGrid.RowSpacing = 20;
            layoutGrid.ColumnSpacing = 20;
            return layoutGrid;
        }
        void FillFormGrid(Grid layoutGrid, bool isNewItemForm) {
            FillLabels(layoutGrid, isNewItemForm);
            FillEditors(layoutGrid, isNewItemForm);
        }
```

```
        void FillLabels(Grid layoutGrid, bool isNewItemForm) {
            Label productName = new Label() { Text = "Product name:",
                                              FontSize = 18, VerticalOptions = LayoutOptions.Center };
            Label productUnitPrice = new Label() { Text = "Price per product unit:",
                                                   FontSize = 18, VerticalOptions = LayoutOptions.Center
            Label quantity = new Label() { Text = "The number of ordered units:",
                                           FontSize = 18, VerticalOptions = LayoutOptions.Start };
            Label total = new Label() { Text = "Total cost:",
                                        FontSize = 18, VerticalOptions = LayoutOptions.Center };
            Label isShipped = null;
            if (!isNewItemForm) {
                isShipped = new Label() { Text = "Has this order been shipped?",
                                          FontSize = 18, VerticalOptions = LayoutOptions.Center };
            }
            layoutGrid.Children.Add(productName, 0, 0);
            layoutGrid.Children.Add(productUnitPrice, 0, 1);
            layoutGrid.Children.Add(quantity, 0, 2);
            layoutGrid.Children.Add(total, 0, 3);
            if (!isNewItemForm) {
                layoutGrid.Children.Add(isShipped, 0, 4);
            }
        }
        void FillEditors(Grid layoutGrid, bool isNewItemForm) {
            View productName;
            // Display a product name in a label in the Edit Item form
            // or change a product name by using a picker in the New Item form.
            if (isNewItemForm) {
                productName = new Picker() { HorizontalOptions = LayoutOptions.FillAndExpand,
                                             VerticalOptions = LayoutOptions.Center};
                InitProductPickerValues(productName as Picker);
            }
            else {
                productName = new Label() { HorizontalOptions = LayoutOptions.FillAndExpand,
                                            VerticalOptions = LayoutOptions.Center,
                                            FontAttributes = FontAttributes.Bold, FontSize = 18 };
                productName.SetBinding(Label.TextProperty,
                                       new Binding(EditValuesContainer.GetBindingPath("Product.Name")));
            }
            // Display a product unit price in a form.
            Int32ToPriceStringConverter priceConverter = new Int32ToPriceStringConverter();
            Label priceLabel = new Label () { HorizontalOptions = LayoutOptions.FillAndExpand,
                                              VerticalOptions = LayoutOptions.Center, FontSize = 18 };
            priceLabel.SetBinding (Label.TextProperty, EditValuesContainer.GetBindingPath ("Product.UnitPr
                                   BindingMode.Default, priceConverter);
            // Modify the number of ordered units via a slider.
            StackLayout quantityStack = new StackLayout () { Orientation = StackOrientation.Vertical,
                                                             VerticalOptions = LayoutOptions.CenterAndExpa
            Label quantityLabel = new Label () { XAlign = TextAlignment.Start,
                                                 VerticalOptions = LayoutOptions.Center, FontSize = 18};
            Slider quantitySlider = new Slider () { Maximum = 99, Minimum = 10 };
            quantitySlider.SetBinding (Slider.ValueProperty, EditValuesContainer.GetBindingPath ("Quantity
                                       BindingMode.TwoWay, new Int32ToDoubleConverter());
            quantityLabel.BindingContext = quantitySlider;
            quantityLabel.SetBinding(Label.TextProperty, "Value", BindingMode.OneWay, new DoubleToInt32Co
            quantityStack.Children.Add(quantityLabel);
            quantityStack.Children.Add(quantitySlider);
            // Display the order total value calculated automatically.
            Label totalLabel = new Label () { XAlign = TextAlignment.Start,
                                              VerticalOptions = LayoutOptions.Center, FontSize = 18 };
            totalLabel.PropertyChanged += OnTotalPropertyChanged;
            totalLabel.SetBinding(Label.TextProperty, EditValuesContainer.GetBindingPath("Total"),
                                  BindingMode.OneWay, priceConverter);
            if (!isNewItemForm) {
                priceLabel.FontAttributes = FontAttributes.Bold;
                quantityLabel.FontAttributes = FontAttributes.Bold;
```

© 2018 Developer Express Inc.

```
                totalLabel.FontAttributes = FontAttributes.Bold;
            }
            // Add created elements to a form.
            layoutGrid.Children.Add(productName, 1, 0);
            layoutGrid.Children.Add(priceLabel, 1, 1);
            layoutGrid.Children.Add(quantityStack, 1, 2);
            layoutGrid.Children.Add(totalLabel, 1, 3);
            // Show a switch in the Edit Item form allowing end-users to set whether or not an order is sh
            Switch isShipped = new Switch() { VerticalOptions = LayoutOptions.Center,
                                              HorizontalOptions = LayoutOptions.Start };
            isShipped.SetBinding(Switch.IsToggledProperty, EditValuesContainer.GetBindingPath("Shipped"));
            if (!isNewItemForm) {
                layoutGrid.Children.Add(isShipped, 1, 4);
            }
        }
        void InitProductPickerValues(Picker picker) {
            EditValuesContainer valuesContainer = BindingContext as EditValuesContainer;
            String productName = valuesContainer.Values["Product.Name"].ToString();
            int selectedItemIndex = -1;
            foreach (Product product in ViewModel.Products) {
                if (productName == product.Name) {
                    selectedItemIndex = picker.Items.Count;
                }
                picker.Items.Add(product.Name);
            }
            picker.SelectedIndexChanged += OnProductNameSelectedIndexChanged;
            if (selectedItemIndex >= 0) {
                picker.SelectedIndex = selectedItemIndex;
            }
        }
        void OnProductNameSelectedIndexChanged(object sender, EventArgs e) {
            EditValuesContainer valuesContainer = BindingContext as EditValuesContainer;
            Picker picker = sender as Picker;
            if (picker.SelectedIndex < 0) {
                return;
            }
            String productName = picker.Items[picker.SelectedIndex];
            valuesContainer.Values["Product.Name"] = productName;
            Product product = ViewModel.Products.Find(p => p.Name == productName);
            valuesContainer.Values["Product.UnitPrice"] = product.UnitPrice;
        }
        void OnTotalPropertyChanged (object sender, System.ComponentModel.PropertyChangedEventArgs e)
        {
            Label label = (sender as Label);
            if (e.PropertyName == Label.TextProperty.PropertyName && !String.IsNullOrWhiteSpace (label.Tex
                Int32 value = (Int32)((label.BindingContext as EditValuesContainer).Values ["Total"]);
                if (value > 2000) {
                    label.TextColor = Color.Green; }
                else {
                    if (value < 500) {
                        label.TextColor = Color.Red; }
                    else {
                        label.TextColor = ThemeManager.Theme.CellCustomizer.Font.Color;}
                }
            }
        }
    }
}
```

**C#**

```
 (MainPage.xaml.cs)
using System;
using System.Collections.Generic;
using Xamarin.Forms;
using DevExpress.Mobile.DataGrid;
namespace CustomEditFormDemo
{
    public partial class MainPage : ContentPage
    {
        TestOrdersRepository model;
        public MainPage ()
        {
            InitializeComponent ();
            model = new TestOrdersRepository ();
            BindingContext = model;
            DataTemplate editFormContentTemplate = new DataTemplate(GetTem
            grid.EditFormContent = editFormContentTemplate;
            grid.RowEditMode = RowEditMode.Popup;
        }
        object GetTemplate() {
            return new CustomEditFormContent() { ViewModel = model };
        }
        void OnInitNewItemRow(object sender, InitNewRowEventArgs e) {
            e.EditableRowData.SetFieldValue("Quantity", model.MinimumQuant
            e.EditableRowData.SetFieldValue("Product.Name", model.Products
        }
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<?xml version="1.0" encoding="UTF-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="CustomEditFormDemo.MainPage"
             xmlns:dxGrid="clr-namespace:DevExpress.Mobile.DataGrid;assemb
      <ContentPage.Padding>
        <OnPlatform x:TypeArguments="Thickness" iOS="0, 20, 0, 0" />
    </ContentPage.Padding>
    <dxGrid:GridControl x:Name="grid" ItemsSource="{Binding Orders}"
                        NewItemRowVisibility="true" InitNewRow="OnInitNew
            <dxGrid:GridControl.Columns>
             <dxGrid:TextColumn FieldName="Product.Name" Caption="Product"
             <dxGrid:NumberColumn FieldName="Product.UnitPrice" Caption="P
             <dxGrid:NumberColumn FieldName="Quantity"/>
             <dxGrid:NumberColumn FieldName="Total"
                                  UnboundType="Integer" UnboundExpression=
                                  DisplayFormat="C0" IsReadOnly="True"/>
             <dxGrid:SwitchColumn FieldName="Shipped"/>
          </dxGrid:GridControl.Columns>
    </dxGrid:GridControl>
</ContentPage>
```

© 2018 Developer Express Inc.

## Inheritance Hierarchy

System.Object
    **EditValuesContainer**

## See Also

EditValuesContainer Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# EditValuesContainer Members

Contains information on a data row currently being edited by an end-user via an edit form.

The following tables list the members exposed by the EditValuesContainer type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | EditValuesContainer | Initializes a new instance of the EditValuesContainer class. |

### Public Properties

| | Name | Description |
|---|---|---|
| | IsNewRow | Indicates whether an edit form is invoked to create a new data row or to edit an existing row. |
| | RowData | Gets an object that contains information on the currently edited row. |
| | Values | Provides access to data values currently set in a row edit form. |

### Public Methods

| | Name | Description |
|---|---|---|
| | ContainsValueForField | For internal use. |
| S | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| S | GetBindingPath | Allows you to bind a custom edit form's editor to a grid's data value. |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | Raise | For internal use. |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

### Public Events

| | Name | Description |
|---|---|---|
| | PropertyChanged | For internal use. |

**See Also**
EditValuesContainer Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# EditValuesContainer Constructor

Initializes a new instance of the EditValuesContainer class.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**
**C#**

```
public EditValuesContainer()
```

**See Also**
EditValuesContainer Class
EditValuesContainer Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# EditValuesContainer Properties

Contains information on a data row currently being edited by an end-user via an edit form.

The following tables list the members exposed by the EditValuesContainer type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | EditValuesContainer | Initializes a new instance of the EditValuesContainer class. |

### Public Properties

| | Name | Description |
|---|---|---|
| | IsNewRow | Indicates whether an edit form is invoked to create a new data row or to edit an existing row. |
| | RowData | Gets an object that contains information on the currently edited row. |
| | Values | Provides access to data values currently set in a row edit form. |

### Public Methods

| | Name | Description |
|---|---|---|
| | ContainsValueForField | For internal use. |
| S | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| S | GetBindingPath | Allows you to bind a custom edit form's editor to a grid's data value. |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | Raise | For internal use. |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

### Public Events

| | Name | Description |
|---|---|---|
| | PropertyChanged | For internal use. |

**See Also**
EditValuesContainer Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# IsNewRow Property

Indicates whether an edit form is invoked to create a new data row or to edit an existing row.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public bool IsNewRow { get; }
```

### Property Value

**true**, if a row currently being processed in the edit form is new; otherwise, **false**.

### Example

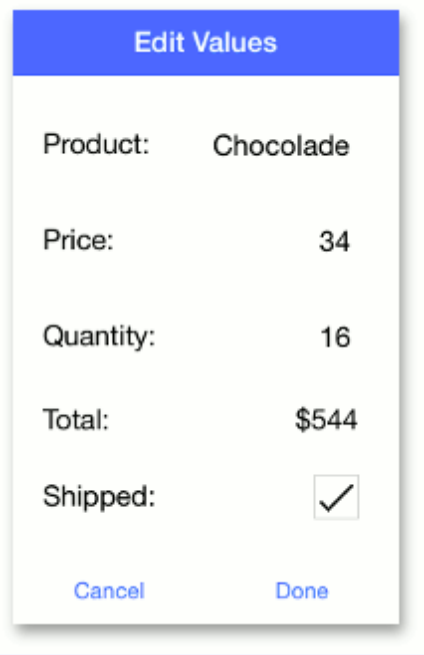
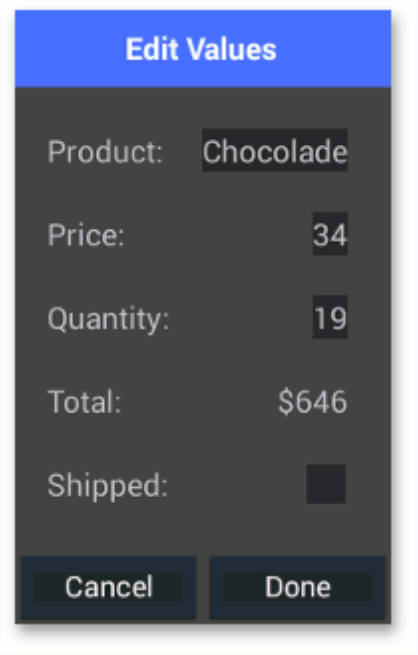
| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T279098. | |

GridControl allows you to substitute the default row edit form with a custom form. This example explains how to do this.

In this application, the grid is bound to a collection of orders. Each data row in the grid represents an individual order and displays the following information:

- Ordered product name (the **Product** column)
- Product unit price (the **Price** column)
- Number of ordered product units (the **Quantity** column)
- Order total value (the **Total** column)
- A value indicating whether an order is shipped (the **Shipped** column)

The default row edit form in this grid looks like the following.

| iOS | Android |
|---|---|
|  |  |



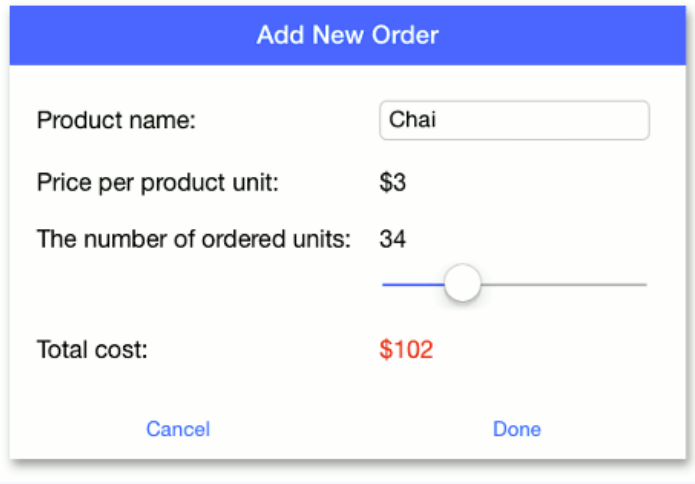
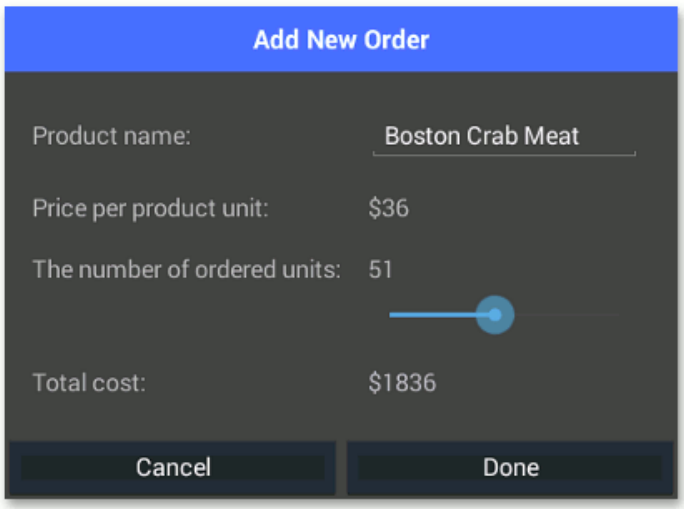


This example demonstrates how to create a custom form to add new data rows and another form to edit existing rows.

- **Add New Order Form**

© 2018 Developer Express Inc.

This form is invoked on tapping New Item Row in the grid and allows an end user to add a new data row (create a new order). On this form, an end user can select a product name via a Picker control, and set the number of ordered product units via a Slider control. A unit price is automatically shown for the selected product, and the order total value is automatically calculated.

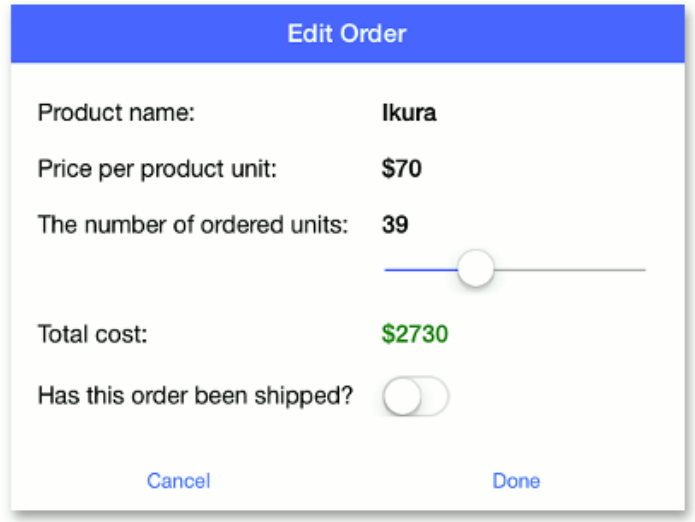
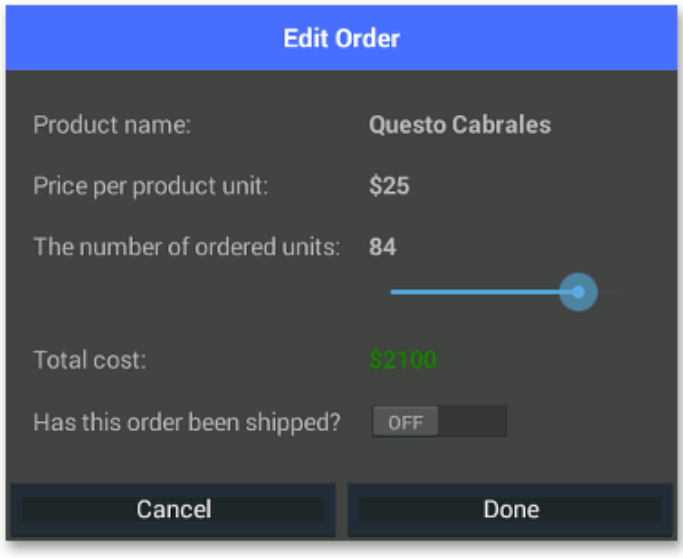
| iOS | Android |
| --- | --- |
|  |  |





- **Edit Order Form**

This form is invoked on double-tapping a cell or selecting **Edit Cell** from the popup menu. It allows an end user to modify existing data rows. On this form, product name and unit price are shown in Lable controls and cannot be changed. An end user can change the number of ordered product units via a Slider control and use the Switch control to set whether the order is shipped.

| iOS | Android |
| --- | --- |
|  |  |





To create a custom row edit form and show it instead of the default form, execute the steps below.

1. Specify a custom form view. In this example, it is a ContentView descendant with a set of labels and editors inside (see the *CustomFormEditorContent.cs* file). This view's BindingContext is an EditValuesContainer object. Labels and editors are bound to data fields via the GetBindingPath method with data field names passed as parameters.

The IsNewRow property identifies whether a from is invoked for a new or existing row.

Custom form captions are implemented by using the mechanism of localization.

2. Create a DataTemplate instance for your custom form view and assign this template to the GridControl.EditFormContent property (see the *MainPage.xaml.cs* file).

© 2018 Developer Express Inc.

3. Set the GridControl.RowEditMode property to RowEditMode.Popup to show a form instead of an in-place row editor.

**C#**

```
 (Converters.cs)
using System;
using Xamarin.Forms;
using System.Globalization;
namespace CustomEditFormDemo {
    class Int32ToStringConverter : IValueConverter {
        object IValueConverter.Convert(object value, Type targetType, object parameter, CultureInfo cultu
            return value.ToString();
        }
        object IValueConverter.ConvertBack(object value, Type targetType, object parameter, CultureInfo cu
            int result = 0;
            try {
                result = Int32.Parse(value as String);
            } catch {
                result = 0;
            }
            return result;
        }
    }
    class Int32ToPriceStringConverter : IValueConverter {
        object IValueConverter.Convert(object value, Type targetType, object parameter, CultureInfo cultu
            return "$" + value.ToString();
        }
        object IValueConverter.ConvertBack(object value, Type targetType, object parameter, CultureInfo cu
            int result = 0;
            try {
                result = Int32.Parse(value as String);
            } catch {
                result = 0;
            }
            return result;
        }
    }
    class Int32ToDoubleConverter : IValueConverter {
        object IValueConverter.Convert(object value, Type targetType, object parameter, CultureInfo cultu
            return Convert.ToDouble(value);
        }
        object IValueConverter.ConvertBack(object value, Type targetType, object parameter, CultureInfo cu
            return Convert.ToInt32(value);
        }
    }
    class DoubleToInt32Converter : IValueConverter {
        object IValueConverter.Convert(object value, Type targetType, object parameter, CultureInfo cultu
            return Convert.ToInt32(value);
        }
        object IValueConverter.ConvertBack(object value, Type targetType, object parameter, CultureInfo cu
            return Convert.ToDouble(value);
        }
    }
}
```

**C#**

```
 (CustomFormEditorContent.cs)
using System;
using System.Collections.Generic;
using Xamarin.Forms;
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.DataGrid.Theme;
using System.Globalization;
using DevExpress.Mobile.DataGrid.Localization;
namespace CustomEditFormDemo {
    class EditFormLocalizer : GridLocalizer {
        public void SetEditItemFormCaption() {
            AddString(GridStringId.EditingForm_LabelCaption, "Edit Order");
        }
        public void SetNewItemFormCaption() {
            AddString(GridStringId.EditingForm_LabelCaption, "Add New Order");
        }
    }
    public class CustomEditFormContent : ContentView {
        public TestOrdersRepository ViewModel { get; set; }
        public CustomEditFormContent() {
            HorizontalOptions = LayoutOptions.FillAndExpand;
            VerticalOptions = LayoutOptions.FillAndExpand;
            GridLocalizer.Active = new EditFormLocalizer();
        }
        protected override void OnBindingContextChanged () {
            base.OnBindingContextChanged ();
            EditFormLocalizer localizer = GridLocalizer.Active as EditFormLocalizer;
            if ((BindingContext as EditValuesContainer).IsNewRow) {
                if (localizer != null) {
                    localizer.SetNewItemFormCaption();
                }
                Content = CreateNewItemFormContent();
            }
            else {
                if (localizer != null) {
                    localizer.SetEditItemFormCaption();
                }
                Content = CreateEditItemFormContent ();
            }
        }
        View CreateNewItemFormContent() {
            Grid layoutGrid = CreateGrid(4);
            FillFormGrid(layoutGrid, true);
            return layoutGrid;
        }
        View CreateEditItemFormContent() {
            Grid layoutGrid = CreateGrid(5);
            FillFormGrid(layoutGrid, false);
            return layoutGrid;
        }
        Grid CreateGrid(int rowsCount) {
            Grid layoutGrid = new Grid();
            layoutGrid.ColumnDefinitions.Add(new ColumnDefinition() { Width = GridLength.Auto });
            layoutGrid.ColumnDefinitions.Add(new ColumnDefinition() { Width = new GridLength(200, GridUnit
            for (int i = 0; i < rowsCount; i++) {
                layoutGrid.RowDefinitions.Add(new RowDefinition() { Height = GridLength.Auto });
            }
            layoutGrid.RowSpacing = 20;
            layoutGrid.ColumnSpacing = 20;
            return layoutGrid;
        }
        void FillFormGrid(Grid layoutGrid, bool isNewItemForm) {
            FillLabels(layoutGrid, isNewItemForm);
            FillEditors(layoutGrid, isNewItemForm);
        }
```

© 2018 Developer Express Inc.

```
        void FillLabels(Grid layoutGrid, bool isNewItemForm) {
            Label productName = new Label() { Text = "Product name:",
                                              FontSize = 18, VerticalOptions = LayoutOptions.Center };
            Label productUnitPrice = new Label() { Text = "Price per product unit:",
                                                   FontSize = 18, VerticalOptions = LayoutOptions.Center
            Label quantity = new Label() { Text = "The number of ordered units:",
                                           FontSize = 18, VerticalOptions = LayoutOptions.Start };
            Label total = new Label() { Text = "Total cost:",
                                        FontSize = 18, VerticalOptions = LayoutOptions.Center };
            Label isShipped = null;
            if (!isNewItemForm) {
                isShipped = new Label() { Text = "Has this order been shipped?",
                                          FontSize = 18, VerticalOptions = LayoutOptions.Center };
            }
            layoutGrid.Children.Add(productName, 0, 0);
            layoutGrid.Children.Add(productUnitPrice, 0, 1);
            layoutGrid.Children.Add(quantity, 0, 2);
            layoutGrid.Children.Add(total, 0, 3);
            if (!isNewItemForm) {
                layoutGrid.Children.Add(isShipped, 0, 4);
            }
        }
        void FillEditors(Grid layoutGrid, bool isNewItemForm) {
            View productName;
            // Display a product name in a label in the Edit Item form
            // or change a product name by using a picker in the New Item form.
            if (isNewItemForm) {
                productName = new Picker() { HorizontalOptions = LayoutOptions.FillAndExpand,
                                             VerticalOptions = LayoutOptions.Center};
                InitProductPickerValues(productName as Picker);
            }
            else {
                productName = new Label() { HorizontalOptions = LayoutOptions.FillAndExpand,
                                            VerticalOptions = LayoutOptions.Center,
                                            FontAttributes = FontAttributes.Bold, FontSize = 18 };
                productName.SetBinding(Label.TextProperty,
                                       new Binding(EditValuesContainer.GetBindingPath("Product.Name")));
            }
            // Display a product unit price in a form.
            Int32ToPriceStringConverter priceConverter = new Int32ToPriceStringConverter();
            Label priceLabel = new Label () { HorizontalOptions = LayoutOptions.FillAndExpand,
                                              VerticalOptions = LayoutOptions.Center, FontSize = 18 };
            priceLabel.SetBinding (Label.TextProperty, EditValuesContainer.GetBindingPath ("Product.UnitPr
                                   BindingMode.Default, priceConverter);
            // Modify the number of ordered units via a slider.
            StackLayout quantityStack = new StackLayout () { Orientation = StackOrientation.Vertical,
                                                             VerticalOptions = LayoutOptions.CenterAndExpa
            Label quantityLabel = new Label () { XAlign = TextAlignment.Start,
                                                 VerticalOptions = LayoutOptions.Center, FontSize = 18};
            Slider quantitySlider = new Slider () { Maximum = 99, Minimum = 10 };
            quantitySlider.SetBinding (Slider.ValueProperty, EditValuesContainer.GetBindingPath ("Quantity
                                       BindingMode.TwoWay, new Int32ToDoubleConverter());
            quantityLabel.BindingContext = quantitySlider;
            quantityLabel.SetBinding(Label.TextProperty, "Value", BindingMode.OneWay, new DoubleToInt32Co
            quantityStack.Children.Add(quantityLabel);
            quantityStack.Children.Add(quantitySlider);
            // Display the order total value calculated automatically.
            Label totalLabel = new Label () { XAlign = TextAlignment.Start,
                                              VerticalOptions = LayoutOptions.Center, FontSize = 18 };
            totalLabel.PropertyChanged += OnTotalPropertyChanged;
            totalLabel.SetBinding(Label.TextProperty, EditValuesContainer.GetBindingPath("Total"),
                                  BindingMode.OneWay, priceConverter);
            if (!isNewItemForm) {
                priceLabel.FontAttributes = FontAttributes.Bold;
                quantityLabel.FontAttributes = FontAttributes.Bold;
```

```
                totalLabel.FontAttributes = FontAttributes.Bold;
            }
            // Add created elements to a form.
            layoutGrid.Children.Add(productName, 1, 0);
            layoutGrid.Children.Add(priceLabel, 1, 1);
            layoutGrid.Children.Add(quantityStack, 1, 2);
            layoutGrid.Children.Add(totalLabel, 1, 3);
            // Show a switch in the Edit Item form allowing end-users to set whether or not an order is sh
            Switch isShipped = new Switch() { VerticalOptions = LayoutOptions.Center,
                                              HorizontalOptions = LayoutOptions.Start };
            isShipped.SetBinding(Switch.IsToggledProperty, EditValuesContainer.GetBindingPath("Shipped"));
            if (!isNewItemForm) {
                layoutGrid.Children.Add(isShipped, 1, 4);
            }
        }
        void InitProductPickerValues(Picker picker) {
            EditValuesContainer valuesContainer = BindingContext as EditValuesContainer;
            String productName = valuesContainer.Values["Product.Name"].ToString();
            int selectedItemIndex = -1;
            foreach (Product product in ViewModel.Products) {
                if (productName == product.Name) {
                    selectedItemIndex = picker.Items.Count;
                }
                picker.Items.Add(product.Name);
            }
            picker.SelectedIndexChanged += OnProductNameSelectedIndexChanged;
            if (selectedItemIndex >= 0) {
                picker.SelectedIndex = selectedItemIndex;
            }
        }
        void OnProductNameSelectedIndexChanged(object sender, EventArgs e) {
            EditValuesContainer valuesContainer = BindingContext as EditValuesContainer;
            Picker picker = sender as Picker;
            if (picker.SelectedIndex < 0) {
                return;
            }
            String productName = picker.Items[picker.SelectedIndex];
            valuesContainer.Values["Product.Name"] = productName;
            Product product = ViewModel.Products.Find(p => p.Name == productName);
            valuesContainer.Values["Product.UnitPrice"] = product.UnitPrice;
        }
        void OnTotalPropertyChanged (object sender, System.ComponentModel.PropertyChangedEventArgs e)
        {
            Label label = (sender as Label);
            if (e.PropertyName == Label.TextProperty.PropertyName && !String.IsNullOrWhiteSpace (label.Tex
                Int32 value = (Int32)((label.BindingContext as EditValuesContainer).Values ["Total"]);
                if (value > 2000) {
                    label.TextColor = Color.Green; }
                else {
                    if (value < 500) {
                        label.TextColor = Color.Red; }
                    else {
                        label.TextColor = ThemeManager.Theme.CellCustomizer.Font.Color;}
                }
            }
        }
    }
}
```

**C#**

```
 (MainPage.xaml.cs)
using System;
using System.Collections.Generic;
using Xamarin.Forms;
using DevExpress.Mobile.DataGrid;
namespace CustomEditFormDemo
{
    public partial class MainPage : ContentPage
    {
        TestOrdersRepository model;
        public MainPage ()
        {
            InitializeComponent ();
            model = new TestOrdersRepository ();
            BindingContext = model;
            DataTemplate editFormContentTemplate = new DataTemplate(GetTem
            grid.EditFormContent = editFormContentTemplate;
            grid.RowEditMode = RowEditMode.Popup;
        }
        object GetTemplate() {
            return new CustomEditFormContent() { ViewModel = model };
        }
        void OnInitNewItemRow(object sender, InitNewRowEventArgs e) {
            e.EditableRowData.SetFieldValue("Quantity", model.MinimumQuant
            e.EditableRowData.SetFieldValue("Product.Name", model.Products
        }
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<?xml version="1.0" encoding="UTF-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="CustomEditFormDemo.MainPage"
             xmlns:dxGrid="clr-namespace:DevExpress.Mobile.DataGrid;assemb
      <ContentPage.Padding>
        <OnPlatform x:TypeArguments="Thickness" iOS="0, 20, 0, 0" />
    </ContentPage.Padding>
    <dxGrid:GridControl x:Name="grid" ItemsSource="{Binding Orders}"
                        NewItemRowVisibility="true" InitNewRow="OnInitNew
            <dxGrid:GridControl.Columns>
             <dxGrid:TextColumn FieldName="Product.Name" Caption="Product"
             <dxGrid:NumberColumn FieldName="Product.UnitPrice" Caption="P
             <dxGrid:NumberColumn FieldName="Quantity"/>
             <dxGrid:NumberColumn FieldName="Total"
                                  UnboundType="Integer" UnboundExpression=
                                  DisplayFormat="C0" IsReadOnly="True"/>
             <dxGrid:SwitchColumn FieldName="Shipped"/>
          </dxGrid:GridControl.Columns>
    </dxGrid:GridControl>
</ContentPage>
```

© 2018 Developer Express Inc.

## See Also

EditValuesContainer Class
EditValuesContainer Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# RowData Property

Gets an object that contains information on the currently edited row.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public IRowData RowData { get; }
```

### Property Value

An object implementing the IRowData interface.

### Remarks

The IRowData.RowHandle property returns the edited row handle. To obtain an object representing a record in the grid's underlying data source, use the IRowData.DataObject property. To get a value in the specified field, use the IRowData.GetFieldValue method.

### See Also

EditValuesContainer Class
EditValuesContainer Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# Values Property

Provides access to data values currently set in a row edit form.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public ObservableDictionaryMaskingDot<Object> Values { get; }
```

### Property Value

A dictionary of edit data values.

### Remarks

Use the **Values** property to obtain data values currently set in an edit form. A value set for a grid's column in the corresponding editor of the form is accessed by this column's field name.

### Example

| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T279098. | |

GridControl allows you to substitute the default row edit form with a custom form. This example explains how to do this.

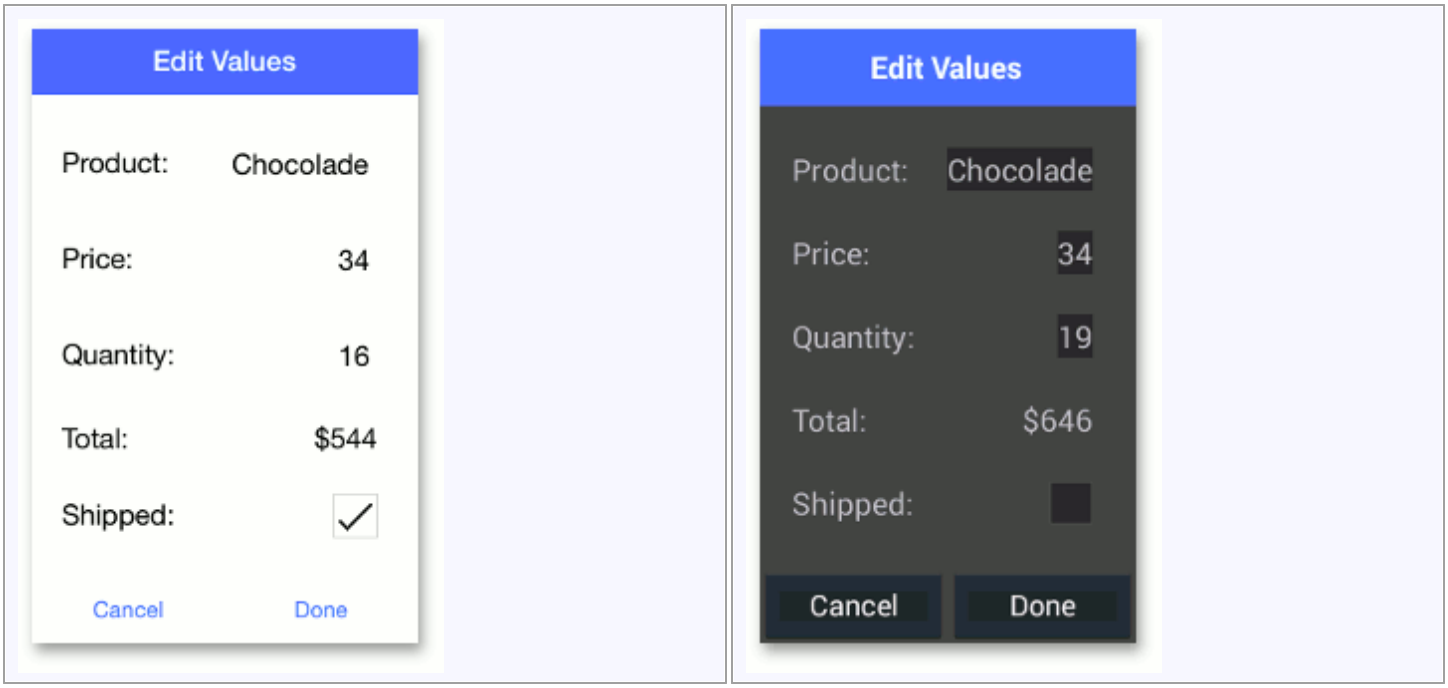
In this application, the grid is bound to a collection of orders. Each data row in the grid represents an individual order and displays the following information:

- Ordered product name (the **Product** column)
- Product unit price (the **Price** column)
- Number of ordered product units (the **Quantity** column)
- Order total value (the **Total** column)
- A value indicating whether an order is shipped (the **Shipped** column)

The default row edit form in this grid looks like the following.

| iOS | Android |
|---|---|

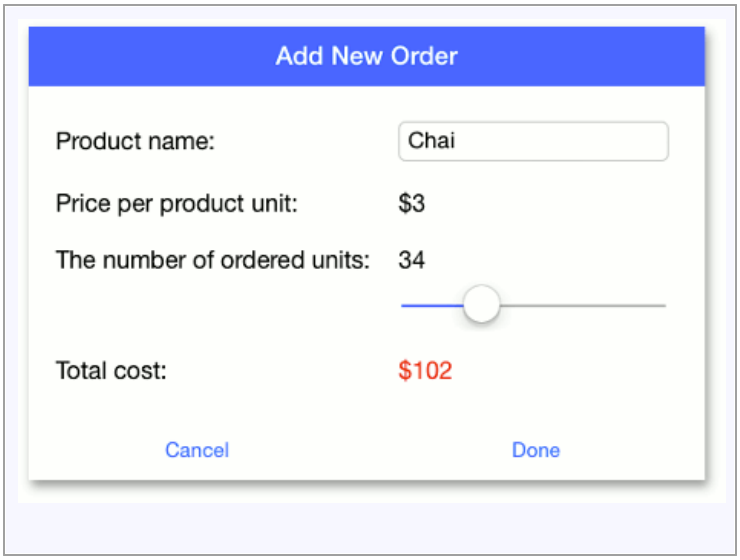
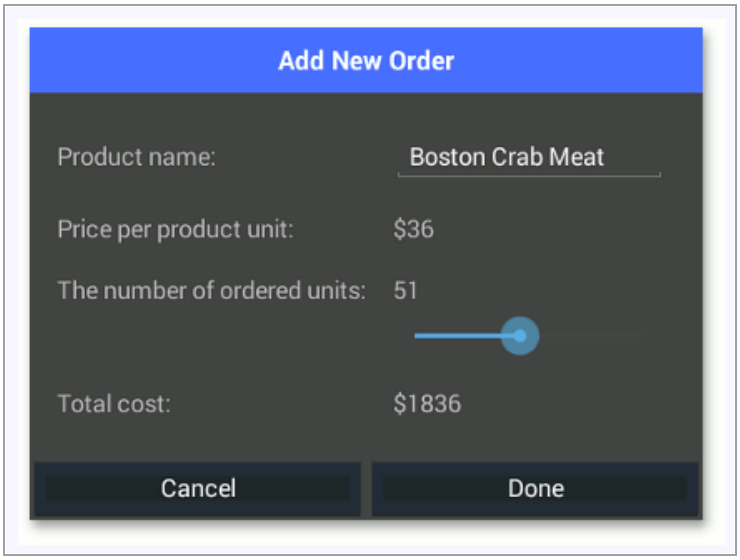
© 2018 Developer Express Inc.




This example demonstrates how to create a custom form to add new data rows and another form to edit existing rows.

- **Add New Order Form**

This form is invoked on tapping New Item Row in the grid and allows an end user to add a new data row (create a new order). On this form, an end user can select a product name via a Picker control, and set the number of ordered product units via a Slider control. A unit price is automatically shown for the selected product, and the order total value is automatically calculated.

| iOS | Android |
|---|---|
|  |  |

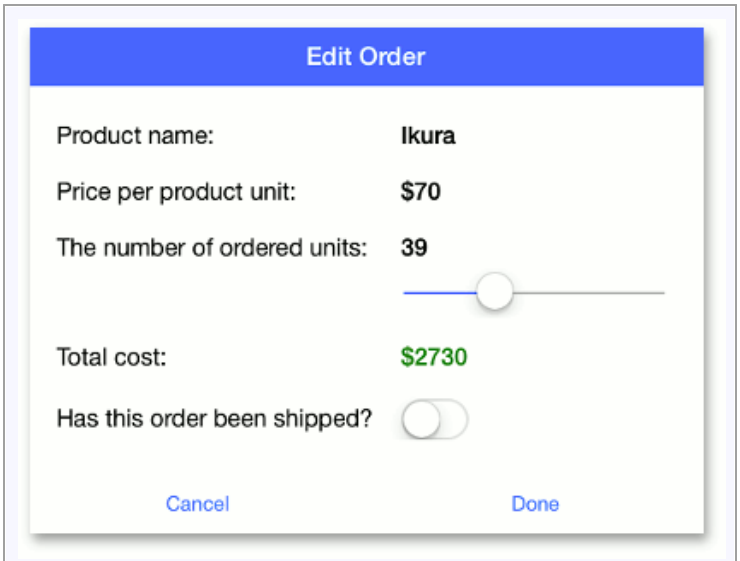


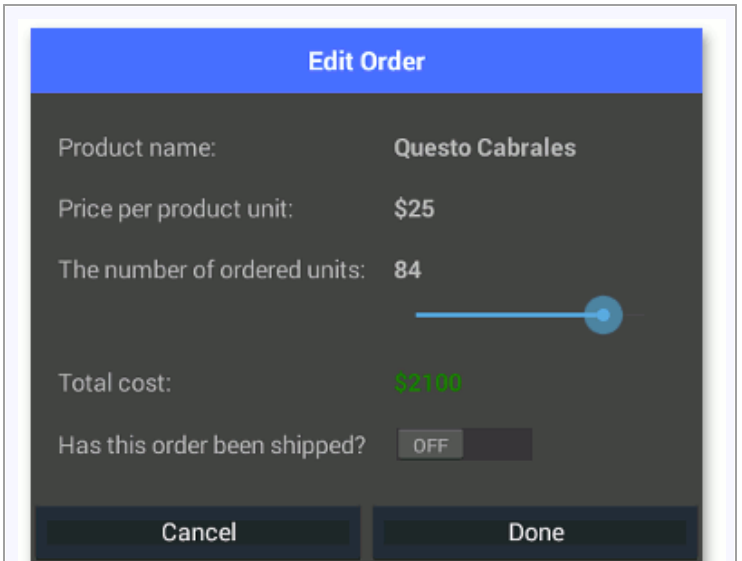
- **Edit Order Form**

This form is invoked on double-tapping a cell or selecting **Edit Cell** from the popup menu. It allows an end user to modify existing data rows. On this form, product name and unit price are shown in Lable controls and cannot be changed. An end user can change the number of ordered product units via a Slider control and use the Switch control to set whether the order is shipped.

| iOS | Android |
|---|---|

© 2018 Developer Express Inc.




To create a custom row edit form and show it instead of the default form, execute the steps below.

1. Specify a custom form view. In this example, it is a ContentView descendant with a set of labels and editors inside (see the *CustomFormEditorContent.cs* file). This view's BindingContext is an EditValuesContainer object. Labels and editors are bound to data fields via the GetBindingPath method with data field names passed as parameters.

The IsNewRow property identifies whether a from is invoked for a new or existing row.

Custom form captions are implemented by using the mechanism of localization.

2. Create a DataTemplate instance for your custom form view and assign this template to the GridControl.EditFormContent property (see the *MainPage.xaml.cs* file).
3. Set the GridControl.RowEditMode property to RowEditMode.Popup to show a form instead of an in-place row editor.

| C# | |
|---|---|

© 2018 Developer Express Inc.

```
 (Converters.cs)
using System;
using Xamarin.Forms;
using System.Globalization;
namespace CustomEditFormDemo {
    class Int32ToStringConverter : IValueConverter {
        object IValueConverter.Convert(object value, Type targetType, object parameter, CultureInfo cultu
            return value.ToString();
        }
        object IValueConverter.ConvertBack(object value, Type targetType, object parameter, CultureInfo cu
            int result = 0;
            try {
                result = Int32.Parse(value as String);
            } catch {
                result = 0;
            }
            return result;
        }
    }
    class Int32ToPriceStringConverter : IValueConverter {
        object IValueConverter.Convert(object value, Type targetType, object parameter, CultureInfo cultu
            return "$" + value.ToString();
        }
        object IValueConverter.ConvertBack(object value, Type targetType, object parameter, CultureInfo cu
            int result = 0;
            try {
                result = Int32.Parse(value as String);
            } catch {
                result = 0;
            }
            return result;
        }
    }
    class Int32ToDoubleConverter : IValueConverter {
        object IValueConverter.Convert(object value, Type targetType, object parameter, CultureInfo cultu
            return Convert.ToDouble(value);
        }
        object IValueConverter.ConvertBack(object value, Type targetType, object parameter, CultureInfo cu
            return Convert.ToInt32(value);
        }
    }
    class DoubleToInt32Converter : IValueConverter {
        object IValueConverter.Convert(object value, Type targetType, object parameter, CultureInfo cultu
            return Convert.ToInt32(value);
        }
        object IValueConverter.ConvertBack(object value, Type targetType, object parameter, CultureInfo cu
            return Convert.ToDouble(value);
        }
    }
}
```

| C# | |
|---|---|

© 2018 Developer Express Inc.

```
 (CustomFormEditorContent.cs)
using System;
using System.Collections.Generic;
using Xamarin.Forms;
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.DataGrid.Theme;
using System.Globalization;
using DevExpress.Mobile.DataGrid.Localization;
namespace CustomEditFormDemo {
    class EditFormLocalizer : GridLocalizer {
        public void SetEditItemFormCaption() {
            AddString(GridStringId.EditingForm_LabelCaption, "Edit Order");
        }
        public void SetNewItemFormCaption() {
            AddString(GridStringId.EditingForm_LabelCaption, "Add New Order");
        }
    }
    public class CustomEditFormContent : ContentView {
        public TestOrdersRepository ViewModel { get; set; }
        public CustomEditFormContent() {
            HorizontalOptions = LayoutOptions.FillAndExpand;
            VerticalOptions = LayoutOptions.FillAndExpand;
            GridLocalizer.Active = new EditFormLocalizer();
        }
        protected override void OnBindingContextChanged () {
            base.OnBindingContextChanged ();
            EditFormLocalizer localizer = GridLocalizer.Active as EditFormLocalizer;
            if ((BindingContext as EditValuesContainer).IsNewRow) {
                if (localizer != null) {
                    localizer.SetNewItemFormCaption();
                }
                Content = CreateNewItemFormContent();
            }
            else {
                if (localizer != null) {
                    localizer.SetEditItemFormCaption();
                }
                Content = CreateEditItemFormContent ();
            }
        }
        View CreateNewItemFormContent() {
            Grid layoutGrid = CreateGrid(4);
            FillFormGrid(layoutGrid, true);
            return layoutGrid;
        }
        View CreateEditItemFormContent() {
            Grid layoutGrid = CreateGrid(5);
            FillFormGrid(layoutGrid, false);
            return layoutGrid;
        }
        Grid CreateGrid(int rowsCount) {
            Grid layoutGrid = new Grid();
            layoutGrid.ColumnDefinitions.Add(new ColumnDefinition() { Width = GridLength.Auto });
            layoutGrid.ColumnDefinitions.Add(new ColumnDefinition() { Width = new GridLength(200, GridUnit
            for (int i = 0; i < rowsCount; i++) {
                layoutGrid.RowDefinitions.Add(new RowDefinition() { Height = GridLength.Auto });
            }
            layoutGrid.RowSpacing = 20;
            layoutGrid.ColumnSpacing = 20;
            return layoutGrid;
        }
        void FillFormGrid(Grid layoutGrid, bool isNewItemForm) {
            FillLabels(layoutGrid, isNewItemForm);
            FillEditors(layoutGrid, isNewItemForm);
        }
```

```
        void FillLabels(Grid layoutGrid, bool isNewItemForm) {
            Label productName = new Label() { Text = "Product name:",
                                              FontSize = 18, VerticalOptions = LayoutOptions.Center };
            Label productUnitPrice = new Label() { Text = "Price per product unit:",
                                                   FontSize = 18, VerticalOptions = LayoutOptions.Center
            Label quantity = new Label() { Text = "The number of ordered units:",
                                           FontSize = 18, VerticalOptions = LayoutOptions.Start };
            Label total = new Label() { Text = "Total cost:",
                                        FontSize = 18, VerticalOptions = LayoutOptions.Center };
            Label isShipped = null;
            if (!isNewItemForm) {
                isShipped = new Label() { Text = "Has this order been shipped?",
                                          FontSize = 18, VerticalOptions = LayoutOptions.Center };
            }
            layoutGrid.Children.Add(productName, 0, 0);
            layoutGrid.Children.Add(productUnitPrice, 0, 1);
            layoutGrid.Children.Add(quantity, 0, 2);
            layoutGrid.Children.Add(total, 0, 3);
            if (!isNewItemForm) {
                layoutGrid.Children.Add(isShipped, 0, 4);
            }
        }
        void FillEditors(Grid layoutGrid, bool isNewItemForm) {
            View productName;
            // Display a product name in a label in the Edit Item form
            // or change a product name by using a picker in the New Item form.
            if (isNewItemForm) {
                productName = new Picker() { HorizontalOptions = LayoutOptions.FillAndExpand,
                                             VerticalOptions = LayoutOptions.Center};
                InitProductPickerValues(productName as Picker);
            }
            else {
                productName = new Label() { HorizontalOptions = LayoutOptions.FillAndExpand,
                                            VerticalOptions = LayoutOptions.Center,
                                            FontAttributes = FontAttributes.Bold, FontSize = 18 };
                productName.SetBinding(Label.TextProperty,
                                       new Binding(EditValuesContainer.GetBindingPath("Product.Name")));
            }
            // Display a product unit price in a form.
            Int32ToPriceStringConverter priceConverter = new Int32ToPriceStringConverter();
            Label priceLabel = new Label () { HorizontalOptions = LayoutOptions.FillAndExpand,
                                              VerticalOptions = LayoutOptions.Center, FontSize = 18 };
            priceLabel.SetBinding (Label.TextProperty, EditValuesContainer.GetBindingPath ("Product.UnitPr
                                   BindingMode.Default, priceConverter);
            // Modify the number of ordered units via a slider.
            StackLayout quantityStack = new StackLayout () { Orientation = StackOrientation.Vertical,
                                                             VerticalOptions = LayoutOptions.CenterAndExpa
            Label quantityLabel = new Label () { XAlign = TextAlignment.Start,
                                                 VerticalOptions = LayoutOptions.Center, FontSize = 18};
            Slider quantitySlider = new Slider () { Maximum = 99, Minimum = 10 };
            quantitySlider.SetBinding (Slider.ValueProperty, EditValuesContainer.GetBindingPath ("Quantity
                                       BindingMode.TwoWay, new Int32ToDoubleConverter());
            quantityLabel.BindingContext = quantitySlider;
            quantityLabel.SetBinding(Label.TextProperty, "Value", BindingMode.OneWay, new DoubleToInt32Co
            quantityStack.Children.Add(quantityLabel);
            quantityStack.Children.Add(quantitySlider);
            // Display the order total value calculated automatically.
            Label totalLabel = new Label () { XAlign = TextAlignment.Start,
                                              VerticalOptions = LayoutOptions.Center, FontSize = 18 };
            totalLabel.PropertyChanged += OnTotalPropertyChanged;
            totalLabel.SetBinding(Label.TextProperty, EditValuesContainer.GetBindingPath("Total"),
                                  BindingMode.OneWay, priceConverter);
            if (!isNewItemForm) {
                priceLabel.FontAttributes = FontAttributes.Bold;
                quantityLabel.FontAttributes = FontAttributes.Bold;
```

© 2018 Developer Express Inc.

```
                totalLabel.FontAttributes = FontAttributes.Bold;
            }
            // Add created elements to a form.
            layoutGrid.Children.Add(productName, 1, 0);
            layoutGrid.Children.Add(priceLabel, 1, 1);
            layoutGrid.Children.Add(quantityStack, 1, 2);
            layoutGrid.Children.Add(totalLabel, 1, 3);
            // Show a switch in the Edit Item form allowing end-users to set whether or not an order is sh
            Switch isShipped = new Switch() { VerticalOptions = LayoutOptions.Center,
                                              HorizontalOptions = LayoutOptions.Start };
            isShipped.SetBinding(Switch.IsToggledProperty, EditValuesContainer.GetBindingPath("Shipped"));
            if (!isNewItemForm) {
                layoutGrid.Children.Add(isShipped, 1, 4);
            }
        }
        void InitProductPickerValues(Picker picker) {
            EditValuesContainer valuesContainer = BindingContext as EditValuesContainer;
            String productName = valuesContainer.Values["Product.Name"].ToString();
            int selectedItemIndex = -1;
            foreach (Product product in ViewModel.Products) {
                if (productName == product.Name) {
                    selectedItemIndex = picker.Items.Count;
                }
                picker.Items.Add(product.Name);
            }
            picker.SelectedIndexChanged += OnProductNameSelectedIndexChanged;
            if (selectedItemIndex >= 0) {
                picker.SelectedIndex = selectedItemIndex;
            }
        }
        void OnProductNameSelectedIndexChanged(object sender, EventArgs e) {
            EditValuesContainer valuesContainer = BindingContext as EditValuesContainer;
            Picker picker = sender as Picker;
            if (picker.SelectedIndex < 0) {
                return;
            }
            String productName = picker.Items[picker.SelectedIndex];
            valuesContainer.Values["Product.Name"] = productName;
            Product product = ViewModel.Products.Find(p => p.Name == productName);
            valuesContainer.Values["Product.UnitPrice"] = product.UnitPrice;
        }
        void OnTotalPropertyChanged (object sender, System.ComponentModel.PropertyChangedEventArgs e)
        {
            Label label = (sender as Label);
            if (e.PropertyName == Label.TextProperty.PropertyName && !String.IsNullOrWhiteSpace (label.Tex
                Int32 value = (Int32)((label.BindingContext as EditValuesContainer).Values ["Total"]);
                if (value > 2000) {
                    label.TextColor = Color.Green; }
                else {
                    if (value < 500) {
                        label.TextColor = Color.Red; }
                    else {
                        label.TextColor = ThemeManager.Theme.CellCustomizer.Font.Color;}
                }
            }
        }
    }
}
```

**C#**

```
 (MainPage.xaml.cs)
using System;
using System.Collections.Generic;
using Xamarin.Forms;
using DevExpress.Mobile.DataGrid;
namespace CustomEditFormDemo
{
    public partial class MainPage : ContentPage
    {
        TestOrdersRepository model;
        public MainPage ()
        {
            InitializeComponent ();
            model = new TestOrdersRepository ();
            BindingContext = model;
            DataTemplate editFormContentTemplate = new DataTemplate(GetTem
            grid.EditFormContent = editFormContentTemplate;
            grid.RowEditMode = RowEditMode.Popup;
        }
        object GetTemplate() {
            return new CustomEditFormContent() { ViewModel = model };
        }
        void OnInitNewItemRow(object sender, InitNewRowEventArgs e) {
            e.EditableRowData.SetFieldValue("Quantity", model.MinimumQuant
            e.EditableRowData.SetFieldValue("Product.Name", model.Products
        }
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<?xml version="1.0" encoding="UTF-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="CustomEditFormDemo.MainPage"
             xmlns:dxGrid="clr-namespace:DevExpress.Mobile.DataGrid;assemb
      <ContentPage.Padding>
        <OnPlatform x:TypeArguments="Thickness" iOS="0, 20, 0, 0" />
    </ContentPage.Padding>
    <dxGrid:GridControl x:Name="grid" ItemsSource="{Binding Orders}"
                        NewItemRowVisibility="true" InitNewRow="OnInitNew
            <dxGrid:GridControl.Columns>
             <dxGrid:TextColumn FieldName="Product.Name" Caption="Product"
             <dxGrid:NumberColumn FieldName="Product.UnitPrice" Caption="P
             <dxGrid:NumberColumn FieldName="Quantity"/>
             <dxGrid:NumberColumn FieldName="Total"
                                  UnboundType="Integer" UnboundExpression=
                                  DisplayFormat="C0" IsReadOnly="True"/>
             <dxGrid:SwitchColumn FieldName="Shipped"/>
          </dxGrid:GridControl.Columns>
    </dxGrid:GridControl>
</ContentPage>
```

© 2018 Developer Express Inc.

## See Also

EditValuesContainer Class
EditValuesContainer Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# EditValuesContainer Events

Contains information on a data row currently being edited by an end-user via an edit form.

The following tables list the members exposed by the EditValuesContainer type.

**Public Events**

| | Name | Description |
|---|---|---|
| | PropertyChanged | For internal use. |

**See Also**
EditValuesContainer Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# PropertyChanged Event

For internal use.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public event PropertyChangedEventHandler PropertyChanged
```

### Event Data

The event handler receives an argument of type System.ComponentModel.PropertyChangedEventArgs containing data related to this event.

The following **PropertyChangedEventArgs** properties provide information specific to this event.

| Property | Description |
|---|---|
| PropertyName | |

### See Also

EditValuesContainer Class
EditValuesContainer Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# EditValuesContainer Methods

Contains information on a data row currently being edited by an end-user via an edit form.

The following tables list the members exposed by the EditValuesContainer type.

**Public Methods**

| | Name | Description |
|---|---|---|
| | ContainsValueForField | For internal use. |
| S | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| S | GetBindingPath | Allows you to bind a custom edit form's editor to a grid's data value. |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | Raise | For internal use. |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

**See Also**
EditValuesContainer Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# ContainsValueForField Method

For internal use.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**
**C#**

```
public bool ContainsValueForField(
   string fieldName
)
```

**Parameters**
*fieldName*

**Return Value**

**See Also**
EditValuesContainer Class
EditValuesContainer Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# GetBindingPath Method

Allows you to bind a custom edit form's editor to a grid's data value.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public static string GetBindingPath(
   string fieldName
)
```

### Parameters

*fieldName*
A grid column's field name (GridColumn.FieldName).

### Return Value

A string specifying an edit value that can be bound to an edit form's editor.

### Remarks

When you create a custom row edit form, bind this form's editors to edit values contained in the Values dictionary. A column containing a cell to which a value entered in the form's editor should be assigned is identified by the column's field name (GridColumn.FieldName). For example, an editor bound to *Values[Name]* shows or sets a cell value of the grid's *Name* column. However, if a column's field name is complex (a field name is constructed in the "String1.String2" form), you need to use the **GetBindingPath** method when binding a form's editor to a grid's edit value. This method masks a dot in a complex field name. For example, *GetBindingPath("Product.Name")* returns *"Values[Product_##_DoT_##_Name]"*.

### Example

| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T279098. | |

GridControl allows you to substitute the default row edit form with a custom form. This example explains how to do this.

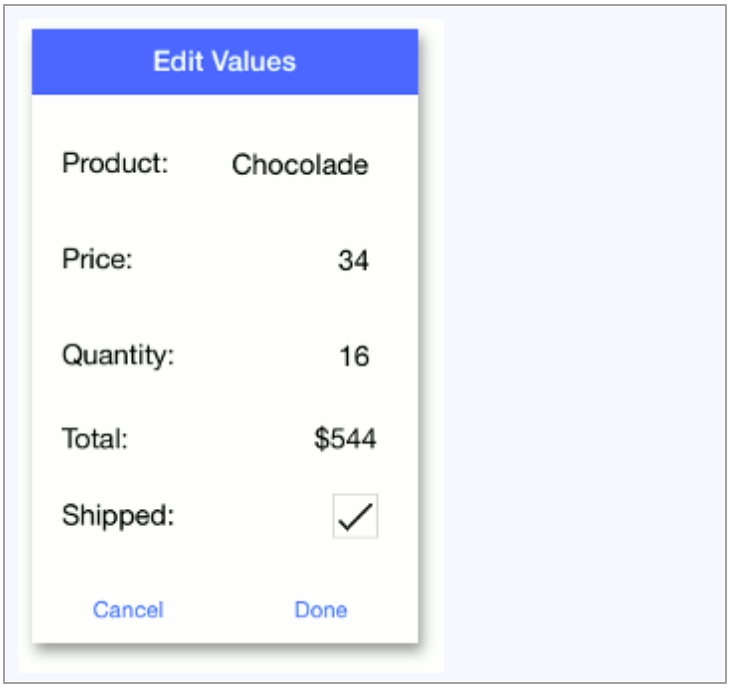
In this application, the grid is bound to a collection of orders. Each data row in the grid represents an individual order and displays the following information:

- Ordered product name (the **Product** column)
- Product unit price (the **Price** column)
- Number of ordered product units (the **Quantity** column)
- Order total value (the **Total** column)
- A value indicating whether an order is shipped (the **Shipped** column)

The default row edit form in this grid looks like the following.

| iOS | Android |
|---|---|

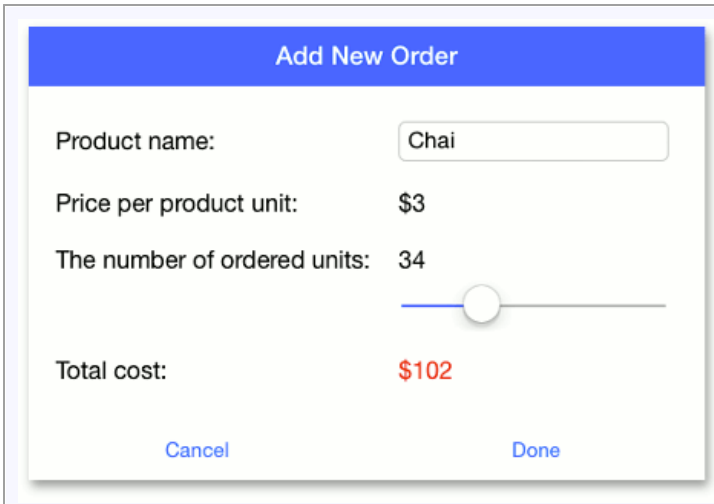
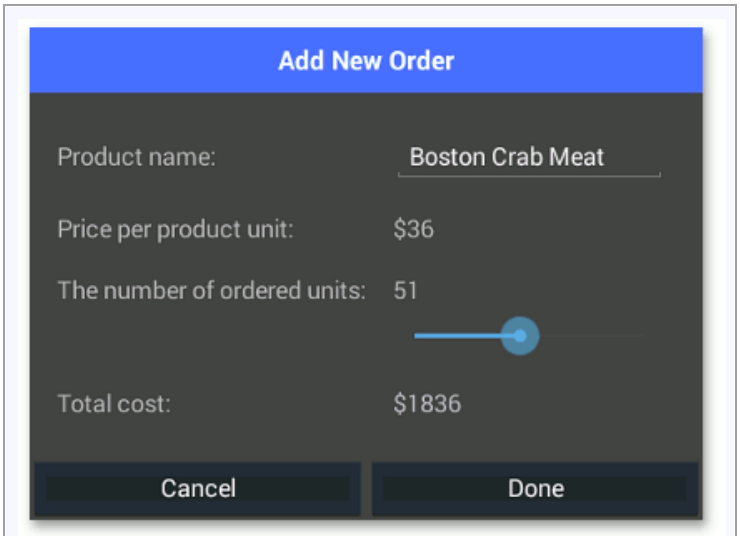
© 2018 Developer Express Inc.




This example demonstrates how to create a custom form to add new data rows and another form to edit existing rows.

- **Add New Order Form**

This form is invoked on tapping New Item Row in the grid and allows an end user to add a new data row (create a new order). On this form, an end user can select a product name via a Picker control, and set the number of ordered product units via a Slider control. A unit price is automatically shown for the selected product, and the order total value is automatically calculated.

| iOS | Android |
| --- | --- |
|  |  |

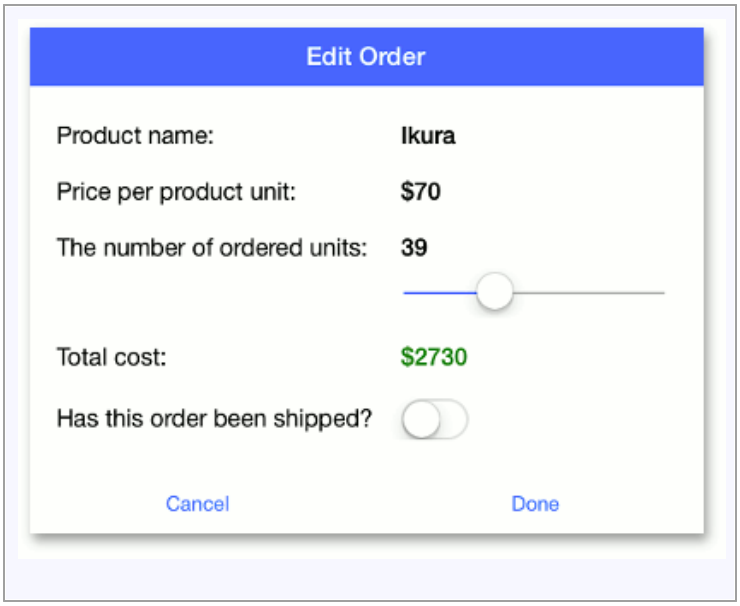


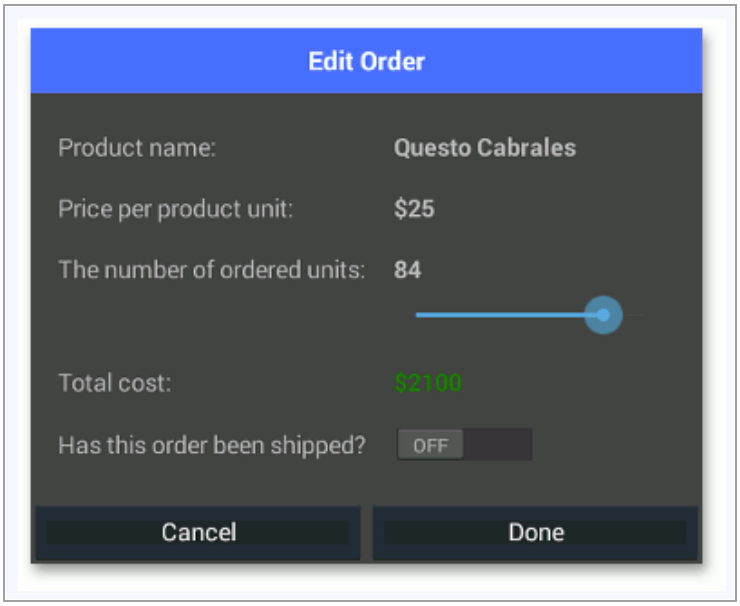
- **Edit Order Form**

This form is invoked on double-tapping a cell or selecting **Edit Cell** from the popup menu. It allows an end user to modify existing data rows. On this form, product name and unit price are shown in Lable controls and cannot be changed. An end user can change the number of ordered product units via a Slider control and use the Switch control to set whether the order is shipped.

| iOS | Android |
| --- | --- |

© 2018 Developer Express Inc.




To create a custom row edit form and show it instead of the default form, execute the steps below.

1. Specify a custom form view. In this example, it is a ContentView descendant with a set of labels and editors inside (see the *CustomFormEditorContent.cs* file). This view's BindingContext is an EditValuesContainer object. Labels and editors are bound to data fields via the GetBindingPath method with data field names passed as parameters.

The IsNewRow property identifies whether a from is invoked for a new or existing row.

Custom form captions are implemented by using the mechanism of localization.

2. Create a DataTemplate instance for your custom form view and assign this template to the GridControl.EditFormContent property (see the *MainPage.xaml.cs* file).
3. Set the GridControl.RowEditMode property to RowEditMode.Popup to show a form instead of an in-place row editor.

| C# | |
|---|---|

```
 (Converters.cs)
using System;
using Xamarin.Forms;
using System.Globalization;
namespace CustomEditFormDemo {
    class Int32ToStringConverter : IValueConverter {
        object IValueConverter.Convert(object value, Type targetType, object parameter, CultureInfo cultu
            return value.ToString();
        }
        object IValueConverter.ConvertBack(object value, Type targetType, object parameter, CultureInfo cu
            int result = 0;
            try {
                result = Int32.Parse(value as String);
            } catch {
                result = 0;
            }
            return result;
        }
    }
    class Int32ToPriceStringConverter : IValueConverter {
        object IValueConverter.Convert(object value, Type targetType, object parameter, CultureInfo cultu
            return "$" + value.ToString();
        }
        object IValueConverter.ConvertBack(object value, Type targetType, object parameter, CultureInfo cu
            int result = 0;
            try {
                result = Int32.Parse(value as String);
            } catch {
                result = 0;
            }
            return result;
        }
    }
    class Int32ToDoubleConverter : IValueConverter {
        object IValueConverter.Convert(object value, Type targetType, object parameter, CultureInfo cultu
            return Convert.ToDouble(value);
        }
        object IValueConverter.ConvertBack(object value, Type targetType, object parameter, CultureInfo cu
            return Convert.ToInt32(value);
        }
    }
    class DoubleToInt32Converter : IValueConverter {
        object IValueConverter.Convert(object value, Type targetType, object parameter, CultureInfo cultu
            return Convert.ToInt32(value);
        }
        object IValueConverter.ConvertBack(object value, Type targetType, object parameter, CultureInfo cu
            return Convert.ToDouble(value);
        }
    }
}
```

| C# | |
|---|---|

© 2018 Developer Express Inc.

```
 (CustomFormEditorContent.cs)
using System;
using System.Collections.Generic;
using Xamarin.Forms;
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.DataGrid.Theme;
using System.Globalization;
using DevExpress.Mobile.DataGrid.Localization;
namespace CustomEditFormDemo {
    class EditFormLocalizer : GridLocalizer {
        public void SetEditItemFormCaption() {
            AddString(GridStringId.EditingForm_LabelCaption, "Edit Order");
        }
        public void SetNewItemFormCaption() {
            AddString(GridStringId.EditingForm_LabelCaption, "Add New Order");
        }
    }
    public class CustomEditFormContent : ContentView {
        public TestOrdersRepository ViewModel { get; set; }
        public CustomEditFormContent() {
            HorizontalOptions = LayoutOptions.FillAndExpand;
            VerticalOptions = LayoutOptions.FillAndExpand;
            GridLocalizer.Active = new EditFormLocalizer();
        }
        protected override void OnBindingContextChanged () {
            base.OnBindingContextChanged ();
            EditFormLocalizer localizer = GridLocalizer.Active as EditFormLocalizer;
            if ((BindingContext as EditValuesContainer).IsNewRow) {
                if (localizer != null) {
                    localizer.SetNewItemFormCaption();
                }
                Content = CreateNewItemFormContent();
            }
            else {
                if (localizer != null) {
                    localizer.SetEditItemFormCaption();
                }
                Content = CreateEditItemFormContent ();
            }
        }
        View CreateNewItemFormContent() {
            Grid layoutGrid = CreateGrid(4);
            FillFormGrid(layoutGrid, true);
            return layoutGrid;
        }
        View CreateEditItemFormContent() {
            Grid layoutGrid = CreateGrid(5);
            FillFormGrid(layoutGrid, false);
            return layoutGrid;
        }
        Grid CreateGrid(int rowsCount) {
            Grid layoutGrid = new Grid();
            layoutGrid.ColumnDefinitions.Add(new ColumnDefinition() { Width = GridLength.Auto });
            layoutGrid.ColumnDefinitions.Add(new ColumnDefinition() { Width = new GridLength(200, GridUnit
            for (int i = 0; i < rowsCount; i++) {
                layoutGrid.RowDefinitions.Add(new RowDefinition() { Height = GridLength.Auto });
            }
            layoutGrid.RowSpacing = 20;
            layoutGrid.ColumnSpacing = 20;
            return layoutGrid;
        }
        void FillFormGrid(Grid layoutGrid, bool isNewItemForm) {
            FillLabels(layoutGrid, isNewItemForm);
            FillEditors(layoutGrid, isNewItemForm);
        }
```

© 2018 Developer Express Inc.

```
        void FillLabels(Grid layoutGrid, bool isNewItemForm) {
            Label productName = new Label() { Text = "Product name:",
                                              FontSize = 18, VerticalOptions = LayoutOptions.Center };
            Label productUnitPrice = new Label() { Text = "Price per product unit:",
                                                   FontSize = 18, VerticalOptions = LayoutOptions.Center
            Label quantity = new Label() { Text = "The number of ordered units:",
                                           FontSize = 18, VerticalOptions = LayoutOptions.Start };
            Label total = new Label() { Text = "Total cost:",
                                        FontSize = 18, VerticalOptions = LayoutOptions.Center };
            Label isShipped = null;
            if (!isNewItemForm) {
                isShipped = new Label() { Text = "Has this order been shipped?",
                                          FontSize = 18, VerticalOptions = LayoutOptions.Center };
            }
            layoutGrid.Children.Add(productName, 0, 0);
            layoutGrid.Children.Add(productUnitPrice, 0, 1);
            layoutGrid.Children.Add(quantity, 0, 2);
            layoutGrid.Children.Add(total, 0, 3);
            if (!isNewItemForm) {
                layoutGrid.Children.Add(isShipped, 0, 4);
            }
        }
        void FillEditors(Grid layoutGrid, bool isNewItemForm) {
            View productName;
            // Display a product name in a label in the Edit Item form
            // or change a product name by using a picker in the New Item form.
            if (isNewItemForm) {
                productName = new Picker() { HorizontalOptions = LayoutOptions.FillAndExpand,
                                             VerticalOptions = LayoutOptions.Center};
                InitProductPickerValues(productName as Picker);
            }
            else {
                productName = new Label() { HorizontalOptions = LayoutOptions.FillAndExpand,
                                            VerticalOptions = LayoutOptions.Center,
                                            FontAttributes = FontAttributes.Bold, FontSize = 18 };
                productName.SetBinding(Label.TextProperty,
                                       new Binding(EditValuesContainer.GetBindingPath("Product.Name")));
            }
            // Display a product unit price in a form.
            Int32ToPriceStringConverter priceConverter = new Int32ToPriceStringConverter();
            Label priceLabel = new Label () { HorizontalOptions = LayoutOptions.FillAndExpand,
                                              VerticalOptions = LayoutOptions.Center, FontSize = 18 };
            priceLabel.SetBinding (Label.TextProperty, EditValuesContainer.GetBindingPath ("Product.UnitPr
                                   BindingMode.Default, priceConverter);
            // Modify the number of ordered units via a slider.
            StackLayout quantityStack = new StackLayout () { Orientation = StackOrientation.Vertical,
                                                             VerticalOptions = LayoutOptions.CenterAndExpa
            Label quantityLabel = new Label () { XAlign = TextAlignment.Start,
                                                 VerticalOptions = LayoutOptions.Center, FontSize = 18};
            Slider quantitySlider = new Slider () { Maximum = 99, Minimum = 10 };
            quantitySlider.SetBinding (Slider.ValueProperty, EditValuesContainer.GetBindingPath ("Quantity
                                       BindingMode.TwoWay, new Int32ToDoubleConverter());
            quantityLabel.BindingContext = quantitySlider;
            quantityLabel.SetBinding(Label.TextProperty, "Value", BindingMode.OneWay, new DoubleToInt32Co
            quantityStack.Children.Add(quantityLabel);
            quantityStack.Children.Add(quantitySlider);
            // Display the order total value calculated automatically.
            Label totalLabel = new Label () { XAlign = TextAlignment.Start,
                                              VerticalOptions = LayoutOptions.Center, FontSize = 18 };
            totalLabel.PropertyChanged += OnTotalPropertyChanged;
            totalLabel.SetBinding(Label.TextProperty, EditValuesContainer.GetBindingPath("Total"),
                                  BindingMode.OneWay, priceConverter);
            if (!isNewItemForm) {
                priceLabel.FontAttributes = FontAttributes.Bold;
                quantityLabel.FontAttributes = FontAttributes.Bold;
```

```
                totalLabel.FontAttributes = FontAttributes.Bold;
            }
            // Add created elements to a form.
            layoutGrid.Children.Add(productName, 1, 0);
            layoutGrid.Children.Add(priceLabel, 1, 1);
            layoutGrid.Children.Add(quantityStack, 1, 2);
            layoutGrid.Children.Add(totalLabel, 1, 3);
            // Show a switch in the Edit Item form allowing end-users to set whether or not an order is sh
            Switch isShipped = new Switch() { VerticalOptions = LayoutOptions.Center,
                                              HorizontalOptions = LayoutOptions.Start };
            isShipped.SetBinding(Switch.IsToggledProperty, EditValuesContainer.GetBindingPath("Shipped"));
            if (!isNewItemForm) {
                layoutGrid.Children.Add(isShipped, 1, 4);
            }
        }
        void InitProductPickerValues(Picker picker) {
            EditValuesContainer valuesContainer = BindingContext as EditValuesContainer;
            String productName = valuesContainer.Values["Product.Name"].ToString();
            int selectedItemIndex = -1;
            foreach (Product product in ViewModel.Products) {
                if (productName == product.Name) {
                    selectedItemIndex = picker.Items.Count;
                }
                picker.Items.Add(product.Name);
            }
            picker.SelectedIndexChanged += OnProductNameSelectedIndexChanged;
            if (selectedItemIndex >= 0) {
                picker.SelectedIndex = selectedItemIndex;
            }
        }
        void OnProductNameSelectedIndexChanged(object sender, EventArgs e) {
            EditValuesContainer valuesContainer = BindingContext as EditValuesContainer;
            Picker picker = sender as Picker;
            if (picker.SelectedIndex < 0) {
                return;
            }
            String productName = picker.Items[picker.SelectedIndex];
            valuesContainer.Values["Product.Name"] = productName;
            Product product = ViewModel.Products.Find(p => p.Name == productName);
            valuesContainer.Values["Product.UnitPrice"] = product.UnitPrice;
        }
        void OnTotalPropertyChanged (object sender, System.ComponentModel.PropertyChangedEventArgs e)
        {
            Label label = (sender as Label);
            if (e.PropertyName == Label.TextProperty.PropertyName && !String.IsNullOrWhiteSpace (label.Tex
                Int32 value = (Int32)((label.BindingContext as EditValuesContainer).Values ["Total"]);
                if (value > 2000) {
                    label.TextColor = Color.Green; }
                else {
                    if (value < 500) {
                        label.TextColor = Color.Red; }
                    else {
                        label.TextColor = ThemeManager.Theme.CellCustomizer.Font.Color;}
                }
            }
        }
    }
}
```

**C#**

```
 (MainPage.xaml.cs)
using System;
using System.Collections.Generic;
using Xamarin.Forms;
using DevExpress.Mobile.DataGrid;
namespace CustomEditFormDemo
{
    public partial class MainPage : ContentPage
    {
        TestOrdersRepository model;
        public MainPage ()
        {
            InitializeComponent ();
            model = new TestOrdersRepository ();
            BindingContext = model;
            DataTemplate editFormContentTemplate = new DataTemplate(GetTem
            grid.EditFormContent = editFormContentTemplate;
            grid.RowEditMode = RowEditMode.Popup;
        }
        object GetTemplate() {
            return new CustomEditFormContent() { ViewModel = model };
        }
        void OnInitNewItemRow(object sender, InitNewRowEventArgs e) {
            e.EditableRowData.SetFieldValue("Quantity", model.MinimumQuant
            e.EditableRowData.SetFieldValue("Product.Name", model.Products
        }
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<?xml version="1.0" encoding="UTF-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="CustomEditFormDemo.MainPage"
             xmlns:dxGrid="clr-namespace:DevExpress.Mobile.DataGrid;assemb
      <ContentPage.Padding>
        <OnPlatform x:TypeArguments="Thickness" iOS="0, 20, 0, 0" />
    </ContentPage.Padding>
    <dxGrid:GridControl x:Name="grid" ItemsSource="{Binding Orders}"
                        NewItemRowVisibility="true" InitNewRow="OnInitNew
            <dxGrid:GridControl.Columns>
             <dxGrid:TextColumn FieldName="Product.Name" Caption="Product"
             <dxGrid:NumberColumn FieldName="Product.UnitPrice" Caption="P
             <dxGrid:NumberColumn FieldName="Quantity"/>
             <dxGrid:NumberColumn FieldName="Total"
                                  UnboundType="Integer" UnboundExpression=
                                  DisplayFormat="C0" IsReadOnly="True"/>
             <dxGrid:SwitchColumn FieldName="Shipped"/>
          </dxGrid:GridControl.Columns>
    </dxGrid:GridControl>
</ContentPage>
```

© 2018 Developer Express Inc.

## See Also

EditValuesContainer Class
EditValuesContainer Members
DevExpress.Mobile.DataGrid Namespace
Row Edit Modes

© 2018 Developer Express Inc.

# Raise Method

For internal use.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**
**C#**

```
public void Raise()
```

**See Also**
EditValuesContainer Class
EditValuesContainer Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# ExpressionConditionBase Class

Serves as a base for classes that represent expression-based format conditions.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public abstract class ExpressionConditionBase : FormatConditionBase
```

### Inheritance Hierarchy

System.Object
  Xamarin.Forms.BindableObject
    FormatConditionBase
      **ExpressionConditionBase**
        Derived classes

### See Also

ExpressionConditionBase Members
DevExpress.Mobile.DataGrid Namespace
Conditional Formatting

© 2018 Developer Express Inc.

# ExpressionConditionBase Members

Serves as a base for classes that represent expression-based format conditions.

The following tables list the members exposed by the ExpressionConditionBase type.

### Public Properties

| | Name | Description |
|---|---|---|
| | ApplyToRow | Specifies whether to apply formatting settings to the entire row. This is a bindable property. |
| | BindingContext | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Expression | Gets or sets the expression specifying to which cells the format should be applied. This is a bindable property. |
| | FieldName | Gets or sets the name of the column which provides values to test against the current format rule, and to which the format should be applied. This is a bindable property. (Inherited from **FormatConditionBase**) |
| | Format | Gets or sets the formatting settings applied to cells whose values take part in conditional formatting. This is a bindable property. |
| | Owner | Gets the format condition's owner. This is a bindable property. (Inherited from **FormatConditionBase**) |
| | PredefinedFormatName | Gets or sets the name of the predefined format to be applied to target cells. This is a bindable property. (Inherited from **FormatConditionBase**) |
| | TypeName | Gets or sets the data type name. This property supports the internal infrastructure and is not intended to be used directly from your code. (Inherited from **FormatConditionBase**) |

### Public Methods

| | Name | Description |
|---|---|---|
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Equals | (Inherited from **System.Object**) |
| S | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ToString | (Inherited from **System.Object**) |

**Public Fields**

| | Name | Description |
|---|---|---|
| S | ApplyToRowProperty | Identifies the ApplyToRow bindable property. |
| S | BindingContextProperty | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ExpressionProperty | Identifies the Expression bindable property. |
| S | FieldNameProperty | Identifies the FormatConditionBase.FieldName bindable property. (Inherited from **FormatConditionBase**) |
| S | FormatProperty | Identifies the Format bindable property. |
| S | OwnerProperty | Identifies the FormatConditionBase.Owner bindable property. (Inherited from **FormatConditionBase**) |
| S | PredefinedFormatNameProperty | Identifies the FormatConditionBase.PredefinedFormatName bindable property. (Inherited from **FormatConditionBase**) |

**Public Events**

| | Name | Description |
|---|---|---|
| | BindingContextChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanging | (Inherited from **Xamarin.Forms.BindableObject**) |

**See Also**
ExpressionConditionBase Members
DevExpress.Mobile.DataGrid Namespace
Conditional Formatting

© 2018 Developer Express Inc.

# ExpressionConditionBase Properties

Serves as a base for classes that represent expression-based format conditions.

The following tables list the members exposed by the ExpressionConditionBase type.

### Public Properties

| | Name | Description |
|---|---|---|
| | ApplyToRow | Specifies whether to apply formatting settings to the entire row. This is a bindable property. |
| | BindingContext | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Expression | Gets or sets the expression specifying to which cells the format should be applied. This is a bindable property. |
| | FieldName | Gets or sets the name of the column which provides values to test against the current format rule, and to which the format should be applied. This is a bindable property. (Inherited from **FormatConditionBase**) |
| | Format | Gets or sets the formatting settings applied to cells whose values take part in conditional formatting. This is a bindable property. |
| | Owner | Gets the format condition's owner. This is a bindable property. (Inherited from **FormatConditionBase**) |
| | PredefinedFormatName | Gets or sets the name of the predefined format to be applied to target cells. This is a bindable property. (Inherited from **FormatConditionBase**) |
| | TypeName | Gets or sets the data type name. This property supports the internal infrastructure and is not intended to be used directly from your code. (Inherited from **FormatConditionBase**) |

### Public Methods

| | Name | Description |
|---|---|---|
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Equals | (Inherited from **System.Object**) |
| S | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ToString | (Inherited from **System.Object**) |

**Public Fields**

| | Name | Description |
|---|---|---|
| S | ApplyToRowProperty | Identifies the ApplyToRow bindable property. |
| S | BindingContextProperty | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ExpressionProperty | Identifies the Expression bindable property. |
| S | FieldNameProperty | Identifies the FormatConditionBase.FieldName bindable property. (Inherited from **FormatConditionBase**) |
| S | FormatProperty | Identifies the Format bindable property. |
| S | OwnerProperty | Identifies the FormatConditionBase.Owner bindable property. (Inherited from **FormatConditionBase**) |
| S | PredefinedFormatNameProperty | Identifies the FormatConditionBase.PredefinedFormatName bindable property. (Inherited from **FormatConditionBase**) |

**Public Events**

| | Name | Description |
|---|---|---|
| | BindingContextChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanging | (Inherited from **Xamarin.Forms.BindableObject**) |

**See Also**
ExpressionConditionBase Members
DevExpress.Mobile.DataGrid Namespace
Conditional Formatting

© 2018 Developer Express Inc.

# ApplyToRow Property

Specifies whether to apply formatting settings to the entire row. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public bool ApplyToRow { get; set; }
```

### Property Value

**true**, to apply formatting settings to the entire row; otherwise, **false**.

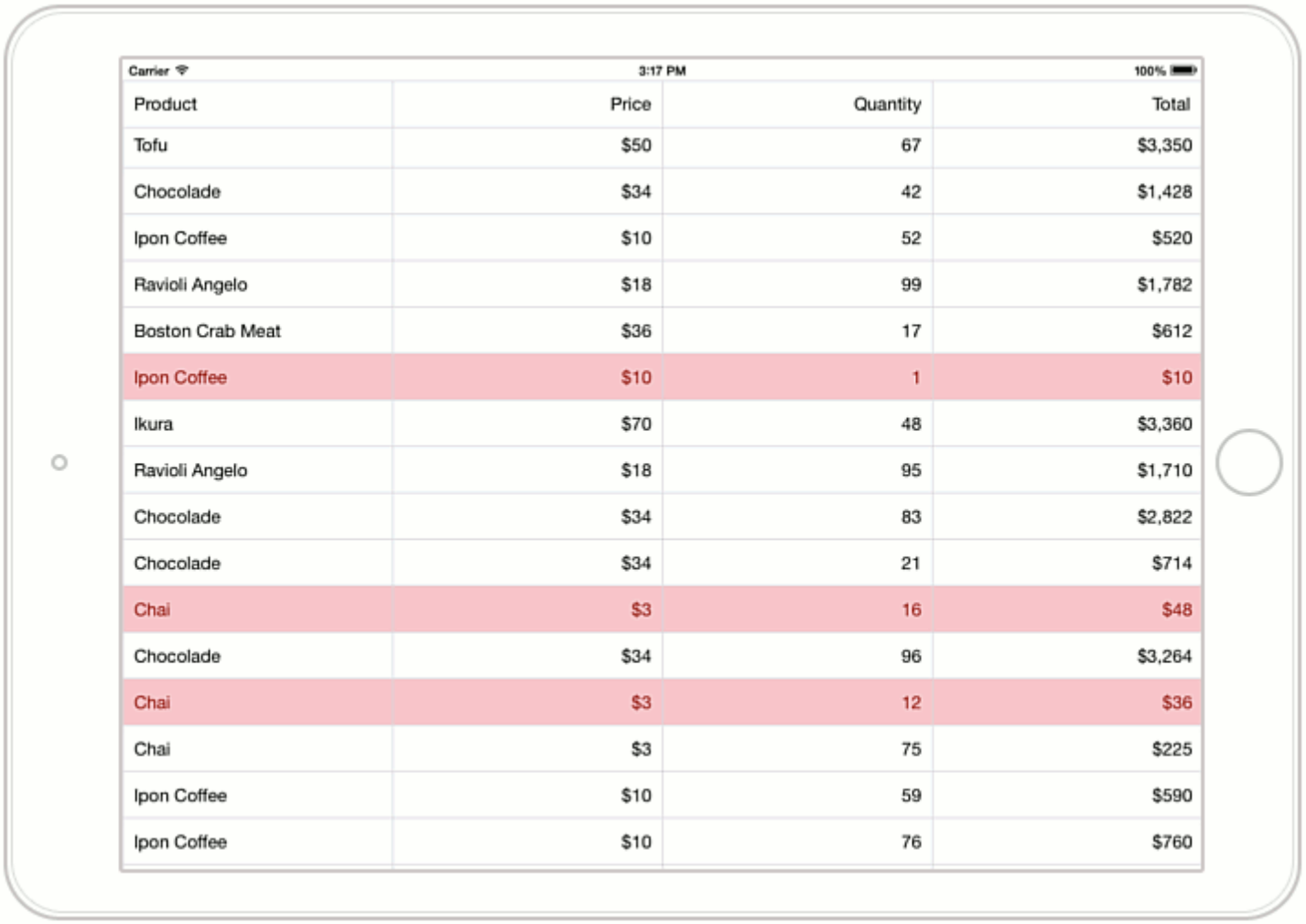
### Remarks

If the **ApplyToRow** property is set to **false**, formatting settings are only applied to the column's cells.

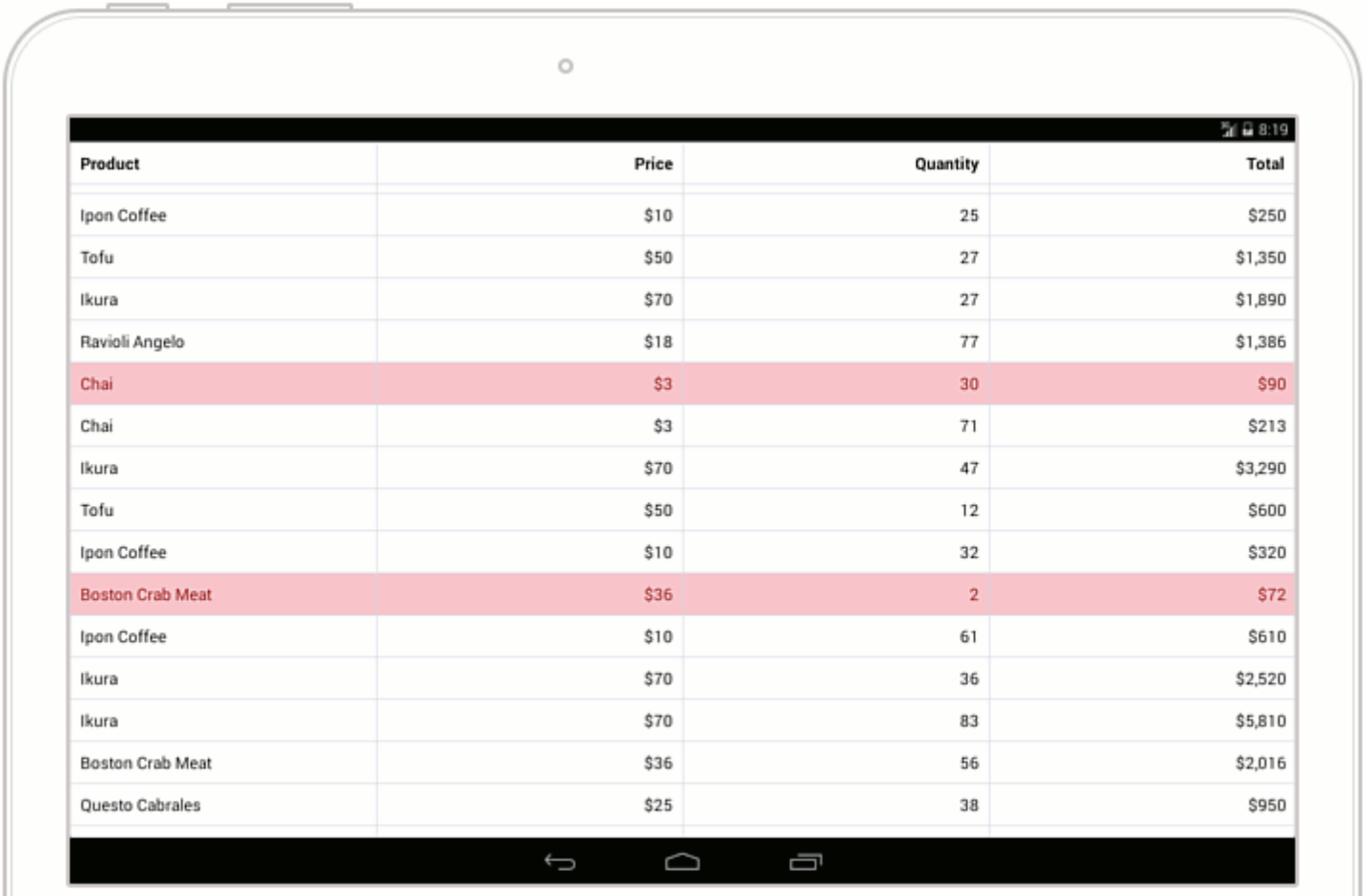
### Example

This example illustrates how to apply a format to rows in a GridControl that match a specific expression.

© 2018 Developer Express Inc.




| Product | Price | Quantity | Total |
|---|---|---|---|
| Tofu | $50 | 67 | $3,350 |
| Chocolade | $34 | 42 | $1,428 |
| Ipon Coffee | $10 | 52 | $520 |
| Ravioli Angelo | $18 | 99 | $1,782 |
| Boston Crab Meat | $36 | 17 | $612 |
| Ipon Coffee | $10 | 1 | $10 |
| Ikura | $70 | 48 | $3,360 |
| Ravioli Angelo | $18 | 95 | $1,710 |
| Chocolade | $34 | 83 | $2,822 |
| Chocolade | $34 | 21 | $714 |
| Chai | $3 | 16 | $48 |
| Chocolade | $34 | 96 | $3,264 |
| Chai | $3 | 12 | $36 |
| Chai | $3 | 75 | $225 |
| Ipon Coffee | $10 | 59 | $590 |
| Ipon Coffee | $10 | 76 | $760 |




| Product | Price | Quantity | Total |
|---|---|---|---|
| Ipon Coffee | $10 | 25 | $250 |
| Tofu | $50 | 27 | $1,350 |
| Ikura | $70 | 27 | $1,890 |
| Ravioli Angelo | $18 | 77 | $1,386 |
| Chai | $3 | 30 | $90 |
| Chai | $3 | 71 | $213 |
| Ikura | $70 | 47 | $3,290 |
| Tofu | $50 | 12 | $600 |
| Ipon Coffee | $10 | 32 | $320 |
| Boston Crab Meat | $36 | 2 | $72 |
| Ipon Coffee | $10 | 61 | $610 |
| Ikura | $70 | 36 | $2,520 |
| Ikura | $70 | 83 | $5,810 |
| Boston Crab Meat | $36 | 56 | $2,016 |
| Questo Cabrales | $25 | 38 | $950 |

An expression is a string that, when parsed and processed, evaluates some value. Expressions consist of column names, constants, operators, and functions. In this example, a Boolean expression is used. If the expression evaluates to **true**, the format is applied. The format highlights rows that have values in the **Total** column that are less than **100**.

1. Create a new FormatCondition class instance.
2. Set this object's FormatConditionBase.FieldName property to **Total**. This column provides values to test against the formatting rule.
3. Assign the expression string to the Expression property.
4. Assign one of the predefined formats to the FormatConditionBase.PredefinedFormatName property. All available format names are listed in the Predefined Format Names document.
5. Set the ApplyToRow property to **true** to apply the format to entire rows instead of individual column cells.
6. To apply a conditional formatting rule represented by the created **FormatCondition** object, add this object to the GridControl.FormatConditions collection.

**C#**

```
using DevExpress.Mobile.DataGrid;
// ...
FormatCondition condition = new FormatCondition();
condition.FieldName = "Total";
condition.Expression = "[Total] < 100";
condition.PredefinedFormatName = "LightRedFillWithDarkRedText";
condition.ApplyToRow = true;
grid.FormatConditions.Add (condition);
```

**See Also**

ExpressionConditionBase Class
ExpressionConditionBase Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# Expression Property

Gets or sets the expression specifying to which cells the format should be applied. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public string Expression { get; set; }
```

### Property Value

A System.String value that specifies the expression used to apply the conditional formatting rule.

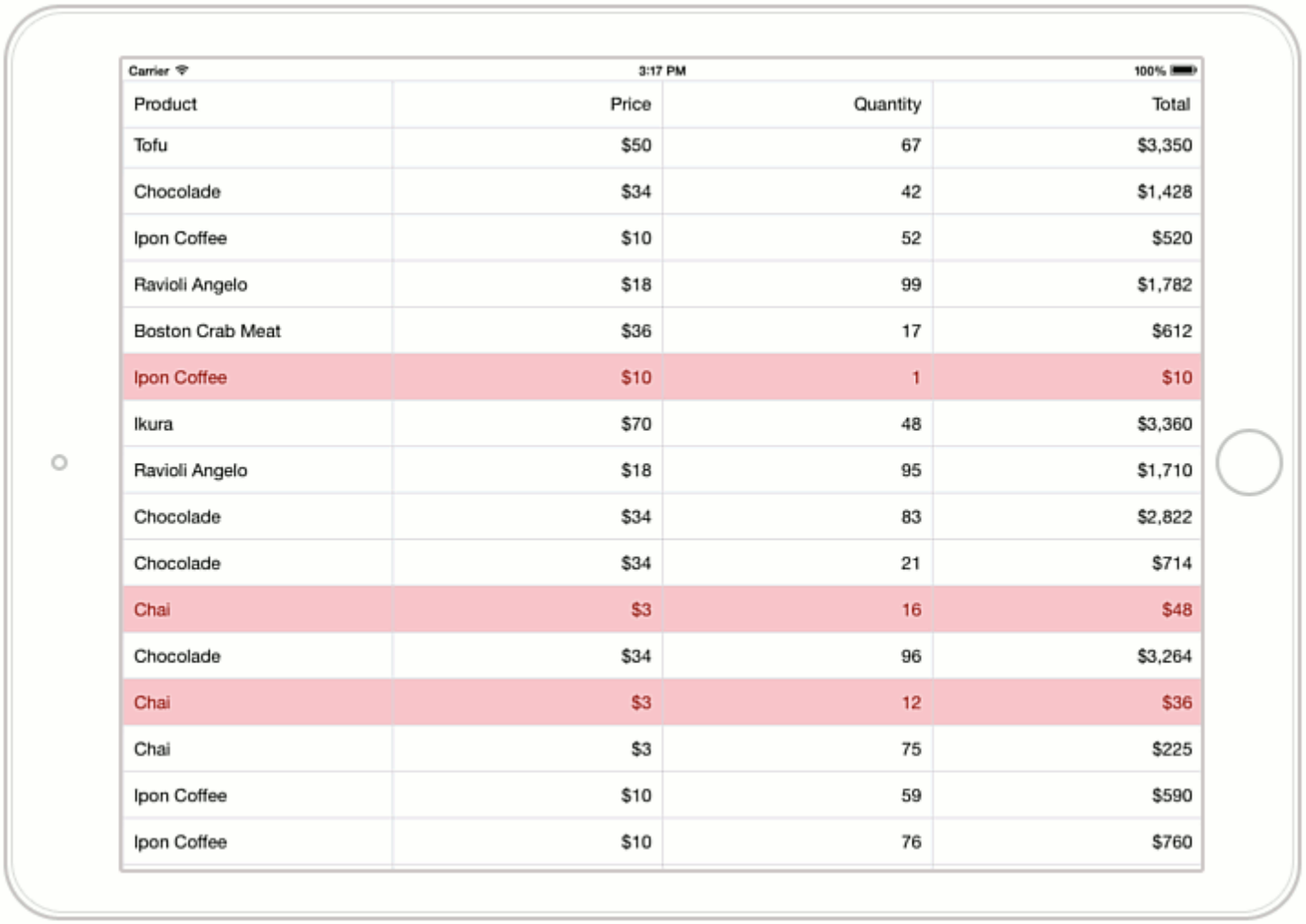
### Remarks

If a cell value meets the expression set to the **Expression** property, the format is applied to a target cell(s).

### Example

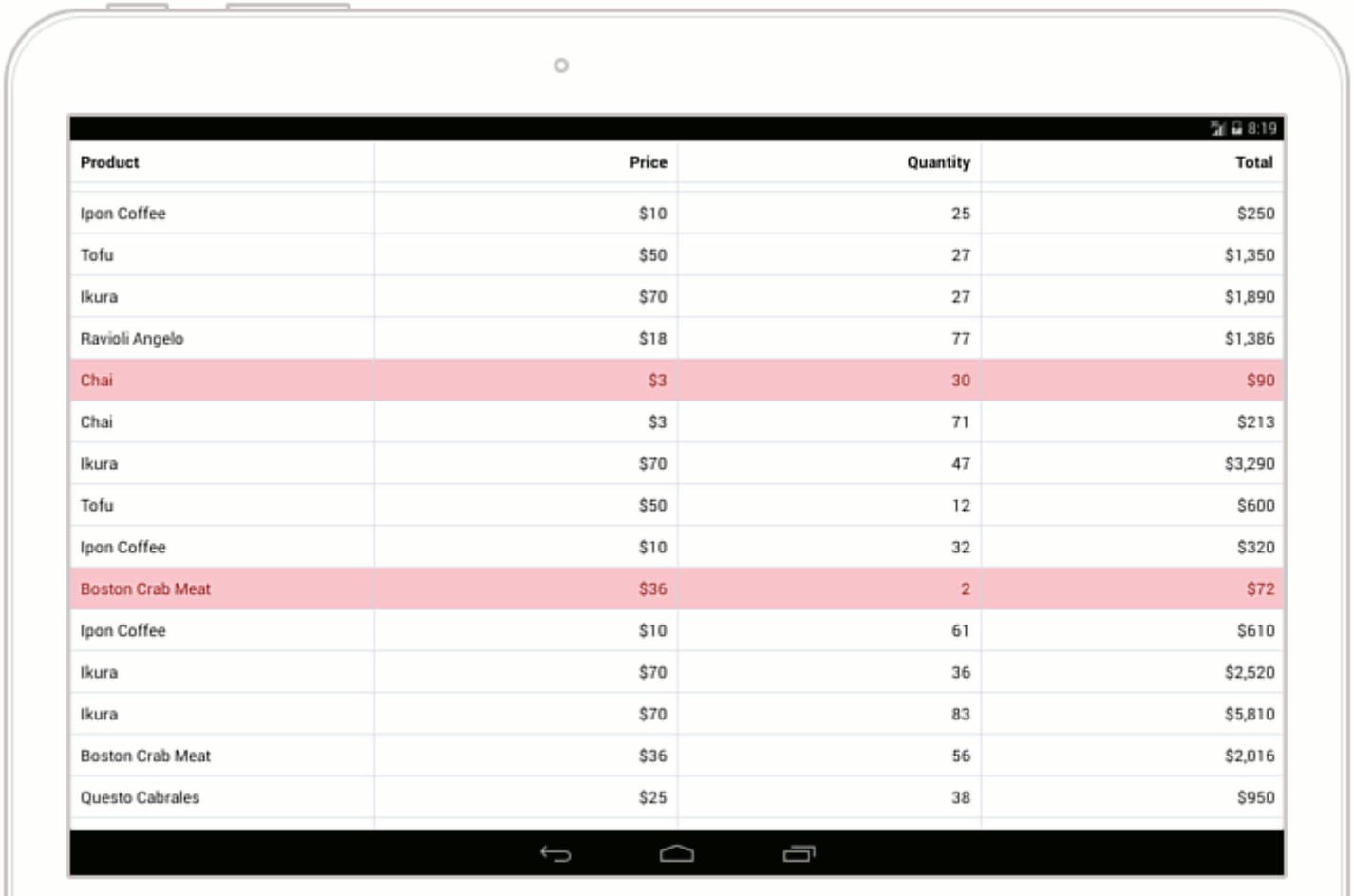
This example illustrates how to apply a format to rows in a GridControl that match a specific expression.

© 2018 Developer Express Inc.



Carrier 3:17 PM 100%

| Product | Price | Quantity | Total |
|---|---|---|---|
| Tofu | $50 | 67 | $3,350 |
| Chocolade | $34 | 42 | $1,428 |
| Ipon Coffee | $10 | 52 | $520 |
| Ravioli Angelo | $18 | 99 | $1,782 |
| Boston Crab Meat | $36 | 17 | $612 |
| Ipon Coffee | $10 | 1 | $10 |
| Ikura | $70 | 48 | $3,360 |
| Ravioli Angelo | $18 | 95 | $1,710 |
| Chocolade | $34 | 83 | $2,822 |
| Chocolade | $34 | 21 | $714 |
| Chai | $3 | 16 | $48 |
| Chocolade | $34 | 96 | $3,264 |
| Chai | $3 | 12 | $36 |
| Chai | $3 | 75 | $225 |
| Ipon Coffee | $10 | 59 | $590 |
| Ipon Coffee | $10 | 76 | $760 |



8:19

| Product | Price | Quantity | Total |
|---|---|---|---|
| Ipon Coffee | $10 | 25 | $250 |
| Tofu | $50 | 27 | $1,350 |
| Ikura | $70 | 27 | $1,890 |
| Ravioli Angelo | $18 | 77 | $1,386 |
| Chai | $3 | 30 | $90 |
| Chai | $3 | 71 | $213 |
| Ikura | $70 | 47 | $3,290 |
| Tofu | $50 | 12 | $600 |
| Ipon Coffee | $10 | 32 | $320 |
| Boston Crab Meat | $36 | 2 | $72 |
| Ipon Coffee | $10 | 61 | $610 |
| Ikura | $70 | 36 | $2,520 |
| Ikura | $70 | 83 | $5,810 |
| Boston Crab Meat | $36 | 56 | $2,016 |
| Questo Cabrales | $25 | 38 | $950 |

An expression is a string that, when parsed and processed, evaluates some value. Expressions consist of column names, constants, operators, and functions. In this example, a Boolean expression is used. If the expression evaluates to **true**, the format is applied. The format highlights rows that have values in the **Total** column that are less than **100**.

1. Create a new FormatCondition class instance.
2. Set this object's FormatConditionBase.FieldName property to **Total**. This column provides values to test against the formatting rule.
3. Assign the expression string to the Expression property.
4. Assign one of the predefined formats to the FormatConditionBase.PredefinedFormatName property. All available format names are listed in the Predefined Format Names document.
5. Set the ApplyToRow property to **true** to apply the format to entire rows instead of individual column cells.
6. To apply a conditional formatting rule represented by the created **FormatCondition** object, add this object to the GridControl.FormatConditions collection.

**C#**

```
using DevExpress.Mobile.DataGrid;
// ...
FormatCondition condition = new FormatCondition();
condition.FieldName = "Total";
condition.Expression = "[Total] < 100";
condition.PredefinedFormatName = "LightRedFillWithDarkRedText";
condition.ApplyToRow = true;
grid.FormatConditions.Add (condition);
```

**See Also**

ExpressionConditionBase Class
ExpressionConditionBase Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# Format Property

Gets or sets the formatting settings applied to cells whose values take part in conditional formatting. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public Format Format { get; set; }
```

### Property Value

A **Format** object that provides formatting settings.

### See Also

ExpressionConditionBase Class
ExpressionConditionBase Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# FixedStyle Enumeration

Lists values that specify a column's behavior when a grid is horizontally scrolled.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public enum FixedStyle
```

### Members

| Name | Description |
|---|---|
| Left | A column is anchored to the grid's left edge. |
| None | A column is not anchored and takes part in horizontal scrolling. |
| Right | A column is anchored to the grid's right edge. |

### Remarks

The FixedStyle enumeration values are used by the GridColumn.FixedStyle property.

### See Also

DevExpress.Mobile.DataGrid Namespace
Fixed Columns

© 2018 Developer Express Inc.

# FormatCondition Class

Represents a format condition based on a specified rule(s) or expression (equal, less, between, etc.).

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

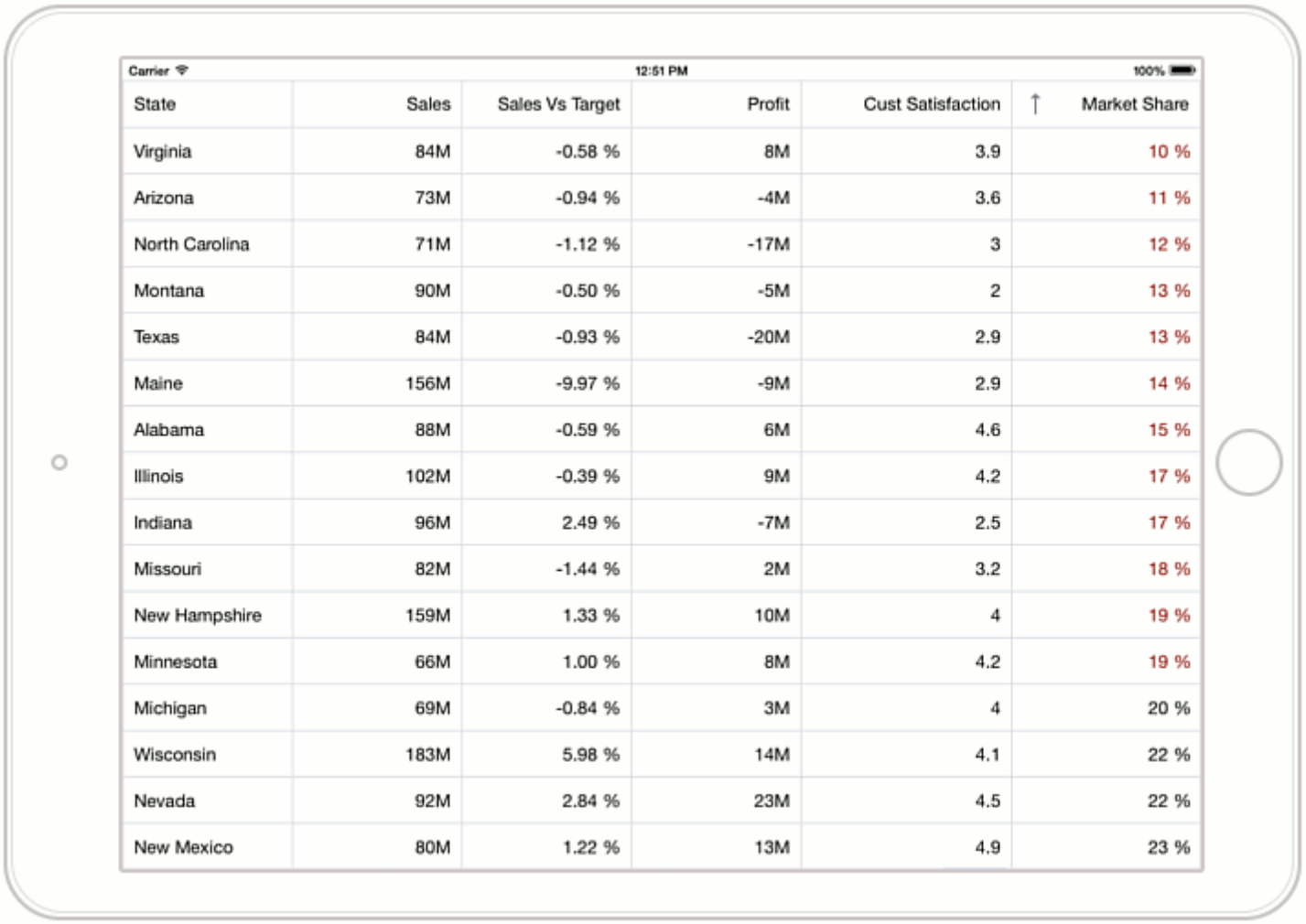
**C#**

```
public class FormatCondition : ExpressionConditionBase
```

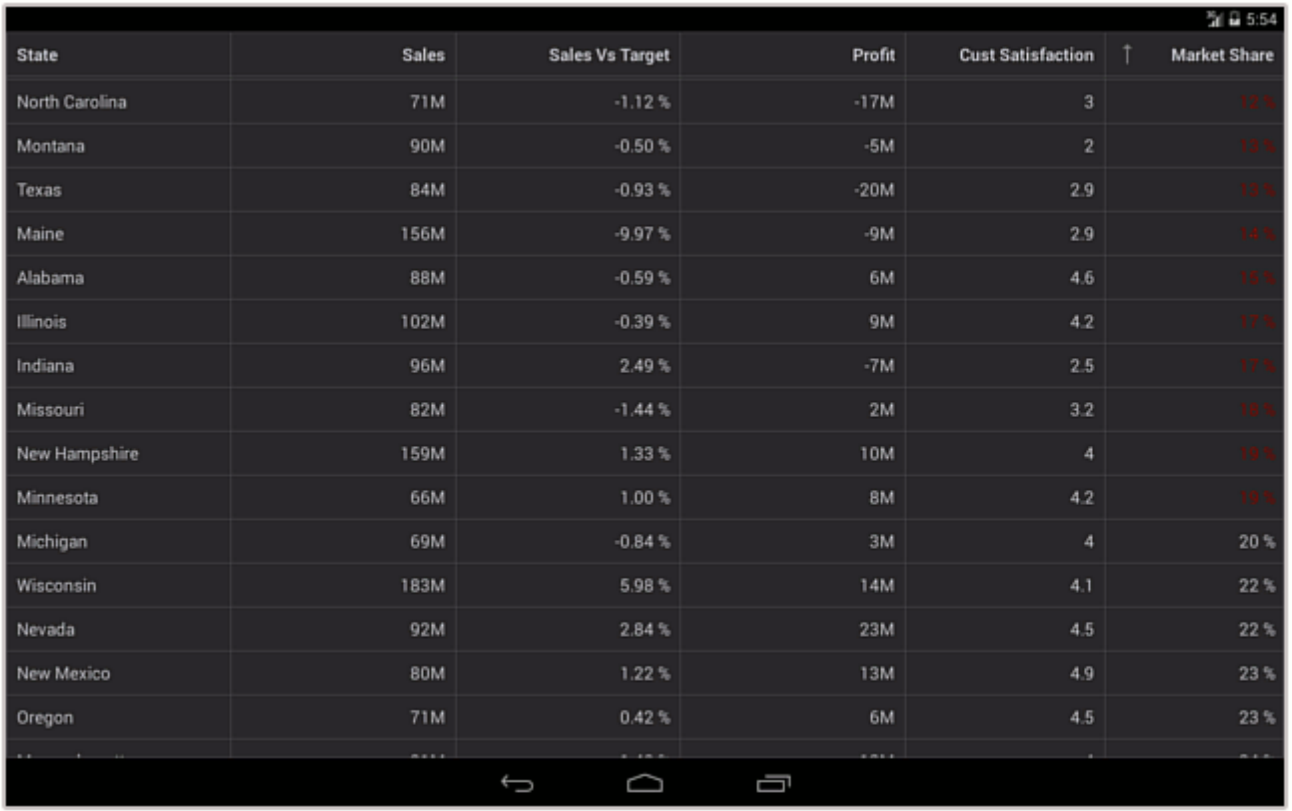
### Example

This example illustrates how to apply conditional formatting to a **Market Share** column in a GridControl to highlight cells that match a specific condition. Cells containing values that are less than 20% will be highlighted.

© 2018 Developer Express Inc.




| State | Sales | Sales Vs Target | Profit | Cust Satisfaction | ↑ Market Share |
|---|---|---|---|---|---|
| Virginia | 84M | -0.58 % | 8M | 3.9 | 10 % |
| Arizona | 73M | -0.94 % | -4M | 3.6 | 11 % |
| North Carolina | 71M | -1.12 % | -17M | 3 | 12 % |
| Montana | 90M | -0.50 % | -5M | 2 | 13 % |
| Texas | 84M | -0.93 % | -20M | 2.9 | 13 % |
| Maine | 156M | -9.97 % | -9M | 2.9 | 14 % |
| Alabama | 88M | -0.59 % | 6M | 4.6 | 15 % |
| Illinois | 102M | -0.39 % | 9M | 4.2 | 17 % |
| Indiana | 96M | 2.49 % | -7M | 2.5 | 17 % |
| Missouri | 82M | -1.44 % | 2M | 3.2 | 18 % |
| New Hampshire | 159M | 1.33 % | 10M | 4 | 19 % |
| Minnesota | 66M | 1.00 % | 8M | 4.2 | 19 % |
| Michigan | 69M | -0.84 % | 3M | 4 | 20 % |
| Wisconsin | 183M | 5.98 % | 14M | 4.1 | 22 % |
| Nevada | 92M | 2.84 % | 23M | 4.5 | 22 % |
| New Mexico | 80M | 1.22 % | 13M | 4.9 | 23 % |




| State | Sales | Sales Vs Target | Profit | Cust Satisfaction | ↑ Market Share |
|---|---|---|---|---|---|
| North Carolina | 71M | -1.12 % | -17M | 3 | 12 % |
| Montana | 90M | -0.50 % | -5M | 2 | 13 % |
| Texas | 84M | -0.93 % | -20M | 2.9 | 13 % |
| Maine | 156M | -9.97 % | -9M | 2.9 | 14 % |
| Alabama | 88M | -0.59 % | 6M | 4.6 | 15 % |
| Illinois | 102M | -0.39 % | 9M | 4.2 | 17 % |
| Indiana | 96M | 2.49 % | -7M | 2.5 | 17 % |
| Missouri | 82M | -1.44 % | 2M | 3.2 | 18 % |
| New Hampshire | 159M | 1.33 % | 10M | 4 | 19 % |
| Minnesota | 66M | 1.00 % | 8M | 4.2 | 19 % |
| Michigan | 69M | -0.84 % | 3M | 4 | 20 % |
| Wisconsin | 183M | 5.98 % | 14M | 4.1 | 22 % |
| Nevada | 92M | 2.84 % | 23M | 4.5 | 22 % |
| New Mexico | 80M | 1.22 % | 13M | 4.9 | 23 % |
| Oregon | 71M | 0.42 % | 6M | 4.5 | 23 % |

1. Create a new FormatCondition class instance.
2. Set this object's FormatConditionBase.FieldName property to **MarketShare**. This column provides values to test against the formatting rule and the specified format will be applied to this column.
3. Assign one of the predefined formats to the FormatConditionBase.PredefinedFormatName property. All available format names are listed in the Predefined Format Names document.
4. Specify the comparison operator by setting the ValueRule property. In this example, the **Less** operator is used for constructing the rule's criteria.
5. Set the Value1 property to 0.20. This means that the format should be applied to cells with market share less than 20%.
6. To apply a conditional formatting rule represented by the created **FormatCondition** object, add this object to the GridControl.FormatConditions collection.

**C#**

```
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.Core.ConditionalFormatting;
// ...
FormatCondition condition = new FormatCondition ();
condition.FieldName = "MarketShare";
condition.PredefinedFormatName = "RedText";
condition.ValueRule = ConditionRule.Less;
condition.Value1 = 0.20;
grid.FormatConditions.Add (condition);
```

### Inheritance Hierarchy

System.Object
  Xamarin.Forms.BindableObject
    FormatConditionBase
      ExpressionConditionBase
        **FormatCondition**

### See Also

FormatCondition Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# FormatCondition Members

Represents a format condition based on a specified rule(s) or expression (equal, less, between, etc.).

The following tables list the members exposed by the FormatCondition type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | FormatCondition | Initializes a new instance of the FormatCondition class with default settings. |

### Public Properties

| | Name | Description |
|---|---|---|
| | ApplyToRow | Specifies whether to apply formatting settings to the entire row. This is a bindable property. (Inherited from **ExpressionConditionBase**) |
| | BindingContext | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Expression | Gets or sets the expression specifying to which cells the format should be applied. This is a bindable property. (Inherited from **ExpressionConditionBase**) |
| | FieldName | Gets or sets the name of the column which provides values to test against the current format rule, and to which the format should be applied. This is a bindable property. (Inherited from **FormatConditionBase**) |
| | Format | Gets or sets the formatting settings applied to cells whose values take part in conditional formatting. This is a bindable property. (Inherited from **ExpressionConditionBase**) |
| | Owner | Gets the format condition's owner. This is a bindable property. (Inherited from **FormatConditionBase**) |
| | PredefinedFormatName | Gets or sets the name of the predefined format to be applied to target cells. This is a bindable property. (Inherited from **FormatConditionBase**) |
| | TypeName | Gets or sets the data type name. This property supports the internal infrastructure and is not intended to be used directly from your code. (Inherited from **FormatConditionBase**) |
| | Value1 | Gets or sets the first constant that is compared with column values. This is a bindable property. |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | Value2 | Gets or sets the second constant that is compared with column values. This is a bindable property. |
| | ValueRule | Specifies the rule for conditional formatting. This is a bindable property. |

**Public Methods**

| | Name | Description |
|---|---|---|
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Equals | (Inherited from **System.Object**) |
| S | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ToString | (Inherited from **System.Object**) |

**Public Fields**

| | Name | Description |
|---|---|---|
| S | ApplyToRowProperty | Identifies the ExpressionConditionBase.ApplyToRow bindable property. (Inherited from **ExpressionConditionBase**) |
| S | BindingContextProperty | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ExpressionProperty | Identifies the ExpressionConditionBase.Expression bindable property. (Inherited from **ExpressionConditionBase**) |
| S | FieldNameProperty | Identifies the FormatConditionBase.FieldName bindable property. (Inherited from **FormatConditionBase**) |
| S | FormatProperty | Identifies the ExpressionConditionBase.Format bindable property. (Inherited from **ExpressionConditionBase**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | OwnerProperty | Identifies the FormatConditionBase.Owner bindable property. (Inherited from **FormatConditionBase**) |
| | PredefinedFormatNameProperty | Identifies the FormatConditionBase.PredefinedFormatName bindable property. (Inherited from **FormatConditionBase**) |
| | Value1Property | Identifies the Value1 bindable property. |
| | Value2Property | Identifies the Value2 bindable property. |
| | ValueRuleProperty | Identifies the ValueRule bindable property. |

**Public Events**

| | Name | Description |
|---|---|---|
| | BindingContextChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanging | (Inherited from **Xamarin.Forms.BindableObject**) |

**See Also**
FormatCondition Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# FormatCondition Constructor

Initializes a new instance of the FormatCondition class with default settings.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**
**C#**

```
public FormatCondition()
```

**See Also**
FormatCondition Class
FormatCondition Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# FormatCondition Properties

Represents a format condition based on a specified rule(s) or expression (equal, less, between, etc.).

The following tables list the members exposed by the FormatCondition type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | FormatCondition | Initializes a new instance of the FormatCondition class with default settings. |

### Public Properties

| | Name | Description |
|---|---|---|
| | ApplyToRow | Specifies whether to apply formatting settings to the entire row. This is a bindable property. (Inherited from **ExpressionConditionBase**) |
| | BindingContext | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Expression | Gets or sets the expression specifying to which cells the format should be applied. This is a bindable property. (Inherited from **ExpressionConditionBase**) |
| | FieldName | Gets or sets the name of the column which provides values to test against the current format rule, and to which the format should be applied. This is a bindable property. (Inherited from **FormatConditionBase**) |
| | Format | Gets or sets the formatting settings applied to cells whose values take part in conditional formatting. This is a bindable property. (Inherited from **ExpressionConditionBase**) |
| | Owner | Gets the format condition's owner. This is a bindable property. (Inherited from **FormatConditionBase**) |
| | PredefinedFormatName | Gets or sets the name of the predefined format to be applied to target cells. This is a bindable property. (Inherited from **FormatConditionBase**) |
| | TypeName | Gets or sets the data type name. This property supports the internal infrastructure and is not intended to be used directly from your code. (Inherited from **FormatConditionBase**) |
| | Value1 | Gets or sets the first constant that is compared with column values. This is a bindable property. |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | Value2 | Gets or sets the second constant that is compared with column values. This is a bindable property. |
| | ValueRule | Specifies the rule for conditional formatting. This is a bindable property. |

### Public Methods

| | Name | Description |
|---|---|---|
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Equals | (Inherited from **System.Object**) |
| S | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ToString | (Inherited from **System.Object**) |

### Public Fields

| | Name | Description |
|---|---|---|
| S | ApplyToRowProperty | Identifies the ExpressionConditionBase.ApplyToRow bindable property. (Inherited from **ExpressionConditionBase**) |
| S | BindingContextProperty | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ExpressionProperty | Identifies the ExpressionConditionBase.Expression bindable property. (Inherited from **ExpressionConditionBase**) |
| S | FieldNameProperty | Identifies the FormatConditionBase.FieldName bindable property. (Inherited from **FormatConditionBase**) |
| S | FormatProperty | Identifies the ExpressionConditionBase.Format bindable property. (Inherited from **ExpressionConditionBase**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | OwnerProperty | Identifies the FormatConditionBase.Owner bindable property. (Inherited from **FormatConditionBase**) |
| | PredefinedFormatNameProperty | Identifies the FormatConditionBase.PredefinedFormatName bindable property. (Inherited from **FormatConditionBase**) |
| | Value1Property | Identifies the Value1 bindable property. |
| | Value2Property | Identifies the Value2 bindable property. |
| | ValueRuleProperty | Identifies the ValueRule bindable property. |

**Public Events**

| | Name | Description |
|---|---|---|
| | BindingContextChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanging | (Inherited from **Xamarin.Forms.BindableObject**) |

**See Also**
FormatCondition Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# Value1 Property

Gets or sets the first constant that is compared with column values. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public object Value1 { get; set; }
```

### Property Value

A value that is compared with column values.

### Remarks

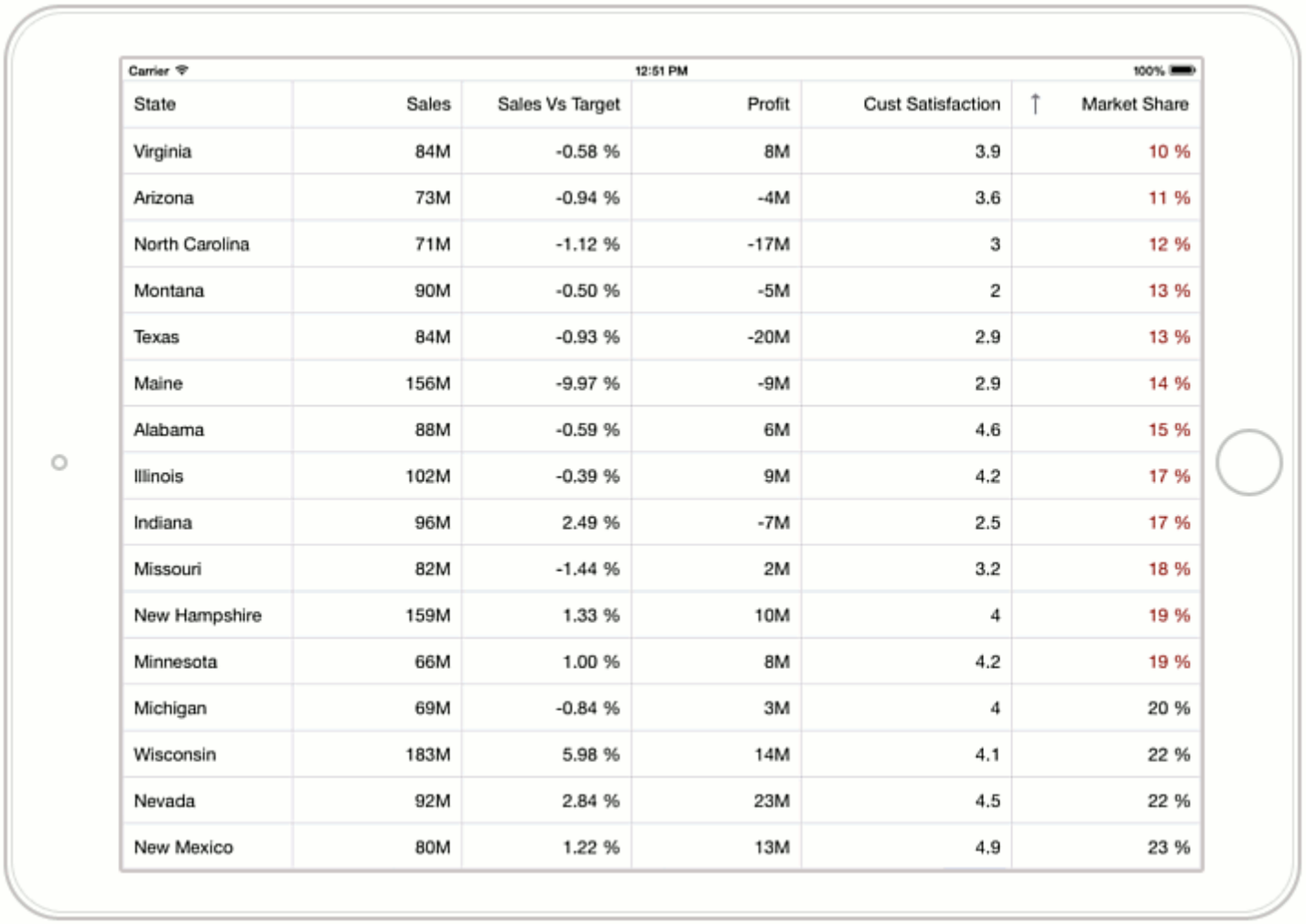
Column cells are compared with the **Value1** value using the ValueRule comparison operator. Specific comparison operators require two values. In this case, use **Value1** along with Value2.

The **Value1** and Value2 properties are not supported if the ValueRule property is set to ConditionRule.None or ConditionRule.Expression.

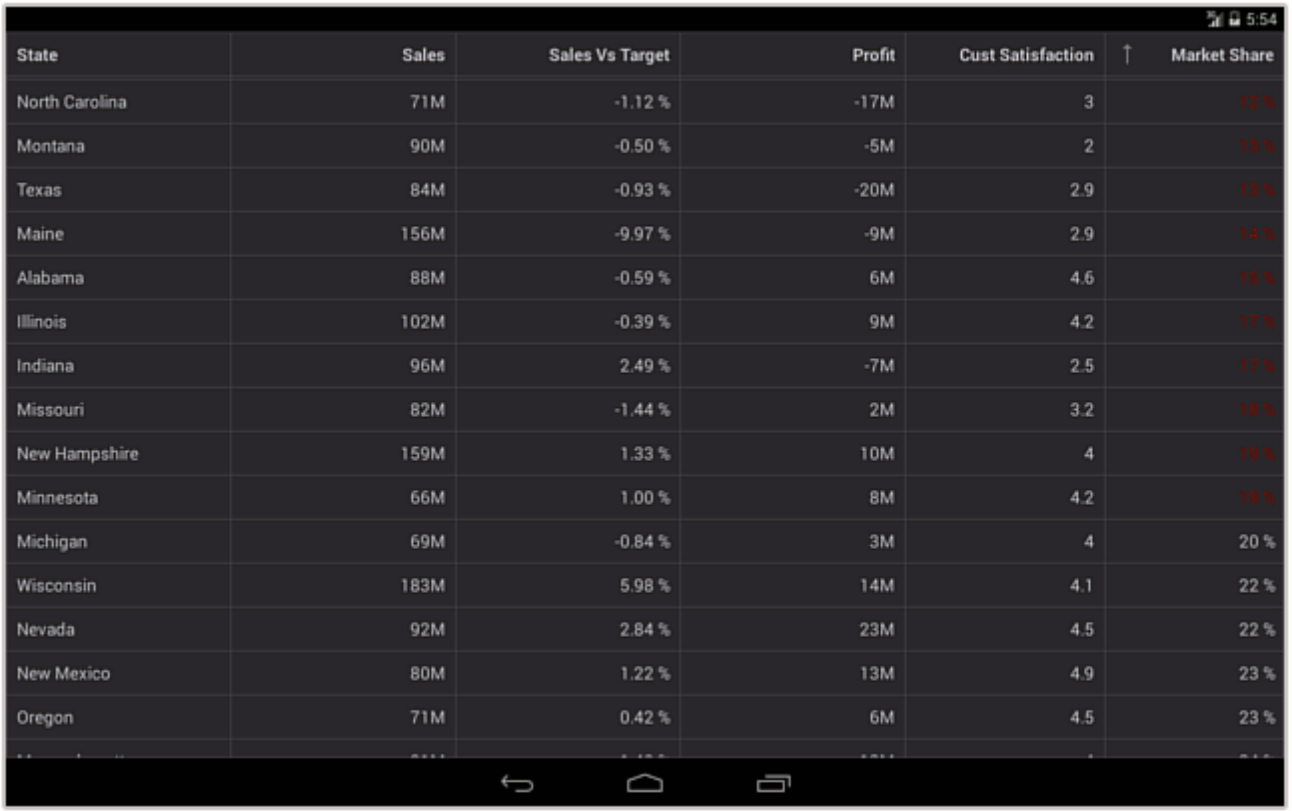
### Example

This example illustrates how to apply conditional formatting to a **Market Share** column in a GridControl to highlight cells that match a specific condition. Cells containing values that are less than 20% will be highlighted.

© 2018 Developer Express Inc.




| State | Sales | Sales Vs Target | Profit | Cust Satisfaction | ↑ Market Share |
|---|---|---|---|---|---|
| Virginia | 84M | -0.58 % | 8M | 3.9 | 10 % |
| Arizona | 73M | -0.94 % | -4M | 3.6 | 11 % |
| North Carolina | 71M | -1.12 % | -17M | 3 | 12 % |
| Montana | 90M | -0.50 % | -5M | 2 | 13 % |
| Texas | 84M | -0.93 % | -20M | 2.9 | 13 % |
| Maine | 156M | -9.97 % | -9M | 2.9 | 14 % |
| Alabama | 88M | -0.59 % | 6M | 4.6 | 15 % |
| Illinois | 102M | -0.39 % | 9M | 4.2 | 17 % |
| Indiana | 96M | 2.49 % | -7M | 2.5 | 17 % |
| Missouri | 82M | -1.44 % | 2M | 3.2 | 18 % |
| New Hampshire | 159M | 1.33 % | 10M | 4 | 19 % |
| Minnesota | 66M | 1.00 % | 8M | 4.2 | 19 % |
| Michigan | 69M | -0.84 % | 3M | 4 | 20 % |
| Wisconsin | 183M | 5.98 % | 14M | 4.1 | 22 % |
| Nevada | 92M | 2.84 % | 23M | 4.5 | 22 % |
| New Mexico | 80M | 1.22 % | 13M | 4.9 | 23 % |




| State | Sales | Sales Vs Target | Profit | Cust Satisfaction | ↑ Market Share |
|---|---|---|---|---|---|
| North Carolina | 71M | -1.12 % | -17M | 3 | 12 % |
| Montana | 90M | -0.50 % | -5M | 2 | 13 % |
| Texas | 84M | -0.93 % | -20M | 2.9 | 13 % |
| Maine | 156M | -9.97 % | -9M | 2.9 | 14 % |
| Alabama | 88M | -0.59 % | 6M | 4.6 | 15 % |
| Illinois | 102M | -0.39 % | 9M | 4.2 | 17 % |
| Indiana | 96M | 2.49 % | -7M | 2.5 | 17 % |
| Missouri | 82M | -1.44 % | 2M | 3.2 | 18 % |
| New Hampshire | 159M | 1.33 % | 10M | 4 | 19 % |
| Minnesota | 66M | 1.00 % | 8M | 4.2 | 19 % |
| Michigan | 69M | -0.84 % | 3M | 4 | 20 % |
| Wisconsin | 183M | 5.98 % | 14M | 4.1 | 22 % |
| Nevada | 92M | 2.84 % | 23M | 4.5 | 22 % |
| New Mexico | 80M | 1.22 % | 13M | 4.9 | 23 % |
| Oregon | 71M | 0.42 % | 6M | 4.5 | 23 % |

1. Create a new FormatCondition class instance.
2. Set this object's FormatConditionBase.FieldName property to **MarketShare**. This column provides values to test against the formatting rule and the specified format will be applied to this column.
3. Assign one of the predefined formats to the FormatConditionBase.PredefinedFormatName property. All available format names are listed in the Predefined Format Names document.
4. Specify the comparison operator by setting the ValueRule property. In this example, the **Less** operator is used for constructing the rule's criteria.
5. Set the Value1 property to 0.20. This means that the format should be applied to cells with market share less than 20%.
6. To apply a conditional formatting rule represented by the created **FormatCondition** object, add this object to the GridControl.FormatConditions collection.

**C#**

```
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.Core.ConditionalFormatting;
// ...
FormatCondition condition = new FormatCondition ();
condition.FieldName = "MarketShare";
condition.PredefinedFormatName = "RedText";
condition.ValueRule = ConditionRule.Less;
condition.Value1 = 0.20;
grid.FormatConditions.Add (condition);
```

**See Also**

FormatCondition Class
FormatCondition Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# Value2 Property

Gets or sets the second constant that is compared with column values. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public object Value2 { get; set; }
```

### Property Value

A value that is compared with column values.

### Remarks

Column cells are compared with the Value1 value using the ValueRule comparison operator. Specific comparison operators require two values. In this case, use Value1 along with **Value2**.

The Value1 and **Value2** properties are not supported if the ValueRule property is set to ConditionRule.None or ConditionRule.Expression.

### See Also

FormatCondition Class
FormatCondition Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# ValueRule Property

Specifies the rule for conditional formatting. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public ConditionRule ValueRule { get; set; }
```

### Property Value

A ConditionRule enumeration value.

### Remarks

The **ValueRule** property specifies how cell values should be compared to the Value1 and Value2 property values in order to determine whether the conditional format should be applied to them.

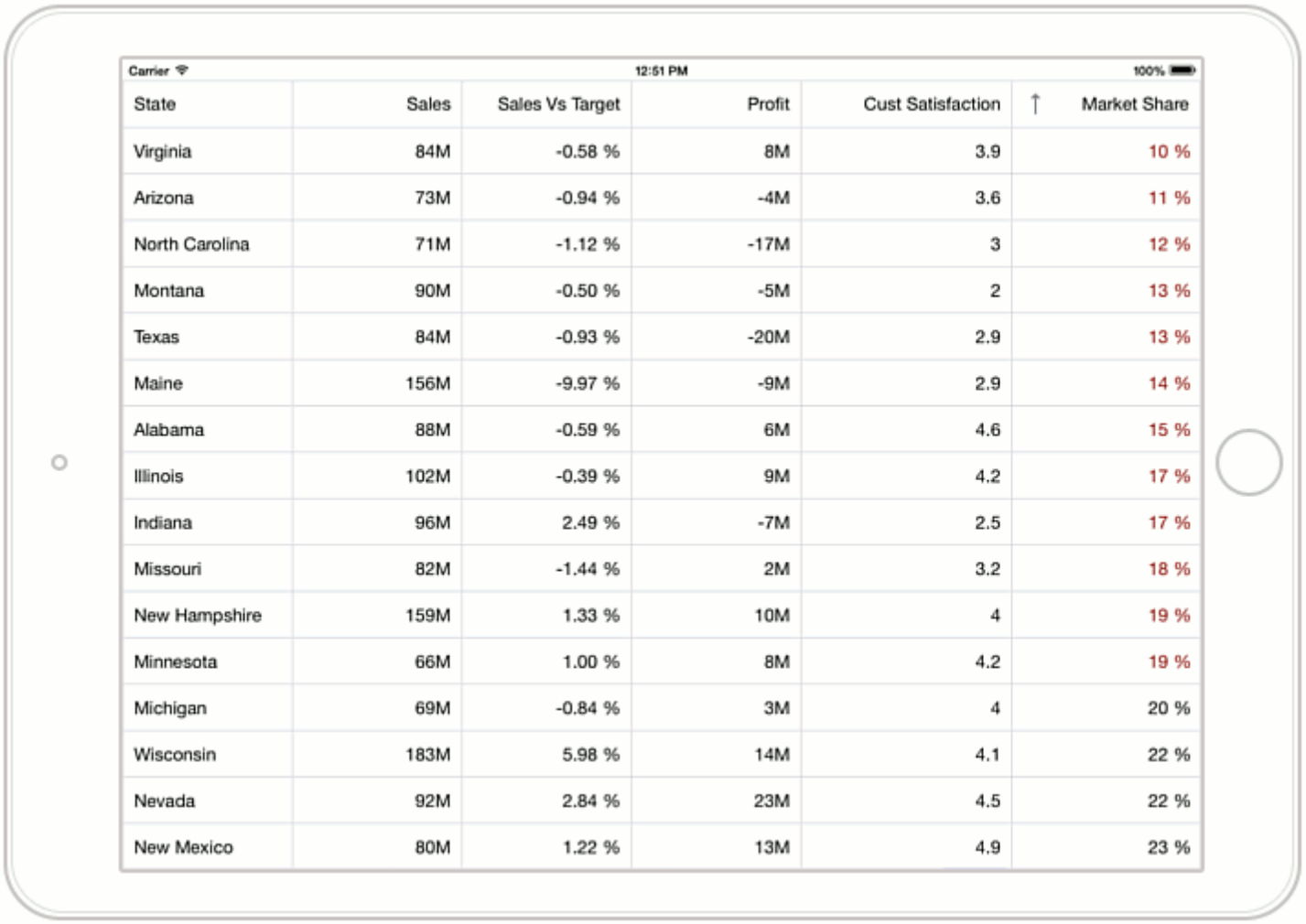
If the **ValueRule** property is set to ConditionRule.None, the format is applied to all cells of the target column.

The **ValueRule** property's default value is ConditionRule.Expression. In this case, a custom rule to format cells is specified by the ExpressionConditionBase.Expression property.

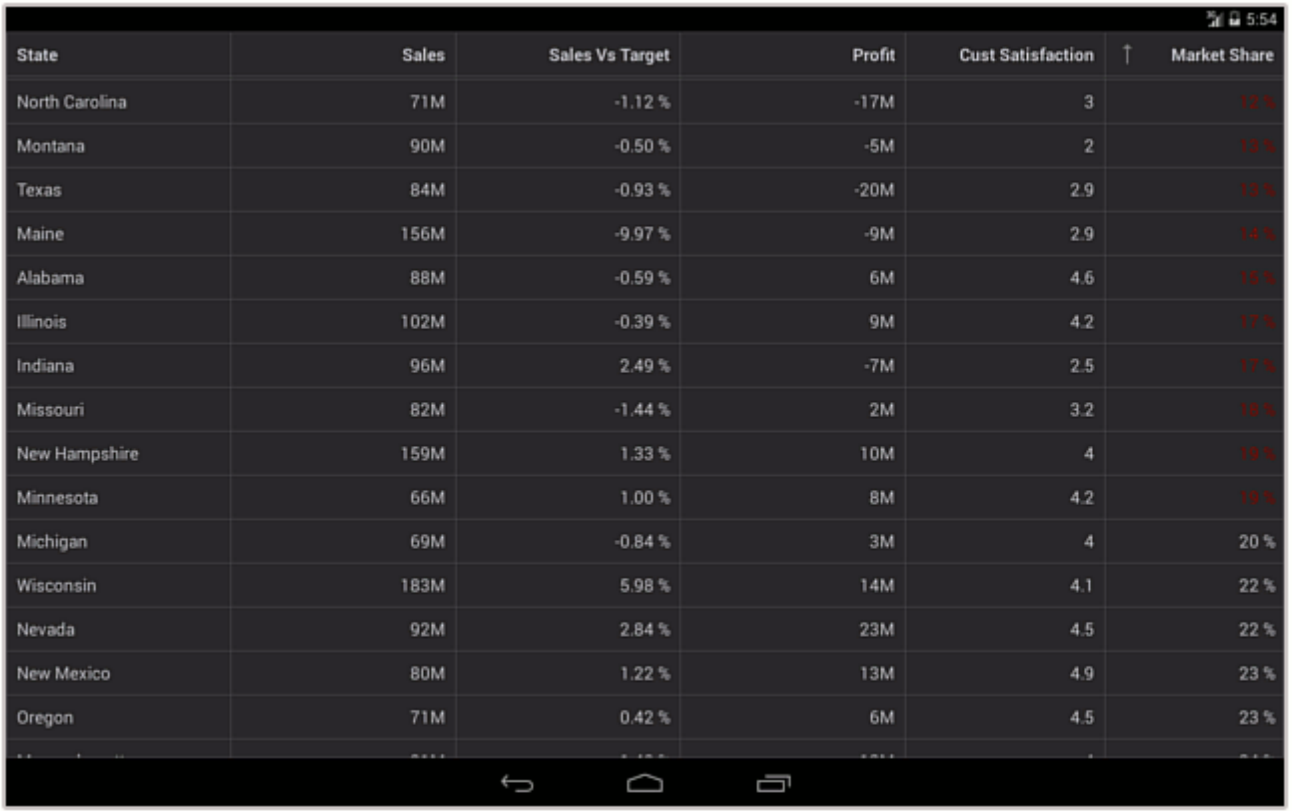
### Example

This example illustrates how to apply conditional formatting to a **Market Share** column in a GridControl to highlight cells that match a specific condition. Cells containing values that are less than 20% will be highlighted.

© 2018 Developer Express Inc.




| State | Sales | Sales Vs Target | Profit | Cust Satisfaction | ↑ Market Share |
|---|---|---|---|---|---|
| Virginia | 84M | -0.58 % | 8M | 3.9 | 10 % |
| Arizona | 73M | -0.94 % | -4M | 3.6 | 11 % |
| North Carolina | 71M | -1.12 % | -17M | 3 | 12 % |
| Montana | 90M | -0.50 % | -5M | 2 | 13 % |
| Texas | 84M | -0.93 % | -20M | 2.9 | 13 % |
| Maine | 156M | -9.97 % | -9M | 2.9 | 14 % |
| Alabama | 88M | -0.59 % | 6M | 4.6 | 15 % |
| Illinois | 102M | -0.39 % | 9M | 4.2 | 17 % |
| Indiana | 96M | 2.49 % | -7M | 2.5 | 17 % |
| Missouri | 82M | -1.44 % | 2M | 3.2 | 18 % |
| New Hampshire | 159M | 1.33 % | 10M | 4 | 19 % |
| Minnesota | 66M | 1.00 % | 8M | 4.2 | 19 % |
| Michigan | 69M | -0.84 % | 3M | 4 | 20 % |
| Wisconsin | 183M | 5.98 % | 14M | 4.1 | 22 % |
| Nevada | 92M | 2.84 % | 23M | 4.5 | 22 % |
| New Mexico | 80M | 1.22 % | 13M | 4.9 | 23 % |




| State | Sales | Sales Vs Target | Profit | Cust Satisfaction | ↑ Market Share |
|---|---|---|---|---|---|
| North Carolina | 71M | -1.12 % | -17M | 3 | 12 % |
| Montana | 90M | -0.50 % | -5M | 2 | 13 % |
| Texas | 84M | -0.93 % | -20M | 2.9 | 13 % |
| Maine | 156M | -9.97 % | -9M | 2.9 | 14 % |
| Alabama | 88M | -0.59 % | 6M | 4.6 | 15 % |
| Illinois | 102M | -0.39 % | 9M | 4.2 | 17 % |
| Indiana | 96M | 2.49 % | -7M | 2.5 | 17 % |
| Missouri | 82M | -1.44 % | 2M | 3.2 | 18 % |
| New Hampshire | 159M | 1.33 % | 10M | 4 | 19 % |
| Minnesota | 66M | 1.00 % | 8M | 4.2 | 19 % |
| Michigan | 69M | -0.84 % | 3M | 4 | 20 % |
| Wisconsin | 183M | 5.98 % | 14M | 4.1 | 22 % |
| Nevada | 92M | 2.84 % | 23M | 4.5 | 22 % |
| New Mexico | 80M | 1.22 % | 13M | 4.9 | 23 % |
| Oregon | 71M | 0.42 % | 6M | 4.5 | 23 % |

1. Create a new FormatCondition class instance.
2. Set this object's FormatConditionBase.FieldName property to **MarketShare**. This column provides values to test against the formatting rule and the specified format will be applied to this column.
3. Assign one of the predefined formats to the FormatConditionBase.PredefinedFormatName property. All available format names are listed in the Predefined Format Names document.
4. Specify the comparison operator by setting the ValueRule property. In this example, the **Less** operator is used for constructing the rule's criteria.
5. Set the Value1 property to 0.20. This means that the format should be applied to cells with market share less than 20%.
6. To apply a conditional formatting rule represented by the created **FormatCondition** object, add this object to the GridControl.FormatConditions collection.

**C#**

```
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.Core.ConditionalFormatting;
// ...
FormatCondition condition = new FormatCondition ();
condition.FieldName = "MarketShare";
condition.PredefinedFormatName = "RedText";
condition.ValueRule = ConditionRule.Less;
condition.Value1 = 0.20;
grid.FormatConditions.Add (condition);
```

**See Also**
FormatCondition Class
FormatCondition Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# FormatConditionBase Class

Serves as a base for classes that represent format conditions.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public abstract class FormatConditionBase : BindableObject
```

### Remarks

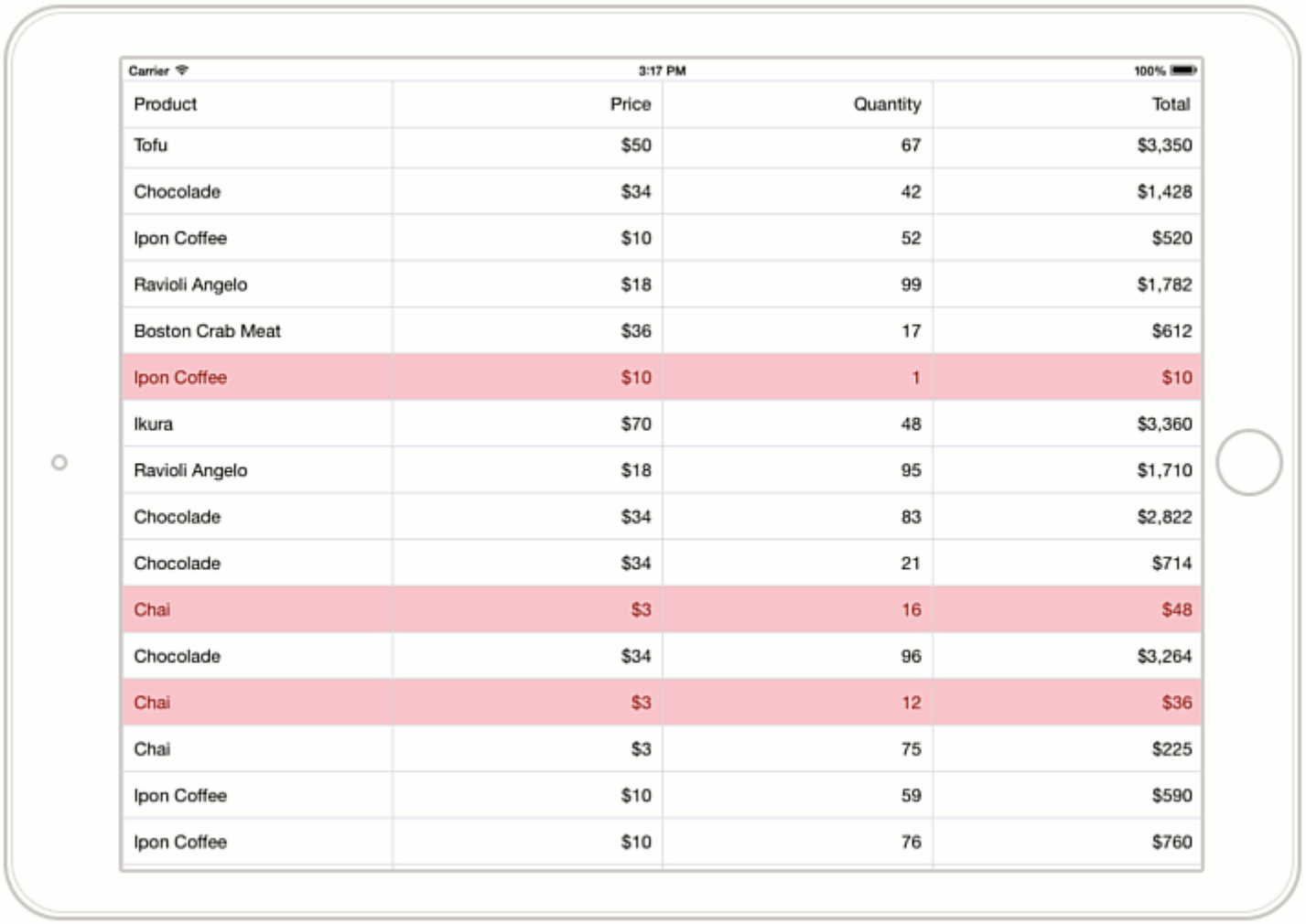
The **FormatConditionBase** class implements the base functionality for classes that represent format conditions.

A format condition defines appearance settings and criteria. When a cell value matches the criteria, the appearance settings are applied to this cell or row containing these cells.

### Example

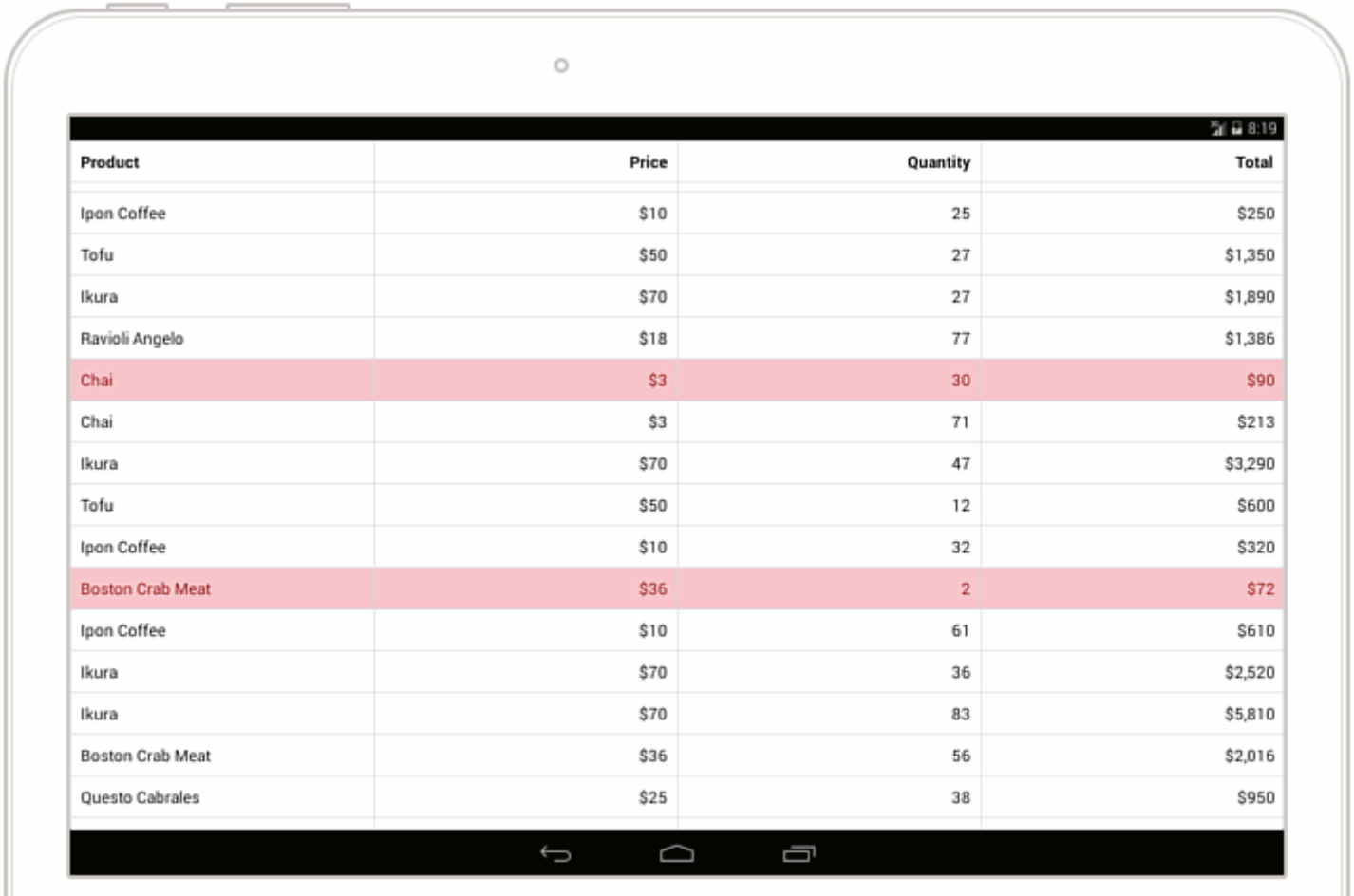
This example illustrates how to apply a format to rows in a GridControl that match a specific expression.

© 2018 Developer Express Inc.



Carrier 3:17 PM 100%

| Product | Price | Quantity | Total |
|---|---|---|---|
| Tofu | $50 | 67 | $3,350 |
| Chocolade | $34 | 42 | $1,428 |
| Ipon Coffee | $10 | 52 | $520 |
| Ravioli Angelo | $18 | 99 | $1,782 |
| Boston Crab Meat | $36 | 17 | $612 |
| Ipon Coffee | $10 | 1 | $10 |
| Ikura | $70 | 48 | $3,360 |
| Ravioli Angelo | $18 | 95 | $1,710 |
| Chocolade | $34 | 83 | $2,822 |
| Chocolade | $34 | 21 | $714 |
| Chai | $3 | 16 | $48 |
| Chocolade | $34 | 96 | $3,264 |
| Chai | $3 | 12 | $36 |
| Chai | $3 | 75 | $225 |
| Ipon Coffee | $10 | 59 | $590 |
| Ipon Coffee | $10 | 76 | $760 |



8:19

| Product | Price | Quantity | Total |
|---|---|---|---|
| Ipon Coffee | $10 | 25 | $250 |
| Tofu | $50 | 27 | $1,350 |
| Ikura | $70 | 27 | $1,890 |
| Ravioli Angelo | $18 | 77 | $1,386 |
| Chai | $3 | 30 | $90 |
| Chai | $3 | 71 | $213 |
| Ikura | $70 | 47 | $3,290 |
| Tofu | $50 | 12 | $600 |
| Ipon Coffee | $10 | 32 | $320 |
| Boston Crab Meat | $36 | 2 | $72 |
| Ipon Coffee | $10 | 61 | $610 |
| Ikura | $70 | 36 | $2,520 |
| Ikura | $70 | 83 | $5,810 |
| Boston Crab Meat | $36 | 56 | $2,016 |
| Questo Cabrales | $25 | 38 | $950 |

An expression is a string that, when parsed and processed, evaluates some value. Expressions consist of column names, constants, operators, and functions. In this example, a Boolean expression is used. If the expression evaluates to **true**, the format is applied. The format highlights rows that have values in the **Total** column that are less than **100**.

1. Create a new FormatCondition class instance.
2. Set this object's FieldName property to **Total**. This column provides values to test against the formatting rule.
3. Assign the expression string to the ExpressionConditionBase.Expression property.
4. Assign one of the predefined formats to the PredefinedFormatName property. All available format names are listed in the Predefined Format Names document.
5. Set the ExpressionConditionBase.ApplyToRow property to **true** to apply the format to entire rows instead of individual column cells.
6. To apply a conditional formatting rule represented by the created **FormatCondition** object, add this object to the GridControl.FormatConditions collection.

**C#**

```
using DevExpress.Mobile.DataGrid;
// ...
FormatCondition condition = new FormatCondition();
condition.FieldName = "Total";
condition.Expression = "[Total] < 100";
condition.PredefinedFormatName = "LightRedFillWithDarkRedText";
condition.ApplyToRow = true;
grid.FormatConditions.Add (condition);
```

### Inheritance Hierarchy

System.Object
  Xamarin.Forms.BindableObject
    **FormatConditionBase**
      Derived classes

### See Also

FormatConditionBase Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# FormatConditionBase Members

Serves as a base for classes that represent format conditions.

The following tables list the members exposed by the FormatConditionBase type.

### Public Properties

| | Name | Description |
|---|---|---|
| | BindingContext | (Inherited from **Xamarin.Forms.BindableObject**) |
| | FieldName | Gets or sets the name of the column which provides values to test against the current format rule, and to which the format should be applied. This is a bindable property. |
| | Owner | Gets the format condition's owner. This is a bindable property. |
| | PredefinedFormatName | Gets or sets the name of the predefined format to be applied to target cells. This is a bindable property. |
| | TypeName | Gets or sets the data type name. This property supports the internal infrastructure and is not intended to be used directly from your code. |

### Public Methods

| | Name | Description |
|---|---|---|
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Equals | (Inherited from **System.Object**) |
| S | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ToString | (Inherited from **System.Object**) |

### Public Fields

© 2018 Developer Express Inc.

| | Name | Description |
|---|---|---|
| | BindingContextProperty | (Inherited from **Xamarin.Forms.BindableObject**) |
| | FieldNameProperty | Identifies the FieldName bindable property. |
| | OwnerProperty | Identifies the Owner bindable property. |
| | PredefinedFormatNameProperty | Identifies the PredefinedFormatName bindable property. |

**Public Events**

| | Name | Description |
|---|---|---|
| | BindingContextChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanging | (Inherited from **Xamarin.Forms.BindableObject**) |

**See Also**
FormatConditionBase Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# FormatConditionBase Properties

Serves as a base for classes that represent format conditions.

The following tables list the members exposed by the FormatConditionBase type.

### Public Properties

| | Name | Description |
|---|---|---|
| | BindingContext | (Inherited from **Xamarin.Forms.BindableObject**) |
| | FieldName | Gets or sets the name of the column which provides values to test against the current format rule, and to which the format should be applied. This is a bindable property. |
| | Owner | Gets the format condition's owner. This is a bindable property. |
| | PredefinedFormatName | Gets or sets the name of the predefined format to be applied to target cells. This is a bindable property. |
| | TypeName | Gets or sets the data type name. This property supports the internal infrastructure and is not intended to be used directly from your code. |

### Public Methods

| | Name | Description |
|---|---|---|
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Equals | (Inherited from **System.Object**) |
| S | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ToString | (Inherited from **System.Object**) |

### Public Fields

© 2018 Developer Express Inc.

| | Name | Description |
|---|---|---|
| | BindingContextProperty | (Inherited from **Xamarin.Forms.BindableObject**) |
| | FieldNameProperty | Identifies the FieldName bindable property. |
| | OwnerProperty | Identifies the Owner bindable property. |
| | PredefinedFormatNameProperty | Identifies the PredefinedFormatName bindable property. |

**Public Events**

| | Name | Description |
|---|---|---|
| | BindingContextChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanging | (Inherited from **Xamarin.Forms.BindableObject**) |

**See Also**
FormatConditionBase Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# FieldName Property

Gets or sets the name of the column which provides values to test against the current format rule, and to which the format should be applied. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public string FieldName { get; set; }
```

### Property Value

A System.String value specifying the column name.

### Example

This example shows how to create conditional formatting rules for the GridControl in XAML.

| State | Sales | Sales Vs Target | Profit | Cust Satisfaction | Market Share |
|---|---|---|---|---|---|
| Montana | 90M | -0.50 % | -5M | 2 | 13 % |
| Indiana | 96M | 2.49 % | -7M | 2.5 | 17 % |
| Maine | 156M | -9.97 % | -9M | 2.9 | 14 % |
| Texas | 84M | -0.93 % | -20M | 2.9 | 13 % |
| Connecticut | 73M | 3.38 % | 8M | 2.9 | 24 % |
| North Carolina | 71M | -1.12 % | -17M | 3 | 12 % |
| Missouri | 82M | -1.44 % | 2M | 3.2 | 18 % |
| Ohio | 251M | -2.88 % | -15M | 3.4 | 29 % |
| Arizona | 73M | -0.94 % | -4M | 3.6 | 11 % |
| Wyoming | 98M | 0.21 % | 4M | 3.7 | 31 % |
| Idaho | 119M | 0.65 % | 14M | 3.9 | 30 % |
| New York | 88M | 1.15 % | 15M | 3.9 | 30 % |
| Virginia | 84M | -0.58 % | 8M | 3.9 | 10 % |
| New Hampshire | 159M | 1.33 % | 10M | 4 | 19 % |
| Massachusetts | 91M | 1.40 % | 12M | 4 | 24 % |
| Michigan | 69M | -0.84 % | 3M | 4 | 20 % |

Formatting rules specified by the FormatConditionBase class descendants (DataBarFormatCondition, FormatCondition, TopBottomRuleFormatCondition and IconSetFormatCondition) are added to the GridControl.FormatConditions collection.

| Xaml | |
|---|---|

© 2018 Developer Express Inc.

```
<?xml version="1.0" encoding="UTF-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="HelloGrid.MainPage"
             xmlns:dxg="clr-namespace:DevExpress.Mobile.DataGrid;assembly=DevExpress.Mobile.Grid.v15.1">
    <dxg:GridControl x:Name = "grid" ItemsSource="{Binding Items}">
         <!-- ... -->
         <dxg:GridControl.FormatConditions>
            <dxg:DataBarFormatCondition FieldName="Sales" PredefinedFormatName="BlueGradientDataBar"/>
            <dxg:FormatCondition FieldName="SalesVsTarget" Expression="[SalesVsTarget]&lt;0" PredefinedFo
            <dxg:FormatCondition FieldName="SalesVsTarget" Expression="[SalesVsTarget]>=0" PredefinedForma
            <dxg:DataBarFormatCondition FieldName="Profit" PredefinedFormatName="GreenGradientDataBar" />
            <dxg:TopBottomRuleFormatCondition FieldName="CustomersSatisfaction" Rule="BottomPercent" Thres
            <dxg:IconSetFormatCondition FieldName="MarketShare" PredefinedFormatName="Arrows3ColoredIconSe
         </dxg:GridControl.FormatConditions>
    </dxg:GridControl>
</ContentPage>
```

**See Also**

FormatConditionBase Class
FormatConditionBase Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# Owner Property

Gets the format condition's owner. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public FormatConditionCollection Owner { get; }
```

### Property Value

A FormatConditionCollection that is the format condition's owner.

### See Also

FormatConditionBase Class
FormatConditionBase Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# PredefinedFormatName Property

Gets or sets the name of the predefined format to be applied to target cells. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public string PredefinedFormatName { get; set; }
```

### Property Value

A System.String value that is the predefined format name.

### Remarks

Available predefined formats are listed in the Predefined Format Names document.

### Example

This example shows how to create conditional formatting rules for the GridControl in XAML.

| State | Sales | Sales Vs Target | Profit | Cust Satisfaction | Market Share |
|---|---|---|---|---|---|
| Montana | 90M | -0.50 % | -5M | 2 | 13 % |
| Indiana | 96M | 2.49 % | -7M | 2.5 | 17 % |
| Maine | 156M | -9.97 % | -9M | 2.9 | 14 % |
| Texas | 84M | -0.93 % | -20M | 2.9 | 13 % |
| Connecticut | 73M | 3.38 % | 8M | 2.9 | 24 % |
| North Carolina | 71M | -1.12 % | -17M | 3 | 12 % |
| Missouri | 82M | -1.44 % | 2M | 3.2 | 18 % |
| Ohio | 251M | -2.88 % | -15M | 3.4 | 29 % |
| Arizona | 73M | -0.94 % | -4M | 3.6 | 11 % |
| Wyoming | 98M | 0.21 % | 4M | 3.7 | 31 % |
| Idaho | 119M | 0.65 % | 14M | 3.9 | 30 % |
| New York | 88M | 1.15 % | 15M | 3.9 | 30 % |
| Virginia | 84M | -0.58 % | 8M | 3.9 | 10 % |
| New Hampshire | 159M | 1.33 % | 10M | 4 | 19 % |
| Massachusetts | 91M | 1.40 % | 12M | 4 | 24 % |
| Michigan | 69M | -0.84 % | 3M | 4 | 20 % |

Formatting rules specified by the FormatConditionBase class descendants (DataBarFormatCondition, FormatCondition, TopBottomRuleFormatCondition and IconSetFormatCondition) are added to the GridControl.FormatConditions collection.

| Xaml | |
|---|---|

© 2018 Developer Express Inc.

```
<?xml version="1.0" encoding="UTF-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="HelloGrid.MainPage"
             xmlns:dxg="clr-namespace:DevExpress.Mobile.DataGrid;assembly=DevExpress.Mobile.Grid.v15.1">
    <dxg:GridControl x:Name = "grid" ItemsSource="{Binding Items}">
         <!-- ... -->
         <dxg:GridControl.FormatConditions>
            <dxg:DataBarFormatCondition FieldName="Sales" PredefinedFormatName="BlueGradientDataBar"/>
            <dxg:FormatCondition FieldName="SalesVsTarget" Expression="[SalesVsTarget]&lt;0" PredefinedFo
            <dxg:FormatCondition FieldName="SalesVsTarget" Expression="[SalesVsTarget]>=0" PredefinedForma
            <dxg:DataBarFormatCondition FieldName="Profit" PredefinedFormatName="GreenGradientDataBar" />
            <dxg:TopBottomRuleFormatCondition FieldName="CustomersSatisfaction" Rule="BottomPercent" Thres
            <dxg:IconSetFormatCondition FieldName="MarketShare" PredefinedFormatName="Arrows3ColoredIconSe
         </dxg:GridControl.FormatConditions>
    </dxg:GridControl>
</ContentPage>
```

**See Also**

FormatConditionBase Class
FormatConditionBase Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# TypeName Property

Gets or sets the data type name. This property supports the internal infrastructure and is not intended to be used directly from your code.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public virtual string TypeName { get; set; }
```

### Property Value

A System.String value.

### Remarks

The **TypeName** property is a utility member that is used for serialization.

### See Also

FormatConditionBase Class
FormatConditionBase Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# FormatConditionCollection Class

A collection of format conditions.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public class FormatConditionCollection : ObservableCollectionCore<FormatConditionBase>
```

### Remarks

To manage the collection of format conditions applied in the grid, use the GridControl.FormatConditions property. See the Conditional Formatting section for examples.

### Example

This example shows how to create conditional formatting rules for the GridControl in XAML.

| State | Sales | Sales Vs Target | Profit | Cust Satisfaction | Market Share |
|---|---|---|---|---|---|
| Montana | 90M | -0.50 % | -5M | 2 | 13 % |
| Indiana | 96M | 2.49 % | -7M | 2.5 | 17 % |
| Maine | 156M | -9.97 % | -9M | 2.9 | 14 % |
| Texas | 84M | -0.93 % | -20M | 2.9 | 13 % |
| Connecticut | 73M | 3.38 % | 8M | 2.9 | 24 % |
| North Carolina | 71M | -1.12 % | -17M | 3 | 12 % |
| Missouri | 82M | -1.44 % | 2M | 3.2 | 18 % |
| Ohio | 251M | -2.88 % | -15M | 3.4 | 29 % |
| Arizona | 73M | -0.94 % | -4M | 3.6 | 11 % |
| Wyoming | 98M | 0.21 % | 4M | 3.7 | 31 % |
| Idaho | 119M | 0.65 % | 14M | 3.9 | 30 % |
| New York | 88M | 1.15 % | 15M | 3.9 | 30 % |
| Virginia | 84M | -0.58 % | 8M | 3.9 | 10 % |
| New Hampshire | 159M | 1.33 % | 10M | 4 | 19 % |
| Massachusetts | 91M | 1.40 % | 12M | 4 | 24 % |
| Michigan | 69M | -0.84 % | 3M | 4 | 20 % |

Formatting rules specified by the FormatConditionBase class descendants (DataBarFormatCondition, FormatCondition, TopBottomRuleFormatCondition and IconSetFormatCondition) are added to the GridControl.FormatConditions collection.

| Xaml | |
|---|---|

```
<?xml version="1.0" encoding="UTF-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="HelloGrid.MainPage"
             xmlns:dxg="clr-namespace:DevExpress.Mobile.DataGrid;assembly=DevExpress.Mobile.Grid.v15.1">
    <dxg:GridControl x:Name = "grid" ItemsSource="{Binding Items}">
         <!-- ... -->
         <dxg:GridControl.FormatConditions>
            <dxg:DataBarFormatCondition FieldName="Sales" PredefinedFormatName="BlueGradientDataBar"/>
            <dxg:FormatCondition FieldName="SalesVsTarget" Expression="[SalesVsTarget]&lt;0" PredefinedFo
            <dxg:FormatCondition FieldName="SalesVsTarget" Expression="[SalesVsTarget]>=0" PredefinedForma
            <dxg:DataBarFormatCondition FieldName="Profit" PredefinedFormatName="GreenGradientDataBar" />
            <dxg:TopBottomRuleFormatCondition FieldName="CustomersSatisfaction" Rule="BottomPercent" Thres
            <dxg:IconSetFormatCondition FieldName="MarketShare" PredefinedFormatName="Arrows3ColoredIconSe
         </dxg:GridControl.FormatConditions>
    </dxg:GridControl>
</ContentPage>
```

### Inheritance Hierarchy

System.Object
  System.Collections.ObjectModel.Collection<DevExpress.Mobile.DataGrid.FormatConditionBase>
    System.Collections.ObjectModel.ObservableCollection<DevExpress.Mobile.DataGrid.FormatConditionBase>
      DevExpress.Core.ObservableCollectionCore<DevExpress.Mobile.DataGrid.FormatConditionBase>
        **FormatConditionCollection**

### See Also

FormatConditionCollection Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# FormatConditionCollection Members

A collection of format conditions.

The following tables list the members exposed by the FormatConditionCollection type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | FormatConditionCollection | Initializes a new instance of the FormatConditionCollection class. |

### Public Properties

| | Name | Description |
|---|---|---|
| | Count | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | IsLockUpdate | (Inherited from **DevExpress.Core.ObservableCollectionCore<T>**) |
| | Item | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | Owner | Gets the format condition collection's owner. |

### Public Methods

| | Name | Description |
|---|---|---|
| | Add | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | Assign | (Inherited from **DevExpress.Core.ObservableCollectionCore<T>**) |
| | BeginUpdate | (Inherited from **DevExpress.Core.ObservableCollectionCore<T>**) |
| | Clear | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | Contains | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | CopyTo | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | EndUpdate | (Inherited from **DevExpress.Core.ObservableCollectionCore<T>**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | Equals | (Inherited from **System.Object**) |
| S | Equals | (Inherited from **System.Object**) |
| | GetCachedIndex | (Inherited from **DevExpress.Core.ObservableCollectionCore<T>**) |
| | GetEnumerator | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetInfoByFieldName | For internal use. |
| | GetType | (Inherited from **System.Object**) |
| | IndexOf | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | Insert | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | Move | (Inherited from **System.Collections.ObjectModel.ObservableCollection<T>**) |
| | NeedsRebuild | (Inherited from **DevExpress.Core.ObservableCollectionCore<T>**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | Remove | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | RemoveAt | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | RemoveSafe | (Inherited from **DevExpress.Core.ObservableCollectionCore<T>**) |
| | ToString | (Inherited from **System.Object**) |

**Public Events**

| | Name | Description |
|---|---|---|
| | CollectionChanged | (Inherited from **System.Collections.ObjectModel.ObservableCollection<T>**) |

**See Also**
FormatConditionCollection Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# FormatConditionCollection Constructor

Initializes a new instance of the FormatConditionCollection class.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**
**C#**

```
public FormatConditionCollection(
   IFormatConditionCollectionOwner owner
)
```

**Parameters**
*owner*
An object implementing the DevExpress.Mobile.DataGrid.Internal.IFormatConditionCollectionOwner interface.

**See Also**
FormatConditionCollection Class
FormatConditionCollection Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# FormatConditionCollection Properties

A collection of format conditions.

The following tables list the members exposed by the FormatConditionCollection type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | FormatConditionCollection | Initializes a new instance of the FormatConditionCollection class. |

### Public Properties

| | Name | Description |
|---|---|---|
| | Count | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | IsLockUpdate | (Inherited from **DevExpress.Core.ObservableCollectionCore<T>**) |
| | Item | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | Owner | Gets the format condition collection's owner. |

### Public Methods

| | Name | Description |
|---|---|---|
| | Add | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | Assign | (Inherited from **DevExpress.Core.ObservableCollectionCore<T>**) |
| | BeginUpdate | (Inherited from **DevExpress.Core.ObservableCollectionCore<T>**) |
| | Clear | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | Contains | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | CopyTo | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | EndUpdate | (Inherited from **DevExpress.Core.ObservableCollectionCore<T>**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | Equals | (Inherited from **System.Object**) |
| S | Equals | (Inherited from **System.Object**) |
| | GetCachedIndex | (Inherited from **DevExpress.Core.ObservableCollectionCore<T>**) |
| | GetEnumerator | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetInfoByFieldName | For internal use. |
| | GetType | (Inherited from **System.Object**) |
| | IndexOf | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | Insert | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | Move | (Inherited from **System.Collections.ObjectModel.ObservableCollection<T>**) |
| | NeedsRebuild | (Inherited from **DevExpress.Core.ObservableCollectionCore<T>**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | Remove | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | RemoveAt | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | RemoveSafe | (Inherited from **DevExpress.Core.ObservableCollectionCore<T>**) |
| | ToString | (Inherited from **System.Object**) |

**Public Events**

| | Name | Description |
|---|---|---|
| | CollectionChanged | (Inherited from **System.Collections.ObjectModel.ObservableCollection<T>**) |

**See Also**
FormatConditionCollection Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# Owner Property

Gets the format condition collection's owner.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public IFormatConditionCollectionOwner Owner { get; }
```

### Property Value

An object implementing the DevExpress.Mobile.DataGrid.Internal.IFormatConditionCollectionOwner interface.

### See Also

FormatConditionCollection Class
FormatConditionCollection Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# FormatConditionCollection Methods

A collection of format conditions.

The following tables list the members exposed by the FormatConditionCollection type.

**Public Methods**

| | Name | Description |
|---|---|---|
| | Add | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | Assign | (Inherited from **DevExpress.Core.ObservableCollectionCore<T>**) |
| | BeginUpdate | (Inherited from **DevExpress.Core.ObservableCollectionCore<T>**) |
| | Clear | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | Contains | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | CopyTo | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | EndUpdate | (Inherited from **DevExpress.Core.ObservableCollectionCore<T>**) |
| | Equals | (Inherited from **System.Object**) |
| S | Equals | (Inherited from **System.Object**) |
| | GetCachedIndex | (Inherited from **DevExpress.Core.ObservableCollectionCore<T>**) |
| | GetEnumerator | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetInfoByFieldName | For internal use. |
| | GetType | (Inherited from **System.Object**) |
| | IndexOf | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | Insert | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | Move | (Inherited from **System.Collections.ObjectModel.ObservableCollection<T>**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | NeedsRebuild | (Inherited from **DevExpress.Core.ObservableCollectionCore<T>**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | Remove | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | RemoveAt | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | RemoveSafe | (Inherited from **DevExpress.Core.ObservableCollectionCore<T>**) |
| | ToString | (Inherited from **System.Object**) |

**See Also**
FormatConditionCollection Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# GetInfoByFieldName Method

For internal use.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public IList<FormatConditionBaseInfo> GetInfoByFieldName(
   string fieldName
)
```

### Parameters

*fieldName*
A string that specifies the column field name.

### Return Value

The **IList<FormatConditionBaseInfo>** object.

### See Also

FormatConditionCollection Class
FormatConditionCollection Members
DevExpress.Mobile.DataGrid Namespace
GridColumn.FieldName

© 2018 Developer Express Inc.

# GridColumn Class

An abstract class that serves as the base class for grid columns.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public abstract class GridColumn : BindableObject, ILayoutCalculatorItem,
  IFieldValueVisitor
```

### Remarks

The **GridColumn** class implements basic grid column functionality. Since this class is abstract, you cannot create an instance of it directly. Use one of its descendants instead.

To learn more on grid columns, see the following documents.

- Columns Overview
- Create Columns and Bind Them to Data Fields
- Unbound Columns
- How to: Create Grid Columns for Data of Different Types

### Inheritance Hierarchy

System.Object
  Xamarin.Forms.BindableObject
    **GridColumn**
      Derived classes

### See Also

GridColumn Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# GridColumn Members

An abstract class that serves as the base class for grid columns.

The following tables list the members exposed by the GridColumn type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | GridColumn | Initializes a new instance of the GridColumn class. |

### Public Properties

| | Name | Description |
|---|---|---|
| | ActualCaption | Gets the column caption as it is displayed in a column header. |
| | ActualWidth | Gets the column width. This is a bindable property. |
| | AllowAutoFilter | Gets or sets whether an end-user can filter data by the column's values using the auto-filter panel. This is a bindable property. |
| | AllowGroup | Gets or sets whether end-users are allowed to group data by the current column. This is a bindable property. |
| | AllowSort | Gets or sets whether end-users are allowed to sort data by the column's values. This is a bindable property. |
| | AutoFilterCondition | Gets or sets the comparison operator used to create filter conditions for the current column via the auto-filter panel. This is a bindable property. |
| | AutoFilterValue | Gets or sets the value in the auto-filter panel for the column. This is a bindable property. |
| | BindingContext | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Caption | Gets or sets the column's display caption. This is a bindable property. |
| | ColumnFilterMode | Gets or sets how column values are filtered via the auto-filter panel. This is a bindable property. |
| | ContentAlignment | Gets or sets the content alignment within column cells. This is a bindable property. |
| | DisplayFormat | Gets or sets formatting for values in the column. This is a bindable property. |
| | FieldName | Gets or sets the column's unique field name. For bound columns, this property refers to a field in the |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | | underlying data source. For unbound columns, this property serves as a column's identifier.<br><br>This is a bindable property. |
| | FixedStyle | Gets or sets whether a column is fixed. This is a bindable property. |
| | GroupInterval | Gets or sets how data rows are grouped when grouping by the current column is applied. This is a bindable property. |
| | HeaderTemplate | Gets or sets a template used to define the column headers visual representation. |
| | ImmediateUpdateAutoFilter | Gets or sets whether the column's filter condition is updated as soon as an end-user modifies the text within the auto filter panel's cell. This is a bindable property. |
| | IsGrouped | Gets or sets whether the grid's data is grouped by the current column. This is a bindable property. |
| | IsReadOnly | Gets or sets whether end-users are allowed to change cell values in the column. This is a bindable property. |
| | IsUnbound | Gets whether the column is unbound. |
| | IsVisible | Gets or sets whether the column is visible in the grid. This is a bindable property. |
| | MinWidth | Gets or sets the minimum column width allowed. This is a bindable property. |
| | SerializationTypeName | Gets the string containing the column type name. |
| | SortIndex | Gets or sets the column's position among sorted columns. This is a bindable property. |
| | SortMode | Specifies how column values are sorted and grouped. This is a bindable property. |
| | SortOrder | Gets or sets the column's sort order. This is a bindable property. |
| | UnboundExpression | Gets or sets an expression used to evaluate values for the current Unbound Columns. This is a bindable property. |
| | UnboundType | Gets or sets the data type and binding mode of the column. This is a bindable property. |
| | Width | Gets or sets the column width. This is a bindable property. |

© 2018 Developer Express Inc.

### Public Methods

| | Name | Description |
|---|---|---|
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ToString | (Inherited from **System.Object**) |

### Public Fields

| | Name | Description |
|---|---|---|
| S | ActualWidthProperty | Identifies the ActualWidth bindable property. |
| S | AllowAutoFilterProperty | Identifies the AllowAutoFilter bindable property. |
| S | AllowGroupProperty | Identifies the AllowGroup bindable property. |
| S | AllowSortProperty | Identifies the AllowSort bindable property. |
| S | AutoFilterConditionProperty | Identifies the AutoFilterCondition bindable property. |
| S | AutoFilterValueProperty | Identifies the AutoFilterValue bindable property. |
| S | BindingContextProperty | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | CaptionProperty | Identifies the Caption bindable property. |
| S | ColumnFilterModeProperty | Identifies the ColumnFilterMode bindable property. |
| S | ContentAlignmentProperty | Identifies the ContentAlignment bindable property. |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | DisplayFormatProperty | Identifies the DisplayFormat bindable property. |
| | FieldNameProperty | Identifies the FieldName bindable property. |
| | FixedStyleProperty | Identifies the FixedStyle bindable property. |
| | GroupIntervalProperty | Identifies the GroupInterval bindable property. |
| | HeaderTemplateProperty | Identifies the HeaderTemplate bindable property. |
| | ImmediateUpdateAutoFilterProperty | Identifies the ImmediateUpdateAutoFilter bindable property. |
| | IsGroupedProperty | Identifies the IsGrouped bindable property. |
| | IsReadOnlyProperty | Identifies the IsReadOnly bindable property. |
| | IsVisibleProperty | Identifies the IsVisible bindable property. |
| | MinWidthProperty | Identifies the MinWidth bindable property. |
| | SortIndexProperty | Identifies the SortIndex bindable property. |
| | SortModeProperty | Identifies the SortMode bindable property. |
| | SortOrderProperty | Identifies the SortOrder bindable property. |
| | UnboundExpressionProperty | Identifies the UnboundExpression bindable property. |
| | UnboundTypeProperty | Identifies the UnboundType bindable property. |
| | WidthProperty | Identifies the Width bindable property. |

**Public Events**

| | Name | Description |
|---|---|---|
| | BindingContextChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanging | (Inherited from **Xamarin.Forms.BindableObject**) |

**See Also**
GridColumn Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# GridColumn Constructor

Initializes a new instance of the GridColumn class.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**
**C#**

```
public GridColumn()
```

**See Also**
GridColumn Class
GridColumn Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# GridColumn Properties

An abstract class that serves as the base class for grid columns.

The following tables list the members exposed by the GridColumn type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | GridColumn | Initializes a new instance of the GridColumn class. |

### Public Properties

| | Name | Description |
|---|---|---|
| | ActualCaption | Gets the column caption as it is displayed in a column header. |
| | ActualWidth | Gets the column width. This is a bindable property. |
| | AllowAutoFilter | Gets or sets whether an end-user can filter data by the column's values using the auto-filter panel. This is a bindable property. |
| | AllowGroup | Gets or sets whether end-users are allowed to group data by the current column. This is a bindable property. |
| | AllowSort | Gets or sets whether end-users are allowed to sort data by the column's values. This is a bindable property. |
| | AutoFilterCondition | Gets or sets the comparison operator used to create filter conditions for the current column via the auto-filter panel. This is a bindable property. |
| | AutoFilterValue | Gets or sets the value in the auto-filter panel for the column. This is a bindable property. |
| | BindingContext | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Caption | Gets or sets the column's display caption. This is a bindable property. |
| | ColumnFilterMode | Gets or sets how column values are filtered via the auto-filter panel. This is a bindable property. |
| | ContentAlignment | Gets or sets the content alignment within column cells. This is a bindable property. |
| | DisplayFormat | Gets or sets formatting for values in the column. This is a bindable property. |
| | FieldName | Gets or sets the column's unique field name. For bound columns, this property refers to a field in the |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | | underlying data source. For unbound columns, this property serves as a column's identifier.<br><br>This is a bindable property. |
| | FixedStyle | Gets or sets whether a column is fixed. This is a bindable property. |
| | GroupInterval | Gets or sets how data rows are grouped when grouping by the current column is applied. This is a bindable property. |
| | HeaderTemplate | Gets or sets a template used to define the column headers visual representation. |
| | ImmediateUpdateAutoFilter | Gets or sets whether the column's filter condition is updated as soon as an end-user modifies the text within the auto filter panel's cell. This is a bindable property. |
| | IsGrouped | Gets or sets whether the grid's data is grouped by the current column. This is a bindable property. |
| | IsReadOnly | Gets or sets whether end-users are allowed to change cell values in the column. This is a bindable property. |
| | IsUnbound | Gets whether the column is unbound. |
| | IsVisible | Gets or sets whether the column is visible in the grid. This is a bindable property. |
| | MinWidth | Gets or sets the minimum column width allowed. This is a bindable property. |
| | SerializationTypeName | Gets the string containing the column type name. |
| | SortIndex | Gets or sets the column's position among sorted columns. This is a bindable property. |
| | SortMode | Specifies how column values are sorted and grouped. This is a bindable property. |
| | SortOrder | Gets or sets the column's sort order. This is a bindable property. |
| | UnboundExpression | Gets or sets an expression used to evaluate values for the current Unbound Columns. This is a bindable property. |
| | UnboundType | Gets or sets the data type and binding mode of the column. This is a bindable property. |
| | Width | Gets or sets the column width. This is a bindable property. |

© 2018 Developer Express Inc.

### Public Methods

| | Name | Description |
|---|---|---|
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ToString | (Inherited from **System.Object**) |

### Public Fields

| | Name | Description |
|---|---|---|
| S | ActualWidthProperty | Identifies the ActualWidth bindable property. |
| S | AllowAutoFilterProperty | Identifies the AllowAutoFilter bindable property. |
| S | AllowGroupProperty | Identifies the AllowGroup bindable property. |
| S | AllowSortProperty | Identifies the AllowSort bindable property. |
| S | AutoFilterConditionProperty | Identifies the AutoFilterCondition bindable property. |
| S | AutoFilterValueProperty | Identifies the AutoFilterValue bindable property. |
| S | BindingContextProperty | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | CaptionProperty | Identifies the Caption bindable property. |
| S | ColumnFilterModeProperty | Identifies the ColumnFilterMode bindable property. |
| S | ContentAlignmentProperty | Identifies the ContentAlignment bindable property. |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | DisplayFormatProperty | Identifies the DisplayFormat bindable property. |
| | FieldNameProperty | Identifies the FieldName bindable property. |
| | FixedStyleProperty | Identifies the FixedStyle bindable property. |
| | GroupIntervalProperty | Identifies the GroupInterval bindable property. |
| | HeaderTemplateProperty | Identifies the HeaderTemplate bindable property. |
| | ImmediateUpdateAutoFilterProperty | Identifies the ImmediateUpdateAutoFilter bindable property. |
| | IsGroupedProperty | Identifies the IsGrouped bindable property. |
| | IsReadOnlyProperty | Identifies the IsReadOnly bindable property. |
| | IsVisibleProperty | Identifies the IsVisible bindable property. |
| | MinWidthProperty | Identifies the MinWidth bindable property. |
| | SortIndexProperty | Identifies the SortIndex bindable property. |
| | SortModeProperty | Identifies the SortMode bindable property. |
| | SortOrderProperty | Identifies the SortOrder bindable property. |
| | UnboundExpressionProperty | Identifies the UnboundExpression bindable property. |
| | UnboundTypeProperty | Identifies the UnboundType bindable property. |
| | WidthProperty | Identifies the Width bindable property. |

**Public Events**

| | Name | Description |
|---|---|---|
| | BindingContextChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanging | (Inherited from **Xamarin.Forms.BindableObject**) |

**See Also**
GridColumn Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# ActualCaption Property

Gets the column caption as it is displayed in a column header.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public string ActualCaption { get; }
```

### Property Value

A System.String value specifying the column's actual caption.

### Remarks

In the grid, column captions are displayed in column headers.

By default, a column caption is generated based on the column's field name (FieldName) - SPACE characters are added between parts of the field name starting with uppercase letters. For example, if the field name is "CustomerName", the display caption will be "Customer Name". To obtain a column caption as it is displayed in a column header, use the **ActualCaption** property.

To override the default display caption of a column, assign the Caption property.

### See Also

GridColumn Class
GridColumn Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# ActualWidth Property

Gets the column width. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public Double ActualWidth { get; }
```

### Property Value

A System.Double value that specifies the column width in density-independent pixels.

### Remarks

In the grid, column widths are adjusted automatically, so that the total width of all columns matches the grid width. Changing the width of one column automatically changes the widths of other columns. To obtain the actual width of columns, use their **ActualWidth** property.

To change the column width, use the Width property. To specify the minimum width of a column, use the MinWidth property.

### See Also

GridColumn Class
GridColumn Members
DevExpress.Mobile.DataGrid Namespace
Columns

© 2018 Developer Express Inc.

# AllowAutoFilter Property

Gets or sets whether an end-user can filter data by the column's values using the auto-filter panel. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public bool AllowAutoFilter { get; set; }
```

### Property Value

**true**, to allow an end-user to filter data by the column's values using the the auto-filter panel; otherwise, **false**.

### Remarks

End-users can filter data on the fly using the auto-filter panel displayed at the top of the grid. When an end-user types the text within this panel, a filter condition is automatically created based upon the values entered and applied to the column.

The **AllowAutoFilter** property allows you to prevent an end-user from filtering data by the current column's values using the auto-filter panel. If this property is set to **false**, the auto-filter panel's cell that corresponds to the current column is disabled.

By default, the auto-filter panel is hidden. To show it, set the GridControl.AutoFilterPanelVisibility property to **true**.

### See Also

GridColumn Class
GridColumn Members
DevExpress.Mobile.DataGrid Namespace
End-User Restrictions

© 2018 Developer Express Inc.

# AllowGroup Property

Gets or sets whether end-users are allowed to group data by the current column. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public DefaultBoolean AllowGroup { get; set; }
```

### Property Value

A DefaultBoolean enumeration value that specifies whether end-users can group data by the current column.

### Remarks

End-users can group data or remove grouping applied in the grid by tapping the corresponding item in a menu that appears when they touch and hold a column. If the grid's GridControl.AllowGroup property is set to **true**, menu items related to grouping are available. Otherwise, end-users cannot group data.

Individual columns provide the **AllowGroup** property that you can set to 'True' or 'False' to override the default behavior specified by the grid. This can be useful when it is required to prevent end-users from grouping data by values of specific columns.

If the **AllowGroup** property is set to 'Default', the capability to group data is controlled by the grid's GridControl.AllowGroup property.

### See Also

GridColumn Class
GridColumn Members
DevExpress.Mobile.DataGrid Namespace
Lesson 4 - Group Data
End-User Restrictions

© 2018 Developer Express Inc.

# AllowSort Property

Gets or sets whether end-users are allowed to sort data by the column's values. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public DefaultBoolean AllowSort { get; set; }
```

### Property Value

A DefaultBoolean enumeration value that specifies whether end-users can sort data by the column's values.

### Remarks

End-users can sort data by tapping a column header or tapping **Sort Ascending** or **Sort Descending** in a menu that appears when they touch and hold a column cell or header. If the grid's GridControl.AllowSort property is set to **true**, both these methods are available. Otherwise, end-users cannot apply data sorting.

Individual columns provide the **AllowSort** property that you can set to 'True' or 'False' to override the default behavior specified by the grid. This can be useful when it is required to prevent end-users from sorting data by values of specific columns.

If the **AllowSort** property is set to 'Default', the capability to sort data is controlled by the grid's GridControl.AllowSort property.

### See Also

GridColumn Class
GridColumn Members
DevExpress.Mobile.DataGrid Namespace
Lesson 6 - Sort Data
End-User Restrictions

© 2018 Developer Express Inc.

# AutoFilterCondition Property

Gets or sets the comparison operator used to create filter conditions for the current column via the auto-filter panel. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public AutoFilterCondition AutoFilterCondition { get; set; }
```

### Property Value

An AutoFilterCondition enumeration value that specifies the type of the comparison operator.

### Remarks

When an end-user types text within the auto-filter panel's cell, a filter condition is created based upon the text entered. This filter is then applied to the grid. The comparison operator used in this filter condition is determined by the column's **AutoFilterCondition** property.

### See Also

GridColumn Class
GridColumn Members
DevExpress.Mobile.DataGrid Namespace
GridControl.AutoFilterPanelVisibility

© 2018 Developer Express Inc.

# AutoFilterValue Property

Gets or sets the value in the auto-filter panel for the column. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public object AutoFilterValue { get; set; }
```

### Property Value

An object that specifies the value in the auto-filter panel.

### Remarks

The auto-filter panel allows end-users to filter data on the fly by typing text directly into the panel. To do this in code, use the **AutoFilterValue** property. The type of the comparison operator used to create filter conditions for the current column is specified by the AutoFilterCondition property.

### See Also

GridColumn Class
GridColumn Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# Caption Property

Gets or sets the column's display caption. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public string Caption { get; set; }
```

### Property Value

A System.String value specifying the column's display caption.

### Remarks

In the grid, column captions are displayed in column headers.

By default, the **Caption** property is set to an empty string. In this case, a friendly caption is generated for a column based on its FieldName property. Friendly captions are generated by adding SPACE characters between parts of the field name starting with uppercase letters. For example, if the field name is "CustomerName", the display caption will be "Customer Name". To obtain a column caption as it is displayed in a column header, use the ActualCaption property.

To override the default display caption, assign it to the **Caption** property.

To control the visibility of column headers in the GridControl, use the GridControl.ColumnHeadersVisibility property.

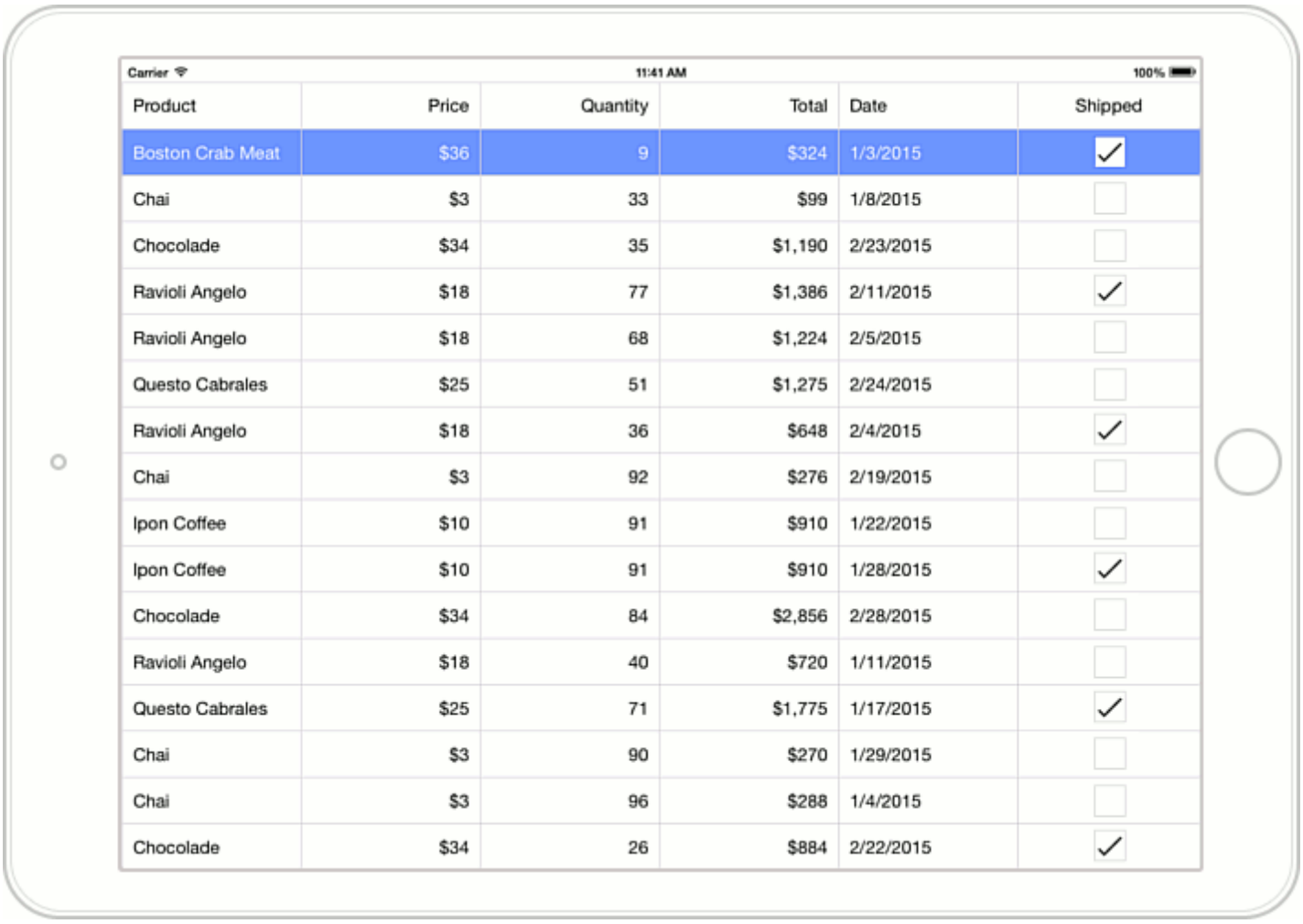
### Example

| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T178764. | |

This example shows how to create and customize grid columns for displaying and editing data of different types (text, numbers, dates and Boolean values). The specified collection contains columns bound to the data source fields (*Product.Name*, *Product.UnitPrice*, *Quantity*, *Date* and *Shipped*) and one unbound column (*Total*) displaying data values calculated according to a formula based on the values of other columns.

The image below illustrates the result.

© 2018 Developer Express Inc.




| Product | Price | Quantity | Total | Date | Shipped |
|---|---|---|---|---|---|
| Boston Crab Meat | $36 | 9 | $324 | 1/3/2015 | ✓ |
| Chai | $3 | 33 | $99 | 1/8/2015 | |
| Chocolade | $34 | 35 | $1,190 | 2/23/2015 | |
| Ravioli Angelo | $18 | 77 | $1,386 | 2/11/2015 | ✓ |
| Ravioli Angelo | $18 | 68 | $1,224 | 2/5/2015 | |
| Questo Cabrales | $25 | 51 | $1,275 | 2/24/2015 | |
| Ravioli Angelo | $18 | 36 | $648 | 2/4/2015 | ✓ |
| Chai | $3 | 92 | $276 | 2/19/2015 | |
| Ipon Coffee | $10 | 91 | $910 | 1/22/2015 | |
| Ipon Coffee | $10 | 91 | $910 | 1/28/2015 | ✓ |
| Chocolade | $34 | 84 | $2,856 | 2/28/2015 | |
| Ravioli Angelo | $18 | 40 | $720 | 1/11/2015 | |
| Questo Cabrales | $25 | 71 | $1,775 | 1/17/2015 | ✓ |
| Chai | $3 | 90 | $270 | 1/29/2015 | |
| Chai | $3 | 96 | $288 | 1/4/2015 | |
| Chocolade | $34 | 26 | $884 | 2/22/2015 | ✓ |

© 2018 Developer Express Inc.

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl.Columns>
        <dxGrid:TextColumn FieldName="Product.Name" Caption = "Product" Width = "170"
                           SortOrder = "Descending" SortIndex = "0"/>
        <dxGrid:NumberColumn FieldName="Product.UnitPrice" Caption = "Price" DisplayFormat="C0"/>
        <dxGrid:NumberColumn FieldName="Quantity"
                             SortOrder = "Ascending" SortIndex = "1"/>
        <dxGrid:NumberColumn FieldName="Total"
                             UnboundType="Integer" UnboundExpression="[Quantity] * [Product.UnitPrice]"
                             DisplayFormat="C0" IsReadOnly="True"/>
        <dxGrid:DateColumn FieldName="Date" DisplayFormat="d"
                           IsGrouped = "true" GroupInterval = "Date"/>
        <dxGrid:SwitchColumn FieldName="Shipped" AllowSort = "False" AllowAutoFilter="false"/>
    </dxGrid:GridControl.Columns>
```

**See Also**

GridColumn Class
GridColumn Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# ColumnFilterMode Property

Gets or sets how column values are filtered via the auto-filter panel. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public ColumnFilterMode ColumnFilterMode { get; set; }
```

### Property Value

A ColumnFilterMode enumeration value that specifies how the column values are filtered.

### Remarks

The grid allows its data to be filtered by column values (edit values) or display text via the auto-filter panel. Filtering by edit values is suitable under most circumstances. Filtering by displayed values can be useful when the column's data is formatted in a specific manner.

If the **ColumnFilterMode** property is set to ColumnFilterMode.Value, an end-user should enter text without formatting characters into the auto-filter panel.

If this property is set to ColumnFilterMode.DisplayText, an end-user should enter text which matches the column's formatted values completely (if necessary, formatting characters should be inserted).

Filter by Values

| Product | Price | Quantity | Total |
|---|---|---|---|
| | | | 22 |
| Questo Cabrales | $25 | 88 | $2,200 |
| Chocolade | $34 | 65 | $2,210 |
| Tofu | $50 | 44 | $2,200 |
| Chocolade | $34 | 67 | $2,278 |

Filter by Display Text

| Product | Price | Quantity | Total |
|---|---|---|---|
| | | | $2,1 |
| Boston Crab Meat | $36 | 61 | $2,196 |
| Chocolade | $34 | 63 | $2,142 |
| Boston Crab Meat | $36 | 59 | $2,124 |

### See Also

GridColumn Class
GridColumn Members
DevExpress.Mobile.DataGrid Namespace
GridControl.AutoFilterPanelVisibility

© 2018 Developer Express Inc.

# ContentAlignment Property

Gets or sets the content alignment within column cells. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public ColumnContentAlignment ContentAlignment { get; set; }
```

### Property Value

A ColumnContentAlignment enumeration value.

### Remarks

The default alignment is different for different column types. For example, TextColumn aligns its values at the start by default, NumberColumn - at the end, SwitchColumn - at the center. To change the default alignment, use the column's **ContentAlignment** property.

The specified alignment is applied to both the data cells content and the column header.

The specified **ContentAlignment** property value is ignored for a column by which grid data is grouped: a group value is always aligned at the start and the group summary value (if any) is aligned at the end in a group row.

### See Also

GridColumn Class
GridColumn Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# DisplayFormat Property

Gets or sets formatting for values in the column. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public string DisplayFormat { get; set; }
```

### Property Value

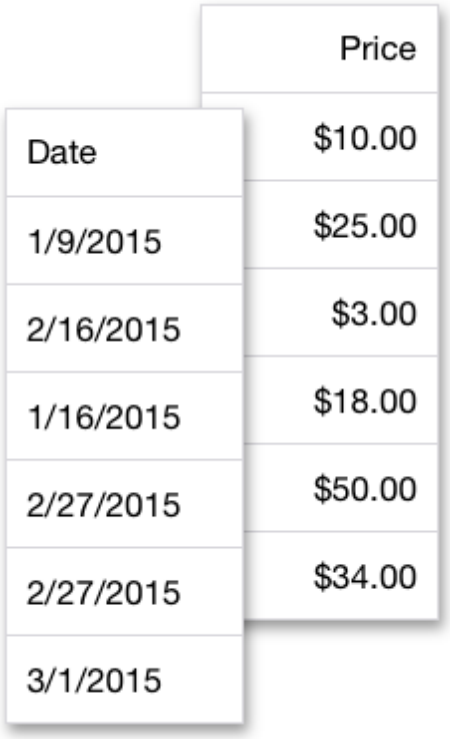
A string that specifies the format pattern.

### Remarks

Use the **DisplayFormat** property to specify the format for numeric and date/time values displayed in the column. To compose format patterns, you can use standard or custom format specifiers (they are listed and described in the **Standard Numeric Format Strings**, **Custom Numeric Format Strings**, **Standard Date and Time Format Strings** and **Custom Date and Time Format Strings** topics in MSDN).

### Example

This example shows how to format column values using the standard format strings. Values in the *Price* column are formatted as a currency amount, values in the *Date* column are formatted using the Short Date Pattern.




**Xaml**

```
<dxGrid:GridControl.Columns>
    <!-- ... -->
    <dxGrid:NumberColumn FieldName="Price" DisplayFormat="c2"/>
    <dxGrid:DateColumn FieldName="Date" DisplayFormat="d"/>
    <!-- ... -->
</dxGrid:GridControl.Columns>
```

### See Also

GridColumn Class

© 2018 Developer Express Inc.

GridColumn Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# FieldName Property

Gets or sets the column's unique field name. For bound columns, this property refers to a field in the underlying data source. For unbound columns, this property serves as a column's identifier.

This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public string FieldName { get; set; }
```

### Property Value

A System.String value that specifies the column's unique field name.

### Remarks

Bound columns retrieve their data from a data source. For these columns, the **FieldName** property identifies the data source field linked to a grid column. The grid column displays data from this field, and it posts data to this field after cell modification.

Unbound columns are not linked to fields in the underlying data source. However, these columns must also have their **FieldName** properties initialized. The field names of unbound columns must not match any field in the underlying data source.

Field names of bound and unbound grid columns must be unique.

To set a caption to be displayed in a column header, use the column's Caption property.

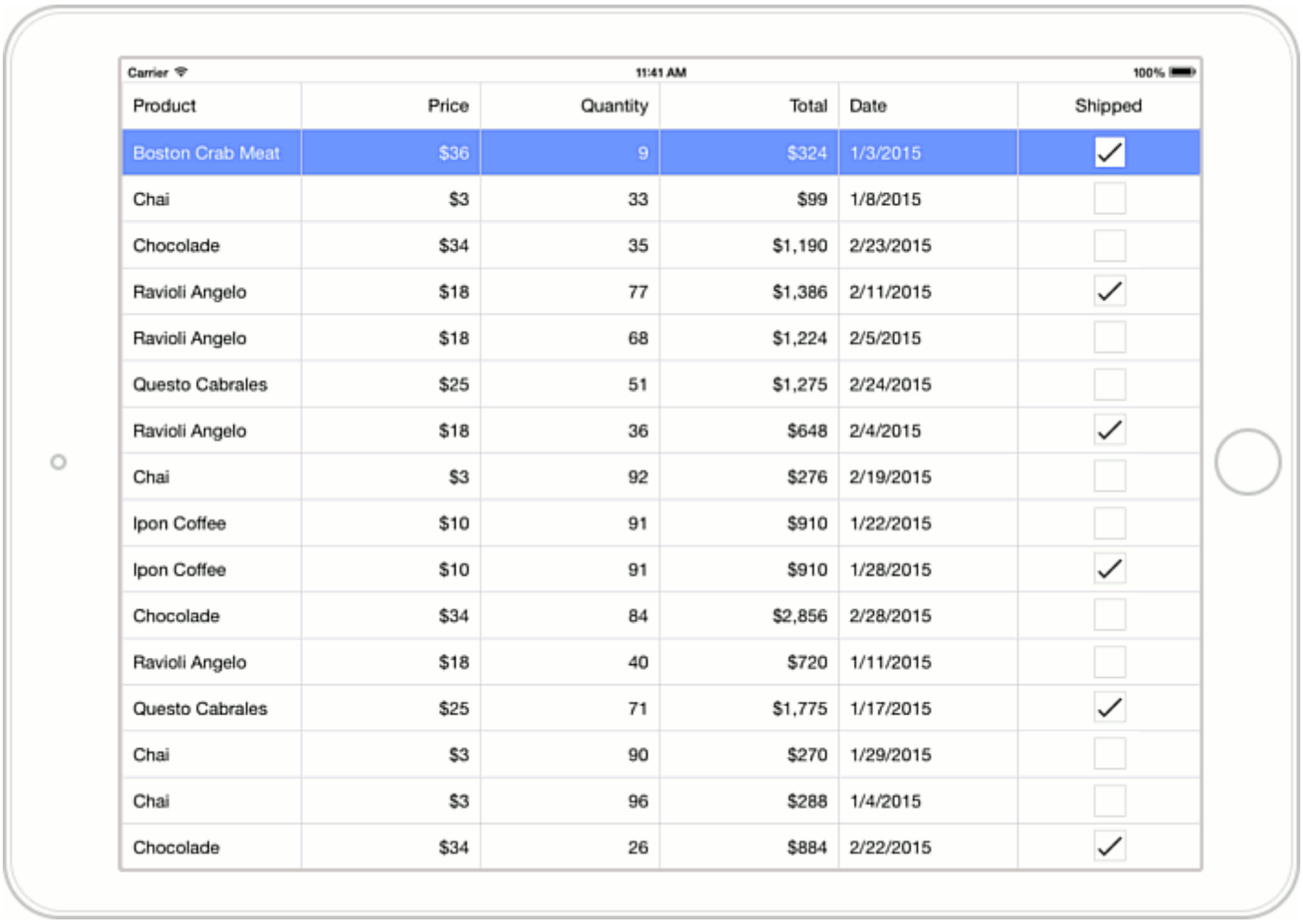
### Example

| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T178764. | |

This example shows how to create and customize grid columns for displaying and editing data of different types (text, numbers, dates and Boolean values). The specified collection contains columns bound to the data source fields (*Product.Name*, *Product.UnitPrice*, *Quantity*, *Date* and *Shipped*) and one unbound column (*Total*) displaying data values calculated according to a formula based on the values of other columns.

The image below illustrates the result.

© 2018 Developer Express Inc.




| Product | Price | Quantity | Total | Date | Shipped |
|---|---|---|---|---|---|
| Boston Crab Meat | $36 | 9 | $324 | 1/3/2015 | ✓ |
| Chai | $3 | 33 | $99 | 1/8/2015 | |
| Chocolade | $34 | 35 | $1,190 | 2/23/2015 | |
| Ravioli Angelo | $18 | 77 | $1,386 | 2/11/2015 | ✓ |
| Ravioli Angelo | $18 | 68 | $1,224 | 2/5/2015 | |
| Questo Cabrales | $25 | 51 | $1,275 | 2/24/2015 | |
| Ravioli Angelo | $18 | 36 | $648 | 2/4/2015 | ✓ |
| Chai | $3 | 92 | $276 | 2/19/2015 | |
| Ipon Coffee | $10 | 91 | $910 | 1/22/2015 | |
| Ipon Coffee | $10 | 91 | $910 | 1/28/2015 | ✓ |
| Chocolade | $34 | 84 | $2,856 | 2/28/2015 | |
| Ravioli Angelo | $18 | 40 | $720 | 1/11/2015 | |
| Questo Cabrales | $25 | 71 | $1,775 | 1/17/2015 | ✓ |
| Chai | $3 | 90 | $270 | 1/29/2015 | |
| Chai | $3 | 96 | $288 | 1/4/2015 | |
| Chocolade | $34 | 26 | $884 | 2/22/2015 | ✓ |

© 2018 Developer Express Inc.

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl.Columns>
        <dxGrid:TextColumn FieldName="Product.Name" Caption = "Product" Width = "170"
                           SortOrder = "Descending" SortIndex = "0"/>
        <dxGrid:NumberColumn FieldName="Product.UnitPrice" Caption = "Price" DisplayFormat="C0"/>
        <dxGrid:NumberColumn FieldName="Quantity"
                             SortOrder = "Ascending" SortIndex = "1"/>
        <dxGrid:NumberColumn FieldName="Total"
                             UnboundType="Integer" UnboundExpression="[Quantity] * [Product.UnitPrice]"
                             DisplayFormat="C0" IsReadOnly="True"/>
        <dxGrid:DateColumn FieldName="Date" DisplayFormat="d"
                           IsGrouped = "true" GroupInterval = "Date"/>
        <dxGrid:SwitchColumn FieldName="Shipped" AllowSort = "False" AllowAutoFilter="false"/>
    </dxGrid:GridControl.Columns>
```

**See Also**

GridColumn Class
GridColumn Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# FixedStyle Property

Gets or sets whether a column is fixed. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public FixedStyle FixedStyle { get; set; }
```

### Property Value

A FixedStyle enumeration value that specifies whether a column is anchored, and how.

### See Also

GridColumn Class
GridColumn Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# GroupInterval Property

Gets or sets how data rows are grouped when grouping by the current column is applied. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public ColumnGroupInterval GroupInterval { get; set; }
```

### Property Value

A ColumnGroupInterval enumeration value that specifies how data rows are grouped.

### Remarks

When data is grouped by a specific column, data rows are combined into groups. By default, each group contains rows which have the same value in the field (column) for which grouping has been applied. Use the **GroupInterval** property to change the logic used to decide which rows should be grouped together. For instance, for a column that displays date/time information, you can set its **GroupInterval** property to ColumnGroupInterval.DateRange. In this case, when grouping is applied against this column, the data rows will be combined into groups according to their date value compared with the current system date ("Today", "Yesterday", etc.).

### See Also

GridColumn Class
GridColumn Members
DevExpress.Mobile.DataGrid Namespace
IsGrouped

© 2018 Developer Express Inc.

# HeaderTemplate Property

Gets or sets a template used to define the column headers visual representation.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public DataTemplate HeaderTemplate { get; set; }
```

### Property Value

A Xamarin.Forms.DataTemplate object.

### Remarks

GridControl provides various ways to customize the appearance of its elements. For simple customizations (such as modifying colors, fonts, etc.) use appropriate customizer objects. If you need to completely modify the appearance of a particular grid element, you can define a custom template for it.

The **HeaderTemplate** property allows you to define a data template for a column header. If you need to define a custom template for column cells, create a TemplateColumn object, assign a custom template to a column's TemplateColumn.DisplayTemplate property and add this templated column to the GridControl.Columns collection.

For more information on using templates, see How to: Create Grid Columns for Data of Different Types.

### See Also

GridColumn Class
GridColumn Members
DevExpress.Mobile.DataGrid Namespace
ThemeBase.HeaderCustomizer
TemplateColumn.EditTemplate

© 2018 Developer Express Inc.

# ImmediateUpdateAutoFilter Property

Gets or sets whether the column's filter condition is updated as soon as an end-user modifies the text within the auto filter panel's cell. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public bool ImmediateUpdateAutoFilter { get; set; }
```

### Property Value

**true** to enable automatic updates; otherwise, **false**.

### Remarks

By default, when an end-user modifies the text within the auto filter panel's cell, the filter condition for this column is immediately updated. To disable automatic updates, set the column's **ImmediateUpdateAutoFilter** property to **false**. In this instance, an end-user should move focus to another cell to force the updates.

### See Also

GridColumn Class
GridColumn Members
DevExpress.Mobile.DataGrid Namespace
GridControl.AutoFilterPanelVisibility

© 2018 Developer Express Inc.

# IsGrouped Property

Gets or sets whether the grid's data is grouped by the current column. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public bool IsGrouped { get; set; }
```

### Property Value

**true**, if data grouping by the current column is applied; otherwise, **false**.

### Remarks

Another way to group data in the grid is to use the GridControl.GroupBy method. End-users can group data by a column using a menu that appears when they touch and hold this column.

To cancel grouping, use the GridControl.ClearGrouping method.

### See Also

GridColumn Class
GridColumn Members
DevExpress.Mobile.DataGrid Namespace
GroupInterval
AllowGroup
Lesson 4 - Group Data

© 2018 Developer Express Inc.

# IsReadOnly Property

Gets or sets whether end-users are allowed to change cell values in the column. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public bool IsReadOnly { get; set; }
```

### Property Value

**true**, to prevent end-users from changing cell values in the column; otherwise, **false**.

### Remarks

End-users can switch the grid to edit mode by double-tapping the required cell or choosing **Edit Cell** in the context menu that appears when they touch and hold a data cell. If the **IsReadOnly** property of a grid column is set to **true**, both these methods are disabled for this column.

To prevent end-users from editing cell values in the whole grid, set the GridControl.AllowEditRows property to **false**.

### See Also

GridColumn Class
GridColumn Members
DevExpress.Mobile.DataGrid Namespace
End-User Restrictions

© 2018 Developer Express Inc.

# IsUnbound Property

Gets whether the column is unbound.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public bool IsUnbound { get; }
```

### Property Value

**true**, if the column is unbound; otherwise, **false**.

### Remarks

Use the **IsUnbound** property to determine whether a column is not bound to any field of the data source. Unbound column values are calculated according to a formula based on the values of other columns. For more details, see Unbound Columns.

### See Also

GridColumn Class
GridColumn Members
DevExpress.Mobile.DataGrid Namespace
Lesson 2 - Create Grid Columns

© 2018 Developer Express Inc.

# IsVisible Property

Gets or sets whether the column is visible in the grid. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public bool IsVisible { get; set; }
```

### Property Value

**true**, if the column is visible; otherwise, **false**.

### Remarks

End-users can hide and show columns using the Column Chooser.

### See Also

GridColumn Class
GridColumn Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# MinWidth Property

Gets or sets the minimum column width allowed. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public Double MinWidth { get; set; }
```

### Property Value

A System.Double value specifying the minimum column width in density-independent pixels.

### Remarks

In the grid, column widths are adjusted automatically, so that the total width of all columns matches the grid width. Changing the width of one column automatically changes the widths of other columns. To set the minimum width that can be assigned to a column, use the **MinWidth** property.

### See Also

GridColumn Class
GridColumn Members
DevExpress.Mobile.DataGrid Namespace
Width
ActualWidth
Columns

© 2018 Developer Express Inc.

# SerializationTypeName Property

Gets the string containing the column type name.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public string SerializationTypeName { get; }
```

### Property Value

A System.String value which contains the type name.

### Remarks

The **SerializationTypeName** property is intended for serialization purposes only. Normally, you don't need to use it in your code.

### See Also

GridColumn Class
GridColumn Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# SortIndex Property

Gets or sets the column's position among sorted columns. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public int SortIndex { get; set; }
```

### Property Value

An integer value specifying the zero-based column's index among sorted columns. **-1** if data is not sorted by this column.

### Remarks

The GridControl allows data sorting by multiple columns if the GridControl.SortMode property is set to GridSortMode.Multiple. Set the SortOrder property for each column to be sorted and specify their **SortIndex** to set the sequence of applying sorting to these columns.

### See Also

GridColumn Class
GridColumn Members
DevExpress.Mobile.DataGrid Namespace
AllowSort
GridControl.SortBy
Lesson 6 - Sort Data

© 2018 Developer Express Inc.

# SortMode Property

Specifies how column values are sorted and grouped. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public ColumnSortMode SortMode { get; set; }
```

### Property Value

A ColumnSortMode enumeration value that specifies how column values are sorted and grouped.

### Remarks

GridControl allows its data to be sorted and grouped by either column values or the display text. Sorting and grouping by column values is suitable under most circumstances. Sorting and grouping by the displayed text can be useful when column data is formatted in a specific manner.

If the **SortMode** property is set to ColumnSortMode.Value, the column values are sorted and grouped by initial values.

If this property is set to ColumnSortMode.DisplayText, the column values are sorted and grouped by columns formatted values.

### See Also

GridColumn Class
GridColumn Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# SortOrder Property

Gets or sets the column's sort order. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public ColumnSortOrder SortOrder { get; set; }
```

### Property Value

A ColumnSortOrder enumeration value specifying the column's sort order.

### Remarks

Get the **SortOrder** property value to determine which sort order is currently applied to the column. Assign a value to this property to sort data by the current column's values.

By default, data in a grid can be sorted by one column at a time (the GridControl.SortMode property is set to GridSortMode.Single). In this case, applying sorting to a column clears sorting previously applied to another column by setting its **SortOrder** property to ColumnSortOrder.None. If sorting data by multiple columns is enabled (the GridControl.SortMode property is set to GridSortMode.Multiple), any previous sorting is not cleared when changing the **SortOrder** property value. The new sort order applies an additional sorting condition to the one previously applied. To specify the sequence of applying sorting to columns, assign the SortIndex of these columns.

Another way to sort data in the grid is to call the GridControl.SortBy method. End-users can sort data by a column using a menu that appears when they touch and hold this column, or by tapping a column header.

To clear sorting applied to grid data, call the GridControl.ClearSorting method.

### See Also

GridColumn Class
GridColumn Members
DevExpress.Mobile.DataGrid Namespace
SortIndex
GridControl.SortMode
AllowSort
Lesson 6 - Sort Data

© 2018 Developer Express Inc.

# UnboundExpression Property

Gets or sets an expression used to evaluate values for the current Unbound Columns. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public string UnboundExpression { get; set; }
```

### Property Value

A string that specifies an expression used to evaluate values for the current column.

### Remarks

The GridControl supports bound and unbound columns. Bound columns obtain their data from data fields of a grid's data source. Unbound columns are not bound to any field of the data source. These columns must be populated manually by handling the GridControl.CustomUnboundColumnData event or using the **UnboundExpression** property.

### Example

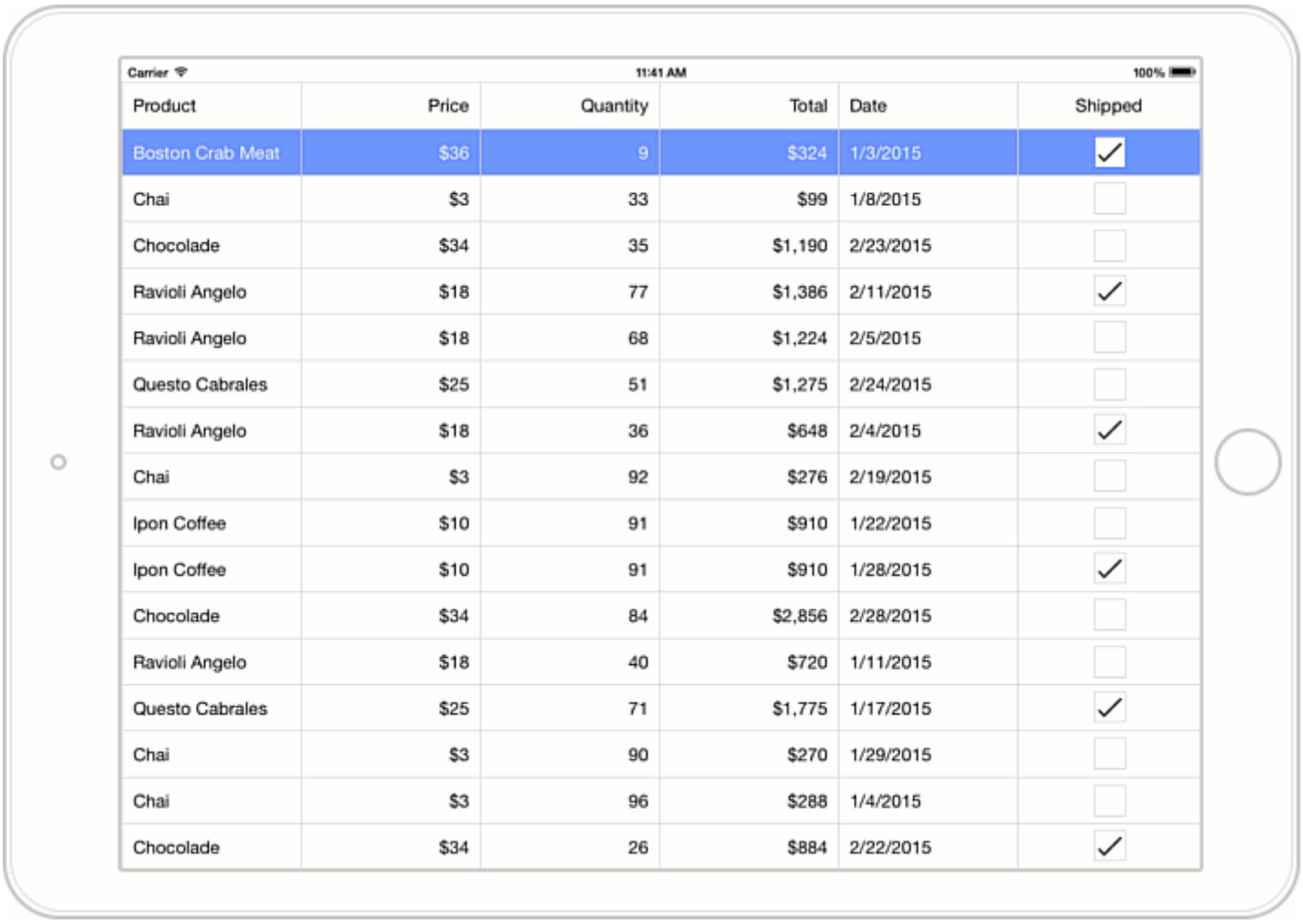
**Show Me**

A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T178764.

This example shows how to create and customize grid columns for displaying and editing data of different types (text, numbers, dates and Boolean values). The specified collection contains columns bound to the data source fields (*Product.Name*, *Product.UnitPrice*, *Quantity*, *Date* and *Shipped*) and one unbound column (*Total*) displaying data values calculated according to a formula based on the values of other columns.

The image below illustrates the result.

© 2018 Developer Express Inc.




| Product | Price | Quantity | Total | Date | Shipped |
|---|---|---|---|---|---|
| Boston Crab Meat | $36 | 9 | $324 | 1/3/2015 | ✓ |
| Chai | $3 | 33 | $99 | 1/8/2015 | |
| Chocolade | $34 | 35 | $1,190 | 2/23/2015 | |
| Ravioli Angelo | $18 | 77 | $1,386 | 2/11/2015 | ✓ |
| Ravioli Angelo | $18 | 68 | $1,224 | 2/5/2015 | |
| Questo Cabrales | $25 | 51 | $1,275 | 2/24/2015 | |
| Ravioli Angelo | $18 | 36 | $648 | 2/4/2015 | ✓ |
| Chai | $3 | 92 | $276 | 2/19/2015 | |
| Ipon Coffee | $10 | 91 | $910 | 1/22/2015 | |
| Ipon Coffee | $10 | 91 | $910 | 1/28/2015 | ✓ |
| Chocolade | $34 | 84 | $2,856 | 2/28/2015 | |
| Ravioli Angelo | $18 | 40 | $720 | 1/11/2015 | |
| Questo Cabrales | $25 | 71 | $1,775 | 1/17/2015 | ✓ |
| Chai | $3 | 90 | $270 | 1/29/2015 | |
| Chai | $3 | 96 | $288 | 1/4/2015 | |
| Chocolade | $34 | 26 | $884 | 2/22/2015 | ✓ |

© 2018 Developer Express Inc.

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl.Columns>
        <dxGrid:TextColumn FieldName="Product.Name" Caption = "Product" Width = "170"
                           SortOrder = "Descending" SortIndex = "0"/>
        <dxGrid:NumberColumn FieldName="Product.UnitPrice" Caption = "Price" DisplayFormat="C0"/>
        <dxGrid:NumberColumn FieldName="Quantity"
                             SortOrder = "Ascending" SortIndex = "1"/>
        <dxGrid:NumberColumn FieldName="Total"
                             UnboundType="Integer" UnboundExpression="[Quantity] * [Product.UnitPrice]"
                             DisplayFormat="C0" IsReadOnly="True"/>
        <dxGrid:DateColumn FieldName="Date" DisplayFormat="d"
                           IsGrouped = "true" GroupInterval = "Date"/>
        <dxGrid:SwitchColumn FieldName="Shipped" AllowSort = "False" AllowAutoFilter="false"/>
    </dxGrid:GridControl.Columns>
```

**See Also**

GridColumn Class
GridColumn Members
DevExpress.Mobile.DataGrid Namespace
Columns

© 2018 Developer Express Inc.

# UnboundType Property

Gets or sets the data type and binding mode of the column. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public UnboundColumnType UnboundType { get; set; }
```

### Property Value

An UnboundColumnType enumeration value specifying the data type and binding mode of the column.

### Remarks

See the Unbound Columns topic for detailed information on unbound columns in GridControl.

### Example

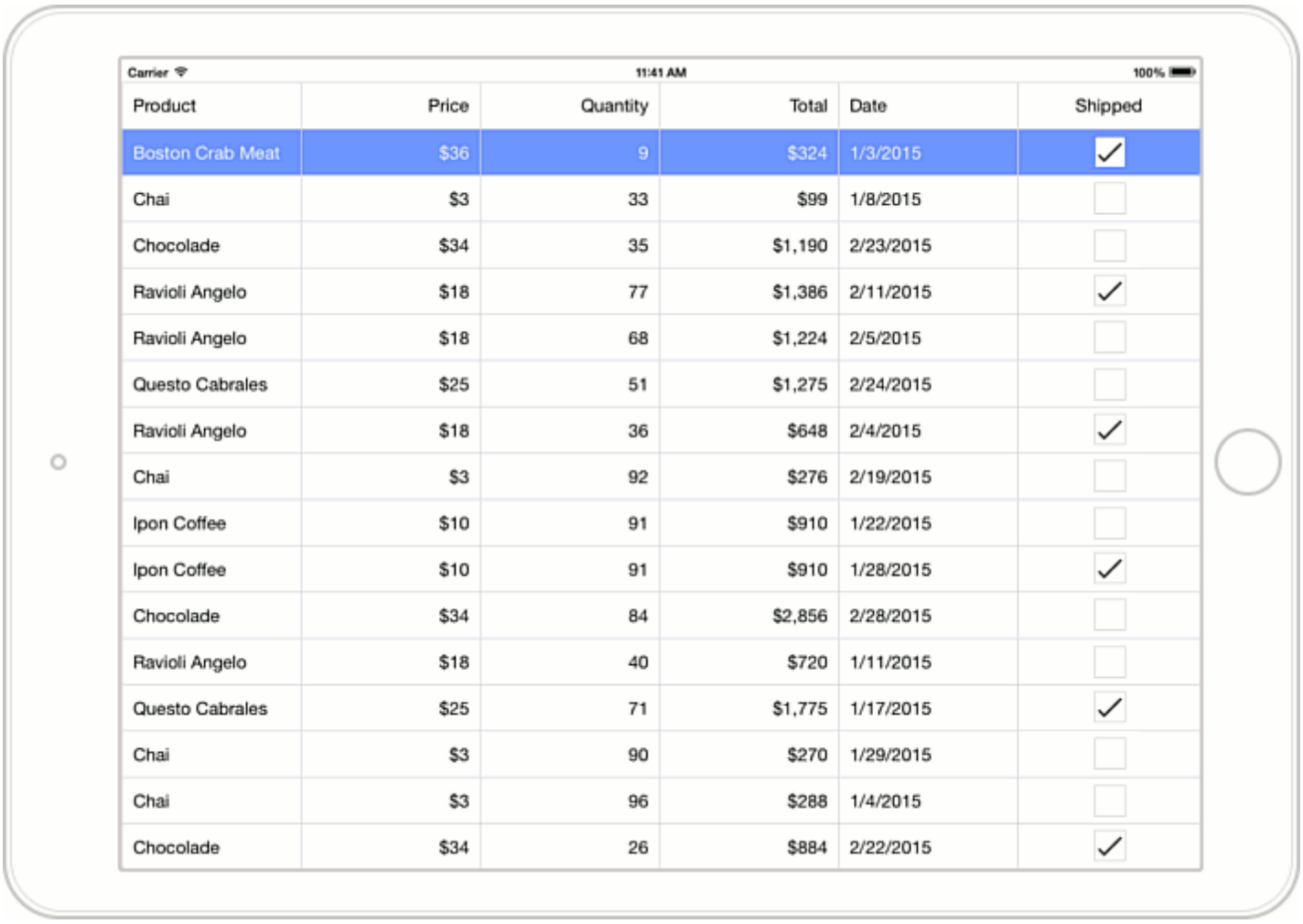
**Show Me**

A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T178764.

This example shows how to create and customize grid columns for displaying and editing data of different types (text, numbers, dates and Boolean values). The specified collection contains columns bound to the data source fields (*Product.Name*, *Product.UnitPrice*, *Quantity*, *Date* and *Shipped*) and one unbound column (*Total*) displaying data values calculated according to a formula based on the values of other columns.

The image below illustrates the result.

© 2018 Developer Express Inc.




| Product | Price | Quantity | Total | Date | Shipped |
|---|---|---|---|---|---|
| Boston Crab Meat | $36 | 9 | $324 | 1/3/2015 | ✓ |
| Chai | $3 | 33 | $99 | 1/8/2015 | |
| Chocolade | $34 | 35 | $1,190 | 2/23/2015 | |
| Ravioli Angelo | $18 | 77 | $1,386 | 2/11/2015 | ✓ |
| Ravioli Angelo | $18 | 68 | $1,224 | 2/5/2015 | |
| Questo Cabrales | $25 | 51 | $1,275 | 2/24/2015 | |
| Ravioli Angelo | $18 | 36 | $648 | 2/4/2015 | ✓ |
| Chai | $3 | 92 | $276 | 2/19/2015 | |
| Ipon Coffee | $10 | 91 | $910 | 1/22/2015 | |
| Ipon Coffee | $10 | 91 | $910 | 1/28/2015 | ✓ |
| Chocolade | $34 | 84 | $2,856 | 2/28/2015 | |
| Ravioli Angelo | $18 | 40 | $720 | 1/11/2015 | |
| Questo Cabrales | $25 | 71 | $1,775 | 1/17/2015 | ✓ |
| Chai | $3 | 90 | $270 | 1/29/2015 | |
| Chai | $3 | 96 | $288 | 1/4/2015 | |
| Chocolade | $34 | 26 | $884 | 2/22/2015 | ✓ |

© 2018 Developer Express Inc.

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl.Columns>
        <dxGrid:TextColumn FieldName="Product.Name" Caption = "Product" Width = "170"
                           SortOrder = "Descending" SortIndex = "0"/>
        <dxGrid:NumberColumn FieldName="Product.UnitPrice" Caption = "Price" DisplayFormat="C0"/>
        <dxGrid:NumberColumn FieldName="Quantity"
                             SortOrder = "Ascending" SortIndex = "1"/>
        <dxGrid:NumberColumn FieldName="Total"
                             UnboundType="Integer" UnboundExpression="[Quantity] * [Product.UnitPrice]"
                             DisplayFormat="C0" IsReadOnly="True"/>
        <dxGrid:DateColumn FieldName="Date" DisplayFormat="d"
                           IsGrouped = "true" GroupInterval = "Date"/>
        <dxGrid:SwitchColumn FieldName="Shipped" AllowSort = "False" AllowAutoFilter="false"/>
    </dxGrid:GridControl.Columns>
```

**See Also**

GridColumn Class
GridColumn Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# Width Property

Gets or sets the column width. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public Double Width { get; set; }
```

### Property Value

A System.Double value that specifies the column width in density-independent pixels.

### Remarks

Use the **Width** property to specify the column width.

The property behavior depends on whether or not the horizontal scrolling is enabled.

#### The Horizontal Scrolling is Enabled

The required column's width is automatically set to the **Width** property's value.

#### The Horizontal Scrolling is Disabled

Column widths are adjusted automatically, so that the total width of all columns matches the grid width. Changing the width of one column automatically changes the widths of other columns. To obtain the actual width of columns, use their ActualWidth property.

If you set the **Width** property only for one column, and there are other columns that don't have this property defined, the **GridControl** will set your required width value for this column and decrease the widths of columns where the width isn't set. However, if you set the **Width** for all columns, or if certain columns whose width is not defined already reached their MinWidth value, the **GridControl** will modify the width of all other columns in proportion to their defined **Width**, but will set it to a value not less than **MinWidth**.

### See Also

GridColumn Class
GridColumn Members
DevExpress.Mobile.DataGrid Namespace
Columns

© 2018 Developer Express Inc.

# GridColumnCollection Class

A grid's column collection.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public class GridColumnCollection : ObservableCollection<GridColumn>
```

### Remarks

The grid stores its columns within the GridControl.Columns collection. The **GridColumnCollection** class provides methods that can be used to add, delete, or access individual columns and perform other common collection management tasks. Individual columns are represented by the GridColumn class descendants and can be accessed by their field name or using indexed notation.

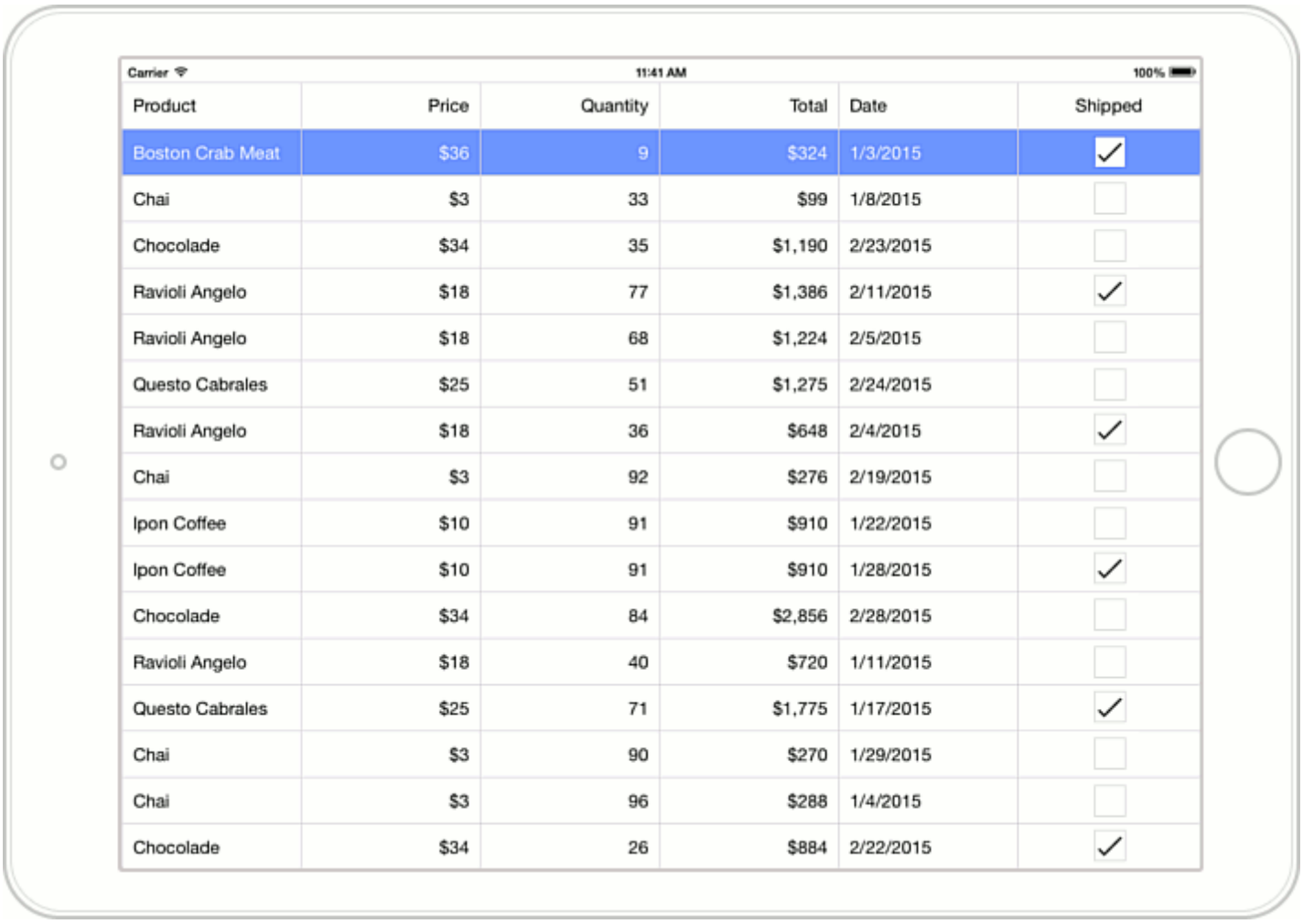
### Example

| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T178764. | |

This example shows how to create and customize grid columns for displaying and editing data of different types (text, numbers, dates and Boolean values). The specified collection contains columns bound to the data source fields (*Product.Name*, *Product.UnitPrice*, *Quantity*, *Date* and *Shipped*) and one unbound column (*Total*) displaying data values calculated according to a formula based on the values of other columns.

The image below illustrates the result.

© 2018 Developer Express Inc.




| Product | Price | Quantity | Total | Date | Shipped |
|---|---|---|---|---|---|
| Boston Crab Meat | $36 | 9 | $324 | 1/3/2015 | ✓ |
| Chai | $3 | 33 | $99 | 1/8/2015 | |
| Chocolade | $34 | 35 | $1,190 | 2/23/2015 | |
| Ravioli Angelo | $18 | 77 | $1,386 | 2/11/2015 | ✓ |
| Ravioli Angelo | $18 | 68 | $1,224 | 2/5/2015 | |
| Questo Cabrales | $25 | 51 | $1,275 | 2/24/2015 | |
| Ravioli Angelo | $18 | 36 | $648 | 2/4/2015 | ✓ |
| Chai | $3 | 92 | $276 | 2/19/2015 | |
| Ipon Coffee | $10 | 91 | $910 | 1/22/2015 | |
| Ipon Coffee | $10 | 91 | $910 | 1/28/2015 | ✓ |
| Chocolade | $34 | 84 | $2,856 | 2/28/2015 | |
| Ravioli Angelo | $18 | 40 | $720 | 1/11/2015 | |
| Questo Cabrales | $25 | 71 | $1,775 | 1/17/2015 | ✓ |
| Chai | $3 | 90 | $270 | 1/29/2015 | |
| Chai | $3 | 96 | $288 | 1/4/2015 | |
| Chocolade | $34 | 26 | $884 | 2/22/2015 | ✓ |

© 2018 Developer Express Inc.

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl.Columns>
        <dxGrid:TextColumn FieldName="Product.Name" Caption = "Product" Width = "170"
                           SortOrder = "Descending" SortIndex = "0"/>
        <dxGrid:NumberColumn FieldName="Product.UnitPrice" Caption = "Price" DisplayFormat="C0"/>
        <dxGrid:NumberColumn FieldName="Quantity"
                             SortOrder = "Ascending" SortIndex = "1"/>
        <dxGrid:NumberColumn FieldName="Total"
                             UnboundType="Integer" UnboundExpression="[Quantity] * [Product.UnitPrice]"
                             DisplayFormat="C0" IsReadOnly="True"/>
        <dxGrid:DateColumn FieldName="Date" DisplayFormat="d"
                           IsGrouped = "true" GroupInterval = "Date"/>
        <dxGrid:SwitchColumn FieldName="Shipped" AllowSort = "False" AllowAutoFilter="false"/>
    </dxGrid:GridControl.Columns>
```

### Inheritance Hierarchy

System.Object
  System.Collections.ObjectModel.Collection<DevExpress.Mobile.DataGrid.GridColumn>
    System.Collections.ObjectModel.ObservableCollection<DevExpress.Mobile.DataGrid.GridColumn>
      **GridColumnCollection**

### See Also

GridColumnCollection Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# GridColumnCollection Members

A grid's column collection.

The following tables list the members exposed by the GridColumnCollection type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | GridColumnCollection | Initializes a new instance of the GridColumnCollection class with default settings. |

### Public Properties

| | Name | Description |
|---|---|---|
| | Count | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | Item | Overloaded. Gets a column by the field name. |

### Public Methods

| | Name | Description |
|---|---|---|
| | Add | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | Clear | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | Contains | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | CopyTo | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetColumnByFieldName | Returns a column specified by its field name. |
| | GetEnumerator | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | IndexOf | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | Insert | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | Move | (Inherited from **System.Collections.ObjectModel.ObservableCollection<T>**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | Remove | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | RemoveAt | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | ToString | (Inherited from **System.Object**) |

**Public Events**

| | Name | Description |
|---|---|---|
| | CollectionChanged | (Inherited from **System.Collections.ObjectModel.ObservableCollection<T>**) |
| | ColumnPropertyChanged | Occurs after a column's property has been changed. |

**See Also**
GridColumnCollection Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# GridColumnCollection Constructor

Initializes a new instance of the GridColumnCollection class with default settings.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**
**C#**

```
public GridColumnCollection()
```

**See Also**
GridColumnCollection Class
GridColumnCollection Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# GridColumnCollection Properties

A grid's column collection.

The following tables list the members exposed by the GridColumnCollection type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | GridColumnCollection | Initializes a new instance of the GridColumnCollection class with default settings. |

### Public Properties

| | Name | Description |
|---|---|---|
| | Count | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | Item | Overloaded. Gets a column by the field name. |

### Public Methods

| | Name | Description |
|---|---|---|
| | Add | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | Clear | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | Contains | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | CopyTo | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetColumnByFieldName | Returns a column specified by its field name. |
| | GetEnumerator | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | IndexOf | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | Insert | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | Move | (Inherited from **System.Collections.ObjectModel.ObservableCollection<T>**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | Remove | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | RemoveAt | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | ToString | (Inherited from **System.Object**) |

**Public Events**

| | Name | Description |
|---|---|---|
| | CollectionChanged | (Inherited from **System.Collections.ObjectModel.ObservableCollection<T>**) |
| | ColumnPropertyChanged | Occurs after a column's property has been changed. |

**See Also**
GridColumnCollection Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# Item Property

Gets a column by the field name.

**Overload List**

| Name | Description |
|---|---|
| GridColumn this[string fieldName] { get; } | Gets a column by the field name. |
| T this[int index] { get; set; } | Inherited from **System.Collections.ObjectModel.Collection<T>** |

**See Also**
GridColumnCollection Class
GridColumnCollection Members
DevExpress.Mobile.DataGrid Namespace
GridColumnCollection.Item Overload List

Gets a column by the field name.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**
**C#**

```
public GridColumn this[
   string fieldName
] { get; }
```

**Parameters**
*fieldName*
A string value specifying the column's field name.

**Property Value**

A GridColumn descendant whose GridColumn.FieldName property value matches the specified field. **null** if the collection does not contain a column with the specified field.

**Remarks**

The property uses the GetColumnByFieldName method internally.

For details on how to access individual columns from the collection, see Identify and Access Columns.

**See Also**
GridColumnCollection Class
GridColumnCollection Members
DevExpress.Mobile.DataGrid Namespace
GridColumnCollection.Item Overload List
GridColumn.FieldName
Columns

© 2018 Developer Express Inc.

# GridColumnCollection Events

A grid's column collection.

The following tables list the members exposed by the GridColumnCollection type.

**Public Events**

| | Name | Description |
|---|---|---|
| | CollectionChanged | (Inherited from **System.Collections.ObjectModel.ObservableCollection<T>**) |
| | ColumnPropertyChanged | Occurs after a column's property has been changed. |

**See Also**
GridColumnCollection Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# ColumnPropertyChanged Event

Occurs after a column's property has been changed.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public event PropertyChangedEventHandler ColumnPropertyChanged
```

### Event Data

The event handler receives an argument of type System.ComponentModel.PropertyChangedEventArgs containing data related to this event.

The following **PropertyChangedEventArgs** properties provide information specific to this event.

| Property | Description |
|---|---|
| PropertyName | |

### See Also

GridColumnCollection Class
GridColumnCollection Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# GridColumnCollection Methods

A grid's column collection.

The following tables list the members exposed by the GridColumnCollection type.

**Public Methods**

| | Name | Description |
|---|---|---|
| | Add | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | Clear | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | Contains | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | CopyTo | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| S | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetColumnByFieldName | Returns a column specified by its field name. |
| | GetEnumerator | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | IndexOf | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | Insert | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | Move | (Inherited from **System.Collections.ObjectModel.ObservableCollection<T>**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | Remove | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | RemoveAt | (Inherited from **System.Collections.ObjectModel.Collection<T>**) |
| | ToString | (Inherited from **System.Object**) |

**See Also**
GridColumnCollection Members

© 2018 Developer Express Inc.

DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# GetColumnByFieldName Method

Returns a column specified by its field name.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public GridColumn GetColumnByFieldName(
    string fieldName
)
```

### Parameters

*fieldName*
A string value specifying the column's field name.

### Return Value

A GridColumn descendant whose GridColumn.FieldName property value matches the specified field. **null** if the collection does not contain a column with the specified field.

### Remarks

For details on how to access individual columns from the collection, see Identify and Access Columns.

### See Also

GridColumnCollection Class
GridColumnCollection Members
DevExpress.Mobile.DataGrid Namespace
Columns

© 2018 Developer Express Inc.

# GridColumnDataEventArgs Class

Provides data for the GridControl.CustomUnboundColumnData event.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public class GridColumnDataEventArgs : object
```

### Remarks

The GridControl.CustomUnboundColumnData event allows you to populate unbound columns with data and save the modified data in the unbound columns back to the data source.

The IsGetData and IsSetData properties can be used to determine whether the event handler should provide data for or save data from an unbound column. The unbound column currently being processed is determined by the Column property. To get the data of the currently processed row or assign a value to any field in this row, use the RowData or EditableRowData property.

### Inheritance Hierarchy

System.Object
  **GridColumnDataEventArgs**

### See Also

GridColumnDataEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# GridColumnDataEventArgs Members

Provides data for the GridControl.CustomUnboundColumnData event.

The following tables list the members exposed by the GridColumnDataEventArgs type.

### Public Properties

| | Name | Description |
|---|---|---|
| | Column | Gets the unbound column currently being processed. |
| | EditableRowData | Gets an object that contains information on the currently processed row and allows you to modify field values. |
| | IsGetData | Gets whether you should provide data for the currently processed cell. |
| | IsSetData | Gets whether the current cell's value should be saved in a data source. |
| | RowData | Gets an object that contains information on the currently processed row. |
| | Source | Gets the grid control that raised the event. |
| | Value | Gets or sets the value of the cell currently being processed. |

### Public Methods

| | Name | Description |
|---|---|---|
| S | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

### See Also

GridColumnDataEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# GridColumnDataEventArgs Properties

Provides data for the GridControl.CustomUnboundColumnData event.

The following tables list the members exposed by the GridColumnDataEventArgs type.

**Public Properties**

| | Name | Description |
|---|---|---|
| | Column | Gets the unbound column currently being processed. |
| | EditableRowData | Gets an object that contains information on the currently processed row and allows you to modify field values. |
| | IsGetData | Gets whether you should provide data for the currently processed cell. |
| | IsSetData | Gets whether the current cell's value should be saved in a data source. |
| | RowData | Gets an object that contains information on the currently processed row. |
| | Source | Gets the grid control that raised the event. |
| | Value | Gets or sets the value of the cell currently being processed. |

**Public Methods**

| | Name | Description |
|---|---|---|
| S | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

**See Also**
GridColumnDataEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# Column Property

Gets the unbound column currently being processed.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public GridColumn Column { get; }
```

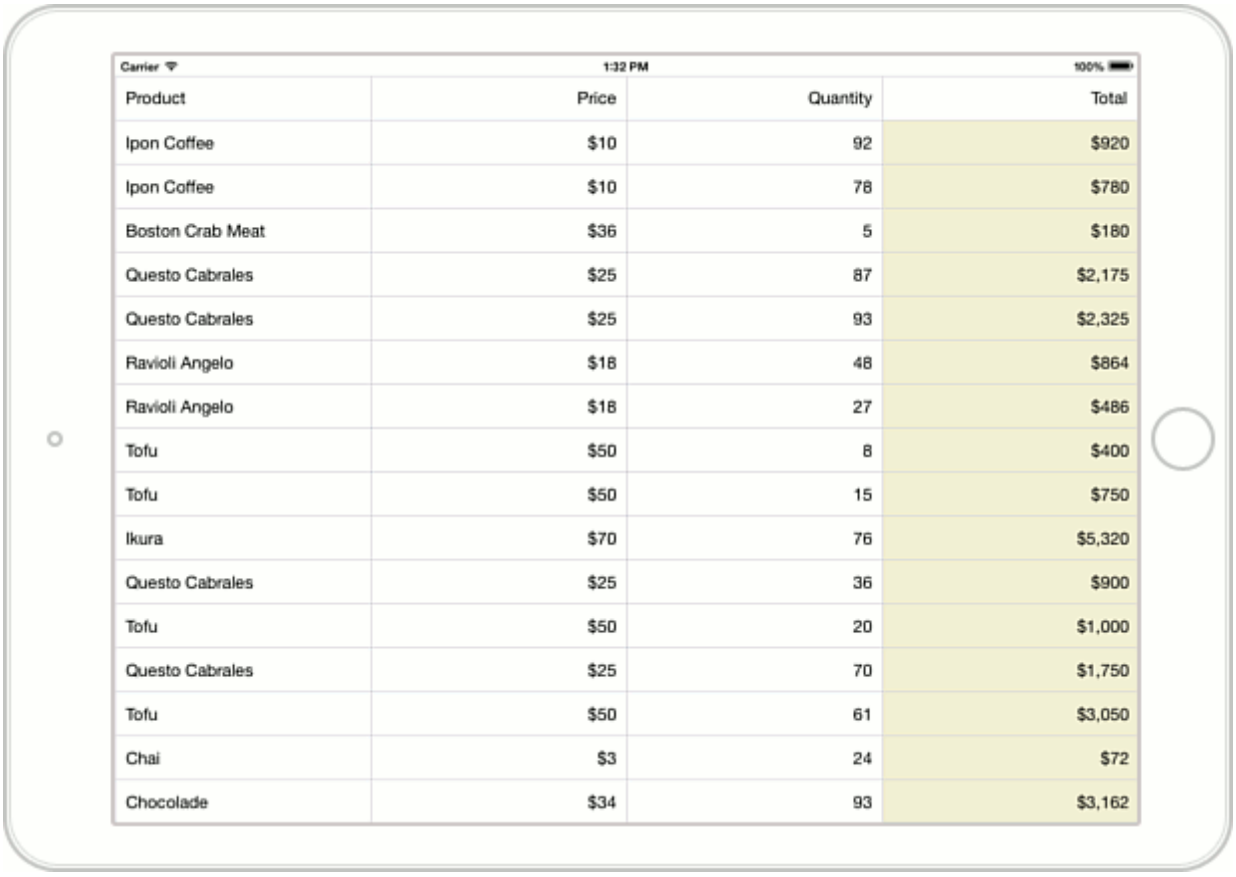
### Property Value

A GridColumn object or descendant specifying the unbound column currently being processed.

### Example

Assume that the GridControl instance is bound to a collection of orders. Each order has the "Product", "UnitPrice"and "Quantity" fields. The example below shows how to add an unbound column to the grid to display the amount of each order according to the expression: UnitPrice*Quantity.

The result is displayed below:




| Product | Price | Quantity | Total |
|---|---|---|---|
| Ipon Coffee | $10 | 92 | $920 |
| Ipon Coffee | $10 | 78 | $780 |
| Boston Crab Meat | $36 | 5 | $180 |
| Questo Cabrales | $25 | 87 | $2,175 |
| Questo Cabrales | $25 | 93 | $2,325 |
| Ravioli Angelo | $18 | 48 | $864 |
| Ravioli Angelo | $18 | 27 | $486 |
| Tofu | $50 | 8 | $400 |
| Tofu | $50 | 15 | $750 |
| Ikura | $70 | 76 | $5,320 |
| Questo Cabrales | $25 | 36 | $900 |
| Tofu | $50 | 20 | $1,000 |
| Questo Cabrales | $25 | 70 | $1,750 |
| Tofu | $50 | 61 | $3,050 |
| Chai | $3 | 24 | $72 |
| Chocolade | $34 | 93 | $3,162 |

**Xaml**

© 2018 Developer Express Inc.

```
<dxGrid:GridControl ItemsSource="{Binding Orders}"
                    CustomUnboundColumnData="OnCustomUnboundColumnData"
                    CustomizeCell="OnCustomizeCell">
    <dxGrid:GridControl.Columns>
         <dxGrid:TextColumn FieldName="Product.Name" Caption="Product"/>
         <dxGrid:NumberColumn FieldName="Product.UnitPrice" Caption="Price" DisplayFormat="C0"/>
         <dxGrid:NumberColumn FieldName="Quantity" />
         <dxGrid:NumberColumn FieldName="Total" IsReadOnly="True"
                              UnboundType="Decimal" DisplayFormat="C0"/>
    </dxGrid:GridControl.Columns>
</dxGrid:GridControl>
```

**C#**

```
// Returns the total for a specific row.
decimal getTotalValue(GridControl grid, int rowHandle) {
    decimal unitPrice = Convert.ToDecimal(grid.GetCellValue(rowHandle, "Product.UnitPrice"));
    decimal quantity = Convert.ToDecimal(grid.GetCellValue(rowHandle, "Quantity"));
    return unitPrice * quantity;
}
// Provides data for the Total column.
void OnCustomUnboundColumnData(object sender, GridColumnDataEventArgs e) {
    if (e.Column.FieldName == "Total" && e.IsGetData)
        e.Value = getTotalValue (e.Source, e.RowData.RowHandle);
}
// Customizes the appearance settings.
void OnCustomizeCell(CustomizeCellEventArgs e) {
if (e.FieldName == "Total") {
        e.BackgroundColor = Color.FromRgb(247, 240, 213);
        e.Handled = true;
    }
}
```

**See Also**

GridColumnDataEventArgs Class
GridColumnDataEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# EditableRowData Property

Gets an object that contains information on the currently processed row and allows you to modify field values.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public IEditableRowData EditableRowData { get; }
```

### Property Value

An object implementing the IEditableRowData interface.

### Remarks

If the GridControl.CustomUnboundColumnData event is raised when an end-user modifies an unbound column's value in the grid (the IsSetData parameter is set to **true**), you can use the **EditableRowData** property in the event handler to access an IEditableRowData object containing data of the processed row and call it's IEditableRowData.SetFieldValue method to change and save any field's value.

### See Also

GridColumnDataEventArgs Class
GridColumnDataEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# IsGetData Property

Gets whether you should provide data for the currently processed cell.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public bool IsGetData { get; }
```

### Property Value

**true** if you need to provide data for the currently processed cell, otherwise **false**.

### Remarks

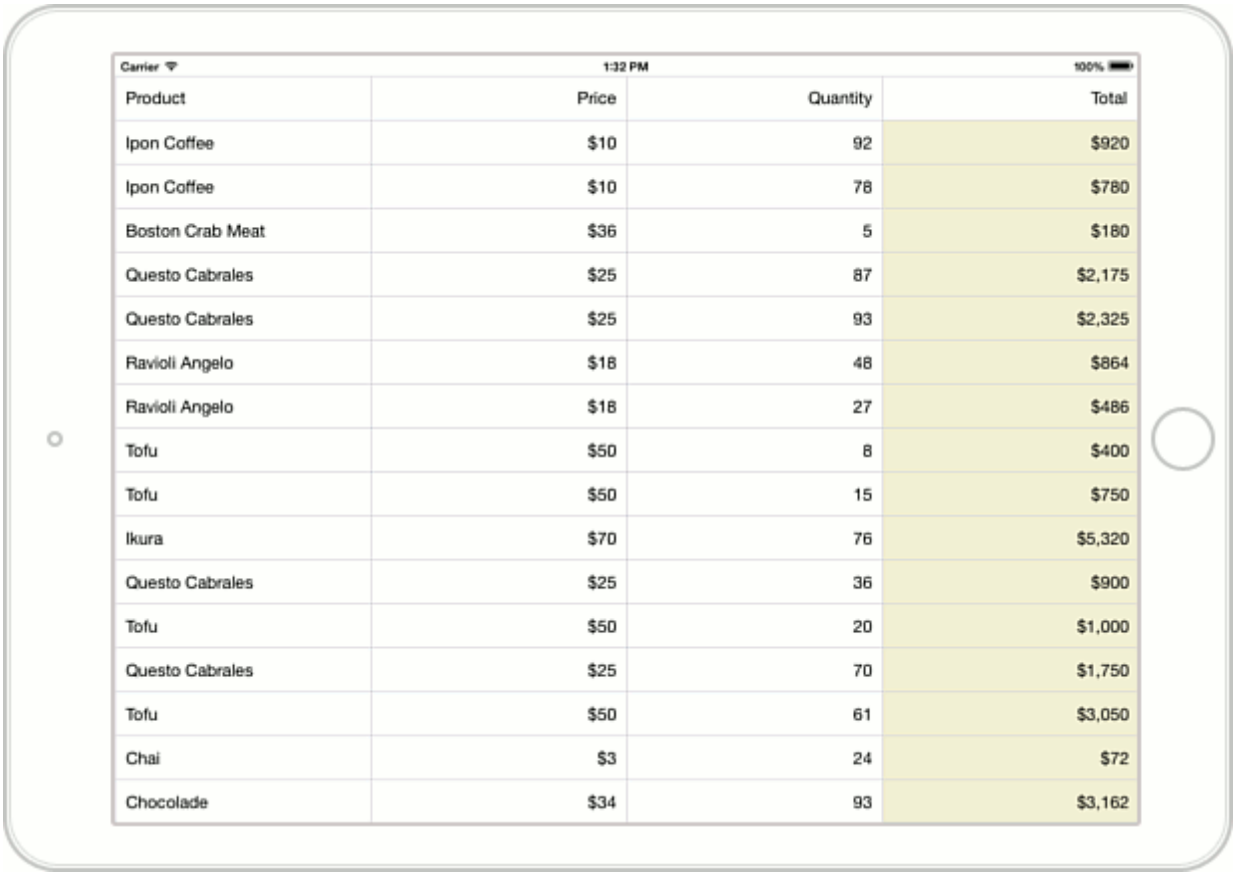
If the **IsGetData** property returns **true** (and consequently the IsSetData property is set to **false**) the data for the cell currently being processed should be supplied via the event handler. To do this, assign a required value (for example, it can be calculated based on values of other columns in the current row) to the event's Value parameter. The RowData property can be used to identify the currently processed row. The current unbound column is identified by the Column property.

If the **IsGetData** property is set to **false** (and consequently the IsSetData property is set to **true**) the Value parameter identifies the value which has been modified for the current cell from the grid. It should be saved to your data source by the event handler.

### Example

Assume that the GridControl instance is bound to a collection of orders. Each order has the "Product", "UnitPrice"and "Quantity" fields. The example below shows how to add an unbound column to the grid to display the amount of each order according to the expression: UnitPrice*Quantity.

The result is displayed below:




© 2018 Developer Express Inc.

**Xaml**

```
<dxGrid:GridControl ItemsSource="{Binding Orders}"
                    CustomUnboundColumnData="OnCustomUnboundColumnData"
                    CustomizeCell="OnCustomizeCell">
    <dxGrid:GridControl.Columns>
         <dxGrid:TextColumn FieldName="Product.Name" Caption="Product"/>
         <dxGrid:NumberColumn FieldName="Product.UnitPrice" Caption="Price" DisplayFormat="C0"/>
         <dxGrid:NumberColumn FieldName="Quantity" />
         <dxGrid:NumberColumn FieldName="Total" IsReadOnly="True"
                              UnboundType="Decimal" DisplayFormat="C0"/>
    </dxGrid:GridControl.Columns>
</dxGrid:GridControl>
```

**C#**

```
// Returns the total for a specific row.
decimal getTotalValue(GridControl grid, int rowHandle) {
    decimal unitPrice = Convert.ToDecimal(grid.GetCellValue(rowHandle, "Product.UnitPrice"));
    decimal quantity = Convert.ToDecimal(grid.GetCellValue(rowHandle, "Quantity"));
    return unitPrice * quantity;
}
// Provides data for the Total column.
void OnCustomUnboundColumnData(object sender, GridColumnDataEventArgs e) {
    if (e.Column.FieldName == "Total" && e.IsGetData)
        e.Value = getTotalValue (e.Source, e.RowData.RowHandle);
}
// Customizes the appearance settings.
void OnCustomizeCell(CustomizeCellEventArgs e) {
if (e.FieldName == "Total") {
        e.BackgroundColor = Color.FromRgb(247, 240, 213);
        e.Handled = true;
    }
}
```

**See Also**

GridColumnDataEventArgs Class
GridColumnDataEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# IsSetData Property

Gets whether the current cell's value should be saved in a data source.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public bool IsSetData { get; }
```

### Property Value

**true** if the current cell's value needs to be saved in a data source.

### Remarks

If the **IsSetData** property is set to **true** (and consequently the IsGetData property is set to **false**) the Value parameter identifies the value which has been set for the current cell in the grid. This value should be saved in your data source by the event handler.

If the **IsSetData** property returns **false** (and consequently the IsSetData property is set to **true**) the data for the currently processed cell should be supplied via the event handler. To do this, assign a required value (for example, it can be calculated based on values of other columns in the current row) to the event's Value parameter.

The RowData property can be used to identify the cell's row. The current cell's column is identified by the Column property.

### See Also

GridColumnDataEventArgs Class
GridColumnDataEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# RowData Property

Gets an object that contains information on the currently processed row.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public IRowData RowData { get; }
```

### Property Value

An object implementing the IRowData interface.

### Remarks

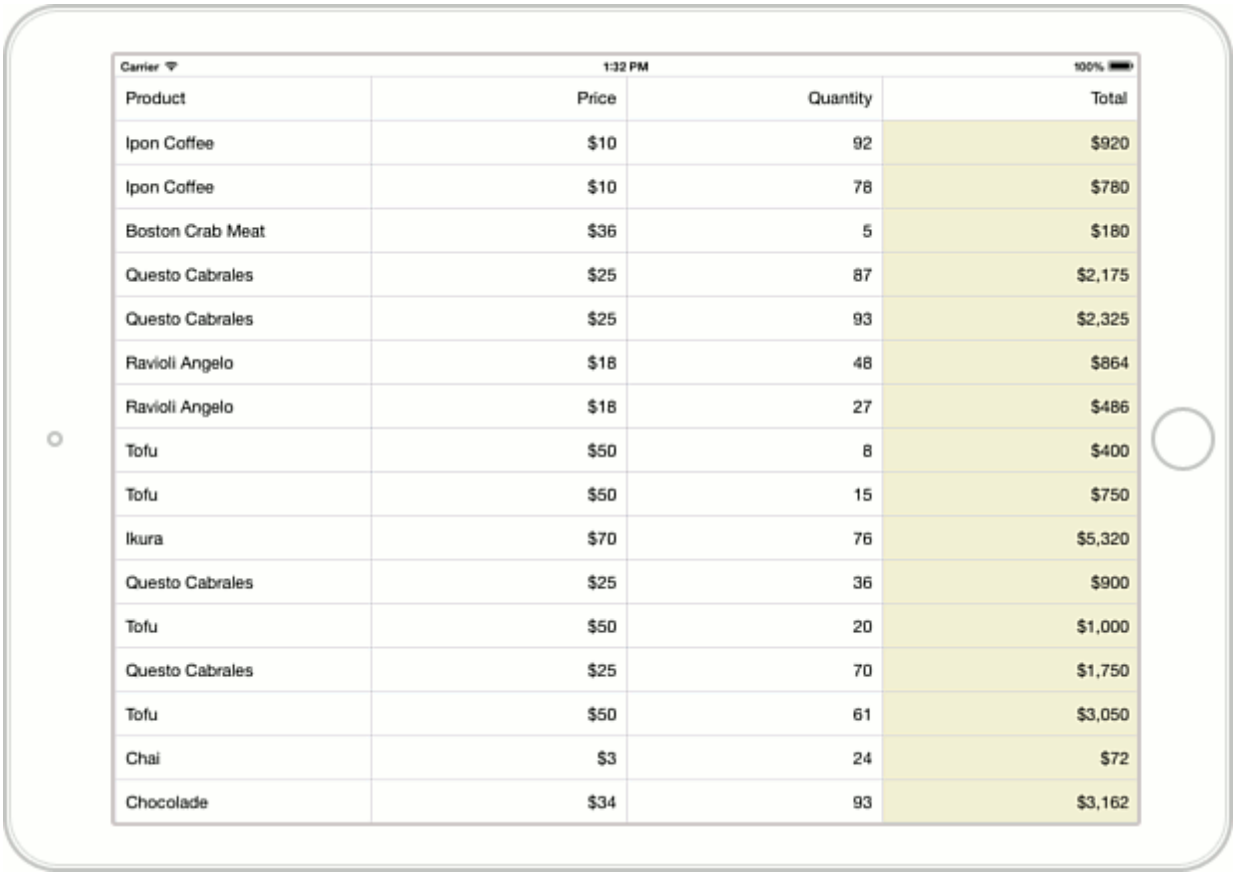
Use the IRowData object's properties and methods to access required data values of the processed row when handling the GridControl.CustomUnboundColumnData event to supply data for an unbound column. The IRowData.RowHandle property returns the processed row handle. To obtain an object representing a record in the grid's underlying data source, use the IRowData.DataObject property. To get a value in the specified field, use the IRowData.GetFieldValue method.

If the GridControl.CustomUnboundColumnData event is raised when an unbound column's value is modified in the grid (IsSetData is **true**), you can change any field's value based on the modified value of the unbound column using the IEditableRowData.SetFieldValue method of an object returned by the EditableRowData property.

### Example

Assume that the GridControl instance is bound to a collection of orders. Each order has the "Product", "UnitPrice"and "Quantity" fields. The example below shows how to add an unbound column to the grid to display the amount of each order according to the expression: UnitPrice*Quantity.

The result is displayed below:




© 2018 Developer Express Inc.

**Xaml**

```
<dxGrid:GridControl ItemsSource="{Binding Orders}"
                    CustomUnboundColumnData="OnCustomUnboundColumnData"
                    CustomizeCell="OnCustomizeCell">
    <dxGrid:GridControl.Columns>
         <dxGrid:TextColumn FieldName="Product.Name" Caption="Product"/>
         <dxGrid:NumberColumn FieldName="Product.UnitPrice" Caption="Price" DisplayFormat="C0"/>
         <dxGrid:NumberColumn FieldName="Quantity" />
         <dxGrid:NumberColumn FieldName="Total" IsReadOnly="True"
                              UnboundType="Decimal" DisplayFormat="C0"/>
    </dxGrid:GridControl.Columns>
</dxGrid:GridControl>
```

**C#**

```
// Returns the total for a specific row.
decimal getTotalValue(GridControl grid, int rowHandle) {
    decimal unitPrice = Convert.ToDecimal(grid.GetCellValue(rowHandle, "Product.UnitPrice"));
    decimal quantity = Convert.ToDecimal(grid.GetCellValue(rowHandle, "Quantity"));
    return unitPrice * quantity;
}
// Provides data for the Total column.
void OnCustomUnboundColumnData(object sender, GridColumnDataEventArgs e) {
    if (e.Column.FieldName == "Total" && e.IsGetData)
        e.Value = getTotalValue (e.Source, e.RowData.RowHandle);
}
// Customizes the appearance settings.
void OnCustomizeCell(CustomizeCellEventArgs e) {
if (e.FieldName == "Total") {
        e.BackgroundColor = Color.FromRgb(247, 240, 213);
        e.Handled = true;
    }
}
```

**See Also**

GridColumnDataEventArgs Class
GridColumnDataEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# Source Property

Gets the grid control that raised the event.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public GridControl Source { get; }
```

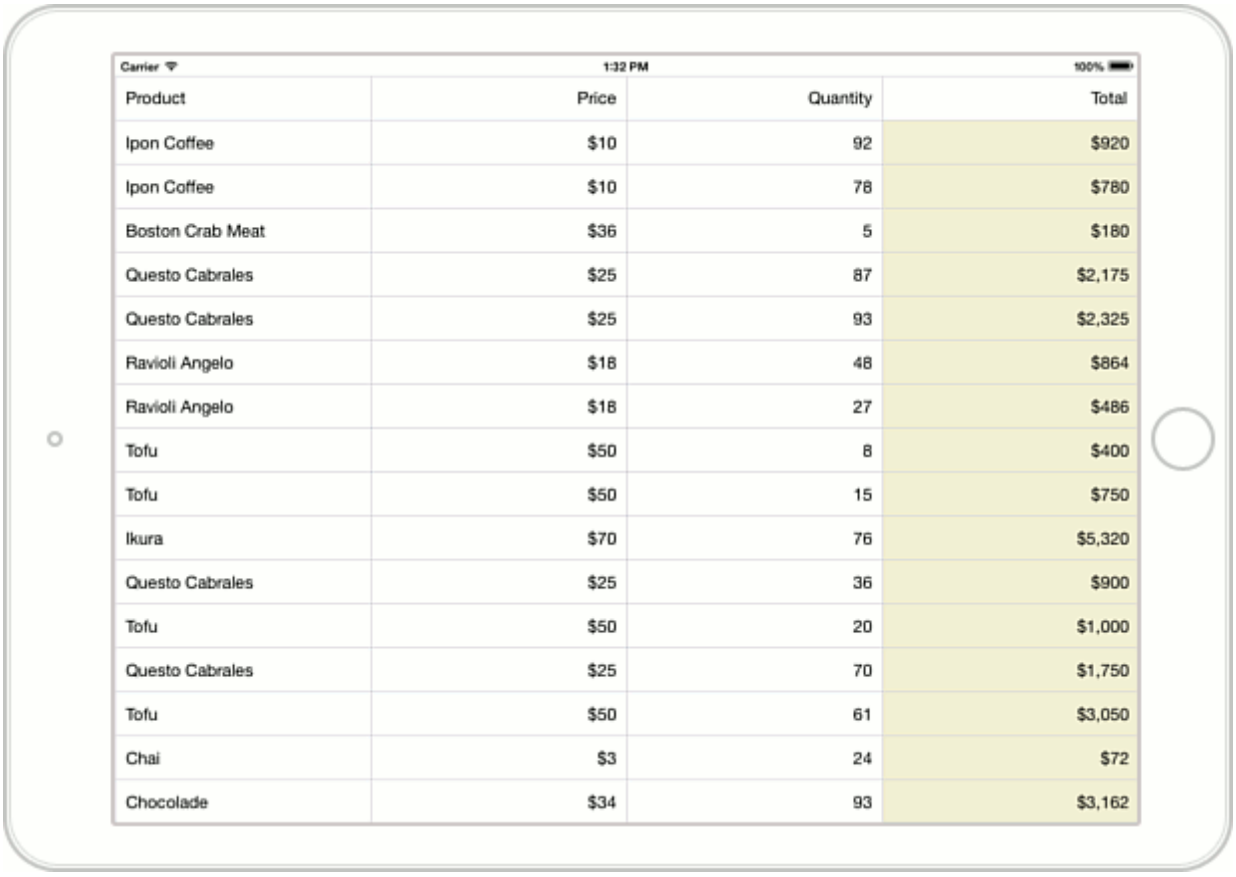
### Property Value

A GridControl object that raised the event.

### Example

Assume that the GridControl instance is bound to a collection of orders. Each order has the "Product", "UnitPrice"and "Quantity" fields. The example below shows how to add an unbound column to the grid to display the amount of each order according to the expression: UnitPrice*Quantity.

The result is displayed below:




**Xaml**

```
<dxGrid:GridControl ItemsSource="{Binding Orders}"
                    CustomUnboundColumnData="OnCustomUnboundColumnData"
                    CustomizeCell="OnCustomizeCell">
    <dxGrid:GridControl.Columns>
         <dxGrid:TextColumn FieldName="Product.Name" Caption="Product"/>
         <dxGrid:NumberColumn FieldName="Product.UnitPrice" Caption="Price" DisplayFormat="C0"/>
         <dxGrid:NumberColumn FieldName="Quantity" />
         <dxGrid:NumberColumn FieldName="Total" IsReadOnly="True"
                              UnboundType="Decimal" DisplayFormat="C0"/>
    </dxGrid:GridControl.Columns>
</dxGrid:GridControl>
```

**C#**

```
// Returns the total for a specific row.
decimal getTotalValue(GridControl grid, int rowHandle) {
    decimal unitPrice = Convert.ToDecimal(grid.GetCellValue(rowHandle, "Product.UnitPrice"));
    decimal quantity = Convert.ToDecimal(grid.GetCellValue(rowHandle, "Quantity"));
    return unitPrice * quantity;
}
// Provides data for the Total column.
void OnCustomUnboundColumnData(object sender, GridColumnDataEventArgs e) {
    if (e.Column.FieldName == "Total" && e.IsGetData)
        e.Value = getTotalValue (e.Source, e.RowData.RowHandle);
}
// Customizes the appearance settings.
void OnCustomizeCell(CustomizeCellEventArgs e) {
if (e.FieldName == "Total") {
        e.BackgroundColor = Color.FromRgb(247, 240, 213);
        e.Handled = true;
    }
}
```

**See Also**

GridColumnDataEventArgs Class
GridColumnDataEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# Value Property

Gets or sets the value of the cell currently being processed.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public object Value { get; set; }
```

### Property Value

An object specifying the value of the cell currently being processed.

### Remarks

If the IsGetData property is set to **true** (and consequently the IsSetData property is set to **false**), a GridControl.CustomUnboundColumnData event handler should provide data for the cell currently being processed. You need to assign a required value (for example, it can be calculated based on values of other columns in the current row) to the **Value** property.

If the IsGetData property is set to **false** (and consequently the IsSetData property is set to **true**), a GridControl.CustomUnboundColumnData event handler should save the value of the **Value** property to the data source. In this case, the event is raised as a result of modifying the cell's value via the grid.

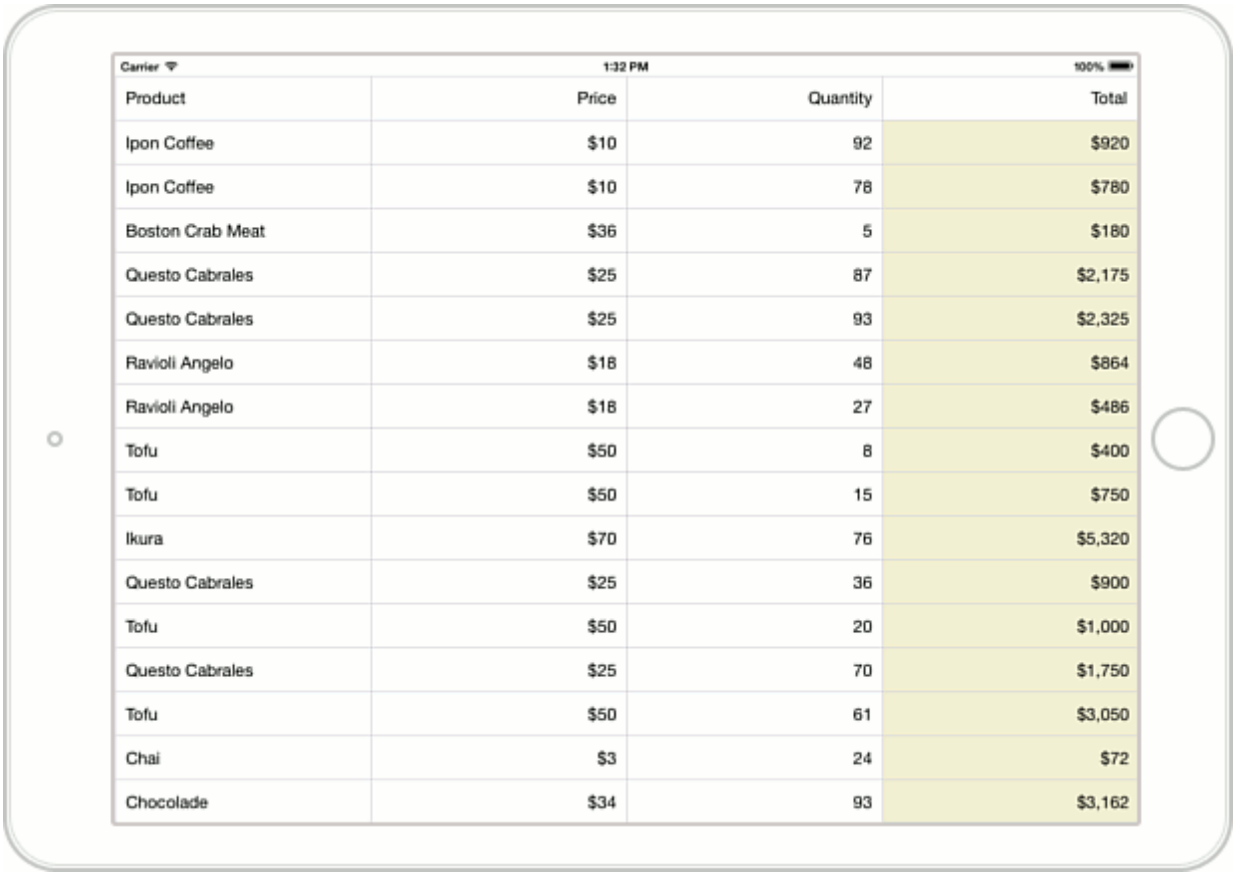
The current unbound column is identified by the Column property. The RowData property can be used to identify the current row.

### Example

Assume that the GridControl instance is bound to a collection of orders. Each order has the "Product", "UnitPrice"and "Quantity" fields. The example below shows how to add an unbound column to the grid to display the amount of each order according to the expression: UnitPrice*Quantity.

The result is displayed below:

© 2018 Developer Express Inc.



| Product | Price | Quantity | Total |
|---|---|---|---|
| Ipon Coffee | $10 | 92 | $920 |
| Ipon Coffee | $10 | 78 | $780 |
| Boston Crab Meat | $36 | 5 | $180 |
| Questo Cabrales | $25 | 87 | $2,175 |
| Questo Cabrales | $25 | 93 | $2,325 |
| Ravioli Angelo | $18 | 48 | $864 |
| Ravioli Angelo | $18 | 27 | $486 |
| Tofu | $50 | 8 | $400 |
| Tofu | $50 | 15 | $750 |
| Ikura | $70 | 76 | $5,320 |
| Questo Cabrales | $25 | 36 | $900 |
| Tofu | $50 | 20 | $1,000 |
| Questo Cabrales | $25 | 70 | $1,750 |
| Tofu | $50 | 61 | $3,050 |
| Chai | $3 | 24 | $72 |
| Chocolade | $34 | 93 | $3,162 |

**Xaml**

```
<dxGrid:GridControl ItemsSource="{Binding Orders}"
                    CustomUnboundColumnData="OnCustomUnboundColumnData"
                    CustomizeCell="OnCustomizeCell">
    <dxGrid:GridControl.Columns>
         <dxGrid:TextColumn FieldName="Product.Name" Caption="Product"/>
         <dxGrid:NumberColumn FieldName="Product.UnitPrice" Caption="Price" DisplayFormat="C0"/>
         <dxGrid:NumberColumn FieldName="Quantity" />
         <dxGrid:NumberColumn FieldName="Total" IsReadOnly="True"
                              UnboundType="Decimal" DisplayFormat="C0"/>
    </dxGrid:GridControl.Columns>
</dxGrid:GridControl>
```

**C#**

```
// Returns the total for a specific row.
decimal getTotalValue(GridControl grid, int rowHandle) {
    decimal unitPrice = Convert.ToDecimal(grid.GetCellValue(rowHandle, "Product.UnitPrice"));
    decimal quantity = Convert.ToDecimal(grid.GetCellValue(rowHandle, "Quantity"));
    return unitPrice * quantity;
}
// Provides data for the Total column.
void OnCustomUnboundColumnData(object sender, GridColumnDataEventArgs e) {
    if (e.Column.FieldName == "Total" && e.IsGetData)
        e.Value = getTotalValue (e.Source, e.RowData.RowHandle);
}
// Customizes the appearance settings.
void OnCustomizeCell(CustomizeCellEventArgs e) {
if (e.FieldName == "Total") {
        e.BackgroundColor = Color.FromRgb(247, 240, 213);
        e.Handled = true;
    }
}
```

© 2018 Developer Express Inc.

**See Also**
GridColumnDataEventArgs Class
GridColumnDataEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# GridColumnDataEventHandler Delegate

Represents a method that will handle the GridControl.CustomUnboundColumnData event.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public delegate void GridColumnDataEventHandler(
   object sender,
   GridColumnDataEventArgs e
)
```

### Parameters

The declaration of your event handler must have the same parameters as the **GridColumnDataEventHandler** delegate declaration.
*sender*
The event source.
*e*
A GridColumnDataEventArgs object that contains event data.

### See Also

DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# GridColumnSummary Class

A data summary item.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public class GridColumnSummary : BindableObject, IToDecimalConverter,
  ICloneable,
  ICancellableReader
```

### Remarks

The GridControl allows you to display concise information about groups of rows or individual data columns. For example, you can display the number of records, or maximum value, etc. This is called a *summary*.

Two summary types are supported: total summaries and group summaries. The grid stores its total summary items within the GridControl.TotalSummaries collection. Group summary items are stored within the GridControl.GroupSummaries collection. Individual summary items are represented by the **GridColumnSummary** class.

When specifying a data summary in the grid, use **GridColumnSummary** object properties to specify the field against whose values a summary should be calculated (FieldName), the summary value format (DisplayFormat) and the aggregate function (Type). Note that you can calculate summaries using predefined aggregate functions (SummaryType.Count, SummaryType.Min, SummaryType.Max, SummaryType.Sum or SummaryType.Average) or your own custom rule (set the Type property to SummaryType.Custom and handle the GridControl.CalculateCustomSummary event).

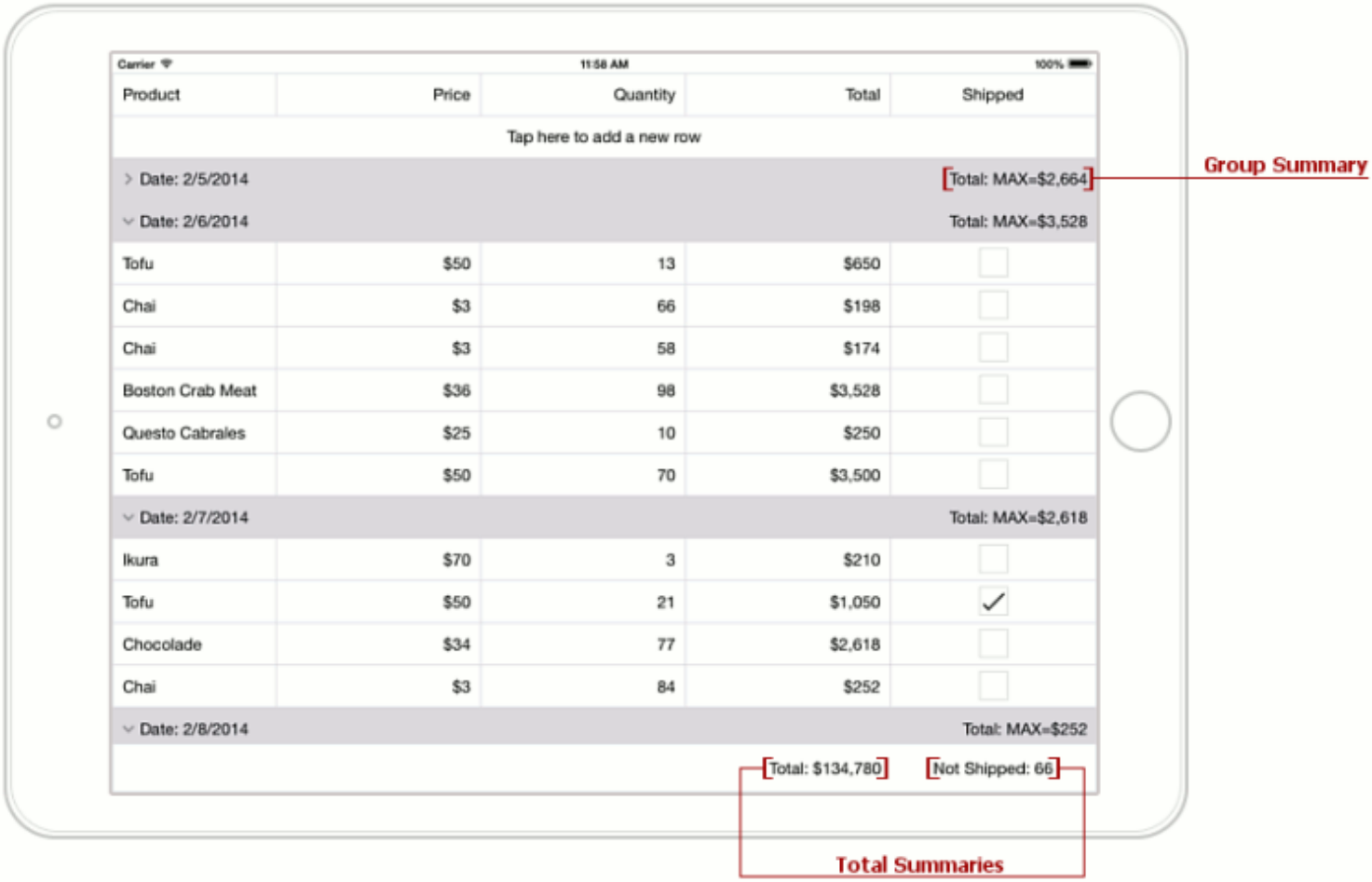
### Example

This example shows how to calculate group and total summaries for grid columns using predefined aggregate functions or a custom rule.

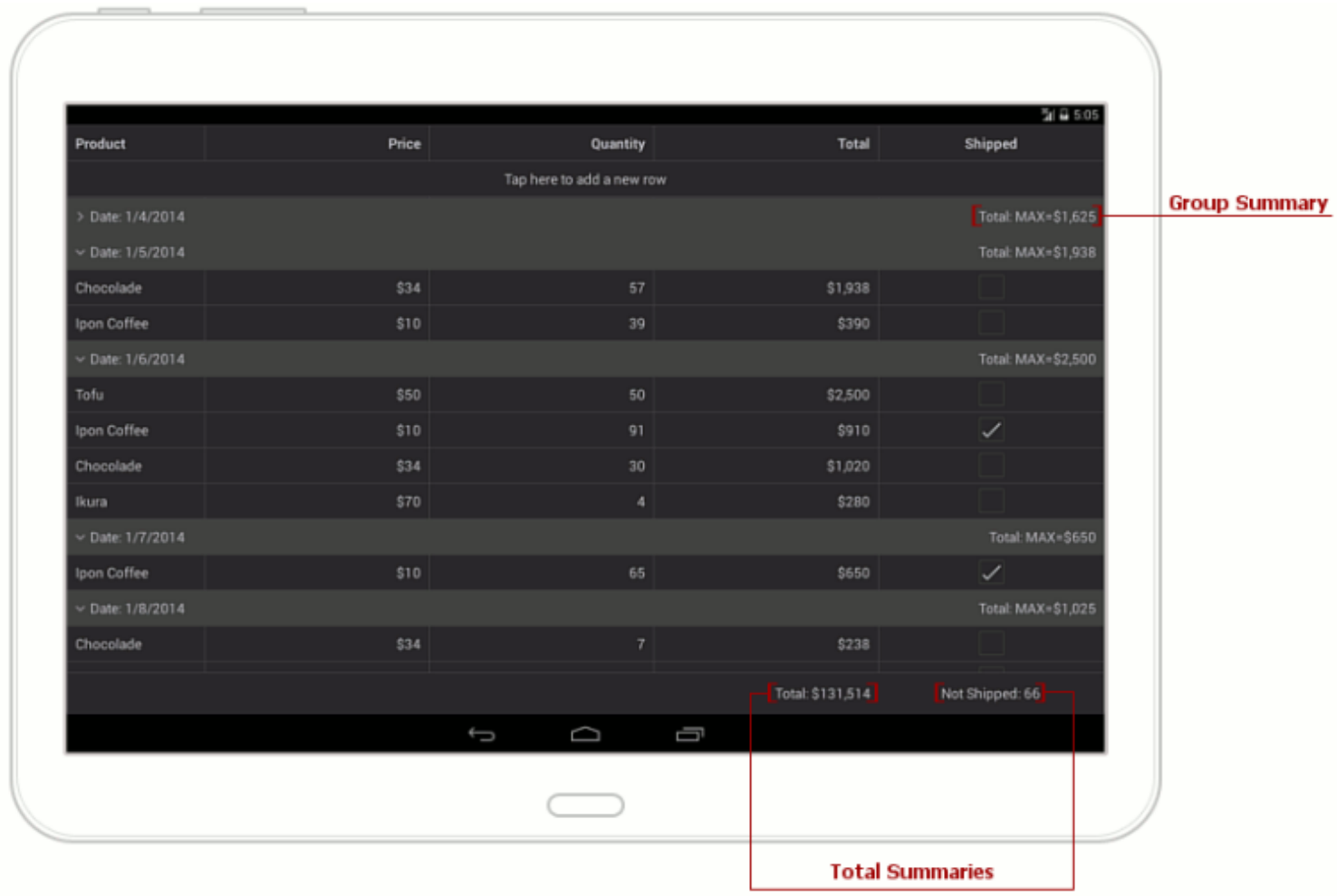
The following data summaries are created.

- Group summary to display the maximum value in a **Total** column for each group of records.
- Total summary to calculate the sum of values by the whole **Total** column.
- Custom total summary to count the number of rows whose value in the **Shipped** column is **false**.

© 2018 Developer Express Inc.




© 2018 Developer Express Inc.




**Xaml**

```
<dxGrid:GridControl x:Name="grid" ItemsSource="{Binding Orders}"
                    NewItemRowVisibility="true"
                    CalculateCustomSummary="OnCalculateCustomSummary">
  <!-- ... -->
  <dxGrid:GridControl.GroupSummaries>
    <dxGrid:GridColumnSummary FieldName="Total" Type="Max"/>
  </dxGrid:GridControl.GroupSummaries>
  <dxGrid:GridControl.TotalSummaries>
    <dxGrid:GridColumnSummary FieldName="Total" Type="Sum" DisplayFormat="Total: {0:C0}"/>
    <dxGrid:GridColumnSummary FieldName="Shipped" Type="Custom" DisplayFormat="Not Shipped: {0}"/>
  </dxGrid:GridControl.TotalSummaries>
  <!-- ... -->
</dxGrid:GridControl>
```

**C#**

```
void OnCalculateCustomSummary(object sender, CustomSummaryEventArgs e) {
    if (e.FieldName.ToString () == "Shipped")
    if (e.IsTotalSummary){
        if (e.SummaryProcess == CustomSummaryProcess.Start) {
            count = 0;
        }
        if (e.SummaryProcess == CustomSummaryProcess.Calculate) {
            if (!(bool)e.FieldValue)
                count++;
            e.TotalValue = count;
        }
    }
}
```

© 2018 Developer Express Inc.

**Inheritance Hierarchy**

System.Object
  Xamarin.Forms.BindableObject
    **GridColumnSummary**

**See Also**
GridColumnSummary Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# GridColumnSummary Members

A data summary item.

The following tables list the members exposed by the GridColumnSummary type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | GridColumnSummary | Overloaded. Initializes a new GridColumnSummary class instance with the specified column's field name. |

### Public Properties

| | Name | Description |
|---|---|---|
| | BindingContext | (Inherited from **Xamarin.Forms.BindableObject**) |
| | DisplayFormat | Gets or sets the pattern used to format the summary value. This is a bindable property. |
| | FieldName | Gets or sets the field name of the column for which a summary should be calculated. This is a bindable property. |
| | IgnoreNullValues | Specifies whether null values should be ignored when a summary is calculated. This is a bindable property. |
| | IsValid | Indicates whether the summary value is calculated. |
| | SummaryValue | Gets the summary value. |
| | Type | Gets or sets the aggregation function type. This is a bindable property. |

### Public Methods

| | Name | Description |
|---|---|---|
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Clone | Creates a copy of the current GridColumnSummary object. |
| | Equals | (Inherited from **System.Object**) |
| S | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetSummaryValueAsync | Gets the summary value asynchronously. |
| | GetType | (Inherited from **System.Object**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Reset | Interrupts the process of summary calculation. |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ToString | (Inherited from **System.Object**) |

**Public Fields**

| | Name | Description |
|---|---|---|
| S | BindingContextProperty | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | DisplayFormatProperty | Identifies the DisplayFormat bindable property. |
| S | FieldNameProperty | Identifies the FieldName bindable property. |
| S | IgnoreNullValuesProperty | Identifies the IgnoreNullValues bindable property. |
| S | TypeProperty | Identifies the Type bindable property. |

**Public Events**

| | Name | Description |
|---|---|---|
| | BindingContextChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanging | (Inherited from **Xamarin.Forms.BindableObject**) |

**See Also**
GridColumnSummary Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# GridColumnSummary Constructor

Initializes a new GridColumnSummary class instance with default settings.

**Overload List**

| Name | Description |
|---|---|
| GridColumnSummary() | Initializes a new GridColumnSummary class instance with default settings. |
| GridColumnSummary(string column) | Initializes a new GridColumnSummary class instance with the specified column's field name. |

**See Also**
GridColumnSummary Class
GridColumnSummary Members
DevExpress.Mobile.DataGrid Namespace
GridColumnSummary Overload List

© 2018 Developer Express Inc.

# GridColumnSummary Constructor

Initializes a new GridColumnSummary class instance with default settings.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**
**C#**

```
public GridColumnSummary()
```

**See Also**
GridColumnSummary Class
GridColumnSummary Members
DevExpress.Mobile.DataGrid Namespace
GridColumnSummary Overload List

© 2018 Developer Express Inc.

# GridColumnSummary Constructor (string)

Initializes a new GridColumnSummary class instance with the specified column's field name.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public GridColumnSummary(
   string column
)
```

### Parameters

*column*
A string value that specifies the field name of a column for which a summary should be calculated. This value is assigned to the FieldName property.

### See Also

GridColumnSummary Class
GridColumnSummary Members
DevExpress.Mobile.DataGrid Namespace
GridColumnSummary Overload List

© 2018 Developer Express Inc.

# GridColumnSummary Properties

A data summary item.

The following tables list the members exposed by the GridColumnSummary type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | GridColumnSummary | Overloaded. Initializes a new GridColumnSummary class instance with the specified column's field name. |

### Public Properties

| | Name | Description |
|---|---|---|
| | BindingContext | (Inherited from **Xamarin.Forms.BindableObject**) |
| | DisplayFormat | Gets or sets the pattern used to format the summary value. This is a bindable property. |
| | FieldName | Gets or sets the field name of the column for which a summary should be calculated. This is a bindable property. |
| | IgnoreNullValues | Specifies whether null values should be ignored when a summary is calculated. This is a bindable property. |
| | IsValid | Indicates whether the summary value is calculated. |
| | SummaryValue | Gets the summary value. |
| | Type | Gets or sets the aggregation function type. This is a bindable property. |

### Public Methods

| | Name | Description |
|---|---|---|
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Clone | Creates a copy of the current GridColumnSummary object. |
| | Equals | (Inherited from **System.Object**) |
| S | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetSummaryValueAsync | Gets the summary value asynchronously. |
| | GetType | (Inherited from **System.Object**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Reset | Interrupts the process of summary calculation. |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ToString | (Inherited from **System.Object**) |

**Public Fields**

| | Name | Description |
|---|---|---|
| S | BindingContextProperty | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | DisplayFormatProperty | Identifies the DisplayFormat bindable property. |
| S | FieldNameProperty | Identifies the FieldName bindable property. |
| S | IgnoreNullValuesProperty | Identifies the IgnoreNullValues bindable property. |
| S | TypeProperty | Identifies the Type bindable property. |

**Public Events**

| | Name | Description |
|---|---|---|
| | BindingContextChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanging | (Inherited from **Xamarin.Forms.BindableObject**) |

**See Also**
GridColumnSummary Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# DisplayFormat Property

Gets or sets the pattern used to format the summary value. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public string DisplayFormat { get; set; }
```

### Property Value

A System.String value that specifies the pattern used to format the summary value.

### Remarks

Use the **DisplayFormat** property to format the summary value and add any text to its textual representation. The general pattern is shown below:

*<custom static text>{0:<format specifier<precision specifier>>}<custom static text>*

- *{0}* - the summary value's placeholder;
- *format specifier* - specifies the formating type (currency, scientific, etc);
- *precision specifier* - specifies the number of characters displayed after the decimal point.

For example, to display currency values, set the **DisplayFormat** property to '**Total: {0:c2}**'. If the summary value is **123**, its formatted equivalent will be '**Total: $123.00**'.

When the format string is not specified for a summary, default formats are used. The following table displays the default format of each summary type for a total and group summary.

| Summary Type | Total Summary Format | Group Summary Format |
|---|---|---|
| Average | AVG={0:columnDisplayFormat} | summaryFieldName: AVG={0:columnDisplayFormat} |
| Count | {0} | {0} |
| Min | MIN={0:columnDisplayFormat} | summaryFieldName: MIN={0:columnDisplayFormat} |
| Max | MAX={0:columnDisplayFormat} | summaryFieldName: MAX={0:columnDisplayFormat} |
| Sum | SUM={0:columnDisplayFormat} | summaryFieldName: SUM={0:columnDisplayFormat} |

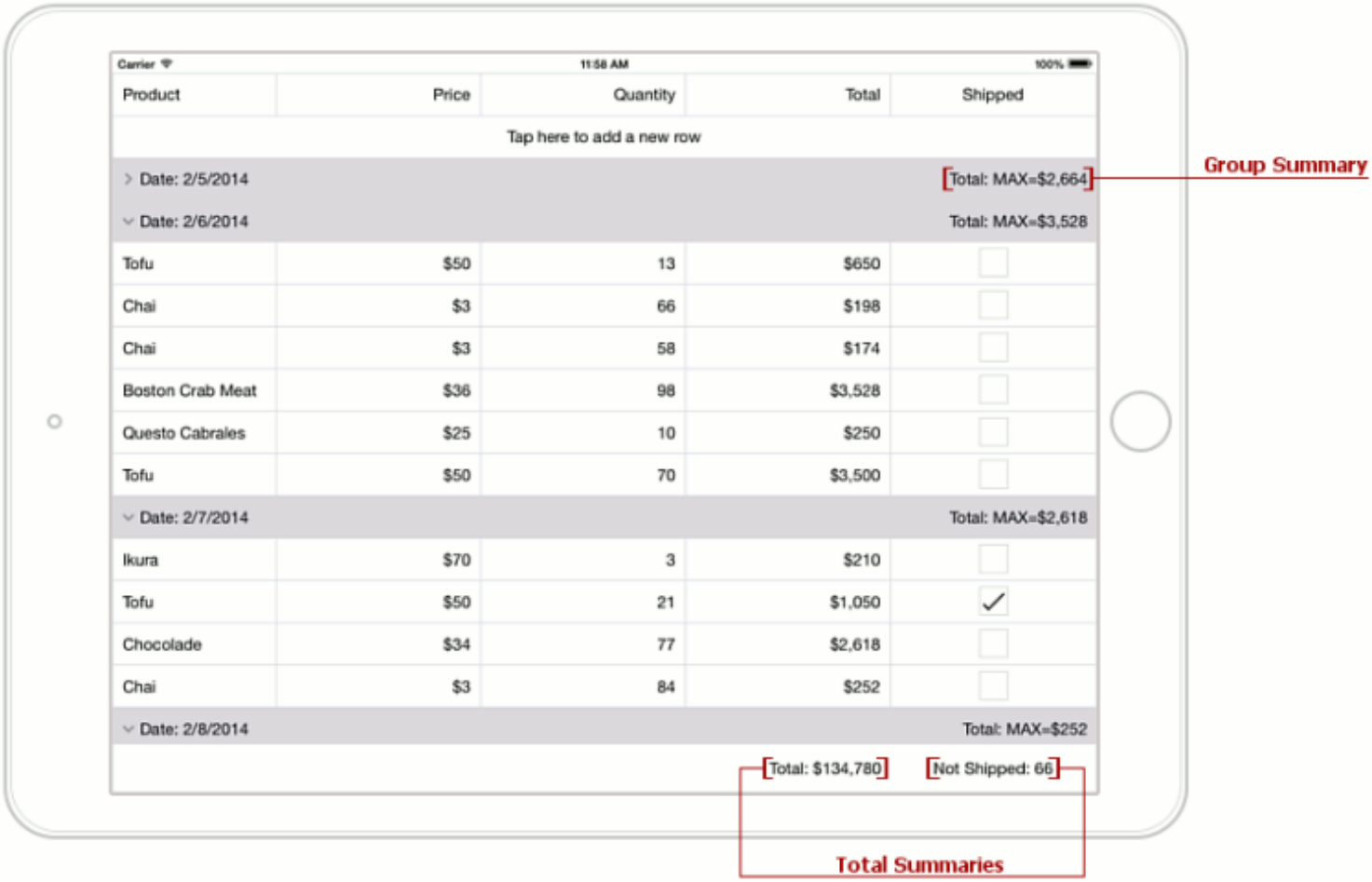
### Example

This example shows how to calculate group and total summaries for grid columns using predefined aggregate functions or a custom rule.

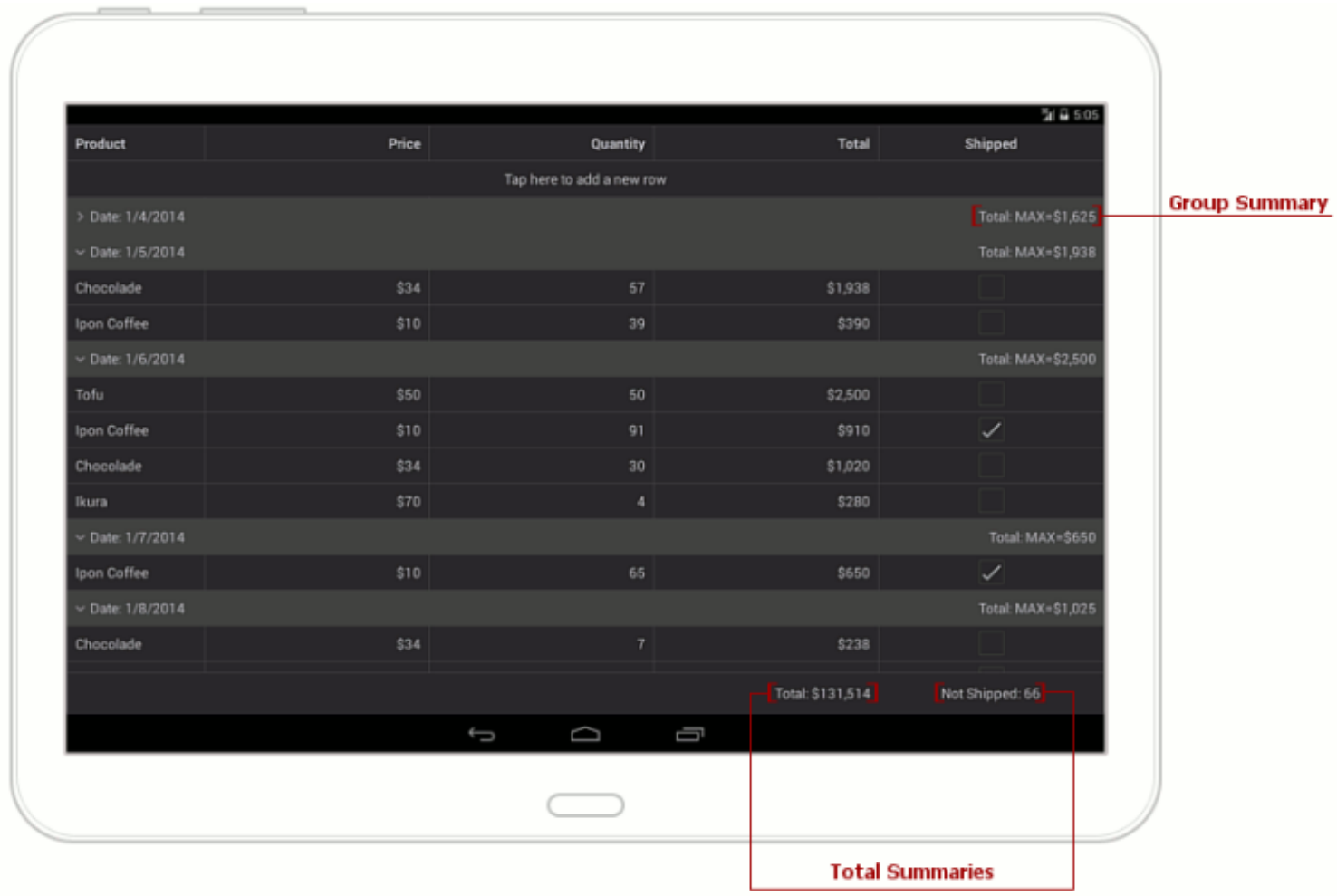
The following data summaries are created.

- Group summary to display the maximum value in a **Total** column for each group of records.
- Total summary to calculate the sum of values by the whole **Total** column.
- Custom total summary to count the number of rows whose value in the **Shipped** column is **false**.

© 2018 Developer Express Inc.




© 2018 Developer Express Inc.




**Xaml**

```
<dxGrid:GridControl x:Name="grid" ItemsSource="{Binding Orders}"
                    NewItemRowVisibility="true"
                    CalculateCustomSummary="OnCalculateCustomSummary">
  <!-- ... -->
  <dxGrid:GridControl.GroupSummaries>
    <dxGrid:GridColumnSummary FieldName="Total" Type="Max"/>
  </dxGrid:GridControl.GroupSummaries>
  <dxGrid:GridControl.TotalSummaries>
    <dxGrid:GridColumnSummary FieldName="Total" Type="Sum" DisplayFormat="Total: {0:C0}"/>
    <dxGrid:GridColumnSummary FieldName="Shipped" Type="Custom" DisplayFormat="Not Shipped: {0}"/>
  </dxGrid:GridControl.TotalSummaries>
  <!-- ... -->
</dxGrid:GridControl>
```

**C#**

```
void OnCalculateCustomSummary(object sender, CustomSummaryEventArgs e) {
    if (e.FieldName.ToString () == "Shipped")
    if (e.IsTotalSummary){
        if (e.SummaryProcess == CustomSummaryProcess.Start) {
            count = 0;
        }
        if (e.SummaryProcess == CustomSummaryProcess.Calculate) {
            if (!(bool)e.FieldValue)
                count++;
            e.TotalValue = count;
        }
    }
}
```

© 2018 Developer Express Inc.

**See Also**
GridColumnSummary Class
GridColumnSummary Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# FieldName Property

Gets or sets the field name of the column for which a summary should be calculated. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public string FieldName { get; set; }
```

### Property Value

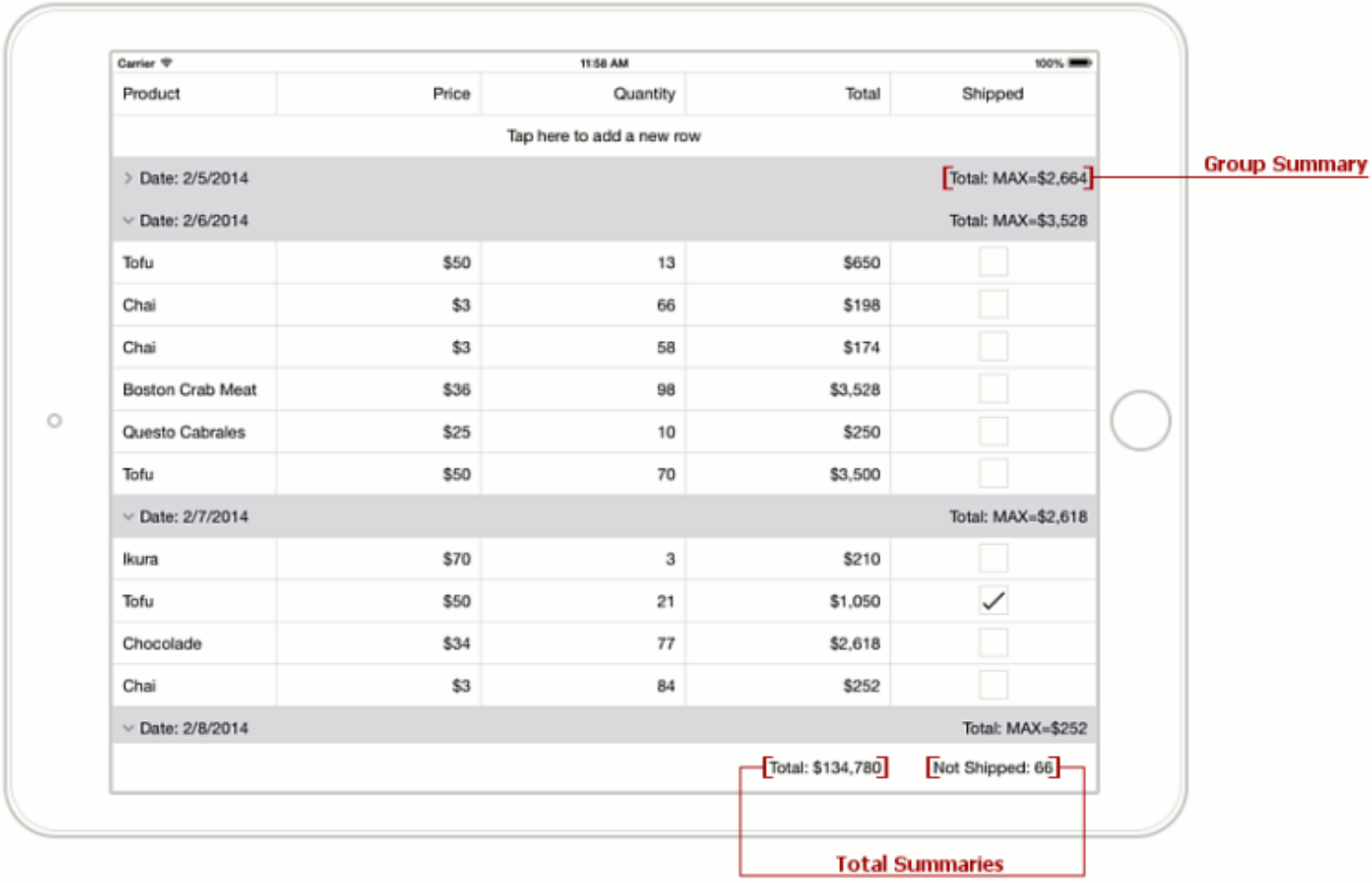
A System.String value that specifies the field name of the column whose values are used for summary calculation.

### Example

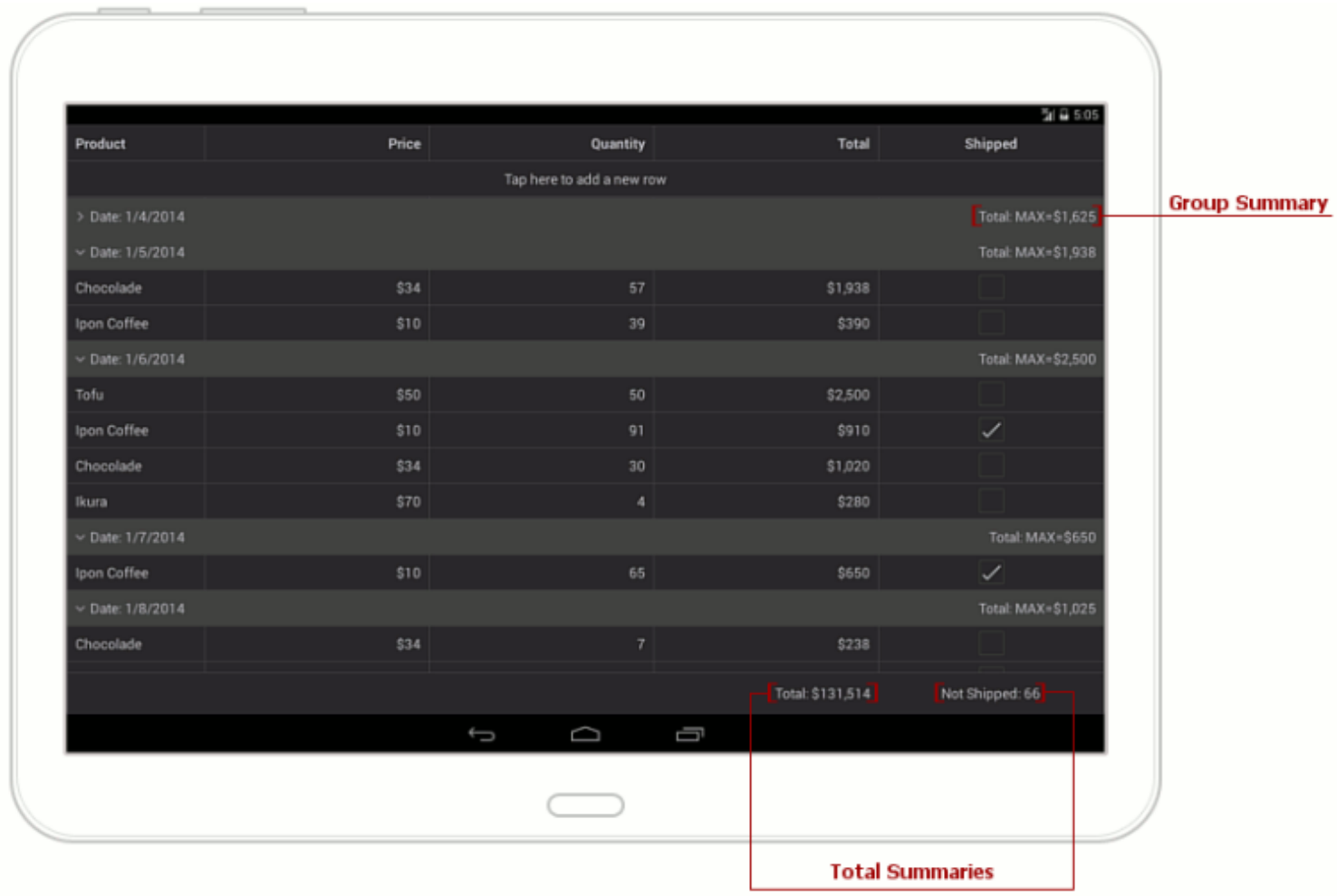
This example shows how to calculate group and total summaries for grid columns using predefined aggregate functions or a custom rule.

The following data summaries are created.

- Group summary to display the maximum value in a **Total** column for each group of records.
- Total summary to calculate the sum of values by the whole **Total** column.
- Custom total summary to count the number of rows whose value in the **Shipped** column is **false**.




© 2018 Developer Express Inc.




**Xaml**

```
<dxGrid:GridControl x:Name="grid" ItemsSource="{Binding Orders}"
                    NewItemRowVisibility="true"
                    CalculateCustomSummary="OnCalculateCustomSummary">
  <!-- ... -->
  <dxGrid:GridControl.GroupSummaries>
    <dxGrid:GridColumnSummary FieldName="Total" Type="Max"/>
  </dxGrid:GridControl.GroupSummaries>
  <dxGrid:GridControl.TotalSummaries>
    <dxGrid:GridColumnSummary FieldName="Total" Type="Sum" DisplayFormat="Total: {0:C0}"/>
    <dxGrid:GridColumnSummary FieldName="Shipped" Type="Custom" DisplayFormat="Not Shipped: {0}"/>
  </dxGrid:GridControl.TotalSummaries>
  <!-- ... -->
</dxGrid:GridControl>
```

**C#**

```
void OnCalculateCustomSummary(object sender, CustomSummaryEventArgs e) {
    if (e.FieldName.ToString () == "Shipped")
    if (e.IsTotalSummary){
        if (e.SummaryProcess == CustomSummaryProcess.Start) {
            count = 0;
        }
        if (e.SummaryProcess == CustomSummaryProcess.Calculate) {
            if (!(bool)e.FieldValue)
                count++;
            e.TotalValue = count;
        }
    }
}
```

© 2018 Developer Express Inc.

**See Also**
GridColumnSummary Class
GridColumnSummary Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# IgnoreNullValues Property

Specifies whether null values should be ignored when a summary is calculated. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public bool IgnoreNullValues { get; set; }
```

### Property Value

**true**, to ignore null values; otherwise, **false**.

### Remarks

Use the **IgnoreNullValues** property to specify whether null values should be ignored when a summary is calculated. If they are included, they will be treated as zeros. This option is useful for functions like SummaryType.Count or SummaryType.Average, because the number of elements counted will depend on it.

For instance, the **Average** summary function will return the following values for the **1**, **null**, **5** set of values, depending on the **IgnoreNullValues** property.

| IgnoreNullValues | Summary Value |
|---|---|
| true | 3 |
| false | 2 |

### See Also

GridColumnSummary Class
GridColumnSummary Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# IsValid Property

Indicates whether the summary value is calculated.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public bool IsValid { get; }
```

### Property Value

**true**, if the summary value is calculated; otherwise, **false**.

### Remarks

You can interrupt the process of summary calculation using the Reset method. In this case, the **IsValid** property returns **false**.

### See Also

GridColumnSummary Class
GridColumnSummary Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# SummaryValue Property

Gets the summary value.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public object SummaryValue { get; }
```

### Property Value

A System.Object specifying the total summary value.

### Remarks

Use one of the following approaches to get a value of the total or group summary.

- **Obtain Total Summary Values**

Total summaries are specified by the GridColumnSummary objects added to the GridControl.TotalSummaries collection. Each total summary is a value calculated for the specified column and displayed under this column in the total summary panel. To obtain a total summary value, use the **SummaryValue** property of the corresponding GridColumnSummary object accessed from the GridControl.TotalSummaries collection.

- **Obtain Group Summary Values**

Group summaries are specified by the GridColumnSummary objects added to the GridControl.GroupSummaries collection. A group summary is calculated for each data group in the grid, and each group summary value is displayed in the corresponding group row. To obtain a group summary value calculated for the specified group, follow the steps below.

1. Call the GridControl.GetGroupInfo method to access an object implementing the IGroupInfo interface and containing information on the specified group.
2. Use the IGroupInfo.Summaries property to get the GridControl.GroupSummaries collection copy whose elements (GridColumnSummary objects) are summaries calculated for the specified data group.
3. Access the required summary object from the IGroupInfo.Summaries collection and use the **SummaryValue** property to get the group summary value calculated for the current group.

### See Also

GridColumnSummary Class
GridColumnSummary Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# Type Property

Gets or sets the aggregation function type. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public SummaryType Type { get; set; }
```

### Property Value

A SummaryType enumeration value that specifies the aggregate function type.

### Remarks

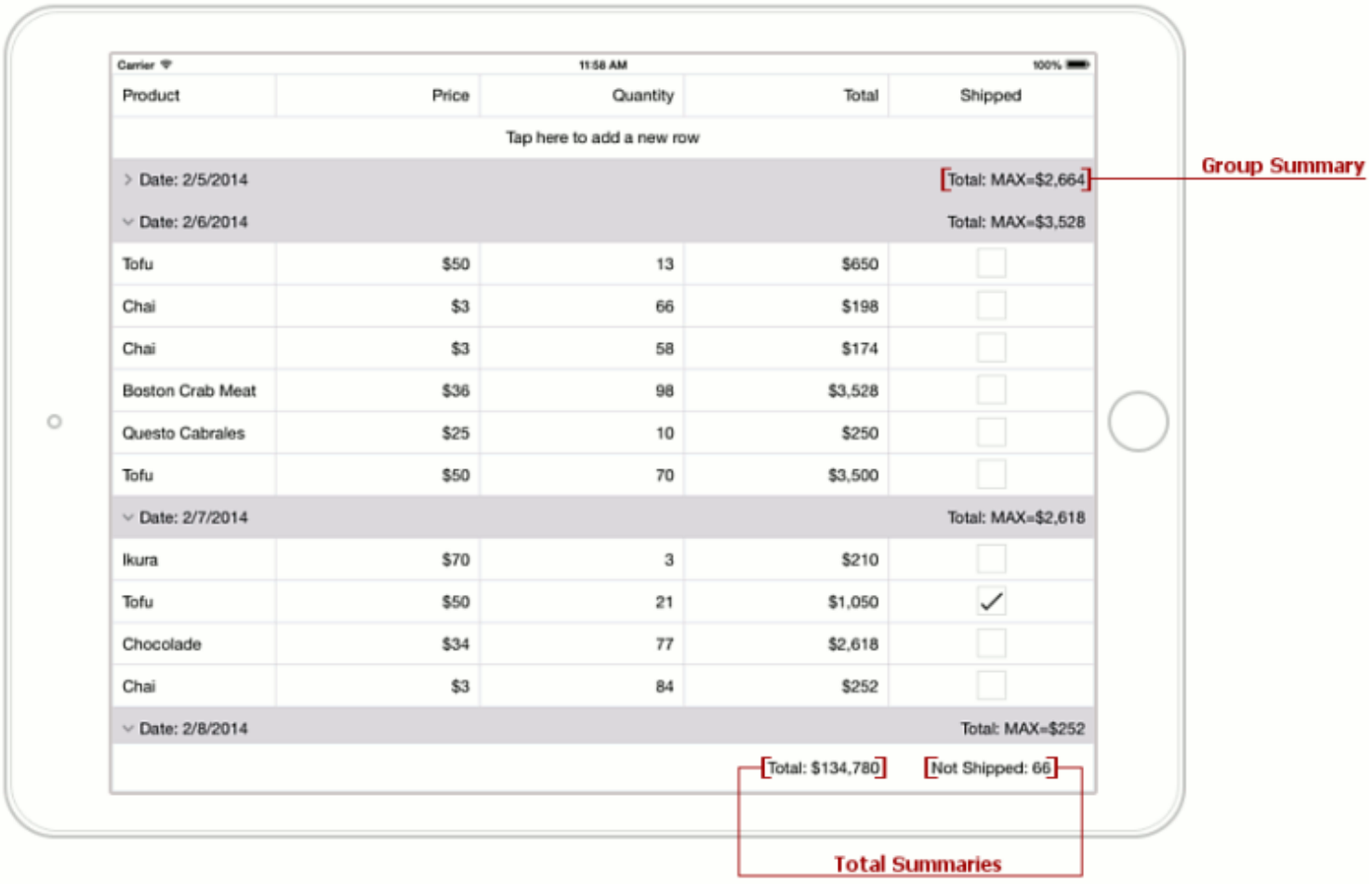
Use the **Type** property to specify which aggregate function should be used to calculate the summary value. There are five predefined aggregate functions for grid data summaries: Sum, Min, Max, Average and Count. To implement your own aggregate function, set the **Type** property to SummaryType.Custom and handle the GridControl.CalculateCustomSummary event.

### Example

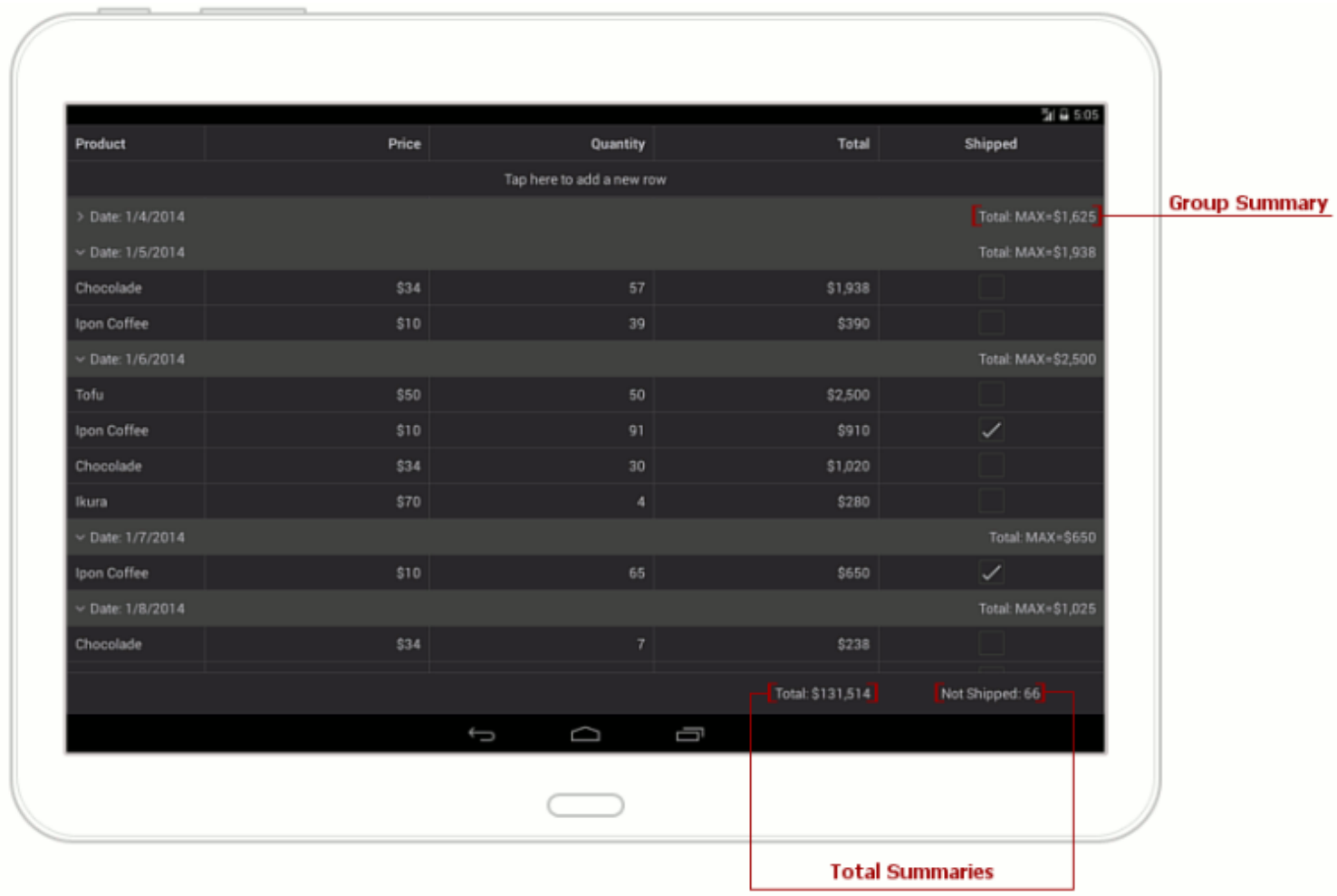
This example shows how to calculate group and total summaries for grid columns using predefined aggregate functions or a custom rule.

The following data summaries are created.

- Group summary to display the maximum value in a **Total** column for each group of records.
- Total summary to calculate the sum of values by the whole **Total** column.
- Custom total summary to count the number of rows whose value in the **Shipped** column is **false**.




© 2018 Developer Express Inc.




**Xaml**

```
<dxGrid:GridControl x:Name="grid" ItemsSource="{Binding Orders}"
                    NewItemRowVisibility="true"
                    CalculateCustomSummary="OnCalculateCustomSummary">
  <!-- ... -->
  <dxGrid:GridControl.GroupSummaries>
    <dxGrid:GridColumnSummary FieldName="Total" Type="Max"/>
  </dxGrid:GridControl.GroupSummaries>
  <dxGrid:GridControl.TotalSummaries>
    <dxGrid:GridColumnSummary FieldName="Total" Type="Sum" DisplayFormat="Total: {0:C0}"/>
    <dxGrid:GridColumnSummary FieldName="Shipped" Type="Custom" DisplayFormat="Not Shipped: {0}"/>
  </dxGrid:GridControl.TotalSummaries>
  <!-- ... -->
</dxGrid:GridControl>
```

**C#**

```
void OnCalculateCustomSummary(object sender, CustomSummaryEventArgs e) {
    if (e.FieldName.ToString () == "Shipped")
    if (e.IsTotalSummary){
        if (e.SummaryProcess == CustomSummaryProcess.Start) {
            count = 0;
        }
        if (e.SummaryProcess == CustomSummaryProcess.Calculate) {
            if (!(bool)e.FieldValue)
                count++;
            e.TotalValue = count;
        }
    }
}
```

© 2018 Developer Express Inc.

**See Also**
GridColumnSummary Class
GridColumnSummary Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# GridColumnSummary Methods

A data summary item.

The following tables list the members exposed by the GridColumnSummary type.

**Public Methods**

| | Name | Description |
|---|---|---|
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Clone | Creates a copy of the current GridColumnSummary object. |
| | Equals | (Inherited from **System.Object**) |
| S | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetSummaryValueAsync | Gets the summary value asynchronously. |
| | GetType | (Inherited from **System.Object**) |
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Reset | Interrupts the process of summary calculation. |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ToString | (Inherited from **System.Object**) |

**See Also**
GridColumnSummary Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# Clone Method

Creates a copy of the current GridColumnSummary object.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax
**C#**
```
public GridColumnSummary Clone()
```

### Return Value

A GridColumnSummary object which is a copy of the current object.

### See Also
GridColumnSummary Class
GridColumnSummary Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# GetSummaryValueAsync Method

Gets the summary value asynchronously.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public Task<Object> GetSummaryValueAsync()
```

### Return Value

A System.Threading.Tasks.Task<System.Object> object whose **Result** property returns the grid's data summary value.

### See Also

GridColumnSummary Class
GridColumnSummary Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# Reset Method

Interrupts the process of summary calculation.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public void Reset()
```

### Remarks

The **Reset** method cancels the summary calculation and sets the IsValid property to **false**.

### See Also

GridColumnSummary Class
GridColumnSummary Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# GridControl Class

The grid control.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

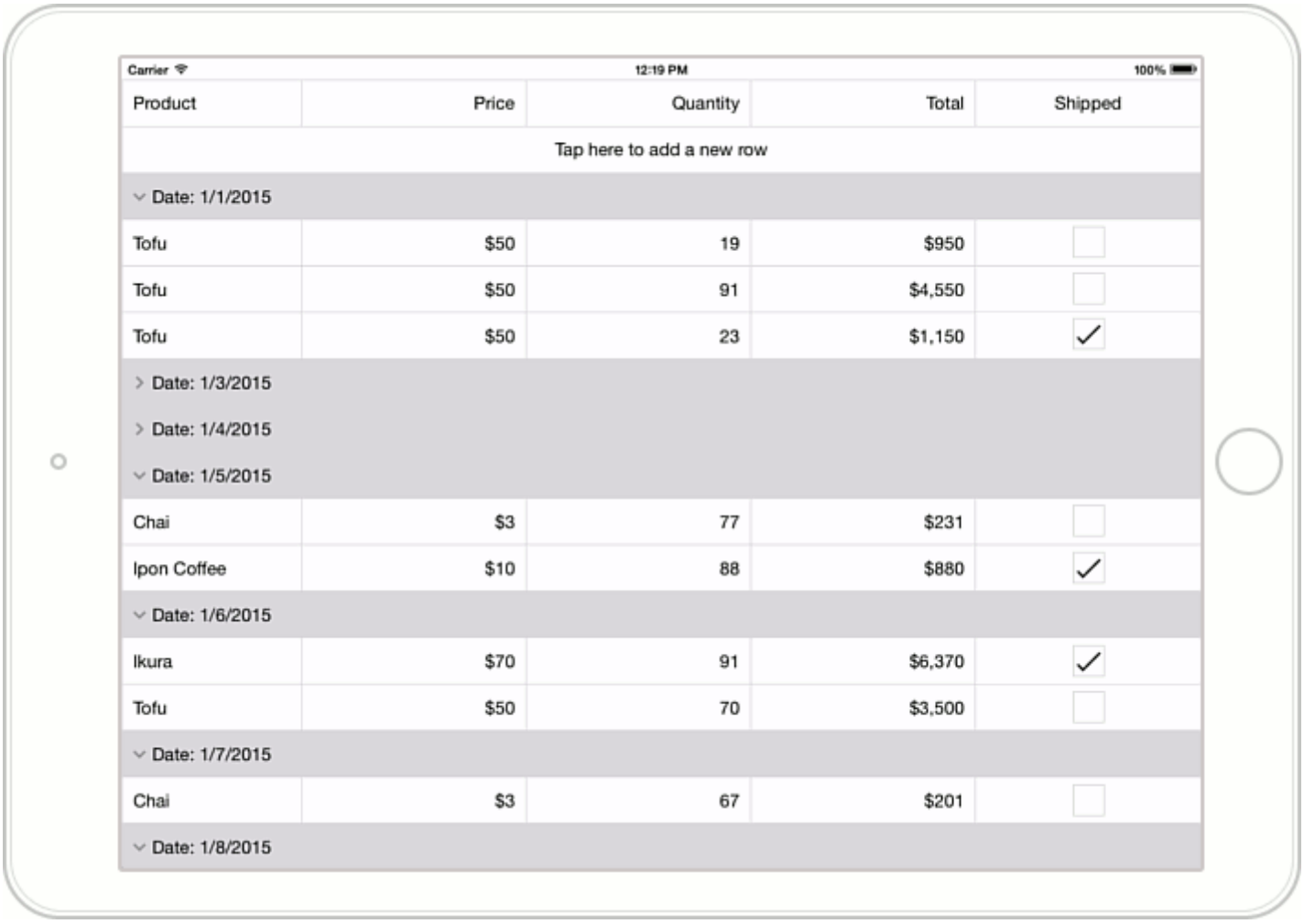
### Syntax

**C#**

```
public class GridControl : GridLayout, ICommandAwareControl<GridCommandId>,
  IFormatConditionCollectionOwner,
  IGestureRecognizerDelegate,
  IServiceProvider,
  ISelectedRowHandleProvider,
  IHitTestAccess,
  IDisposable,
  IThemeChangingHandler,
  IAndroidThemeChanger,
  IXtraSerializableLayoutEx,
  IHorizontalScrollingData,
  ICustomCellTextProvider
```

### Remarks

**GridControl** is a data bound control that provides a set of data shaping and manipulation features, including data sorting, grouping, filtering and summary calculation.



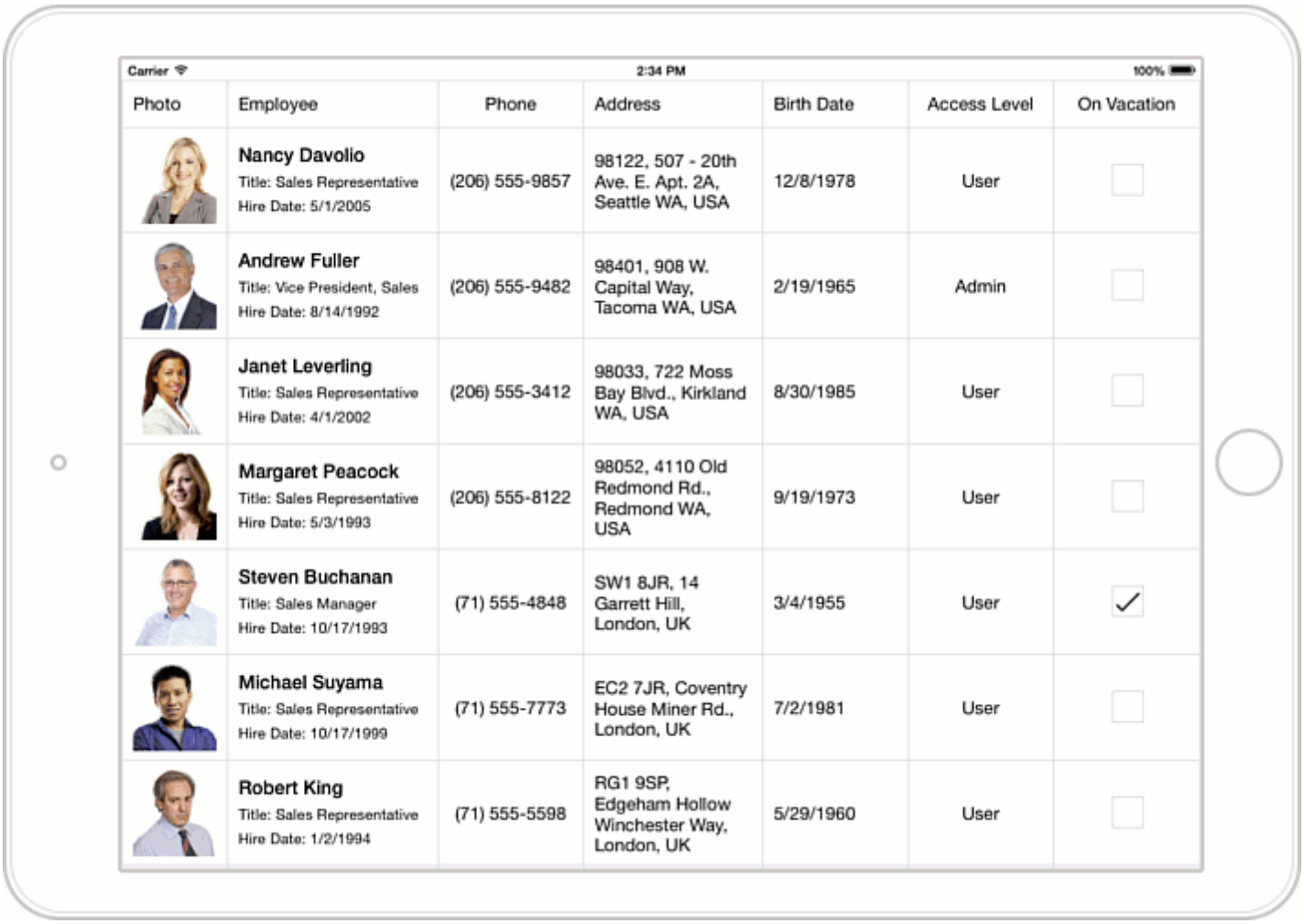

To get started with **GridControl**, see the Getting Started tutorial.

© 2018 Developer Express Inc.

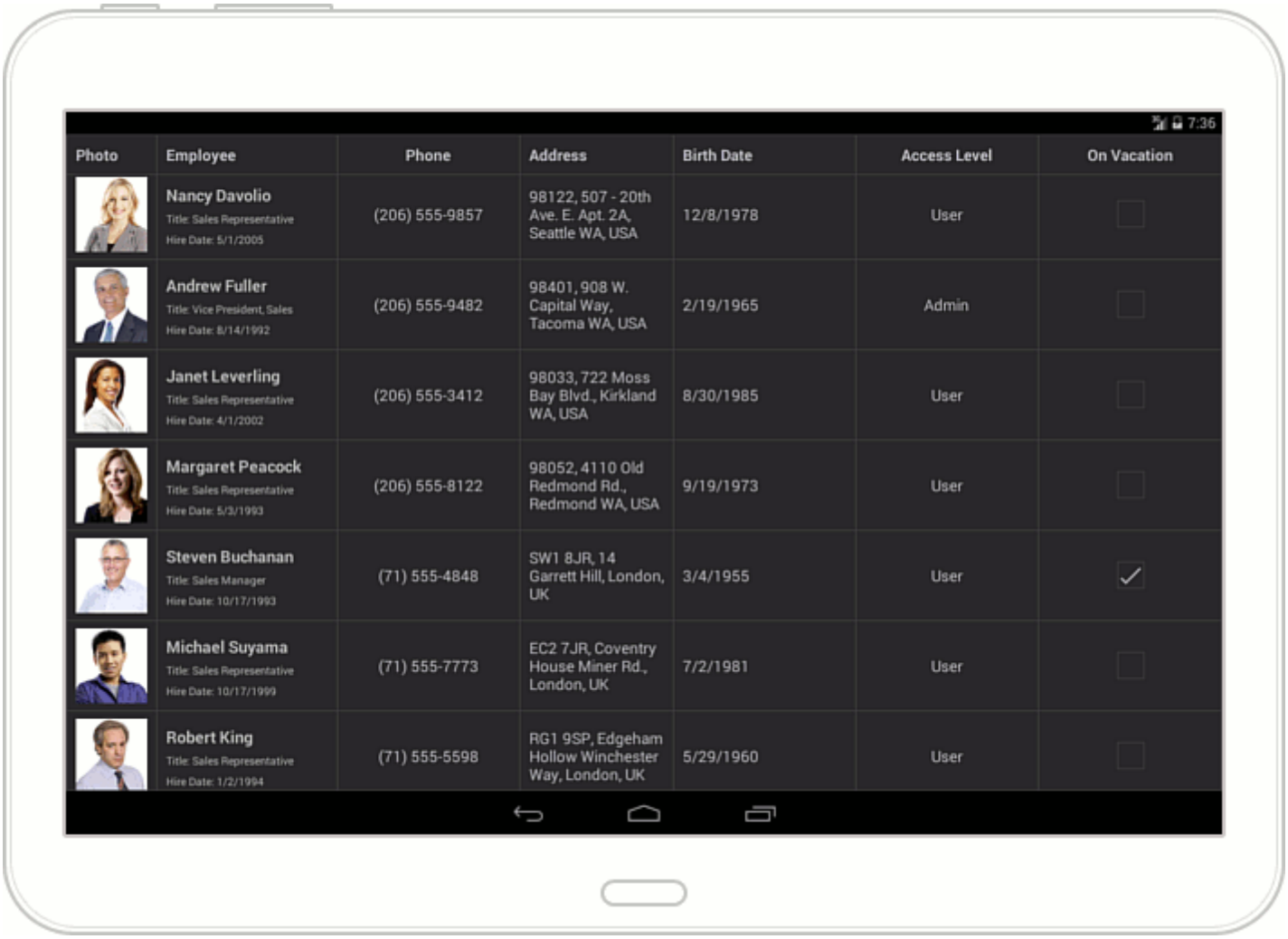
**Example**

| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T208848. | |

This example shows how to add columns to GridControl to display and edit data of different types. The grid is bound to a collection of **Employee** objects. Each **Employee** object contains an employee's photo (image), name, position, phone, address (strings), hire and birth dates (DateTime values), a value that specifies an employee's access level, and a Boolean value indicating whether an employee is on vacation.




© 2018 Developer Express Inc.




Grid columns are stored in the Columns collection. An individual column is specified by a GridColumn descendant, which is appropriate to the type of data contained in this column. In this example, to display information on employees and allow end-users to edit it, the following columns are created in the grid.

| Grid Column | Description |
|---|---|
| Photo<br>ImageColumn | This column displays employee photos which are images added to a project as embedded resources. |
| Employee<br>TemplateColumn | A cell in this column displays three employee properties: Name, Position and HireDate. A cell appearance is defined via a template. Each cell contains three Xamarin.Forms.Label elements within a Xamarin.Forms.Grid. Each Label element is bound to a property of the **Employee** class.<br><br>A cell template data context is specified by the CellData object. Its CellData.Value property provides access to a value of a data field assigned to the column's GridColumn.FieldName property. In this example, a column cell displays not only this field value but also values of two more fields. So, it is required to specify the whole data row as a binding source for the template. To do this, set the BindingContext property of the Grid to CellData.Source. Note that BindingContext specified for the Grid is inherited by all its children. This means that all Label elements have the same BindingContext as the Grid, and you can specify |

© 2018 Developer Express Inc.

| | |
|---|---|
| | their simple bindings to properties of that object (Employee).<br><br>Use the GridColumn.AllowSort, GridColumn.AllowGroup and GridColumn.IsReadOnly properties to prevent end-users from sorting and grouping data in the grid by this column and disable data editing. |
| Phone<br>NumberColumn | The employee's **Phone** property is of the string type. The keyboard allowing text input is automatically displayed when an end-user activates this column cell to edit an employee's phone. |
| Address<br>TextColumn | This column is bound to the employee's **Address** string property. The keyboard allowing text input is automatically displayed when an end-user activates this column cell to edit an employee's address. |
| Birth Date<br>DateColumn | This column displays and allows editing employee birth days. The GridColumn.DisplayFormat property specifies the format of displaying dates. |
| Access Level<br>PickerColumn | The employee's **Access** property is of the **AccessLevel** type which is an enumeration with values **Admin** and **User**. To allow editing an employee's access level by selecting one of two enumeration values, this property is bound to the grid's picker column. |
| On Vacation<br>SwitchColumn | The employee's **OnVacation** property is of the Boolean type, so it is bound to the grid's switch column which allows editing a cell value by selecting between two states. |

**C#**

```
 (Employee.cs)
public enum AccessLevel {
    Admin,
    User
}
public class Employee : ModelObject {
    string name;
    public string Name {
        get { return name; }
        set {
            name = value;
            if (Photo == null)
                Photo = ImageSource.FromResource (value.Replace (" ", String.Empty));
        }
    }
    public Employee(string name) {
        this.Name = name;
    }
    public ImageSource Photo { get; set; }
    public DateTime BirthDate { get; set; }
    public DateTime HireDate { get; set; }
    public string Position { get; set; }
    public string Address { get; set; }
    public string Phone { get; set; }
    public AccessLevel Access { get; set; }
    public bool OnVacation { get; set; }
}
```

**C#**

```
 (EmployeesRepository.cs)
public abstract class EmployeesRepository {
    readonly ObservableCollection<Employee> employees;
    public EmployeesRepository() {
        this.employees = new ObservableCollection<Employee>();
    }
    public ObservableCollection<Employee> Employees {
        get { return employees; }
    }
}
public class TestEmployeesRepository : EmployeesRepository {
    public TestEmployeesRepository() : base() {
        GenerateCustomers();
    }
    void GenerateCustomers() {
        Employees.Add(
            new Employee("Nancy Davolio") {
                BirthDate = new DateTime(1978, 12, 8),
                HireDate = new DateTime(2005, 5, 1),
                Position = "Sales Representative",
                Address = "98122, 507 - 20th Ave. E. Apt. 2A, Seattle WA, USA",
                Phone = "(206) 555-9857",
                Access = AccessLevel.User,
                OnVacation = false
            }
        );
        Employees.Add(
            new Employee("Andrew Fuller") {
                BirthDate = new DateTime(1965, 2, 19),
                HireDate = new DateTime(1992, 8, 14),
                Position = "Vice President, Sales",
                Address = "98401, 908 W. Capital Way, Tacoma WA, USA",
                Phone = "(206) 555-9482",
                Access = AccessLevel.Admin,
                OnVacation = false
            }
        );
        Employees.Add(
            new Employee("Janet Leverling") {
                BirthDate = new DateTime(1985, 8, 30),
                HireDate = new DateTime(2002, 4, 1),
                Position = "Sales Representative",
                Address = "98033, 722 Moss Bay Blvd., Kirkland WA, USA",
                Phone = "(206) 555-3412",
                Access = AccessLevel.User,
                OnVacation = false
            }
        );
        Employees.Add(
            new Employee("Margaret Peacock") {
                BirthDate = new DateTime(1973, 9, 19),
                HireDate = new DateTime(1993, 5, 3),
                Position = "Sales Representative",
                Address = "98052, 4110 Old Redmond Rd., Redmond WA, USA",
                Phone = "(206) 555-8122",
                Access = AccessLevel.User,
                OnVacation = false
            }
        );
        Employees.Add(
            new Employee("Steven Buchanan") {
                BirthDate = new DateTime(1955, 3, 4),
                HireDate = new DateTime(1993, 10, 17),
                Position = "Sales Manager",
                Address = "SW1 8JR, 14 Garrett Hill, London, UK",
```

© 2018 Developer Express Inc.

```
                Phone = "(71) 555-4848",
                Access = AccessLevel.User,
                OnVacation = true
            }
        );
        Employees.Add(
            new Employee("Michael Suyama") {
                BirthDate = new DateTime(1981, 7, 2),
                HireDate = new DateTime(1999, 10, 17),
                Position = "Sales Representative",
                Address = "EC2 7JR, Coventry House Miner Rd., London, UK",
                Phone = "(71) 555-7773",
                Access = AccessLevel.User,
                OnVacation = false
            }
        );
        Employees.Add(
            new Employee("Robert King") {
                BirthDate = new DateTime(1960, 5, 29),
                HireDate = new DateTime(1994, 1, 2),
                Position = "Sales Representative",
                Address = "RG1 9SP, Edgeham Hollow Winchester Way, London, UK",
                Phone = "(71) 555-5598",
                Access = AccessLevel.User,
                OnVacation = false
            }
        );
        Employees.Add(
            new Employee("Laura Callahan") {
                BirthDate = new DateTime(1985, 1, 9),
                HireDate = new DateTime(2004, 3, 5),
                Position = "Inside Sales Coordinator",
                Address = "98105, 4726 - 11th Ave. N.E., Seattle WA, USA",
                Phone = "(206) 555-1189",
                Access = AccessLevel.User,
                OnVacation = true
            }
        );
        Employees.Add(
            new Employee("Anne Dodsworth") {
                BirthDate = new DateTime(1980, 1, 27),
                HireDate = new DateTime(2004, 11, 15),
                Position = "Sales Representative",
                Address = "WG2 7LT, 7 Houndstooth Rd., London, UK",
                Phone = "(71) 555-4444",
                Access = AccessLevel.User,
                OnVacation = false
            }
        );
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl ItemsSource="{Binding Employees}" RowHeight="100">
    <dxGrid:GridControl.Columns>
        <dxGrid:ImageColumn FieldName="Photo" Width="100" />
        <dxGrid:TemplateColumn FieldName="Name" Caption="Employee" Width="
                               IsReadOnly="true" AllowGroup="False" AllowS
          <dxGrid:TemplateColumn.DisplayTemplate>
            <DataTemplate>
              <Grid BindingContext="{Binding Source}" Padding="0,15,0,15">
                <Grid.RowDefinitions>
                  <RowDefinition Height="Auto" />
                  <RowDefinition Height="Auto" />
                  <RowDefinition Height="Auto" />
                </Grid.RowDefinitions>
                <Label Text="{Binding Name}" Font="Bold, 18" Grid.Row="0"
                <Label Text="{Binding Position, StringFormat = 'Title: {0}
                <Label Text="{Binding HireDate, StringFormat = 'Hire Date:
              </Grid>
            </DataTemplate>
          </dxGrid:TemplateColumn.DisplayTemplate>
        </dxGrid:TemplateColumn>
        <dxGrid:NumberColumn FieldName="Phone" ContentAlignment="Center"/>
        <dxGrid:TextColumn FieldName="Address" Width="170"/>
        <dxGrid:DateColumn FieldName="BirthDate" DisplayFormat="d"/>
        <dxGrid:PickerColumn FieldName="Access" Caption="Access Level" Con
        <dxGrid:SwitchColumn FieldName="OnVacation"/>
    </dxGrid:GridControl.Columns>
</dxGrid:GridControl>
```

## Inheritance Hierarchy

System.Object
  Xamarin.Forms.BindableObject
    Xamarin.Forms.Element
      Xamarin.Forms.VisualElement
        Xamarin.Forms.View
          Xamarin.Forms.Layout
            Xamarin.Forms.Layout<Xamarin.Forms.View>
              Xamarin.Forms.Grid
                DevExpress.Mobile.DataGrid.Internal.GridLayout
                  **GridControl**

## See Also

GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# GridControl Members

The grid control.

The following tables list the members exposed by the GridControl type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | GridControl | Initializes a new instance of the GridControl class with default settings. |

### Public Properties

| | Name | Description |
|---|---|---|
| | ActualFilterString | Returns the string that specifies a filter currently used in the grid. |
| | AllowDeleteRows | Gets or sets whether end-users are allowed to delete data rows. This is a bindable property. |
| | AllowEditRows | Gets or sets whether end-users are allowed to change cell values in the grid. This is a bindable property. |
| | AllowGroup | Gets or sets whether end-users are allowed to group data in the grid. This is a bindable property. |
| | AllowGroupCollapse | Gets or sets whether end-users are allowed to collapse and expand group rows. This is a bindable property. |
| | AllowHorizontalScrollingVirtualization | Gets or sets a value indicating whether virtualization is enabled for horizontal scrolling. |
| | AllowResizeColumns | Gets or sets whether end-users are allowed to change column widths. This is a bindable property. |
| | AllowSort | Gets or sets whether end-users are allowed to sort data in the grid. This is a bindable property. |
| | AnchorX | (Inherited from **Xamarin.Forms.VisualElement**) |
| | AnchorY | (Inherited from **Xamarin.Forms.VisualElement**) |
| | AutoFilterPanelHeight | Gets or sets the auto-filter panel height. This is a bindable property. |
| | AutoFilterPanelVisibility | Gets or sets whether the auto-filter panel is displayed in the grid. This is a bindable property. |
| | AutoGenerateColumnsMode | Gets or sets how columns should be automatically created for fields of the underlying data source, depending on whether the grid contains any columns. |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | BackgroundColor | (Inherited from **Xamarin.Forms.VisualElement**) |
| | Behaviors | (Inherited from **Xamarin.Forms.VisualElement**) |
| | BindingContext | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Bounds | (Inherited from **Xamarin.Forms.VisualElement**) |
| | Children | (Inherited from **Xamarin.Forms.Grid**) |
| | ClassId | (Inherited from **Xamarin.Forms.Element**) |
| | ColumnDefinitions | (Inherited from **Xamarin.Forms.Grid**) |
| | ColumnHeadersHeight | Gets or sets the height of column headers in the grid. This is a bindable property. |
| | ColumnHeadersVisibility | Gets or sets whether column headers are shown in the grid. This is a bindable property. |
| | Columns | Provides access to the grid's collection of columns. |
| | ColumnsAutoWidth | Gets or sets whether column widths are automatically changed, so that the total column width matches the grid's width. This is a bindable property. |
| | ColumnSpacing | (Inherited from **Xamarin.Forms.Grid**) |
| | EditFormContent | Gets or sets the template used to define the visual presentation of the grid's row edit form. |
| | FilterExpression | Gets or sets the grid filter expression. |
| | FilterPanelHeight | Gets or sets the filter panel height. This is a bindable property. |
| | FilterPanelVisibility | Gets or sets when the filter panel is shown at the bottom of the grid. This is a bindable property. |
| | FilterString | Gets or sets the grid filter expression string. This is a bindable property. |
| | FormatConditions | Contains format conditions applied to the grid. |
| | GestureRecognizers | (Inherited from **Xamarin.Forms.View**) |
| | GroupsInitiallyExpanded | Gets or sets whether groups should appear expanded or collapsed immediately after a grouping is applied in the grid. |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | GroupSummaries | Provides access to group summary items. |
| | Height | (Inherited from **Xamarin.Forms.VisualElement**) |
| | HeightRequest | (Inherited from **Xamarin.Forms.VisualElement**) |
| | HighlightMenuTargetElements | Specifies whether the grid should highlight a row and column for which an end-user invokes a context menu. This is a bindable property. |
| | HorizontalOptions | (Inherited from **Xamarin.Forms.View**) |
| | HorizontalScrollOffset | Gets or sets a horizontal scrolling offset. |
| | Id | (Inherited from **Xamarin.Forms.Element**) |
| | InputTransparent | (Inherited from **Xamarin.Forms.VisualElement**) |
| | IsClippedToBounds | (Inherited from **Xamarin.Forms.Layout**) |
| | IsColumnChooserEnabled | Gets or sets whether the column chooser is available to end-users. This is a bindable property. |
| | IsColumnMenuEnabled | Gets or sets whether a context menu is shown when an end-user touches and holds a column header. This is a bindable property. |
| | IsEnabled | (Inherited from **Xamarin.Forms.VisualElement**) |
| | IsFocused | (Inherited from **Xamarin.Forms.VisualElement**) |
| | IsGrouped | Gets whether the grid's data is grouped. |
| | IsGroupRowMenuEnabled | Gets or sets whether a context menu is shown when an end-user touches and holds a group row. This is a bindable property. |
| | IsLoadedInVisualTree | For internal use. |
| | IsLoadMoreEnabled | Gets or sets whether the load-more functionality is enabled in the grid. This is a bindable property. |
| | IsPullToRefreshEnabled | Gets or sets whether the pull-to-refresh functionality is enabled in the grid. This is a bindable property. |
| | IsReadOnly | Gets or sets whether the grid is read-only. This is a bindable property. |
| | IsRowCellMenuEnabled | Gets or sets whether a context menu is shown when an end-user touches |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | | and holds a data row cell. This is a bindable property. |
| | IsTotalSummaryMenuEnabled | Gets or sets whether a context menu is shown when an end-user touches and holds a total summary panel. This is a bindable property. |
| | IsVisible | (Inherited from **Xamarin.Forms.VisualElement**) |
| | ItemsSource | Gets or sets the grid's data source. This is a bindable property. |
| | LeftSwipeButtons | Provides access to objects containing information on buttons shown on the left side of a data row when an end-user swipes left to right over a data row. |
| | LoadMoreCommand | Gets or sets the command executed when an end-user scrolls to the bottom of the grid to load new data items. This is a bindable property. |
| | MinimumHeightRequest | (Inherited from **Xamarin.Forms.VisualElement**) |
| | MinimumWidthRequest | (Inherited from **Xamarin.Forms.VisualElement**) |
| | Navigation | (Inherited from **Xamarin.Forms.VisualElement**) |
| | NewItemRowVisibility | Gets or sets whether the **New Item Row** is shown in the grid to allow end-users to enter new data records. This is a bindable property. |
| | Opacity | (Inherited from **Xamarin.Forms.VisualElement**) |
| | OptionsExportCsv | Gets or sets how grid data is exported to CSV format. |
| | OptionsExportXls | Gets or sets how grid data is exported to XLS format. |
| | OptionsExportXlsx | Gets or sets how grid data is exported to XLSX format. |
| | Padding | (Inherited from **Xamarin.Forms.Layout**) |
| | Parent | (Inherited from **Xamarin.Forms.Element**) |
| | ParentView | (Inherited from **Xamarin.Forms.Element**) |
| | PullToRefreshCommand | Gets or sets the command executed when an end-user pulls the grid down to refresh data. This is a bindable property. |
| | Resources | (Inherited from **Xamarin.Forms.VisualElement**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | RightSwipeButtons | Provides access to objects containing information on buttons shown on the right side of a data row when an end-user swipes right to left over a data row. |
| | Rotation | (Inherited from **Xamarin.Forms.VisualElement**) |
| | RotationX | (Inherited from **Xamarin.Forms.VisualElement**) |
| | RotationY | (Inherited from **Xamarin.Forms.VisualElement**) |
| | RowCount | Gets the number of data rows in the grid. |
| | RowDefinitions | (Inherited from **Xamarin.Forms.Grid**) |
| | RowEditMode | Specifies whether the grid invokes an in-place editor or edit form when an end-user starts to edit a data cell or add a new row. |
| | RowHeight | Gets or sets the height of rows in the grid. This is a bindable property. |
| | RowSpacing | (Inherited from **Xamarin.Forms.Grid**) |
| | RowTapCommand | Gets or sets the command executed when an end-user taps a row within the grid. This is a bindable property. |
| | Scale | (Inherited from **Xamarin.Forms.VisualElement**) |
| | SelectedDataObject | Returns an object that specifies a data source record to which a data row currently selected in the grid corresponds. This is a bindable property. |
| | SelectedRowHandle | Gets or sets the selected row handle. This is a bindable property. |
| | SortMode | Specifies whether the grid allows data sorting by multiple columns or by a single column only. This is a bindable property. |
| | Style | (Inherited from **Xamarin.Forms.VisualElement**) |
| | StyleId | (Inherited from **Xamarin.Forms.Element**) |
| | SwipeButtonCommand | Gets or sets the command executed when a swipe button is tapped. This is a bindable property. |
| | TotalSummaries | Provides access to a collection of total summary items. |
| | TotalSummaryHeight | Gets or sets the total summary panel height. This is a bindable property. |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | TotalSummaryVisibility | Gets or sets whether the total summary panel is displayed. This is a bindable property. |
| | TranslationX | (Inherited from **Xamarin.Forms.VisualElement**) |
| | TranslationY | (Inherited from **Xamarin.Forms.VisualElement**) |
| | Triggers | (Inherited from **Xamarin.Forms.VisualElement**) |
| | VerticalOptions | (Inherited from **Xamarin.Forms.View**) |
| | VisibleColumns | Gets the collection of visible columns. |
| | Width | (Inherited from **Xamarin.Forms.VisualElement**) |
| | WidthRequest | (Inherited from **Xamarin.Forms.VisualElement**) |
| | X | (Inherited from **Xamarin.Forms.VisualElement**) |
| | Y | (Inherited from **Xamarin.Forms.VisualElement**) |

**Public Methods**

| | Name | Description |
|---|---|---|
| | BatchBegin | (Inherited from **Xamarin.Forms.VisualElement**) |
| | BatchCommit | (Inherited from **Xamarin.Forms.VisualElement**) |
| | ClearFilter | Clears the filter applied to data in the grid. |
| | ClearGrouping | Ungroups data in the grid. |
| | ClearSorting | Clears the sorting applied to the grid. |
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | CloseEditor | Closes a data row editor (an in-place editor or edit form, depending on the edit mode set in the grid). |
| | CollapseAllGroups | Collapses all group rows. |
| | CollapseGroupRow | Collapses the specified group row. |
| | DeleteRow | Deletes the specified data row. |
| | Dispose | Releases all resources used by the GridControl. |
| | Equals | (Inherited from **System.Object**) |
| S | Equals | (Inherited from **System.Object**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | ExpandAllGroups | Expands all group rows. |
| | ExpandGroupRow | Expands the specified group row. |
| | ExportToExcel | Exports data from the grid to the specified stream in the XLS, XLSX or CSV format. |
| | FindRow | Returns the handle of the first row that matches the specified criteria. |
| | FindRowByValue | Overloaded. Searches for the value in the column and returns the handle of the corresponding row. |
| | Focus | (Inherited from **Xamarin.Forms.VisualElement**) |
| | ForceLayout | (Inherited from **Xamarin.Forms.Layout**) |
| | GetCellDisplayText | Overloaded. Returns the text displayed within the specified cell. |
| | GetCellValue | Overloaded. Returns the value of the specified data cell in the grid. |
| S | GetColumn | (Inherited from **Xamarin.Forms.Grid**) |
| S | GetColumnSpan | (Inherited from **Xamarin.Forms.Grid**) |
| | GetGroupInfo | Provides access to information on the specified data group. |
| | GetGroupRowValue | Returns the value of the specified group row. |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetRow | Overloaded. Gets an object that provides access to the data of a row specified by its handle. |
| S | GetRow | (Inherited from **Xamarin.Forms.Grid**) |
| | GetRowHandle | Returns the handle of the row which corresponds to the specified record in the data source. |
| S | GetRowSpan | (Inherited from **Xamarin.Forms.Grid**) |
| | GetService | Gets the service object of the specified type. |
| | GetService<T> | Gets the specified service. |
| | GetSizeRequest | (Inherited from **Xamarin.Forms.Layout**) |
| | GetSourceRowIndex | Returns the index of the data source record to which the specified row handle corresponds. |
| | GetType | (Inherited from **System.Object**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | GroupBy | Overloaded. Groups data in the grid by the specified column. |
| | HideColumnChooser | Overloaded. Hides the column chooser. |
| | IsGroupCollapsed | Indicates whether the specified group row is collapsed. |
| | IsGroupRow | Indicates whether the specified handle corresponds to a group row. |
| | Layout | (Inherited from **Xamarin.Forms.VisualElement**) |
| S | LayoutChildIntoBoundingRegion | (Inherited from **Xamarin.Forms.Layout**) |
| | LoadInVisualTree | For internal use. |
| | LowerChild | (Inherited from **Xamarin.Forms.Layout**) |
| | OpenEditor | Invokes an editor (an in-place editor or edit form, depending on the edit mode set in the grid) for a data row containing the specified cell. |
| | RaiseChild | (Inherited from **Xamarin.Forms.Layout**) |
| | Redraw | Forces the control to immediately redraw itself. |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | RefreshData | Updates the grid data. |
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | RemoveDynamicResource | (Inherited from **Xamarin.Forms.Element**) |
| | RestoreLayoutFromStream | Restores a grid's layout from the specified stream. |
| | RestoreLayoutFromXml | Restores a grid's layout from the specified XML string. |
| | SaveLayoutToStream | Saves a grid's layout to the specified stream. |
| | SaveLayoutToXml | Saves a gird's layout to an XML string. |
| | ScrollToRow | Scrolls the grid to make the specified row visible. |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetCellValue | Overloaded. Assigns a value to the specified cell. |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | SetColumn | (Inherited from **Xamarin.Forms.Grid**) |
| | SetColumnSpan | (Inherited from **Xamarin.Forms.Grid**) |
| | SetDynamicResource | (Inherited from **Xamarin.Forms.Element**) |
| | SetRow | (Inherited from **Xamarin.Forms.Grid**) |
| | SetRowSpan | (Inherited from **Xamarin.Forms.Grid**) |
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ShowColumnChooser | Invokes the column chooser. |
| | SortBy | Overloaded. Sorts data by the values of the column bound to the specified data field in the ascending order. |
| | ToString | (Inherited from **System.Object**) |
| | Unfocus | (Inherited from **Xamarin.Forms.VisualElement**) |

**Public Fields**

| | Name | Description |
|---|---|---|
| | ActualFilterStringProperty | Identifies the ActualFilterString bindable property. |
| | AllowDeleteRowsProperty | Identifies the AllowDeleteRows bindable property. |
| | AllowEditRowsProperty | Identifies the AllowEditRows bindable property. |
| | AllowGroupCollapseProperty | Identifies the AllowGroupCollapse bindable property. |
| | AllowGroupProperty | Identifies the AllowGroup bindable property. |
| | AllowResizeColumnsProperty | Identifies the AllowResizeColumns bindable property. |
| | AllowSortProperty | Identifies the AllowSort bindable property. |
| | AnchorXProperty | (Inherited from **Xamarin.Forms.VisualElement**) |
| | AnchorYProperty | (Inherited from **Xamarin.Forms.VisualElement**) |
| | AutoFilterPanelHeightProperty | Identifies the AutoFilterPanelHeight bindable property. |
| | AutoFilterPanelVisibilityProperty | Identifies the AutoFilterPanelVisibility bindable property. |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| S | AutoGenerateColumnsModeProperty | Identifies the AutoGenerateColumnsMode bindable property. |
| S | BackgroundColorProperty | (Inherited from **Xamarin.Forms.VisualElement**) |
| S | BehaviorsProperty | (Inherited from **Xamarin.Forms.VisualElement**) |
| S | BindingContextProperty | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ClassIdProperty | (Inherited from **Xamarin.Forms.Element**) |
| S | ColumnDefinitionsProperty | (Inherited from **Xamarin.Forms.Grid**) |
| S | ColumnHeadersHeightProperty | Identifies the ColumnHeadersHeight bindable property. |
| S | ColumnHeadersVisibilityProperty | Identifies the ColumnHeadersVisibility bindable property. |
| S | ColumnProperty | (Inherited from **Xamarin.Forms.Grid**) |
| S | ColumnsAutoWidthProperty | Identifies the ColumnsAutoWidth bindable property. |
| S | ColumnSpacingProperty | (Inherited from **Xamarin.Forms.Grid**) |
| S | ColumnSpanProperty | (Inherited from **Xamarin.Forms.Grid**) |
| S | DefaultColumnHeaderHeight | The default column header height. Returns **44**. |
| S | DefaultColumnWidth | The default column width. Returns **120**. |
| S | DefaultRowHeight | The default row height. Returns **44**. |
| S | FilterPanelHeightProperty | Identifies the FilterPanelHeight bindable property. |
| S | FilterPanelVisibilityProperty | Identifies the FilterPanelVisibility bindable property. |
| S | FilterStringProperty | Identifies the FilterString bindable property. |
| S | GroupsInitiallyExpandedProperty | Identifies the GroupsInitiallyExpanded bindable property. |
| S | HeightProperty | (Inherited from **Xamarin.Forms.VisualElement**) |
| S | HeightRequestProperty | (Inherited from **Xamarin.Forms.VisualElement**) |
| S | HighlightMenuTargetElementsProperty | Identifies the HighlightMenuTargetElements bindable property. |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | HorizontalOptionsProperty | (Inherited from **Xamarin.Forms.View**) |
| | InputTransparentProperty | (Inherited from **Xamarin.Forms.VisualElement**) |
| | InvalidRowHandle | An invalid row handle. This value is returned when an attempt to obtain a particular row fails. |
| | IsClippedToBoundsProperty | (Inherited from **Xamarin.Forms.Layout**) |
| | IsColumnChooserEnabledProperty | Identifies the IsColumnChooserEnabled bindable property. |
| | IsColumnMenuEnabledProperty | Identifies the IsColumnMenuEnabled bindable property. |
| | IsEnabledProperty | (Inherited from **Xamarin.Forms.VisualElement**) |
| | IsFocusedProperty | (Inherited from **Xamarin.Forms.VisualElement**) |
| | IsGroupRowMenuEnabledProperty | Identifies the IsGroupRowMenuEnabled bindable property. |
| | IsLoadMoreEnabledProperty | Identifies the IsLoadMoreEnabled bindable property. |
| | IsPullToRefreshEnabledProperty | Identifies the IsPullToRefreshEnabled bindable property. |
| | IsReadOnlyProperty | Identifies the IsReadOnly bindable property. |
| | IsRowCellMenuEnabledProperty | Identifies the IsRowCellMenuEnabled bindable property. |
| | IsTotalSummaryMenuEnabledProperty | Identifies the IsTotalSummaryMenuEnabled bindable property. |
| | IsVisibleProperty | (Inherited from **Xamarin.Forms.VisualElement**) |
| | ItemsSourceProperty | Identifies the ItemsSource bindable property. |
| | LoadMoreCommandProperty | Identifies the LoadMoreCommand bindable property. |
| | MinimumHeightRequestProperty | (Inherited from **Xamarin.Forms.VisualElement**) |
| | MinimumWidthRequestProperty | (Inherited from **Xamarin.Forms.VisualElement**) |
| | NavigationProperty | (Inherited from **Xamarin.Forms.VisualElement**) |
| | NewItemRowVisibilityProperty | Identifies the NewItemRowVisibility bindable property. |
| | OpacityProperty | (Inherited from **Xamarin.Forms.VisualElement**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| S | OptionsExportCsvProperty | Identifies the OptionsExportCsv bindable property. |
| S | OptionsExportXlsProperty | Identifies the OptionsExportXls bindable property. |
| S | OptionsExportXlsxProperty | Identifies the OptionsExportXlsx bindable property. |
| S | PaddingProperty | (Inherited from **Xamarin.Forms.Layout**) |
| S | PullToRefreshCommandProperty | Identifies the PullToRefreshCommand bindable property. |
| S | RotationProperty | (Inherited from **Xamarin.Forms.VisualElement**) |
| S | RotationXProperty | (Inherited from **Xamarin.Forms.VisualElement**) |
| S | RotationYProperty | (Inherited from **Xamarin.Forms.VisualElement**) |
| S | RowDefinitionsProperty | (Inherited from **Xamarin.Forms.Grid**) |
| S | RowEditModeProperty | Identifies the RowEditMode bindable property. |
| S | RowHeightProperty | Identifies the RowHeight bindable property. |
| S | RowProperty | (Inherited from **Xamarin.Forms.Grid**) |
| S | RowSpacingProperty | (Inherited from **Xamarin.Forms.Grid**) |
| S | RowSpanProperty | (Inherited from **Xamarin.Forms.Grid**) |
| S | RowTapCommandProperty | Identifies the RowTapCommand bindable property. |
| S | ScaleProperty | (Inherited from **Xamarin.Forms.VisualElement**) |
| S | SelectedDataObjectProperty | Identifies the SelectedDataObject bindable property. |
| S | SelectedRowHandleProperty | Identifies the SelectedRowHandle bindable property. |
| S | SortModeProperty | Identifies the SortMode bindable property. |
| S | StyleProperty | (Inherited from **Xamarin.Forms.VisualElement**) |
| S | SwipeButtonCommandProperty | Identifies the SwipeButtonCommand bindable property. |
| S | TotalSummaryHeightProperty | Identifies the TotalSummaryHeight bindable property. |
| S | TotalSummaryVisibilityProperty | Identifies the TotalSummaryVisibility bindable property. |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | TranslationXProperty | (Inherited from **Xamarin.Forms.VisualElement**) |
| | TranslationYProperty | (Inherited from **Xamarin.Forms.VisualElement**) |
| | TriggersProperty | (Inherited from **Xamarin.Forms.VisualElement**) |
| | VerticalOptionsProperty | (Inherited from **Xamarin.Forms.View**) |
| | WidthProperty | (Inherited from **Xamarin.Forms.VisualElement**) |
| | WidthRequestProperty | (Inherited from **Xamarin.Forms.VisualElement**) |
| | XProperty | (Inherited from **Xamarin.Forms.VisualElement**) |
| | YProperty | (Inherited from **Xamarin.Forms.VisualElement**) |

**Public Events**

| | Name | Description |
|---|---|---|
| | AutoGeneratingColumn | Occurs each time an individual column is auto-generated. |
| | BindingContextChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | CalculateCustomSummary | Allows you to calculate summary values manually. |
| | ChildAdded | (Inherited from **Xamarin.Forms.Element**) |
| | ChildRemoved | (Inherited from **Xamarin.Forms.Element**) |
| | ChildrenReordered | (Inherited from **Xamarin.Forms.VisualElement**) |
| | CustomizeCell | Allows you to customize the appearance of cells in the grid. |
| | CustomizeCellDisplayText | Enables custom display text to be provided for any data cell. |
| | CustomUnboundColumnData | Enables you to populate unbound columns with data, and save changes made by end-users in unbound columns. |
| | DescendantAdded | (Inherited from **Xamarin.Forms.Element**) |
| | DescendantRemoved | (Inherited from **Xamarin.Forms.Element**) |
| | EndRowEdit | Occurs after a row editor (an in-place editor or edit form, depending on the edit mode set in the grid) is closed and |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | | modified cell values are applied or canceled. |
| | Focused | (Inherited from **Xamarin.Forms.VisualElement**) |
| | GroupRowCollapsed | Occurs after a group row has been collapsed. |
| | GroupRowCollapsing | Occurs before a group row is collapsed. |
| | GroupRowExpanded | Occurs after a group row has been expanded. |
| | GroupRowExpanding | Occurs before a group row is expanded. |
| | HorizontalScrollOffsetChanged | Occurs after the HorizontalScrollOffset property has been changed. |
| | InitNewRow | Enables you to initialize newly added rows. |
| | LayoutChanged | (Inherited from **Xamarin.Forms.Layout**) |
| | LoadMore | Occurs when an end-user scrolls to the bottom of the grid to load new data items. |
| | MeasureInvalidated | (Inherited from **Xamarin.Forms.VisualElement**) |
| | PopupMenuCustomization | Enables you to customize the grid's popup menus. |
| | PropertyChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanging | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PullToRefresh | Occurs when an end-user pulls the grid down to refresh its content. |
| | RowTap | Occurs when a data row is tapped. |
| | SelectionChanged | Occurs after the selection has been changed. |
| | SizeChanged | (Inherited from **Xamarin.Forms.VisualElement**) |
| | SwipeButtonClick | Occurs when a swipe button is tapped. |
| | SwipeButtonShowing | Occurs before a swipe button is shown when an end-user swipes left to right, or right to left, over a data row. |
| | Unfocused | (Inherited from **Xamarin.Forms.VisualElement**) |

**See Also**
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# GridControl Constructor

Initializes a new instance of the GridControl class with default settings.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**
**C#**

```
public GridControl()
```

**See Also**
GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# GridControl Properties

The grid control.

The following tables list the members exposed by the GridControl type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | GridControl | Initializes a new instance of the GridControl class with default settings. |

### Public Properties

| | Name | Description |
|---|---|---|
| | ActualFilterString | Returns the string that specifies a filter currently used in the grid. |
| | AllowDeleteRows | Gets or sets whether end-users are allowed to delete data rows. This is a bindable property. |
| | AllowEditRows | Gets or sets whether end-users are allowed to change cell values in the grid. This is a bindable property. |
| | AllowGroup | Gets or sets whether end-users are allowed to group data in the grid. This is a bindable property. |
| | AllowGroupCollapse | Gets or sets whether end-users are allowed to collapse and expand group rows. This is a bindable property. |
| | AllowHorizontalScrollingVirtualization | Gets or sets a value indicating whether virtualization is enabled for horizontal scrolling. |
| | AllowResizeColumns | Gets or sets whether end-users are allowed to change column widths. This is a bindable property. |
| | AllowSort | Gets or sets whether end-users are allowed to sort data in the grid. This is a bindable property. |
| | AnchorX | (Inherited from **Xamarin.Forms.VisualElement**) |
| | AnchorY | (Inherited from **Xamarin.Forms.VisualElement**) |
| | AutoFilterPanelHeight | Gets or sets the auto-filter panel height. This is a bindable property. |
| | AutoFilterPanelVisibility | Gets or sets whether the auto-filter panel is displayed in the grid. This is a bindable property. |
| | AutoGenerateColumnsMode | Gets or sets how columns should be automatically created for fields of the underlying data source, depending on whether the grid contains any columns. |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | BackgroundColor | (Inherited from **Xamarin.Forms.VisualElement**) |
| | Behaviors | (Inherited from **Xamarin.Forms.VisualElement**) |
| | BindingContext | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Bounds | (Inherited from **Xamarin.Forms.VisualElement**) |
| | Children | (Inherited from **Xamarin.Forms.Grid**) |
| | ClassId | (Inherited from **Xamarin.Forms.Element**) |
| | ColumnDefinitions | (Inherited from **Xamarin.Forms.Grid**) |
| | ColumnHeadersHeight | Gets or sets the height of column headers in the grid. This is a bindable property. |
| | ColumnHeadersVisibility | Gets or sets whether column headers are shown in the grid. This is a bindable property. |
| | Columns | Provides access to the grid's collection of columns. |
| | ColumnsAutoWidth | Gets or sets whether column widths are automatically changed, so that the total column width matches the grid's width. This is a bindable property. |
| | ColumnSpacing | (Inherited from **Xamarin.Forms.Grid**) |
| | EditFormContent | Gets or sets the template used to define the visual presentation of the grid's row edit form. |
| | FilterExpression | Gets or sets the grid filter expression. |
| | FilterPanelHeight | Gets or sets the filter panel height. This is a bindable property. |
| | FilterPanelVisibility | Gets or sets when the filter panel is shown at the bottom of the grid. This is a bindable property. |
| | FilterString | Gets or sets the grid filter expression string. This is a bindable property. |
| | FormatConditions | Contains format conditions applied to the grid. |
| | GestureRecognizers | (Inherited from **Xamarin.Forms.View**) |
| | GroupsInitiallyExpanded | Gets or sets whether groups should appear expanded or collapsed immediately after a grouping is applied in the grid. |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | GroupSummaries | Provides access to group summary items. |
| | Height | (Inherited from **Xamarin.Forms.VisualElement**) |
| | HeightRequest | (Inherited from **Xamarin.Forms.VisualElement**) |
| | HighlightMenuTargetElements | Specifies whether the grid should highlight a row and column for which an end-user invokes a context menu. This is a bindable property. |
| | HorizontalOptions | (Inherited from **Xamarin.Forms.View**) |
| | HorizontalScrollOffset | Gets or sets a horizontal scrolling offset. |
| | Id | (Inherited from **Xamarin.Forms.Element**) |
| | InputTransparent | (Inherited from **Xamarin.Forms.VisualElement**) |
| | IsClippedToBounds | (Inherited from **Xamarin.Forms.Layout**) |
| | IsColumnChooserEnabled | Gets or sets whether the column chooser is available to end-users. This is a bindable property. |
| | IsColumnMenuEnabled | Gets or sets whether a context menu is shown when an end-user touches and holds a column header. This is a bindable property. |
| | IsEnabled | (Inherited from **Xamarin.Forms.VisualElement**) |
| | IsFocused | (Inherited from **Xamarin.Forms.VisualElement**) |
| | IsGrouped | Gets whether the grid's data is grouped. |
| | IsGroupRowMenuEnabled | Gets or sets whether a context menu is shown when an end-user touches and holds a group row. This is a bindable property. |
| | IsLoadedInVisualTree | For internal use. |
| | IsLoadMoreEnabled | Gets or sets whether the load-more functionality is enabled in the grid. This is a bindable property. |
| | IsPullToRefreshEnabled | Gets or sets whether the pull-to-refresh functionality is enabled in the grid. This is a bindable property. |
| | IsReadOnly | Gets or sets whether the grid is read-only. This is a bindable property. |
| | IsRowCellMenuEnabled | Gets or sets whether a context menu is shown when an end-user touches |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | | and holds a data row cell. This is a bindable property. |
| | IsTotalSummaryMenuEnabled | Gets or sets whether a context menu is shown when an end-user touches and holds a total summary panel. This is a bindable property. |
| | IsVisible | (Inherited from **Xamarin.Forms.VisualElement**) |
| | ItemsSource | Gets or sets the grid's data source. This is a bindable property. |
| | LeftSwipeButtons | Provides access to objects containing information on buttons shown on the left side of a data row when an end-user swipes left to right over a data row. |
| | LoadMoreCommand | Gets or sets the command executed when an end-user scrolls to the bottom of the grid to load new data items. This is a bindable property. |
| | MinimumHeightRequest | (Inherited from **Xamarin.Forms.VisualElement**) |
| | MinimumWidthRequest | (Inherited from **Xamarin.Forms.VisualElement**) |
| | Navigation | (Inherited from **Xamarin.Forms.VisualElement**) |
| | NewItemRowVisibility | Gets or sets whether the **New Item Row** is shown in the grid to allow end-users to enter new data records. This is a bindable property. |
| | Opacity | (Inherited from **Xamarin.Forms.VisualElement**) |
| | OptionsExportCsv | Gets or sets how grid data is exported to CSV format. |
| | OptionsExportXls | Gets or sets how grid data is exported to XLS format. |
| | OptionsExportXlsx | Gets or sets how grid data is exported to XLSX format. |
| | Padding | (Inherited from **Xamarin.Forms.Layout**) |
| | Parent | (Inherited from **Xamarin.Forms.Element**) |
| | ParentView | (Inherited from **Xamarin.Forms.Element**) |
| | PullToRefreshCommand | Gets or sets the command executed when an end-user pulls the grid down to refresh data. This is a bindable property. |
| | Resources | (Inherited from **Xamarin.Forms.VisualElement**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | RightSwipeButtons | Provides access to objects containing information on buttons shown on the right side of a data row when an end-user swipes right to left over a data row. |
| | Rotation | (Inherited from **Xamarin.Forms.VisualElement**) |
| | RotationX | (Inherited from **Xamarin.Forms.VisualElement**) |
| | RotationY | (Inherited from **Xamarin.Forms.VisualElement**) |
| | RowCount | Gets the number of data rows in the grid. |
| | RowDefinitions | (Inherited from **Xamarin.Forms.Grid**) |
| | RowEditMode | Specifies whether the grid invokes an in-place editor or edit form when an end-user starts to edit a data cell or add a new row. |
| | RowHeight | Gets or sets the height of rows in the grid. This is a bindable property. |
| | RowSpacing | (Inherited from **Xamarin.Forms.Grid**) |
| | RowTapCommand | Gets or sets the command executed when an end-user taps a row within the grid. This is a bindable property. |
| | Scale | (Inherited from **Xamarin.Forms.VisualElement**) |
| | SelectedDataObject | Returns an object that specifies a data source record to which a data row currently selected in the grid corresponds. This is a bindable property. |
| | SelectedRowHandle | Gets or sets the selected row handle. This is a bindable property. |
| | SortMode | Specifies whether the grid allows data sorting by multiple columns or by a single column only. This is a bindable property. |
| | Style | (Inherited from **Xamarin.Forms.VisualElement**) |
| | StyleId | (Inherited from **Xamarin.Forms.Element**) |
| | SwipeButtonCommand | Gets or sets the command executed when a swipe button is tapped. This is a bindable property. |
| | TotalSummaries | Provides access to a collection of total summary items. |
| | TotalSummaryHeight | Gets or sets the total summary panel height. This is a bindable property. |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | TotalSummaryVisibility | Gets or sets whether the total summary panel is displayed. This is a bindable property. |
| | TranslationX | (Inherited from **Xamarin.Forms.VisualElement**) |
| | TranslationY | (Inherited from **Xamarin.Forms.VisualElement**) |
| | Triggers | (Inherited from **Xamarin.Forms.VisualElement**) |
| | VerticalOptions | (Inherited from **Xamarin.Forms.View**) |
| | VisibleColumns | Gets the collection of visible columns. |
| | Width | (Inherited from **Xamarin.Forms.VisualElement**) |
| | WidthRequest | (Inherited from **Xamarin.Forms.VisualElement**) |
| | X | (Inherited from **Xamarin.Forms.VisualElement**) |
| | Y | (Inherited from **Xamarin.Forms.VisualElement**) |

**Public Methods**

| | Name | Description |
|---|---|---|
| | BatchBegin | (Inherited from **Xamarin.Forms.VisualElement**) |
| | BatchCommit | (Inherited from **Xamarin.Forms.VisualElement**) |
| | ClearFilter | Clears the filter applied to data in the grid. |
| | ClearGrouping | Ungroups data in the grid. |
| | ClearSorting | Clears the sorting applied to the grid. |
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | CloseEditor | Closes a data row editor (an in-place editor or edit form, depending on the edit mode set in the grid). |
| | CollapseAllGroups | Collapses all group rows. |
| | CollapseGroupRow | Collapses the specified group row. |
| | DeleteRow | Deletes the specified data row. |
| | Dispose | Releases all resources used by the GridControl. |
| | Equals | (Inherited from **System.Object**) |
| S | Equals | (Inherited from **System.Object**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | ExpandAllGroups | Expands all group rows. |
| | ExpandGroupRow | Expands the specified group row. |
| | ExportToExcel | Exports data from the grid to the specified stream in the XLS, XLSX or CSV format. |
| | FindRow | Returns the handle of the first row that matches the specified criteria. |
| | FindRowByValue | Overloaded. Searches for the value in the column and returns the handle of the corresponding row. |
| | Focus | (Inherited from **Xamarin.Forms.VisualElement**) |
| | ForceLayout | (Inherited from **Xamarin.Forms.Layout**) |
| | GetCellDisplayText | Overloaded. Returns the text displayed within the specified cell. |
| | GetCellValue | Overloaded. Returns the value of the specified data cell in the grid. |
| S | GetColumn | (Inherited from **Xamarin.Forms.Grid**) |
| S | GetColumnSpan | (Inherited from **Xamarin.Forms.Grid**) |
| | GetGroupInfo | Provides access to information on the specified data group. |
| | GetGroupRowValue | Returns the value of the specified group row. |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetRow | Overloaded. Gets an object that provides access to the data of a row specified by its handle. |
| S | GetRow | (Inherited from **Xamarin.Forms.Grid**) |
| | GetRowHandle | Returns the handle of the row which corresponds to the specified record in the data source. |
| S | GetRowSpan | (Inherited from **Xamarin.Forms.Grid**) |
| | GetService | Gets the service object of the specified type. |
| | GetService<T> | Gets the specified service. |
| | GetSizeRequest | (Inherited from **Xamarin.Forms.Layout**) |
| | GetSourceRowIndex | Returns the index of the data source record to which the specified row handle corresponds. |
| | GetType | (Inherited from **System.Object**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | GroupBy | Overloaded. Groups data in the grid by the specified column. |
| | HideColumnChooser | Overloaded. Hides the column chooser. |
| | IsGroupCollapsed | Indicates whether the specified group row is collapsed. |
| | IsGroupRow | Indicates whether the specified handle corresponds to a group row. |
| | Layout | (Inherited from **Xamarin.Forms.VisualElement**) |
| S | LayoutChildIntoBoundingRegion | (Inherited from **Xamarin.Forms.Layout**) |
| | LoadInVisualTree | For internal use. |
| | LowerChild | (Inherited from **Xamarin.Forms.Layout**) |
| | OpenEditor | Invokes an editor (an in-place editor or edit form, depending on the edit mode set in the grid) for a data row containing the specified cell. |
| | RaiseChild | (Inherited from **Xamarin.Forms.Layout**) |
| | Redraw | Forces the control to immediately redraw itself. |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | RefreshData | Updates the grid data. |
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | RemoveDynamicResource | (Inherited from **Xamarin.Forms.Element**) |
| | RestoreLayoutFromStream | Restores a grid's layout from the specified stream. |
| | RestoreLayoutFromXml | Restores a grid's layout from the specified XML string. |
| | SaveLayoutToStream | Saves a grid's layout to the specified stream. |
| | SaveLayoutToXml | Saves a gird's layout to an XML string. |
| | ScrollToRow | Scrolls the grid to make the specified row visible. |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetCellValue | Overloaded. Assigns a value to the specified cell. |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | SetColumn | (Inherited from **Xamarin.Forms.Grid**) |
| | SetColumnSpan | (Inherited from **Xamarin.Forms.Grid**) |
| | SetDynamicResource | (Inherited from **Xamarin.Forms.Element**) |
| | SetRow | (Inherited from **Xamarin.Forms.Grid**) |
| | SetRowSpan | (Inherited from **Xamarin.Forms.Grid**) |
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ShowColumnChooser | Invokes the column chooser. |
| | SortBy | Overloaded. Sorts data by the values of the column bound to the specified data field in the ascending order. |
| | ToString | (Inherited from **System.Object**) |
| | Unfocus | (Inherited from **Xamarin.Forms.VisualElement**) |

**Public Fields**

| | Name | Description |
|---|---|---|
| | ActualFilterStringProperty | Identifies the ActualFilterString bindable property. |
| | AllowDeleteRowsProperty | Identifies the AllowDeleteRows bindable property. |
| | AllowEditRowsProperty | Identifies the AllowEditRows bindable property. |
| | AllowGroupCollapseProperty | Identifies the AllowGroupCollapse bindable property. |
| | AllowGroupProperty | Identifies the AllowGroup bindable property. |
| | AllowResizeColumnsProperty | Identifies the AllowResizeColumns bindable property. |
| | AllowSortProperty | Identifies the AllowSort bindable property. |
| | AnchorXProperty | (Inherited from **Xamarin.Forms.VisualElement**) |
| | AnchorYProperty | (Inherited from **Xamarin.Forms.VisualElement**) |
| | AutoFilterPanelHeightProperty | Identifies the AutoFilterPanelHeight bindable property. |
| | AutoFilterPanelVisibilityProperty | Identifies the AutoFilterPanelVisibility bindable property. |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| S | AutoGenerateColumnsModeProperty | Identifies the AutoGenerateColumnsMode bindable property. |
| S | BackgroundColorProperty | (Inherited from **Xamarin.Forms.VisualElement**) |
| S | BehaviorsProperty | (Inherited from **Xamarin.Forms.VisualElement**) |
| S | BindingContextProperty | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ClassIdProperty | (Inherited from **Xamarin.Forms.Element**) |
| S | ColumnDefinitionsProperty | (Inherited from **Xamarin.Forms.Grid**) |
| S | ColumnHeadersHeightProperty | Identifies the ColumnHeadersHeight bindable property. |
| S | ColumnHeadersVisibilityProperty | Identifies the ColumnHeadersVisibility bindable property. |
| S | ColumnProperty | (Inherited from **Xamarin.Forms.Grid**) |
| S | ColumnsAutoWidthProperty | Identifies the ColumnsAutoWidth bindable property. |
| S | ColumnSpacingProperty | (Inherited from **Xamarin.Forms.Grid**) |
| S | ColumnSpanProperty | (Inherited from **Xamarin.Forms.Grid**) |
| S | DefaultColumnHeaderHeight | The default column header height. Returns **44**. |
| S | DefaultColumnWidth | The default column width. Returns **120**. |
| S | DefaultRowHeight | The default row height. Returns **44**. |
| S | FilterPanelHeightProperty | Identifies the FilterPanelHeight bindable property. |
| S | FilterPanelVisibilityProperty | Identifies the FilterPanelVisibility bindable property. |
| S | FilterStringProperty | Identifies the FilterString bindable property. |
| S | GroupsInitiallyExpandedProperty | Identifies the GroupsInitiallyExpanded bindable property. |
| S | HeightProperty | (Inherited from **Xamarin.Forms.VisualElement**) |
| S | HeightRequestProperty | (Inherited from **Xamarin.Forms.VisualElement**) |
| S | HighlightMenuTargetElementsProperty | Identifies the HighlightMenuTargetElements bindable property. |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | HorizontalOptionsProperty | (Inherited from **Xamarin.Forms.View**) |
| | InputTransparentProperty | (Inherited from **Xamarin.Forms.VisualElement**) |
| | InvalidRowHandle | An invalid row handle. This value is returned when an attempt to obtain a particular row fails. |
| | IsClippedToBoundsProperty | (Inherited from **Xamarin.Forms.Layout**) |
| | IsColumnChooserEnabledProperty | Identifies the IsColumnChooserEnabled bindable property. |
| | IsColumnMenuEnabledProperty | Identifies the IsColumnMenuEnabled bindable property. |
| | IsEnabledProperty | (Inherited from **Xamarin.Forms.VisualElement**) |
| | IsFocusedProperty | (Inherited from **Xamarin.Forms.VisualElement**) |
| | IsGroupRowMenuEnabledProperty | Identifies the IsGroupRowMenuEnabled bindable property. |
| | IsLoadMoreEnabledProperty | Identifies the IsLoadMoreEnabled bindable property. |
| | IsPullToRefreshEnabledProperty | Identifies the IsPullToRefreshEnabled bindable property. |
| | IsReadOnlyProperty | Identifies the IsReadOnly bindable property. |
| | IsRowCellMenuEnabledProperty | Identifies the IsRowCellMenuEnabled bindable property. |
| | IsTotalSummaryMenuEnabledProperty | Identifies the IsTotalSummaryMenuEnabled bindable property. |
| | IsVisibleProperty | (Inherited from **Xamarin.Forms.VisualElement**) |
| | ItemsSourceProperty | Identifies the ItemsSource bindable property. |
| | LoadMoreCommandProperty | Identifies the LoadMoreCommand bindable property. |
| | MinimumHeightRequestProperty | (Inherited from **Xamarin.Forms.VisualElement**) |
| | MinimumWidthRequestProperty | (Inherited from **Xamarin.Forms.VisualElement**) |
| | NavigationProperty | (Inherited from **Xamarin.Forms.VisualElement**) |
| | NewItemRowVisibilityProperty | Identifies the NewItemRowVisibility bindable property. |
| | OpacityProperty | (Inherited from **Xamarin.Forms.VisualElement**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| S | OptionsExportCsvProperty | Identifies the OptionsExportCsv bindable property. |
| S | OptionsExportXlsProperty | Identifies the OptionsExportXls bindable property. |
| S | OptionsExportXlsxProperty | Identifies the OptionsExportXlsx bindable property. |
| S | PaddingProperty | (Inherited from **Xamarin.Forms.Layout**) |
| S | PullToRefreshCommandProperty | Identifies the PullToRefreshCommand bindable property. |
| S | RotationProperty | (Inherited from **Xamarin.Forms.VisualElement**) |
| S | RotationXProperty | (Inherited from **Xamarin.Forms.VisualElement**) |
| S | RotationYProperty | (Inherited from **Xamarin.Forms.VisualElement**) |
| S | RowDefinitionsProperty | (Inherited from **Xamarin.Forms.Grid**) |
| S | RowEditModeProperty | Identifies the RowEditMode bindable property. |
| S | RowHeightProperty | Identifies the RowHeight bindable property. |
| S | RowProperty | (Inherited from **Xamarin.Forms.Grid**) |
| S | RowSpacingProperty | (Inherited from **Xamarin.Forms.Grid**) |
| S | RowSpanProperty | (Inherited from **Xamarin.Forms.Grid**) |
| S | RowTapCommandProperty | Identifies the RowTapCommand bindable property. |
| S | ScaleProperty | (Inherited from **Xamarin.Forms.VisualElement**) |
| S | SelectedDataObjectProperty | Identifies the SelectedDataObject bindable property. |
| S | SelectedRowHandleProperty | Identifies the SelectedRowHandle bindable property. |
| S | SortModeProperty | Identifies the SortMode bindable property. |
| S | StyleProperty | (Inherited from **Xamarin.Forms.VisualElement**) |
| S | SwipeButtonCommandProperty | Identifies the SwipeButtonCommand bindable property. |
| S | TotalSummaryHeightProperty | Identifies the TotalSummaryHeight bindable property. |
| S | TotalSummaryVisibilityProperty | Identifies the TotalSummaryVisibility bindable property. |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | TranslationXProperty | (Inherited from **Xamarin.Forms.VisualElement**) |
| | TranslationYProperty | (Inherited from **Xamarin.Forms.VisualElement**) |
| | TriggersProperty | (Inherited from **Xamarin.Forms.VisualElement**) |
| | VerticalOptionsProperty | (Inherited from **Xamarin.Forms.View**) |
| | WidthProperty | (Inherited from **Xamarin.Forms.VisualElement**) |
| | WidthRequestProperty | (Inherited from **Xamarin.Forms.VisualElement**) |
| | XProperty | (Inherited from **Xamarin.Forms.VisualElement**) |
| | YProperty | (Inherited from **Xamarin.Forms.VisualElement**) |

**Public Events**

| | Name | Description |
|---|---|---|
| | AutoGeneratingColumn | Occurs each time an individual column is auto-generated. |
| | BindingContextChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | CalculateCustomSummary | Allows you to calculate summary values manually. |
| | ChildAdded | (Inherited from **Xamarin.Forms.Element**) |
| | ChildRemoved | (Inherited from **Xamarin.Forms.Element**) |
| | ChildrenReordered | (Inherited from **Xamarin.Forms.VisualElement**) |
| | CustomizeCell | Allows you to customize the appearance of cells in the grid. |
| | CustomizeCellDisplayText | Enables custom display text to be provided for any data cell. |
| | CustomUnboundColumnData | Enables you to populate unbound columns with data, and save changes made by end-users in unbound columns. |
| | DescendantAdded | (Inherited from **Xamarin.Forms.Element**) |
| | DescendantRemoved | (Inherited from **Xamarin.Forms.Element**) |
| | EndRowEdit | Occurs after a row editor (an in-place editor or edit form, depending on the edit mode set in the grid) is closed and |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | | modified cell values are applied or canceled. |
| | Focused | (Inherited from **Xamarin.Forms.VisualElement**) |
| | GroupRowCollapsed | Occurs after a group row has been collapsed. |
| | GroupRowCollapsing | Occurs before a group row is collapsed. |
| | GroupRowExpanded | Occurs after a group row has been expanded. |
| | GroupRowExpanding | Occurs before a group row is expanded. |
| | HorizontalScrollOffsetChanged | Occurs after the HorizontalScrollOffset property has been changed. |
| | InitNewRow | Enables you to initialize newly added rows. |
| | LayoutChanged | (Inherited from **Xamarin.Forms.Layout**) |
| | LoadMore | Occurs when an end-user scrolls to the bottom of the grid to load new data items. |
| | MeasureInvalidated | (Inherited from **Xamarin.Forms.VisualElement**) |
| | PopupMenuCustomization | Enables you to customize the grid's popup menus. |
| | PropertyChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanging | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PullToRefresh | Occurs when an end-user pulls the grid down to refresh its content. |
| | RowTap | Occurs when a data row is tapped. |
| | SelectionChanged | Occurs after the selection has been changed. |
| | SizeChanged | (Inherited from **Xamarin.Forms.VisualElement**) |
| | SwipeButtonClick | Occurs when a swipe button is tapped. |
| | SwipeButtonShowing | Occurs before a swipe button is shown when an end-user swipes left to right, or right to left, over a data row. |
| | Unfocused | (Inherited from **Xamarin.Forms.VisualElement**) |

**See Also**
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# ActualFilterString Property

Returns the string that specifies a filter currently used in the grid.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public string ActualFilterString { get; }
```

### Property Value

A System.String value that specifies the grid's current filter.

### Remarks

A filter applied to data in the grid is composed of the following criteria.

- **Column Auto Filters**

End-users can apply auto filters to individual columns using the auto-filter panel that is displayed under column headers if the AutoFilterPanelVisibility property is set to **true**. To programmatically specify a column auto filter, use the GridColumn.AutoFilterValue property. The type of the comparison operator used to create filter conditions for a column is specified by the GridColumn.AutoFilterCondition property.

- **Grid Filter Expression**

A grid filter expression is specified using the FilterString or FilterExpression property. This expression can consist of multiple conditions applied to multiple columns.

To get the string that specifies the filter currently used in the grid (including both column auto filters and grid filter expression), use the **ActualFilterString** property. By default, this string is shown in the filter panel that is automatically displayed at the bottom of the grid after a filter has been created. This panel provides buttons allowing end-users to disable/enable and clear the filter. To control the visibility of the filter panel, use the FilterPanelVisibility property.

To clear the grid's filter, use the ClearFilter property.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# AllowDeleteRows Property

Gets or sets whether end-users are allowed to delete data rows. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public bool AllowDeleteRows { get; set; }
```

### Property Value

**true**, to allow end-users to delete rows; otherwise, **false**.

### Remarks

End-users can delete a row by tapping **Delete Row** in a menu that appears when they touch and hold a data row. If the **AllowDeleteRows** property is set to **true**, this menu item is available. Otherwise, end-users cannot delete data rows.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
DeleteRow
End-User Restrictions

© 2018 Developer Express Inc.

# AllowEditRows Property

Gets or sets whether end-users are allowed to change cell values in the grid. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public bool AllowEditRows { get; set; }
```

### Property Value

**true**, to allow data editing; otherwise, **false**.

### Remarks

End-users can switch the grid to edit mode by double-tapping the required cell or tapping the **Edit Cell** in a menu that appears when they touch and hold a data row. If the **AllowEditRows** property is set to **true**, both these methods are available. Otherwise, end-users cannot edit data.

To prevent end-users from editing values of a specific column only, use the GridColumn.IsReadOnly property of the corresponding column object.

The IsReadOnly property allows you to prevent end-users from editing data values in the grid as well as from adding and deleting data rows.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
End-User Restrictions

© 2018 Developer Express Inc.

# AllowGroup Property

Gets or sets whether end-users are allowed to group data in the grid. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public bool AllowGroup { get; set; }
```

### Property Value

**true**, to allow end-users to group data; otherwise, **false**.

### Remarks

End-users can group data or remove grouping applied in the grid by tapping the corresponding item in a menu that appears when they touch and hold a column. If the **AllowGroup** property is set to **true**, menu items related to grouping are available. Otherwise, end-users cannot group data.

To override the default behavior specified by the grid's **AllowGroup** property for an individual column, use the GridColumn.AllowGroup property of the corresponding column object.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
Lesson 4 - Group Data
End-User Restrictions

© 2018 Developer Express Inc.

# AllowGroupCollapse Property

Gets or sets whether end-users are allowed to collapse and expand group rows. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public bool AllowGroupCollapse { get; set; }
```

### Property Value

**true**, to allow end-users to collapse/expand group rows; otherwise, **false**.

### Remarks

By default, an end-user can collapse and expand data groups by tapping group rows. Set the **AllowGroupCollapse** property to **false** to disable group collapsing on the grid.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
End-User Restrictions

© 2018 Developer Express Inc.

# AllowHorizontalScrollingVirtualization Property

Gets or sets a value indicating whether virtualization is enabled for horizontal scrolling.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public bool AllowHorizontalScrollingVirtualization { get; set; }
```

### Property Value

**true** to enable virtualization; **false** to disable it. The default is **true**.

### Remarks

If the ColumnsAutoWidth property is set to **false** and the number of columns is great, you can speed up a grid's performance by enabling the **AllowHorizontalScrollingVirtualization** option. If this option is enabled, a grid renders only those columns that are displayed onscreen.

Each time a grid is horizontally scrolled, visible (displayed onscreen) columns are rendered again. Otherwise, if the **AllowHorizontalScrollingVirtualization** property is set to **false**, the grid renders all columns.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# AllowResizeColumns Property

Gets or sets whether end-users are allowed to change column widths. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public bool AllowResizeColumns { get; set; }
```

### Property Value

**true**, to allow end-users to resize columns; otherwise, **false**.

### Remarks

By default, end-users can change column widths with pinch and spread gestures. Set the **AllowResizeColumns** property to **false** to prevent end-users from resizing columns.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
End-User Restrictions

© 2018 Developer Express Inc.

# AllowSort Property

Gets or sets whether end-users are allowed to sort data in the grid. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public bool AllowSort { get; set; }
```

### Property Value

**true**, to allow end-users to sort data; otherwise, **false**.

### Remarks

End-users can sort data by tapping a column header or tapping **Sort Ascending** or **Sort Descending** in the menu that appears when they touch and hold a column cell or header. If the **AllowSort** property is set to **true**, both these methods are available. Otherwise, end-users cannot apply data sorting.

To override the default behavior specified by the grid's **AllowSort** property for an individual column, use the GridColumn.AllowSort property of the corresponding column object.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
Lesson 6 - Sort Data
End-User Restrictions

© 2018 Developer Express Inc.

# AutoFilterPanelHeight Property

Gets or sets the auto-filter panel height. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public Double AutoFilterPanelHeight { get; set; }
```

### Property Value

A System.Double value specifying the auto-filter panel height in density-independent pixels.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
AutoFilterPanelVisibility

© 2018 Developer Express Inc.

# AutoFilterPanelVisibility Property

Gets or sets whether the auto-filter panel is displayed in the grid. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public bool AutoFilterPanelVisibility { get; set; }
```

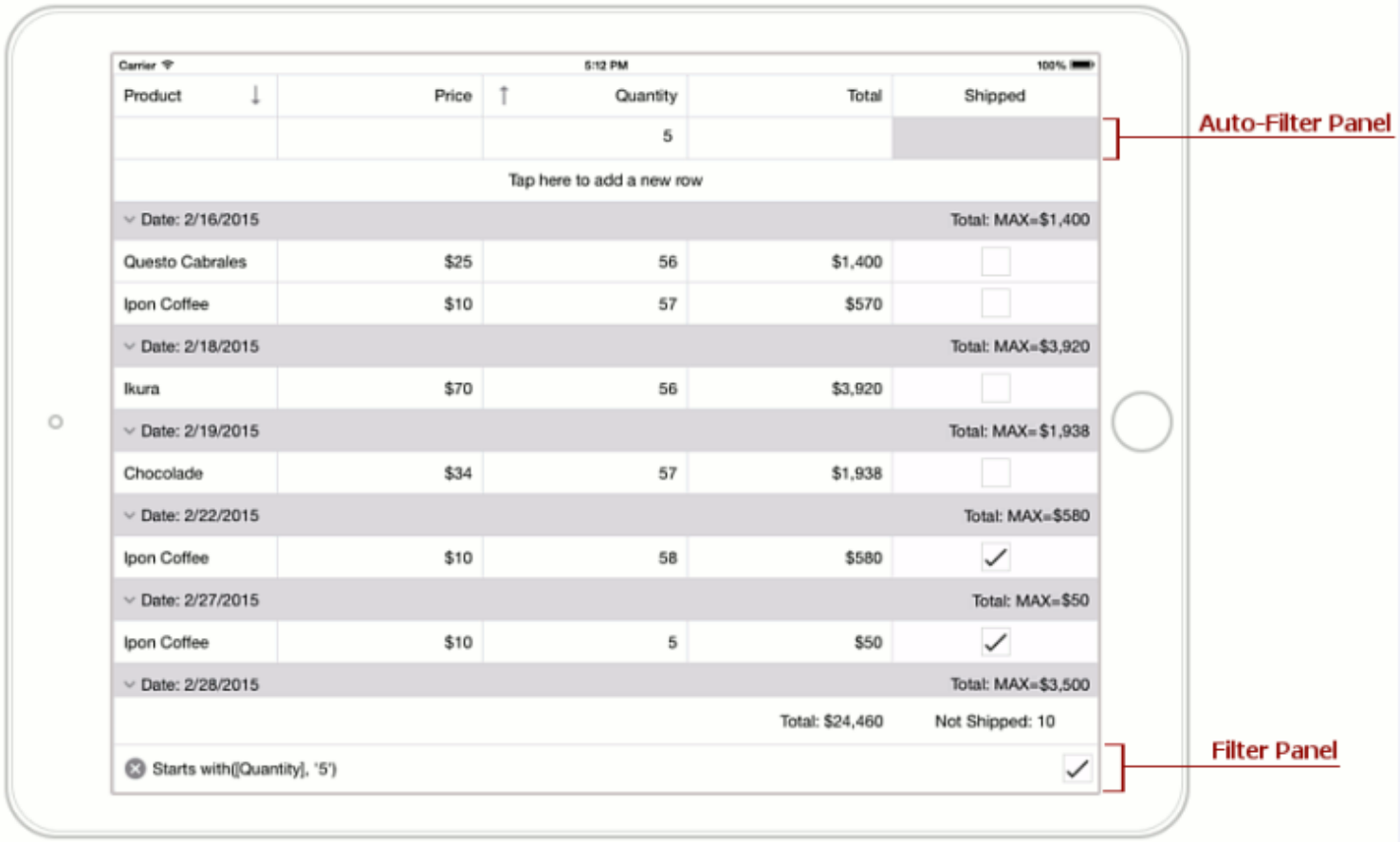
### Property Value

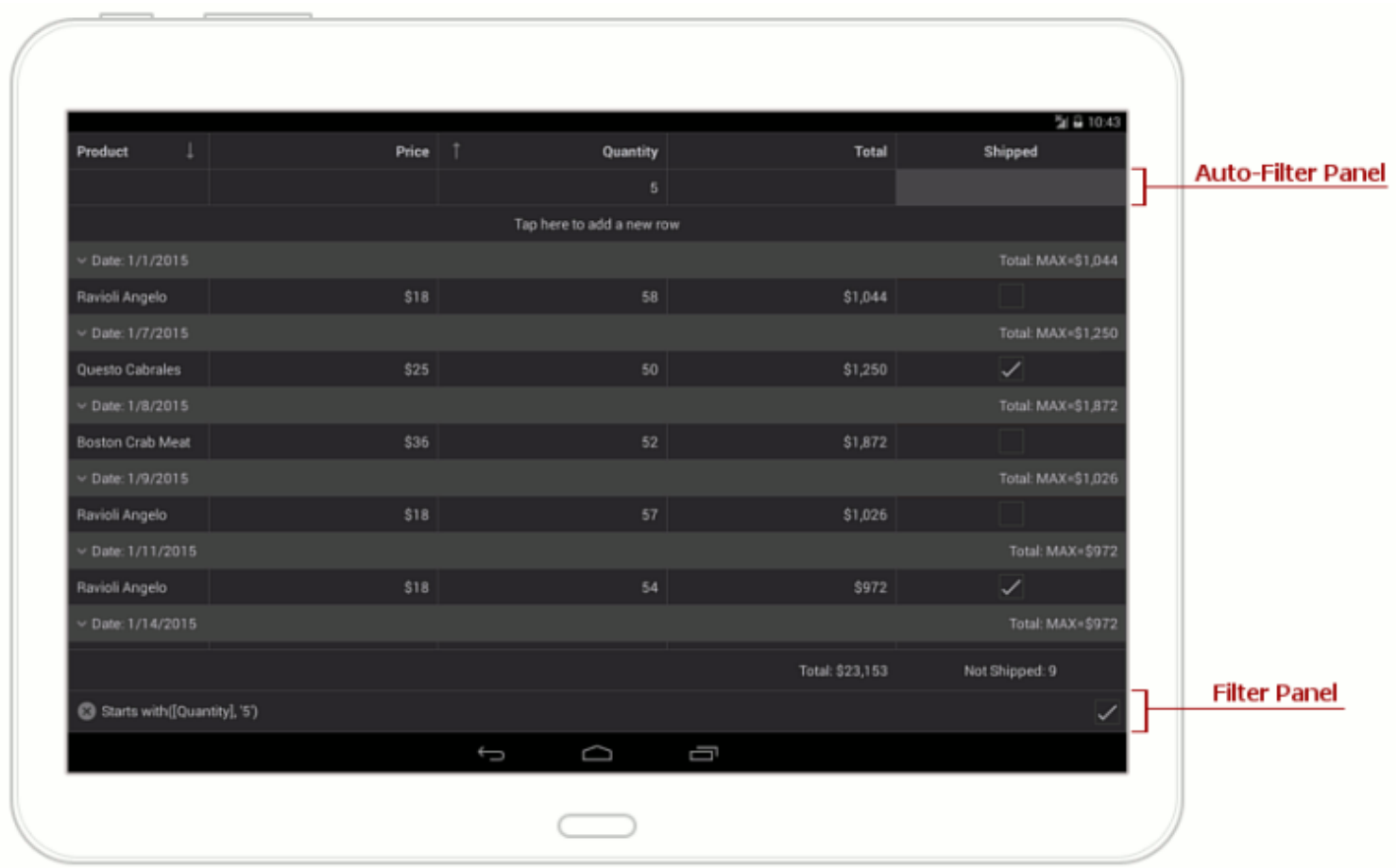
**true**, to display the auto-filter panel; otherwise, **false**.

### Remarks

The auto-filter panel allows end-users to filter data on the fly. When an end-user types text inside a cell in the panel, a filter condition is created based upon the entered value and applied to the column. To remove a filter, an end-user can clear the text from the auto-filter panel.

By default, the auto-filter panel is hidden. To display it under column headers in the grid, set the **AutoFilterPanelVisibility** property to **true**.






You can use the following properties to specify the auto-filter panel settings.

- **Availability**
  To specify whether column values can be filtered using the auto-filter panel, use the GridColumn.AllowAutoFilter property. If this property is set to **false**, the corresponding cell within the auto-filter panel is disabled.
- **Comparison Operator Type**

To specify the comparison operator type (Contains, Equals or Like), use the GridColumn.AutoFilterCondition property.

- **Data**
  To obtain or set a value in the auto-filter panel for a column, use the corresponding column's GridColumn.AutoFilterValue property.
- **Auto Update Filters**

By default, when an end-user modifies the text within the auto filter panel's cell, the filter condition for this column is updated immediately. To disable automatic updates, set the column's GridColumn.ImmediateUpdateAutoFilter property to **false**. In this instance, an end-user should move focus to another cell to force updates.

**See Also**
GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
FilterPanelVisibility
FilterExpression
FilterString
ActualFilterString

© 2018 Developer Express Inc.

# AutoGenerateColumnsMode Property

Gets or sets how columns should be automatically created for fields of the underlying data source, depending on whether the grid contains any columns.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public AutoGenerateColumnsMode AutoGenerateColumnsMode { get; set; }
```

### Property Value

An AutoGenerateColumnsMode enumeration value that specifies the way columns are generated.

### Remarks

Set the **AutoGenerateColumnsMode** property to AutoGenerateColumnsMode.None to prevent columns from being automatically created.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
Create Columns and Bind Them to Data Fields

© 2018 Developer Express Inc.

# ColumnHeadersHeight Property

Gets or sets the height of column headers in the grid. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public Double ColumnHeadersHeight { get; set; }
```

### Property Value

A System.Double value specifying the height of column headers in density-independent pixels.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# ColumnHeadersVisibility Property

Gets or sets whether column headers are shown in the grid. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public bool ColumnHeadersVisibility { get; set; }
```

### Property Value

**true**, to display column headers; otherwise, **false**.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# Columns Property

Provides access to the grid's collection of columns.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public GridColumnCollection Columns { get; }
```

### Property Value

A GridColumnCollection object that specifies a collection of columns within the grid.

### Remarks

The grid stores its columns in the **Columns** collection. This collection provides methods that allow you to add new and remove existing columns. Individual columns are specified by GridColumn descendants, and can be accessed using indexer notation or by the field name (GridColumn.FieldName). To access only columns that are currently displayed within the grid, use the VisibleColumns property.

To learn more on grid columns, see the following topics.

- Columns Overview
- Create Columns and Bind Them to Data Fields
- Unbound Columns
- Identify and Access Columns

### Example

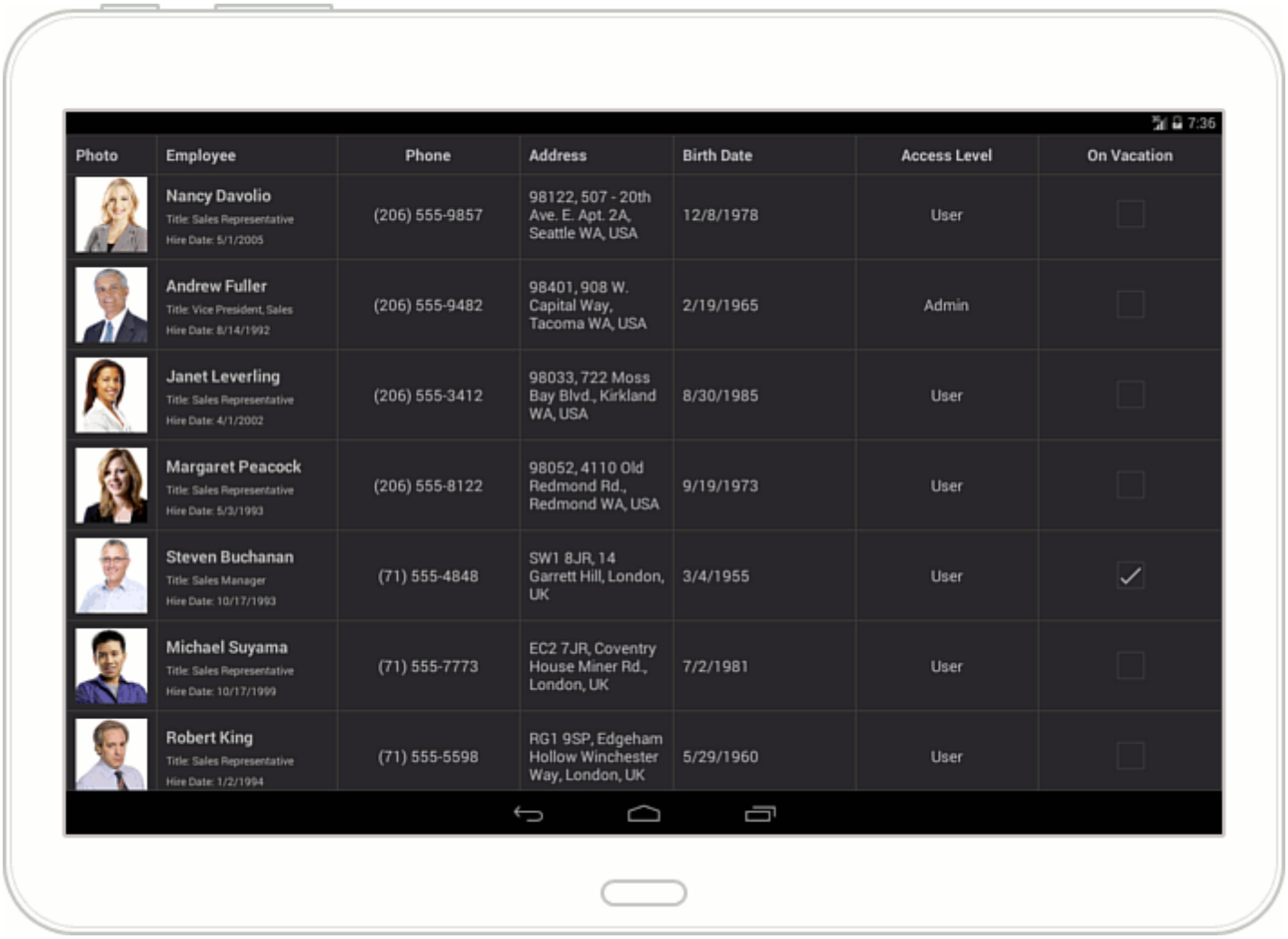
| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T208848. | |

This example shows how to add columns to GridControl to display and edit data of different types. The grid is bound to a collection of **Employee** objects. Each **Employee** object contains an employee's photo (image), name, position, phone, address (strings), hire and birth dates (DateTime values), a value that specifies an employee's access level, and a Boolean value indicating whether an employee is on vacation.

© 2018 Developer Express Inc.

| Photo | Employee | Phone | Address | Birth Date | Access Level | On Vacation |
|---|---|---|---|---|---|---|
| | **Nancy Davolio**<br>Title: Sales Representative<br>Hire Date: 5/1/2005 | (206) 555-9857 | 98122, 507 - 20th Ave. E. Apt. 2A, Seattle WA, USA | 12/8/1978 | User | ☐ |
| | **Andrew Fuller**<br>Title: Vice President, Sales<br>Hire Date: 8/14/1992 | (206) 555-9482 | 98401, 908 W. Capital Way, Tacoma WA, USA | 2/19/1965 | Admin | ☐ |
| | **Janet Leverling**<br>Title: Sales Representative<br>Hire Date: 4/1/2002 | (206) 555-3412 | 98033, 722 Moss Bay Blvd., Kirkland WA, USA | 8/30/1985 | User | ☐ |
| | **Margaret Peacock**<br>Title: Sales Representative<br>Hire Date: 5/3/1993 | (206) 555-8122 | 98052, 4110 Old Redmond Rd., Redmond WA, USA | 9/19/1973 | User | ☐ |
| | **Steven Buchanan**<br>Title: Sales Manager<br>Hire Date: 10/17/1993 | (71) 555-4848 | SW1 8JR, 14 Garrett Hill, London, UK | 3/4/1955 | User | ☑ |
| | **Michael Suyama**<br>Title: Sales Representative<br>Hire Date: 10/17/1999 | (71) 555-7773 | EC2 7JR, Coventry House Miner Rd., London, UK | 7/2/1981 | User | ☐ |
| | **Robert King**<br>Title: Sales Representative<br>Hire Date: 1/2/1994 | (71) 555-5598 | RG1 9SP, Edgeham Hollow Winchester Way, London, UK | 5/29/1960 | User | ☐ |

© 2018 Developer Express Inc.




Grid columns are stored in the Columns collection. An individual column is specified by a GridColumn descendant, which is appropriate to the type of data contained in this column. In this example, to display information on employees and allow end-users to edit it, the following columns are created in the grid.

| Grid Column | Description |
|---|---|
| Photo<br>ImageColumn | This column displays employee photos which are images added to a project as embedded resources. |
| Employee<br>TemplateColumn | A cell in this column displays three employee properties: Name, Position and HireDate. A cell appearance is defined via a template. Each cell contains three Xamarin.Forms.Label elements within a Xamarin.Forms.Grid. Each Label element is bound to a property of the **Employee** class.<br><br>A cell template data context is specified by the CellData object. Its CellData.Value property provides access to a value of a data field assigned to the column's GridColumn.FieldName property. In this example, a column cell displays not only this field value but also values of two more fields. So, it is required to specify the whole data row as a binding source for the template. To do this, set the BindingContext property of the Grid to CellData.Source. Note that BindingContext specified for the Grid is inherited by all its children. This means that all Label elements have the same BindingContext as the Grid, and you can specify |

© 2018 Developer Express Inc.

| | |
|---|---|
| | their simple bindings to properties of that object (Employee).<br><br>Use the GridColumn.AllowSort, GridColumn.AllowGroup and GridColumn.IsReadOnly properties to prevent end-users from sorting and grouping data in the grid by this column and disable data editing. |
| Phone<br>NumberColumn | The employee's **Phone** property is of the string type. The keyboard allowing text input is automatically displayed when an end-user activates this column cell to edit an employee's phone. |
| Address<br>TextColumn | This column is bound to the employee's **Address** string property. The keyboard allowing text input is automatically displayed when an end-user activates this column cell to edit an employee's address. |
| Birth Date<br>DateColumn | This column displays and allows editing employee birth days. The GridColumn.DisplayFormat property specifies the format of displaying dates. |
| Access Level<br>PickerColumn | The employee's **Access** property is of the **AccessLevel** type which is an enumeration with values **Admin** and **User**. To allow editing an employee's access level by selecting one of two enumeration values, this property is bound to the grid's picker column. |
| On Vacation<br>SwitchColumn | The employee's **OnVacation** property is of the Boolean type, so it is bound to the grid's switch column which allows editing a cell value by selecting between two states. |

**C#**

```
 (Employee.cs)
public enum AccessLevel {
    Admin,
    User
}
public class Employee : ModelObject {
    string name;
    public string Name {
        get { return name; }
        set {
            name = value;
            if (Photo == null)
                Photo = ImageSource.FromResource (value.Replace (" ", String.Empty));
        }
    }
    public Employee(string name) {
        this.Name = name;
    }
    public ImageSource Photo { get; set; }
    public DateTime BirthDate { get; set; }
    public DateTime HireDate { get; set; }
    public string Position { get; set; }
    public string Address { get; set; }
    public string Phone { get; set; }
    public AccessLevel Access { get; set; }
    public bool OnVacation { get; set; }
}
```

**C#**

```
 (EmployeesRepository.cs)
public abstract class EmployeesRepository {
    readonly ObservableCollection<Employee> employees;
    public EmployeesRepository() {
        this.employees = new ObservableCollection<Employee>();
    }
    public ObservableCollection<Employee> Employees {
        get { return employees; }
    }
}
public class TestEmployeesRepository : EmployeesRepository {
    public TestEmployeesRepository() : base() {
        GenerateCustomers();
    }
    void GenerateCustomers() {
        Employees.Add(
            new Employee("Nancy Davolio") {
                BirthDate = new DateTime(1978, 12, 8),
                HireDate = new DateTime(2005, 5, 1),
                Position = "Sales Representative",
                Address = "98122, 507 - 20th Ave. E. Apt. 2A, Seattle WA, USA",
                Phone = "(206) 555-9857",
                Access = AccessLevel.User,
                OnVacation = false
            }
        );
        Employees.Add(
            new Employee("Andrew Fuller") {
                BirthDate = new DateTime(1965, 2, 19),
                HireDate = new DateTime(1992, 8, 14),
                Position = "Vice President, Sales",
                Address = "98401, 908 W. Capital Way, Tacoma WA, USA",
                Phone = "(206) 555-9482",
                Access = AccessLevel.Admin,
                OnVacation = false
            }
        );
        Employees.Add(
            new Employee("Janet Leverling") {
                BirthDate = new DateTime(1985, 8, 30),
                HireDate = new DateTime(2002, 4, 1),
                Position = "Sales Representative",
                Address = "98033, 722 Moss Bay Blvd., Kirkland WA, USA",
                Phone = "(206) 555-3412",
                Access = AccessLevel.User,
                OnVacation = false
            }
        );
        Employees.Add(
            new Employee("Margaret Peacock") {
                BirthDate = new DateTime(1973, 9, 19),
                HireDate = new DateTime(1993, 5, 3),
                Position = "Sales Representative",
                Address = "98052, 4110 Old Redmond Rd., Redmond WA, USA",
                Phone = "(206) 555-8122",
                Access = AccessLevel.User,
                OnVacation = false
            }
        );
        Employees.Add(
            new Employee("Steven Buchanan") {
                BirthDate = new DateTime(1955, 3, 4),
                HireDate = new DateTime(1993, 10, 17),
                Position = "Sales Manager",
                Address = "SW1 8JR, 14 Garrett Hill, London, UK",
```

© 2018 Developer Express Inc.

```
                Phone = "(71) 555-4848",
                Access = AccessLevel.User,
                OnVacation = true
            }
        );
        Employees.Add(
            new Employee("Michael Suyama") {
                BirthDate = new DateTime(1981, 7, 2),
                HireDate = new DateTime(1999, 10, 17),
                Position = "Sales Representative",
                Address = "EC2 7JR, Coventry House Miner Rd., London, UK",
                Phone = "(71) 555-7773",
                Access = AccessLevel.User,
                OnVacation = false
            }
        );
        Employees.Add(
            new Employee("Robert King") {
                BirthDate = new DateTime(1960, 5, 29),
                HireDate = new DateTime(1994, 1, 2),
                Position = "Sales Representative",
                Address = "RG1 9SP, Edgeham Hollow Winchester Way, London, UK",
                Phone = "(71) 555-5598",
                Access = AccessLevel.User,
                OnVacation = false
            }
        );
        Employees.Add(
            new Employee("Laura Callahan") {
                BirthDate = new DateTime(1985, 1, 9),
                HireDate = new DateTime(2004, 3, 5),
                Position = "Inside Sales Coordinator",
                Address = "98105, 4726 - 11th Ave. N.E., Seattle WA, USA",
                Phone = "(206) 555-1189",
                Access = AccessLevel.User,
                OnVacation = true
            }
        );
        Employees.Add(
            new Employee("Anne Dodsworth") {
                BirthDate = new DateTime(1980, 1, 27),
                HireDate = new DateTime(2004, 11, 15),
                Position = "Sales Representative",
                Address = "WG2 7LT, 7 Houndstooth Rd., London, UK",
                Phone = "(71) 555-4444",
                Access = AccessLevel.User,
                OnVacation = false
            }
        );
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl ItemsSource="{Binding Employees}" RowHeight="100">
    <dxGrid:GridControl.Columns>
        <dxGrid:ImageColumn FieldName="Photo" Width="100" />
        <dxGrid:TemplateColumn FieldName="Name" Caption="Employee" Width="
                               IsReadOnly="true" AllowGroup="False" AllowS
          <dxGrid:TemplateColumn.DisplayTemplate>
            <DataTemplate>
              <Grid BindingContext="{Binding Source}" Padding="0,15,0,15">
                <Grid.RowDefinitions>
                  <RowDefinition Height="Auto" />
                  <RowDefinition Height="Auto" />
                  <RowDefinition Height="Auto" />
                </Grid.RowDefinitions>
                <Label Text="{Binding Name}" Font="Bold, 18" Grid.Row="0"
                <Label Text="{Binding Position, StringFormat = 'Title: {0}
                <Label Text="{Binding HireDate, StringFormat = 'Hire Date:
              </Grid>
            </DataTemplate>
          </dxGrid:TemplateColumn.DisplayTemplate>
        </dxGrid:TemplateColumn>
        <dxGrid:NumberColumn FieldName="Phone" ContentAlignment="Center"/>
        <dxGrid:TextColumn FieldName="Address" Width="170"/>
        <dxGrid:DateColumn FieldName="BirthDate" DisplayFormat="d"/>
        <dxGrid:PickerColumn FieldName="Access" Caption="Access Level" Con
        <dxGrid:SwitchColumn FieldName="OnVacation"/>
    </dxGrid:GridControl.Columns>
</dxGrid:GridControl>
```

## See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# ColumnsAutoWidth Property

Gets or sets whether column widths are automatically changed, so that the total column width matches the grid's width. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public bool ColumnsAutoWidth { get; set; }
```

### Property Value

**true**, to enable the column auto width feature; otherwise, **false**. The default is **true**.

### Remarks

Column widths can be changed automatically, so that the total column width matches the grid's width. In this instance, changing the width of one column automatically changes the widths of other columns.

If the automatic column width calculation is disabled, a column's width is specified by its GridColumn.Width property. If the total column width exceeds the grid's width, a horizontal scrolling feature is enabled, allowing an end-user to scroll grid columns. If you want a column (or columns) to always be displayed on screen regardless of scrolling, manipulate its GridColumn.FixedStyle property.

For more information, see the Fixed Columns document.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# EditFormContent Property

Gets or sets the template used to define the visual presentation of the grid's row edit form.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public DataTemplate EditFormContent { get; set; }
```

### Property Value

A Xamarin.Forms.DataTemplate object.

### Remarks

Use the **EditFormContent** property to replace a GridControl's default row edit form with your custom form.

### Example

| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T279098. | |

GridControl allows you to substitute the default row edit form with a custom form. This example explains how to do this.

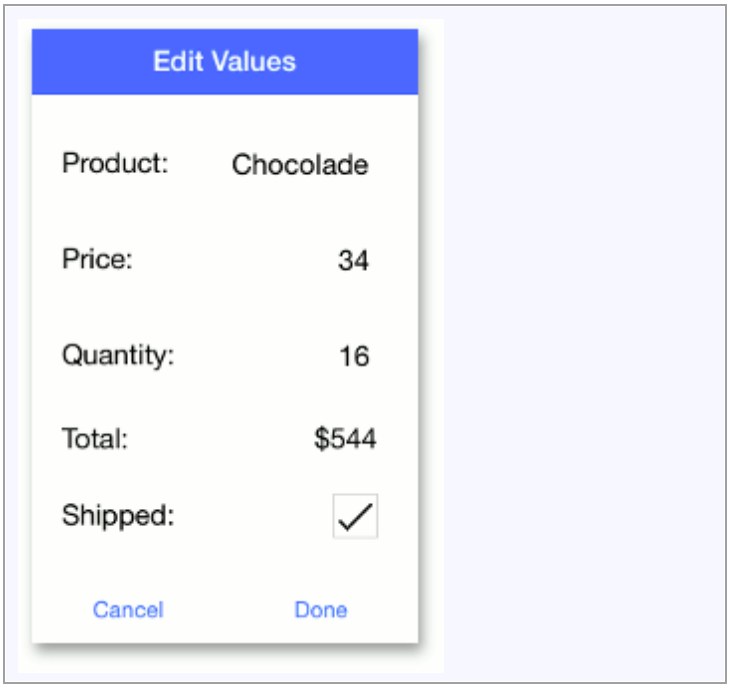
In this application, the grid is bound to a collection of orders. Each data row in the grid represents an individual order and displays the following information:

- Ordered product name (the **Product** column)
- Product unit price (the **Price** column)
- Number of ordered product units (the **Quantity** column)
- Order total value (the **Total** column)
- A value indicating whether an order is shipped (the **Shipped** column)

The default row edit form in this grid looks like the following.

| iOS | Android |
|---|---|

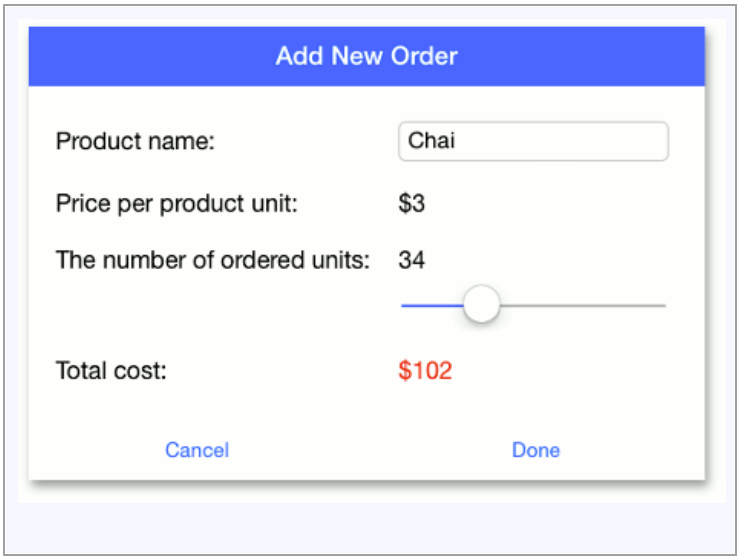
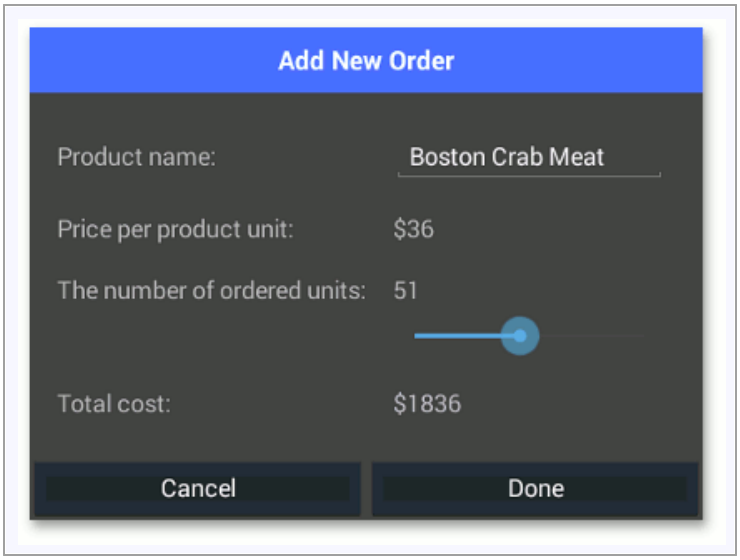
© 2018 Developer Express Inc.




This example demonstrates how to create a custom form to add new data rows and another form to edit existing rows.

- **Add New Order Form**

This form is invoked on tapping New Item Row in the grid and allows an end user to add a new data row (create a new order). On this form, an end user can select a product name via a Picker control, and set the number of ordered product units via a Slider control. A unit price is automatically shown for the selected product, and the order total value is automatically calculated.

| iOS | Android |
| --- | --- |
|  |  |

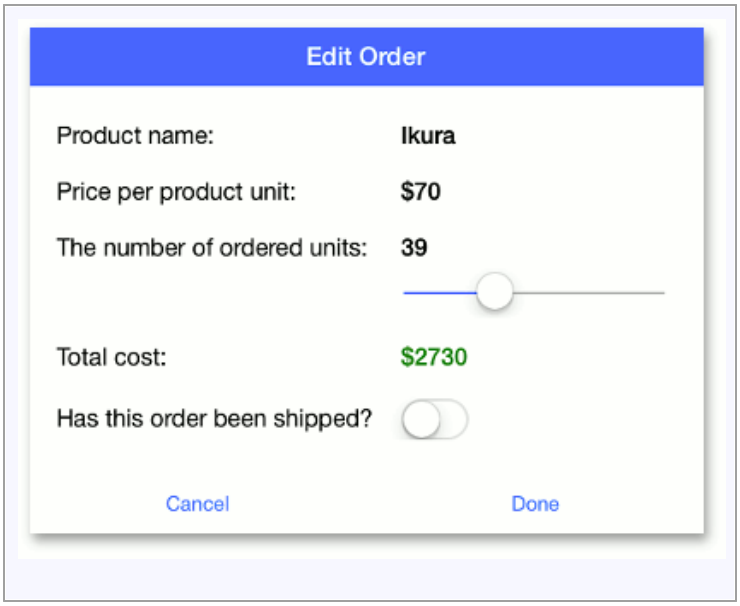


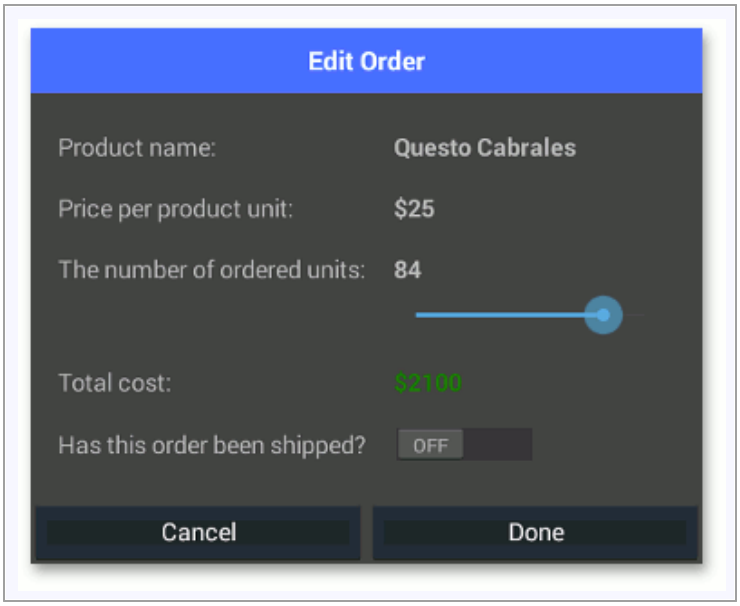
- **Edit Order Form**

This form is invoked on double-tapping a cell or selecting **Edit Cell** from the popup menu. It allows an end user to modify existing data rows. On this form, product name and unit price are shown in Lable controls and cannot be changed. An end user can change the number of ordered product units via a Slider control and use the Switch control to set whether the order is shipped.

| iOS | Android |
| --- | --- |

© 2018 Developer Express Inc.




To create a custom row edit form and show it instead of the default form, execute the steps below.

1. Specify a custom form view. In this example, it is a ContentView descendant with a set of labels and editors inside (see the *CustomFormEditorContent.cs* file). This view's BindingContext is an EditValuesContainer object. Labels and editors are bound to data fields via the EditValuesContainer.GetBindingPath method with data field names passed as parameters.

The EditValuesContainer.IsNewRow property identifies whether a from is invoked for a new or existing row.

Custom form captions are implemented by using the mechanism of localization.

2. Create a DataTemplate instance for your custom form view and assign this template to the EditFormContent property (see the *MainPage.xaml.cs* file).
3. Set the RowEditMode property to RowEditMode.Popup to show a form instead of an in-place row editor.

| C# | |
|---|---|

© 2018 Developer Express Inc.

```
 (Converters.cs)
using System;
using Xamarin.Forms;
using System.Globalization;
namespace CustomEditFormDemo {
    class Int32ToStringConverter : IValueConverter {
        object IValueConverter.Convert(object value, Type targetType, object parameter, CultureInfo cultu
            return value.ToString();
        }
        object IValueConverter.ConvertBack(object value, Type targetType, object parameter, CultureInfo cu
            int result = 0;
            try {
                result = Int32.Parse(value as String);
            } catch {
                result = 0;
            }
            return result;
        }
    }
    class Int32ToPriceStringConverter : IValueConverter {
        object IValueConverter.Convert(object value, Type targetType, object parameter, CultureInfo cultu
            return "$" + value.ToString();
        }
        object IValueConverter.ConvertBack(object value, Type targetType, object parameter, CultureInfo cu
            int result = 0;
            try {
                result = Int32.Parse(value as String);
            } catch {
                result = 0;
            }
            return result;
        }
    }
    class Int32ToDoubleConverter : IValueConverter {
        object IValueConverter.Convert(object value, Type targetType, object parameter, CultureInfo cultu
            return Convert.ToDouble(value);
        }
        object IValueConverter.ConvertBack(object value, Type targetType, object parameter, CultureInfo cu
            return Convert.ToInt32(value);
        }
    }
    class DoubleToInt32Converter : IValueConverter {
        object IValueConverter.Convert(object value, Type targetType, object parameter, CultureInfo cultu
            return Convert.ToInt32(value);
        }
        object IValueConverter.ConvertBack(object value, Type targetType, object parameter, CultureInfo cu
            return Convert.ToDouble(value);
        }
    }
}
```

**C#**

© 2018 Developer Express Inc.

```
 (CustomFormEditorContent.cs)
using System;
using System.Collections.Generic;
using Xamarin.Forms;
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.DataGrid.Theme;
using System.Globalization;
using DevExpress.Mobile.DataGrid.Localization;
namespace CustomEditFormDemo {
    class EditFormLocalizer : GridLocalizer {
        public void SetEditItemFormCaption() {
            AddString(GridStringId.EditingForm_LabelCaption, "Edit Order");
        }
        public void SetNewItemFormCaption() {
            AddString(GridStringId.EditingForm_LabelCaption, "Add New Order");
        }
    }
    public class CustomEditFormContent : ContentView {
        public TestOrdersRepository ViewModel { get; set; }
        public CustomEditFormContent() {
            HorizontalOptions = LayoutOptions.FillAndExpand;
            VerticalOptions = LayoutOptions.FillAndExpand;
            GridLocalizer.Active = new EditFormLocalizer();
        }
        protected override void OnBindingContextChanged () {
            base.OnBindingContextChanged ();
            EditFormLocalizer localizer = GridLocalizer.Active as EditFormLocalizer;
            if ((BindingContext as EditValuesContainer).IsNewRow) {
                if (localizer != null) {
                    localizer.SetNewItemFormCaption();
                }
                Content = CreateNewItemFormContent();
            }
            else {
                if (localizer != null) {
                    localizer.SetEditItemFormCaption();
                }
                Content = CreateEditItemFormContent ();
            }
        }
        View CreateNewItemFormContent() {
            Grid layoutGrid = CreateGrid(4);
            FillFormGrid(layoutGrid, true);
            return layoutGrid;
        }
        View CreateEditItemFormContent() {
            Grid layoutGrid = CreateGrid(5);
            FillFormGrid(layoutGrid, false);
            return layoutGrid;
        }
        Grid CreateGrid(int rowsCount) {
            Grid layoutGrid = new Grid();
            layoutGrid.ColumnDefinitions.Add(new ColumnDefinition() { Width = GridLength.Auto });
            layoutGrid.ColumnDefinitions.Add(new ColumnDefinition() { Width = new GridLength(200, GridUnit
            for (int i = 0; i < rowsCount; i++) {
                layoutGrid.RowDefinitions.Add(new RowDefinition() { Height = GridLength.Auto });
            }
            layoutGrid.RowSpacing = 20;
            layoutGrid.ColumnSpacing = 20;
            return layoutGrid;
        }
        void FillFormGrid(Grid layoutGrid, bool isNewItemForm) {
            FillLabels(layoutGrid, isNewItemForm);
            FillEditors(layoutGrid, isNewItemForm);
        }
```

```
        void FillLabels(Grid layoutGrid, bool isNewItemForm) {
            Label productName = new Label() { Text = "Product name:",
                                              FontSize = 18, VerticalOptions = LayoutOptions.Center };
            Label productUnitPrice = new Label() { Text = "Price per product unit:",
                                                   FontSize = 18, VerticalOptions = LayoutOptions.Center
            Label quantity = new Label() { Text = "The number of ordered units:",
                                           FontSize = 18, VerticalOptions = LayoutOptions.Start };
            Label total = new Label() { Text = "Total cost:",
                                        FontSize = 18, VerticalOptions = LayoutOptions.Center };
            Label isShipped = null;
            if (!isNewItemForm) {
                isShipped = new Label() { Text = "Has this order been shipped?",
                                          FontSize = 18, VerticalOptions = LayoutOptions.Center };
            }
            layoutGrid.Children.Add(productName, 0, 0);
            layoutGrid.Children.Add(productUnitPrice, 0, 1);
            layoutGrid.Children.Add(quantity, 0, 2);
            layoutGrid.Children.Add(total, 0, 3);
            if (!isNewItemForm) {
                layoutGrid.Children.Add(isShipped, 0, 4);
            }
        }
        void FillEditors(Grid layoutGrid, bool isNewItemForm) {
            View productName;
            // Display a product name in a label in the Edit Item form
            // or change a product name by using a picker in the New Item form.
            if (isNewItemForm) {
                productName = new Picker() { HorizontalOptions = LayoutOptions.FillAndExpand,
                                             VerticalOptions = LayoutOptions.Center};
                InitProductPickerValues(productName as Picker);
            }
            else {
                productName = new Label() { HorizontalOptions = LayoutOptions.FillAndExpand,
                                            VerticalOptions = LayoutOptions.Center,
                                            FontAttributes = FontAttributes.Bold, FontSize = 18 };
                productName.SetBinding(Label.TextProperty,
                                       new Binding(EditValuesContainer.GetBindingPath("Product.Name")));
            }
            // Display a product unit price in a form.
            Int32ToPriceStringConverter priceConverter = new Int32ToPriceStringConverter();
            Label priceLabel = new Label () { HorizontalOptions = LayoutOptions.FillAndExpand,
                                              VerticalOptions = LayoutOptions.Center, FontSize = 18 };
            priceLabel.SetBinding (Label.TextProperty, EditValuesContainer.GetBindingPath ("Product.UnitPr
                                   BindingMode.Default, priceConverter);
            // Modify the number of ordered units via a slider.
            StackLayout quantityStack = new StackLayout () { Orientation = StackOrientation.Vertical,
                                                             VerticalOptions = LayoutOptions.CenterAndExpa
            Label quantityLabel = new Label () { XAlign = TextAlignment.Start,
                                                 VerticalOptions = LayoutOptions.Center, FontSize = 18};
            Slider quantitySlider = new Slider () { Maximum = 99, Minimum = 10 };
            quantitySlider.SetBinding (Slider.ValueProperty, EditValuesContainer.GetBindingPath ("Quantity
                                       BindingMode.TwoWay, new Int32ToDoubleConverter());
            quantityLabel.BindingContext = quantitySlider;
            quantityLabel.SetBinding(Label.TextProperty, "Value", BindingMode.OneWay, new DoubleToInt32Co
            quantityStack.Children.Add(quantityLabel);
            quantityStack.Children.Add(quantitySlider);
            // Display the order total value calculated automatically.
            Label totalLabel = new Label () { XAlign = TextAlignment.Start,
                                              VerticalOptions = LayoutOptions.Center, FontSize = 18 };
            totalLabel.PropertyChanged += OnTotalPropertyChanged;
            totalLabel.SetBinding(Label.TextProperty, EditValuesContainer.GetBindingPath("Total"),
                                  BindingMode.OneWay, priceConverter);
            if (!isNewItemForm) {
                priceLabel.FontAttributes = FontAttributes.Bold;
                quantityLabel.FontAttributes = FontAttributes.Bold;
```

© 2018 Developer Express Inc.

```
                totalLabel.FontAttributes = FontAttributes.Bold;
            }
            // Add created elements to a form.
            layoutGrid.Children.Add(productName, 1, 0);
            layoutGrid.Children.Add(priceLabel, 1, 1);
            layoutGrid.Children.Add(quantityStack, 1, 2);
            layoutGrid.Children.Add(totalLabel, 1, 3);
            // Show a switch in the Edit Item form allowing end-users to set whether or not an order is sh
            Switch isShipped = new Switch() { VerticalOptions = LayoutOptions.Center,
                                              HorizontalOptions = LayoutOptions.Start };
            isShipped.SetBinding(Switch.IsToggledProperty, EditValuesContainer.GetBindingPath("Shipped"));
            if (!isNewItemForm) {
                layoutGrid.Children.Add(isShipped, 1, 4);
            }
        }
        void InitProductPickerValues(Picker picker) {
            EditValuesContainer valuesContainer = BindingContext as EditValuesContainer;
            String productName = valuesContainer.Values["Product.Name"].ToString();
            int selectedItemIndex = -1;
            foreach (Product product in ViewModel.Products) {
                if (productName == product.Name) {
                    selectedItemIndex = picker.Items.Count;
                }
                picker.Items.Add(product.Name);
            }
            picker.SelectedIndexChanged += OnProductNameSelectedIndexChanged;
            if (selectedItemIndex >= 0) {
                picker.SelectedIndex = selectedItemIndex;
            }
        }
        void OnProductNameSelectedIndexChanged(object sender, EventArgs e) {
            EditValuesContainer valuesContainer = BindingContext as EditValuesContainer;
            Picker picker = sender as Picker;
            if (picker.SelectedIndex < 0) {
                return;
            }
            String productName = picker.Items[picker.SelectedIndex];
            valuesContainer.Values["Product.Name"] = productName;
            Product product = ViewModel.Products.Find(p => p.Name == productName);
            valuesContainer.Values["Product.UnitPrice"] = product.UnitPrice;
        }
        void OnTotalPropertyChanged (object sender, System.ComponentModel.PropertyChangedEventArgs e)
        {
            Label label = (sender as Label);
            if (e.PropertyName == Label.TextProperty.PropertyName && !String.IsNullOrWhiteSpace (label.Tex
                Int32 value = (Int32)((label.BindingContext as EditValuesContainer).Values ["Total"]);
                if (value > 2000) {
                    label.TextColor = Color.Green; }
                else {
                    if (value < 500) {
                        label.TextColor = Color.Red; }
                    else {
                        label.TextColor = ThemeManager.Theme.CellCustomizer.Font.Color;}
                }
            }
        }
    }
}
```

**C#**

```
 (MainPage.xaml.cs)
using System;
using System.Collections.Generic;
using Xamarin.Forms;
using DevExpress.Mobile.DataGrid;
namespace CustomEditFormDemo
{
    public partial class MainPage : ContentPage
    {
        TestOrdersRepository model;
        public MainPage ()
        {
            InitializeComponent ();
            model = new TestOrdersRepository ();
            BindingContext = model;
            DataTemplate editFormContentTemplate = new DataTemplate(GetTem
            grid.EditFormContent = editFormContentTemplate;
            grid.RowEditMode = RowEditMode.Popup;
        }
        object GetTemplate() {
            return new CustomEditFormContent() { ViewModel = model };
        }
        void OnInitNewItemRow(object sender, InitNewRowEventArgs e) {
            e.EditableRowData.SetFieldValue("Quantity", model.MinimumQuant
            e.EditableRowData.SetFieldValue("Product.Name", model.Products
        }
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<?xml version="1.0" encoding="UTF-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="CustomEditFormDemo.MainPage"
             xmlns:dxGrid="clr-namespace:DevExpress.Mobile.DataGrid;assemb
      <ContentPage.Padding>
        <OnPlatform x:TypeArguments="Thickness" iOS="0, 20, 0, 0" />
    </ContentPage.Padding>
    <dxGrid:GridControl x:Name="grid" ItemsSource="{Binding Orders}"
                        NewItemRowVisibility="true" InitNewRow="OnInitNew
            <dxGrid:GridControl.Columns>
             <dxGrid:TextColumn FieldName="Product.Name" Caption="Product"
             <dxGrid:NumberColumn FieldName="Product.UnitPrice" Caption="P
             <dxGrid:NumberColumn FieldName="Quantity"/>
             <dxGrid:NumberColumn FieldName="Total"
                                  UnboundType="Integer" UnboundExpression=
                                  DisplayFormat="C0" IsReadOnly="True"/>
             <dxGrid:SwitchColumn FieldName="Shipped"/>
          </dxGrid:GridControl.Columns>
    </dxGrid:GridControl>
</ContentPage>
```

© 2018 Developer Express Inc.

## See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# FilterExpression Property

Gets or sets the grid filter expression.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public CriteriaOperator FilterExpression { get; set; }
```

### Property Value

A **CriteriaOperator** descendant which specifies filter criteria.

### Remarks

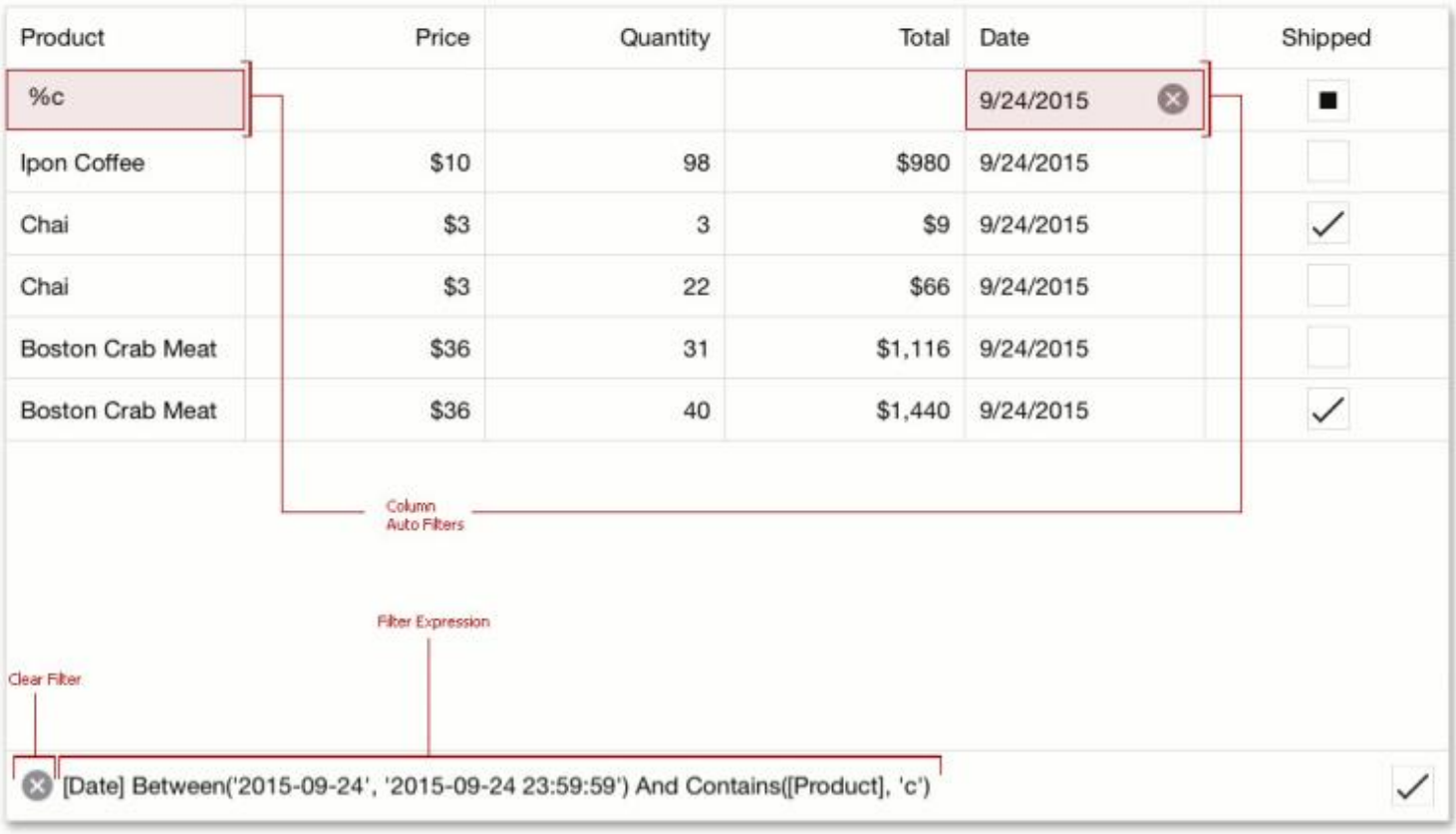
A filter applied to data in the grid is composed of the following criteria.

- **Column Auto Filters**

End-users can apply auto filters to individual columns using the auto-filter panel that is displayed under column headers if the AutoFilterPanelVisibility property is set to **true**. To programmatically specify a column auto filter, use the GridColumn.AutoFilterValue property. The type of comparison operator used to create filter conditions for a column is specified by the GridColumn.AutoFilterCondition property.

- **Grid Filter Expression**

A grid filter expression is specified via the **FilterExpression** or FilterString property. This expression can consist of multiple conditions applied to multiple columns.




To get a string that specifies a filter currently used in the grid, use the ActualFilterString property. By default, this string is shown in the filter panel that is automatically displayed at the bottom of the grid after a filter has been created. This panel provides buttons allowing end-users to disable/enable and clear the filter. To control the visibility of the filter panel, use the FilterPanelVisibility property.

To clear the grid's filter from code, use the ClearFilter property.

© 2018 Developer Express Inc.

**See Also**
GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# FilterPanelHeight Property

Gets or sets the filter panel height. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public Double FilterPanelHeight { get; set; }
```

### Property Value

A System.Double value specifying the filter panel height in density-independent pixels.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
FilterPanelVisibility

© 2018 Developer Express Inc.

# FilterPanelVisibility Property

Gets or sets when the filter panel is shown at the bottom of the grid. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public VisibilityState FilterPanelVisibility { get; set; }
```

### Property Value

A VisibilityState enumeration value which specifies when the filter panel is shown.

### Remarks

By default, the filter panel is automatically shown at the bottom of the grid after a filter has been applied. The filter panel displays a string which specifies the filter criteria currently applied in the grid (ActualFilterString), and contains buttons to disable/enable and clear the filter. Removing the filter hides this panel. To make this panel always visible or always hidden, use the **FilterPanelVisibility** property.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
AutoFilterPanelVisibility

© 2018 Developer Express Inc.

# FilterString Property

Gets or sets the grid filter expression string. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public string FilterString { get; set; }
```

### Property Value

A System.String value that specifies the grid filter expression.

### Remarks

A filter applied to data in the grid is composed of the following criteria.

- **Column Auto Filters**

End-users can apply auto filters to individual columns using the auto-filter panel that is displayed under column headers if the AutoFilterPanelVisibility property is set to **true**. To programmatically specify a column auto filter, use the GridColumn.AutoFilterValue property. The type of the comparison operator used to create filter conditions for a column is specified by the GridColumn.AutoFilterCondition property.

- **Grid Filter Expression**

A grid filter expression is specified via the **FilterString** or FilterExpression property. This expression can consist of multiple conditions applied to multiple columns.

To get a string that specifies a filter currently used in the grid, use the ActualFilterString property. By default, this string is shown in the filter panel that is automatically displayed at the bottom of the grid after a filter has been created. This panel provides buttons allowing end-users to disable/enable and clear the filter. To control the visibility of the filter panel, use the FilterPanelVisibility property.

To clear the grid's filter, use the ClearFilter property.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# FormatConditions Property

Contains format conditions applied to the grid.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public FormatConditionCollection FormatConditions { get; }
```

### Property Value

A FormatConditionCollection object that is the collection of format conditions applied to the grid's columns.

### Example

This example shows how to create conditional formatting rules for the GridControl in XAML.

| State | Sales | Sales Vs Target | Profit | Cust Satisfaction | Market Share |
|---|---|---|---|---|---|
| Montana | 90M | -0.50 % | -5M | 2 | 13 % |
| Indiana | 96M | 2.49 % | -7M | 2.5 | 17 % |
| Maine | 156M | -9.97 % | -9M | 2.9 | 14 % |
| Texas | 84M | -0.93 % | -20M | 2.9 | 13 % |
| Connecticut | 73M | 3.38 % | 8M | 2.9 | 24 % |
| North Carolina | 71M | -1.12 % | -17M | 3 | 12 % |
| Missouri | 82M | -1.44 % | 2M | 3.2 | 18 % |
| Ohio | 251M | -2.88 % | -15M | 3.4 | 29 % |
| Arizona | 73M | -0.94 % | -4M | 3.6 | 11 % |
| Wyoming | 98M | 0.21 % | 4M | 3.7 | 31 % |
| Idaho | 119M | 0.65 % | 14M | 3.9 | 30 % |
| New York | 88M | 1.15 % | 15M | 3.9 | 30 % |
| Virginia | 84M | -0.58 % | 8M | 3.9 | 10 % |
| New Hampshire | 159M | 1.33 % | 10M | 4 | 19 % |
| Massachusetts | 91M | 1.40 % | 12M | 4 | 24 % |
| Michigan | 69M | -0.84 % | 3M | 4 | 20 % |

Formatting rules specified by the FormatConditionBase class descendants (DataBarFormatCondition, FormatCondition, TopBottomRuleFormatCondition and IconSetFormatCondition) are added to the FormatConditions collection.

| Xaml | |
|---|---|

© 2018 Developer Express Inc.

```
<?xml version="1.0" encoding="UTF-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="HelloGrid.MainPage"
             xmlns:dxg="clr-namespace:DevExpress.Mobile.DataGrid;assembly=DevExpress.Mobile.Grid.v15.1">
    <dxg:GridControl x:Name = "grid" ItemsSource="{Binding Items}">
         <!-- ... -->
         <dxg:GridControl.FormatConditions>
            <dxg:DataBarFormatCondition FieldName="Sales" PredefinedFormatName="BlueGradientDataBar"/>
            <dxg:FormatCondition FieldName="SalesVsTarget" Expression="[SalesVsTarget]&lt;0" PredefinedFo
            <dxg:FormatCondition FieldName="SalesVsTarget" Expression="[SalesVsTarget]>=0" PredefinedForma
            <dxg:DataBarFormatCondition FieldName="Profit" PredefinedFormatName="GreenGradientDataBar" />
            <dxg:TopBottomRuleFormatCondition FieldName="CustomersSatisfaction" Rule="BottomPercent" Thres
            <dxg:IconSetFormatCondition FieldName="MarketShare" PredefinedFormatName="Arrows3ColoredIconSe
         </dxg:GridControl.FormatConditions>
    </dxg:GridControl>
</ContentPage>
```

**See Also**

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# GroupsInitiallyExpanded Property

Gets or sets whether groups should appear expanded or collapsed immediately after a grouping is applied in the grid.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public bool GroupsInitiallyExpanded { get; set; }
```

### Property Value

**true**, if groups are initially expanded; otherwise, **false**.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
ExpandAllGroups
CollapseAllGroups

© 2018 Developer Express Inc.

# GroupSummaries Property

Provides access to group summary items.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public ObservableCollection<GridColumnSummary> GroupSummaries { get; }
```

### Property Value

A collection of group summary items.

### Remarks

Group summaries are displayed within group rows when data grouping is applied in the grid. Group summary items are specified by the GridColumnSummary objects and are stored within the **GroupSummaries** collection. This collection provides methods and properties allowing you to add, remove and access summary items.

The collection of total summaries can be accessed using the TotalSummaries property.

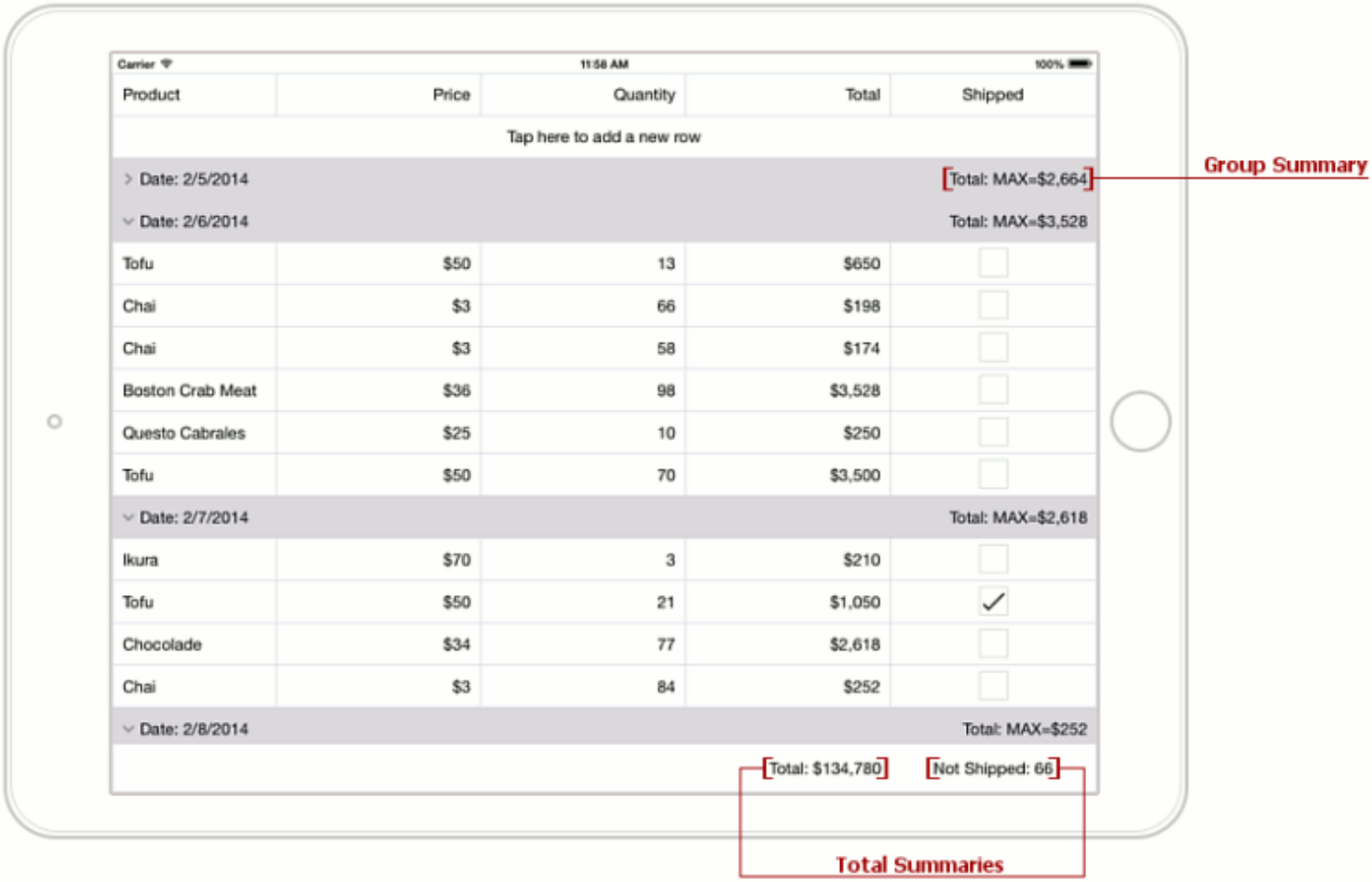
### Example

This example shows how to calculate group and total summaries for grid columns using predefined aggregate functions or a custom rule.

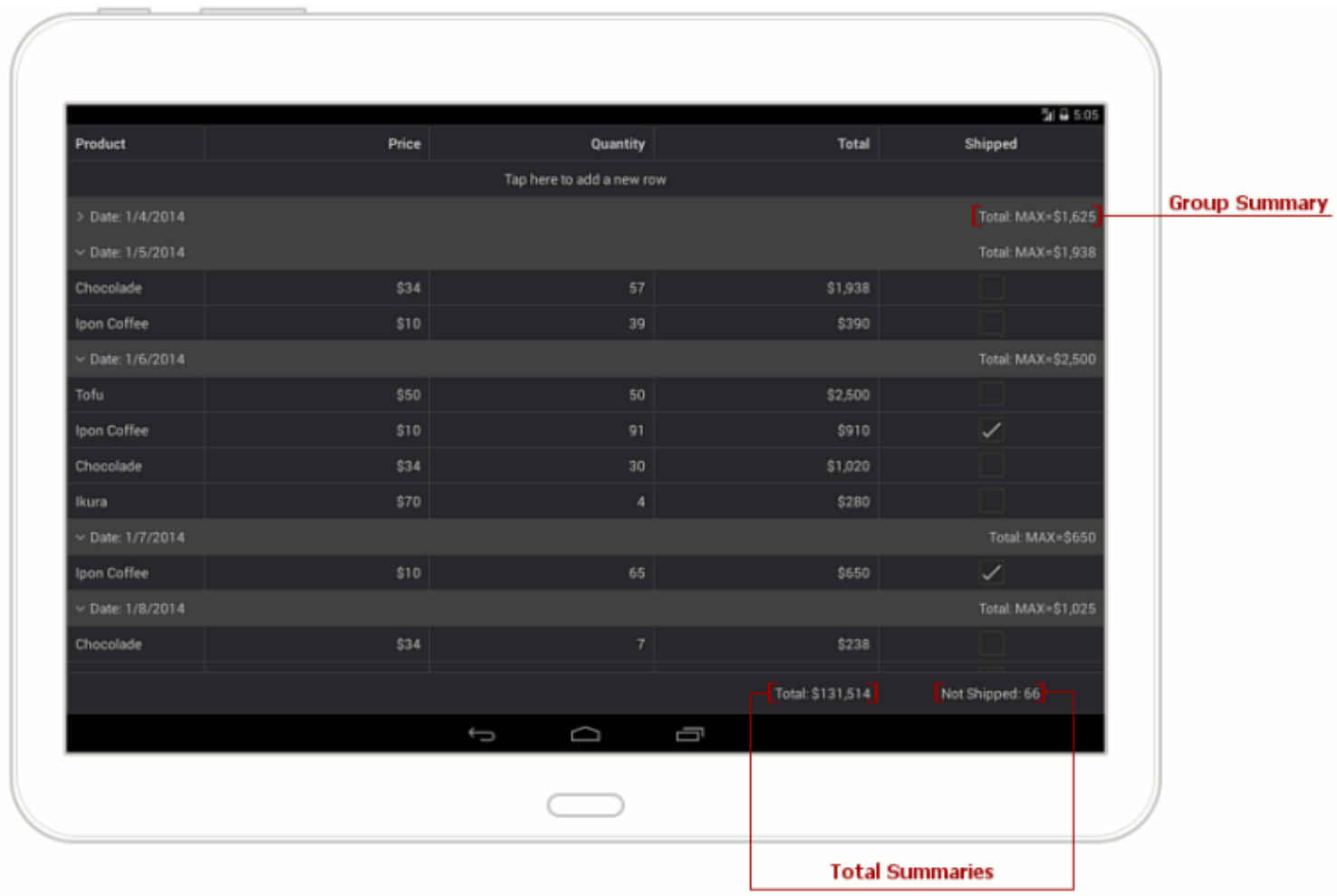
The following data summaries are created.

- Group summary to display the maximum value in a **Total** column for each group of records.
- Total summary to calculate the sum of values by the whole **Total** column.
- Custom total summary to count the number of rows whose value in the **Shipped** column is **false**.

© 2018 Developer Express Inc.




© 2018 Developer Express Inc.




**Xaml**

```
<dxGrid:GridControl x:Name="grid" ItemsSource="{Binding Orders}"
                    NewItemRowVisibility="true"
                    CalculateCustomSummary="OnCalculateCustomSummary">
  <!-- ... -->
  <dxGrid:GridControl.GroupSummaries>
    <dxGrid:GridColumnSummary FieldName="Total" Type="Max"/>
  </dxGrid:GridControl.GroupSummaries>
  <dxGrid:GridControl.TotalSummaries>
    <dxGrid:GridColumnSummary FieldName="Total" Type="Sum" DisplayFormat="Total: {0:C0}"/>
    <dxGrid:GridColumnSummary FieldName="Shipped" Type="Custom" DisplayFormat="Not Shipped: {0}"/>
  </dxGrid:GridControl.TotalSummaries>
  <!-- ... -->
</dxGrid:GridControl>
```

**C#**

```
void OnCalculateCustomSummary(object sender, CustomSummaryEventArgs e) {
    if (e.FieldName.ToString () == "Shipped")
    if (e.IsTotalSummary){
        if (e.SummaryProcess == CustomSummaryProcess.Start) {
            count = 0;
        }
        if (e.SummaryProcess == CustomSummaryProcess.Calculate) {
            if (!(bool)e.FieldValue)
                count++;
            e.TotalValue = count;
        }
    }
}
```

© 2018 Developer Express Inc.

**See Also**
GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
TotalSummaries

© 2018 Developer Express Inc.

# HighlightMenuTargetElements Property

Specifies whether the grid should highlight a row and column for which an end-user invokes a context menu. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public bool HighlightMenuTargetElements { get; set; }
```

### Property Value

**true**, to highlight grid elements; otherwise, **false**.

### Remarks

By default, when an end-user touches and holds a data cell, column header, group row or total summary panel, a context menu is shown, allowing an end-user to manage and shape data in the grid. Depending on the type, a context menu can contain commands to edit and delete a row, as well as to group, sort or create data summaries by a column. To highlight a row and column for which the context menu is invoked, set the **HighlightMenuTargetElements** property to **true**.

© 2018 Developer Express Inc.

HighlightMenuTargetElements = true

HighlightMenuTargetElements = false

**See Also**
GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# HorizontalScrollOffset Property

Gets or sets a horizontal scrolling offset.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public Double HorizontalScrollOffset { get; set; }
```

### Property Value

A double value, which specifies the scrolling offset (in density-independent pixels).

### Remarks

By setting various values to this property, you can scroll grid columns directly from your code. The offset is measured from the right edge of the last column fixed to the left (if any) or from a grid's left edge (if there are no fixed columns).

The code below sets the **Position** column's width as a scrolling offset.

**C#**

```
using DevExpress.Mobile.DataGrid;
     // ...
     grid.HorizontalScrollOffset = grid.Columns["Position"].Width;
     // ...
```

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
Fixed Columns

© 2018 Developer Express Inc.

# IsColumnChooserEnabled Property

Gets or sets whether the column chooser is available to end-users. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

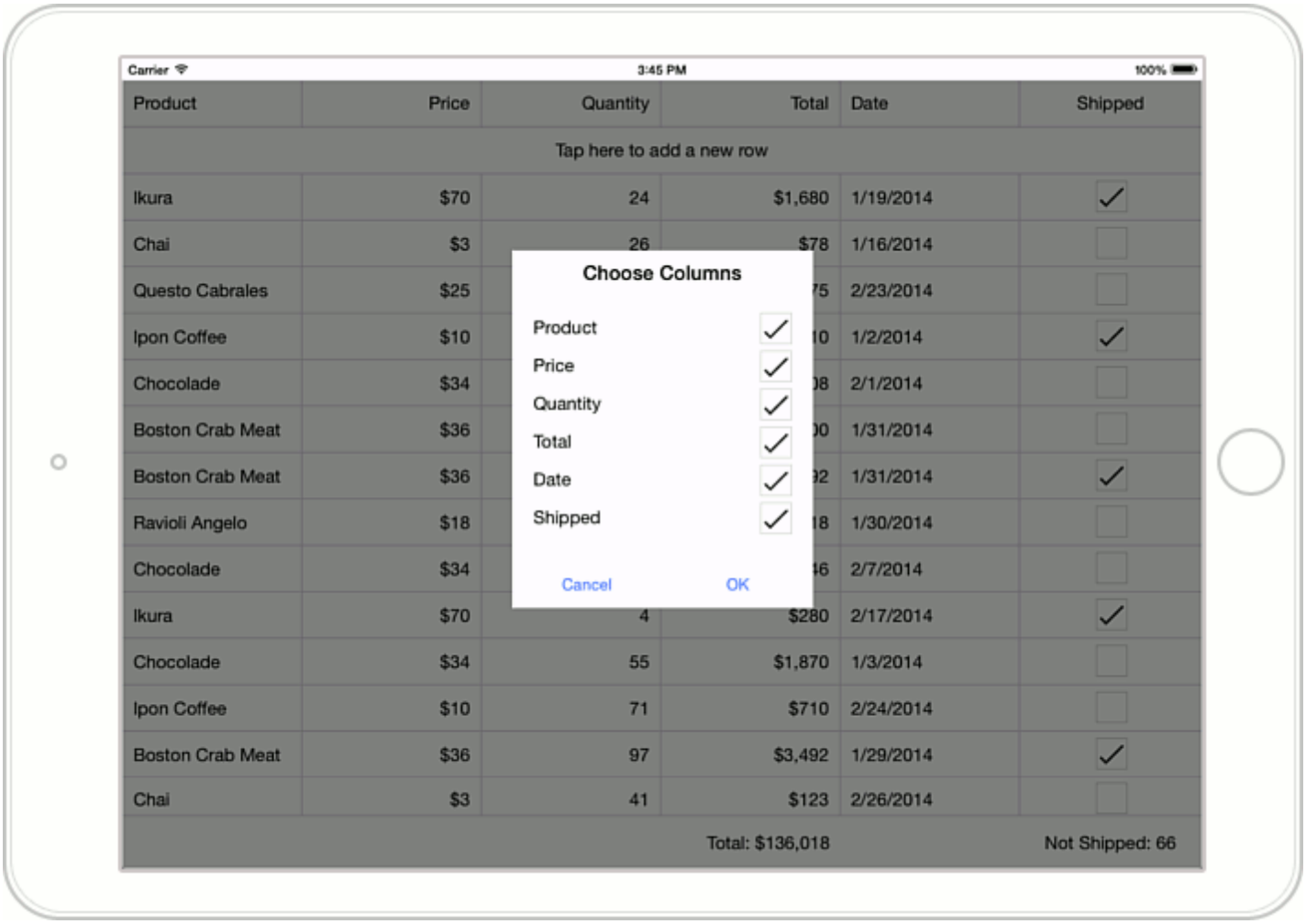
```
public bool IsColumnChooserEnabled { get; set; }
```

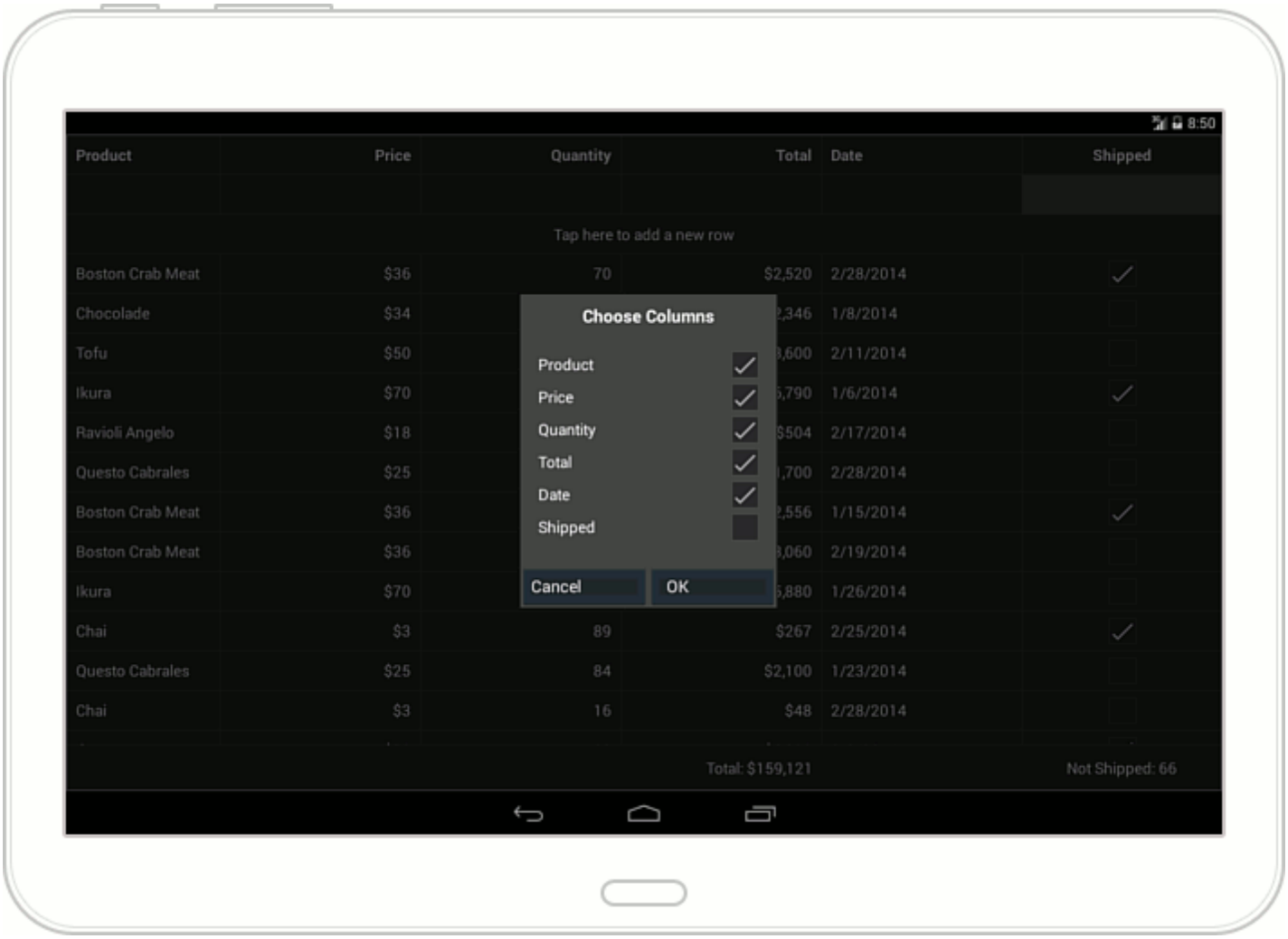
### Property Value

**true**, to enable the column chooser; otherwise, **false**.

### Remarks

The column chooser provides end-users with the capability to show and hide columns from the grid at runtime. By default, the column chooser is shown when an end-user touches and holds a column header and then taps the corresponding item in the invoked menu. If you set the **IsColumnChooserEnabled** property to **false**, this menu item is not displayed.






**See Also**
GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
ShowColumnChooser
HideColumnChooser

© 2018 Developer Express Inc.

# IsColumnMenuEnabled Property

Gets or sets whether a context menu is shown when an end-user touches and holds a column header. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public bool IsColumnMenuEnabled { get; set; }
```

### Property Value

**true**, to enable the column context menu; otherwise, **false**.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# IsGrouped Property

Gets whether the grid's data is grouped.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public bool IsGrouped { get; }
```

### Property Value

**true**, if data grouping is applied; otherwise, **false**.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
GridColumn.GroupInterval
GridColumn.IsGrouped
GroupBy
ClearGrouping
Lesson 4 - Group Data

© 2018 Developer Express Inc.

# IsGroupRowMenuEnabled Property

Gets or sets whether a context menu is shown when an end-user touches and holds a group row. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public bool IsGroupRowMenuEnabled { get; set; }
```

### Property Value

**true**, to enable the group row context menu; otherwise, **false**.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# IsLoadedInVisualTree Property

For internal use.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**
**C#**

```
public bool IsLoadedInVisualTree { get; }
```

**Property Value**

**See Also**
GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# IsLoadMoreEnabled Property

Gets or sets whether the load-more functionality is enabled in the grid. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public bool IsLoadMoreEnabled { get; set; }
```

### Property Value

**true**, to enable the load-more feature; otherwise, **false**.

### Remarks

The GridControl supports dynamic loading of new data records from the bound data source when an end-user scrolls to the bottom of the grid. To enable this functionality, set the **IsLoadMoreEnabled** property to **true**. Then, either define a command in a view model and bind it to the grid using the LoadMoreCommand property, or handle the LoadMore event.

### Example

**Show Me**

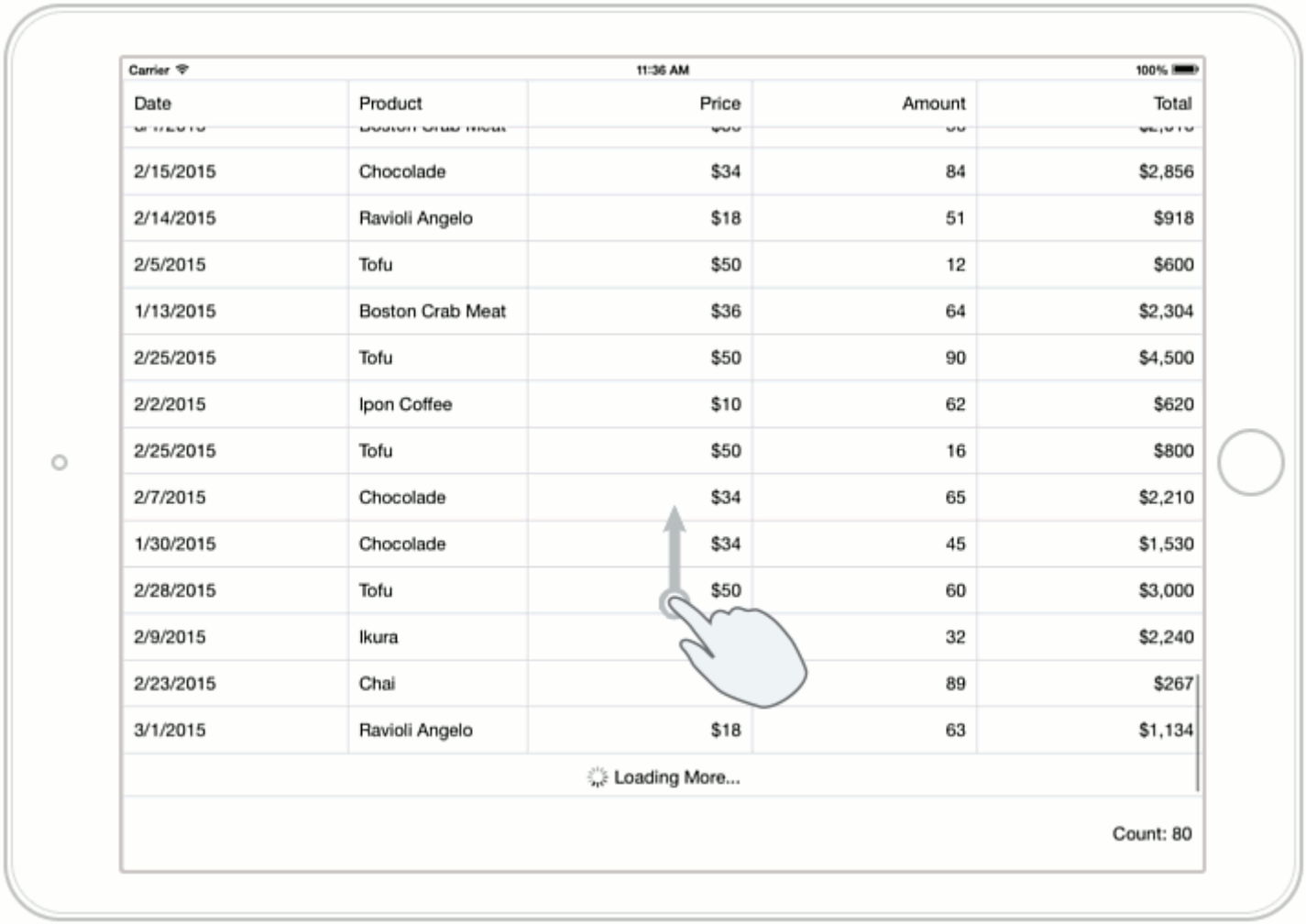
A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T223734.

This example shows how to implement the grid's load-more feature - when an end-user scrolls to the bottom of the grid, a set of new data items is added to the end of the grid. To add this functionality, follow the steps below.

1. Set the IsLoadMoreEnabled property to **true** to enable the grid's load-more feature.
2. Create a command to be executed when an end-user scrolls to the bottom of the grid.
3. Bind the created command to the grid using the LoadMoreCommand property.

In this example, data items for each next load (ten new orders) are generated randomly in code. The maximum number of loads an end-user is allowed to perform is **10**. The count of data items currently loaded to the grid is displayed by the total summary that is automatically updated after each load.

© 2018 Developer Express Inc.


Carrier
11:36 AM
100%
Date
Product
Price
Amount
Total
2/15/2015
Chocolade
$34
84
$2,856
2/14/2015
Ravioli Angelo
$18
51
$918
2/5/2015
Tofu
$50
12
$600
1/13/2015
Boston Crab Meat
$36
64
$2,304
2/25/2015
Tofu
$50
90
$4,500
2/2/2015
Ipon Coffee
$10
62
$620
2/25/2015
Tofu
$50
16
$800
2/7/2015
Chocolade
$34
65
$2,210
1/30/2015
Chocolade
$34
45
$1,530
2/28/2015
Tofu
$50
60
$3,000
2/9/2015
Ikura
32
$2,240
2/23/2015
Chai
89
$267
3/1/2015
Ravioli Angelo
$18
63
$1,134
Loading More...
Count: 80

4:42
Date
Product
Price
Amount
Total
2/7/2015
Chocolade
$34
62
$2,108
2/8/2015
Questo Cabrales
$25
87
$2,175
1/2/2015
Chai
$3
49
$147
1/9/2015
Tofu
$50
11
$550
2/2/2015
Questo Cabrales
$25
33
$825
1/17/2015
Ikura
$70
66
$4,620
2/1/2015
Tofu
$50
19
$950
2/26/2015
Ravioli Angelo
$18
67
$1,206
1/13/2015
Chocolade
$34
35
$1,190
2/17/2015
Tofu
$50
56
$2,800
2/27/2015
Ravioli Angelo
1
$18
2/24/2015
Tofu
19
$950
2/16/2015
Questo Cabrales
$25
55
$1,375
Loading More...
Count: 50

C#

```
 (MainPage.xaml.cs)
public partial class MainPage : ContentPage
{
    OrdersRepository repository;
    public static readonly BindableProperty LoadMoreCommandProperty = BindableProperty.Create<MainPage, Lo
    public static readonly BindableProperty OrdersProperty = BindableProperty.Create<MainPage, ObservableC
    public LoadMoreDataCommand LoadMoreCommand {
        get { return (LoadMoreDataCommand)GetValue(LoadMoreCommandProperty); }
        set { SetValue(LoadMoreCommandProperty, value); }
    }
    public ObservableCollection<Order> Orders {
        get { return (ObservableCollection<Order>)GetValue(OrdersProperty); }
        set { SetValue(OrdersProperty, value); }
    }
    public MainPage () {
        InitializeComponent ();
        BindingContext = this;
        this.repository = new TestOrdersRepository();
        this.Orders = repository.Orders;
        LoadMoreCommand = new LoadMoreDataCommand(ExecuteLoadMoreCommand);
    }
    void ExecuteLoadMoreCommand() {
        repository.LoadMoreOrders();
        Orders = repository.Orders;
    }
}
public class LoadMoreDataCommand : ICommand {
    readonly Action execute;
    int numOfLoadMore;
    public event EventHandler CanExecuteChanged;
    public LoadMoreDataCommand(Action execute) {
        this.execute = execute;
    }
    public bool CanExecute(object parameter) {
        if (numOfLoadMore < 10) {
            return true;
        }
        return false;
    }
    public void Execute(object parameter) {
        numOfLoadMore++;
        this.execute();
    }
}
```

C#

© 2018 Developer Express Inc.

```
 (OrdersRepository.cs)
public abstract class OrdersRepository {
    public ObservableCollection<Order> Orders { get; private set; }
    public OrdersRepository() {
        this.Orders = new ObservableCollection<Order>();
    }
    protected abstract Order GenerateOrder(int number);
    internal void LoadMoreOrders() {
        ObservableCollection<Order> newOrders = new ObservableCollection<Order>();
        foreach (var order in Orders)
            newOrders.Add(order);
        for (int i = 0; i < 10; i++)
            newOrders.Add(GenerateOrder(i));
        Orders = newOrders;
    }
}
public class TestOrdersRepository : OrdersRepository {
    const int orderCount = 50;
    readonly List<Product> products;
    readonly Random random;
    public TestOrdersRepository() : base() {
        this.random = new Random((int)DateTime.Now.Ticks);
        this.products = new List<Product>();
        GenerateProducts();
        for (int i = 0; i < orderCount; i++)
            Orders.Add(GenerateOrder(i));
    }
    protected override Order GenerateOrder(int number) {
        Order order = new Order(new DateTime(2015, 1, 1).AddDays(random.Next(0, 60)),
            number % 3 == 0, RandomItem<Product>(products), random.Next(1, 100));
        return order;
    }
    T RandomItem<T>(IList<T> list) {
        int index = (int)(random.NextDouble() * 0.99 * (list.Count));
        return list[index];
    }
    void GenerateProducts() {
        products.Add(new Product("Tofu", 50));
        products.Add(new Product("Chocolade", 34));
        products.Add(new Product("Ikura", 70));
        products.Add(new Product("Chai", 3));
        products.Add(new Product("Boston Crab Meat", 36));
        products.Add(new Product("Ravioli Angelo", 18));
        products.Add(new Product("Ipon Coffee", 10));
        products.Add(new Product("Questo Cabrales", 25));
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl ItemsSource="{Binding Orders}"
                    LoadMoreCommand="{Binding LoadMoreCommand}" IsLoadMore
```

## See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# IsPullToRefreshEnabled Property

Gets or sets whether the pull-to-refresh functionality is enabled in the grid. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

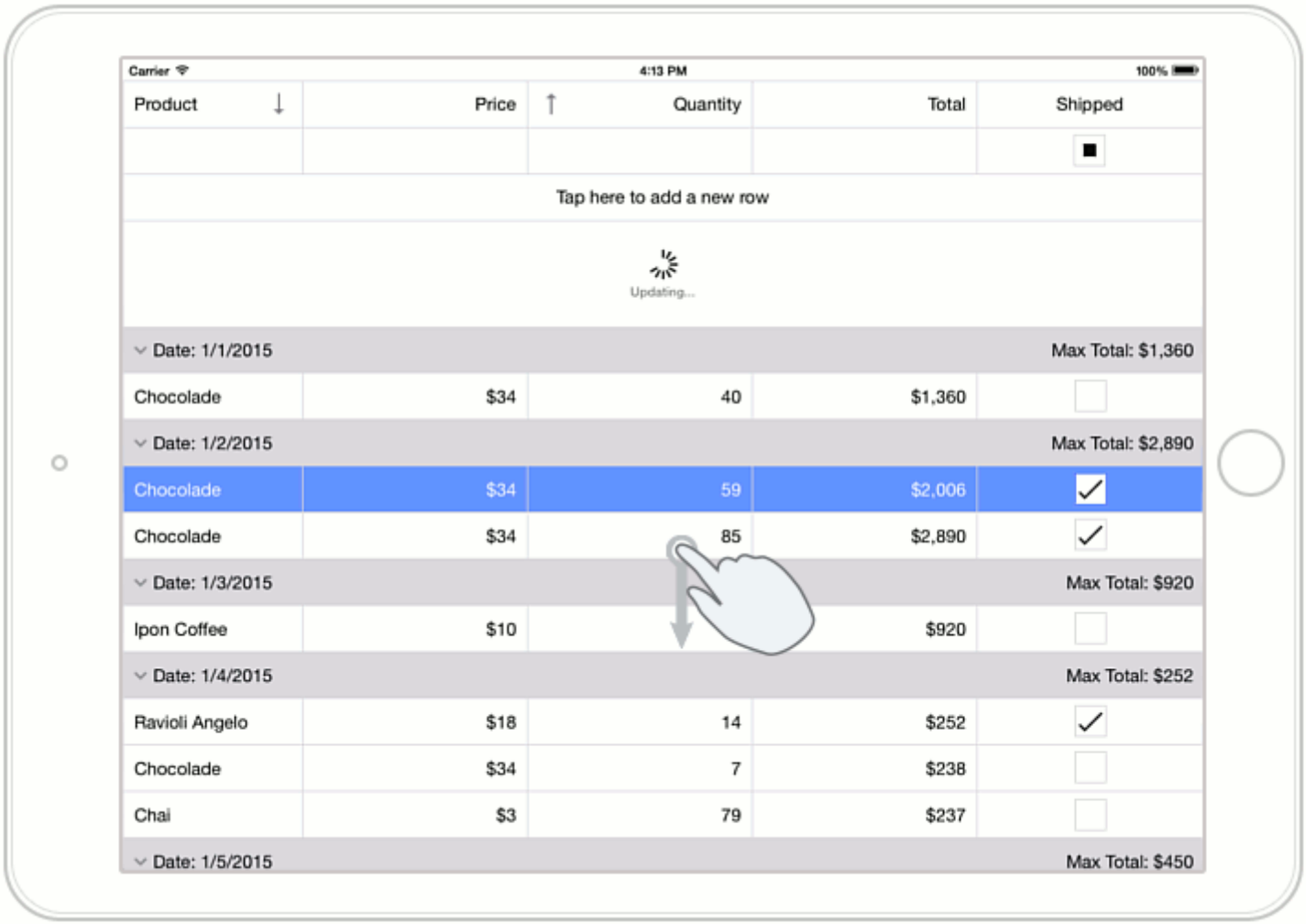
```
public bool IsPullToRefreshEnabled { get; set; }
```

### Property Value

**true**, to enable the pull-to-refresh feature; otherwise, **false**.

### Remarks

The GridControl's pull-to-refresh feature allows end-users to request a grid content update with the pull-down gesture.




To enable this functionality, set the **IsPullToRefreshEnabled** property to **true**. Then, either define a refresh command in a view model and bind it to the grid using the PullToRefreshCommand property, or handle the PullToRefresh event.

### Example

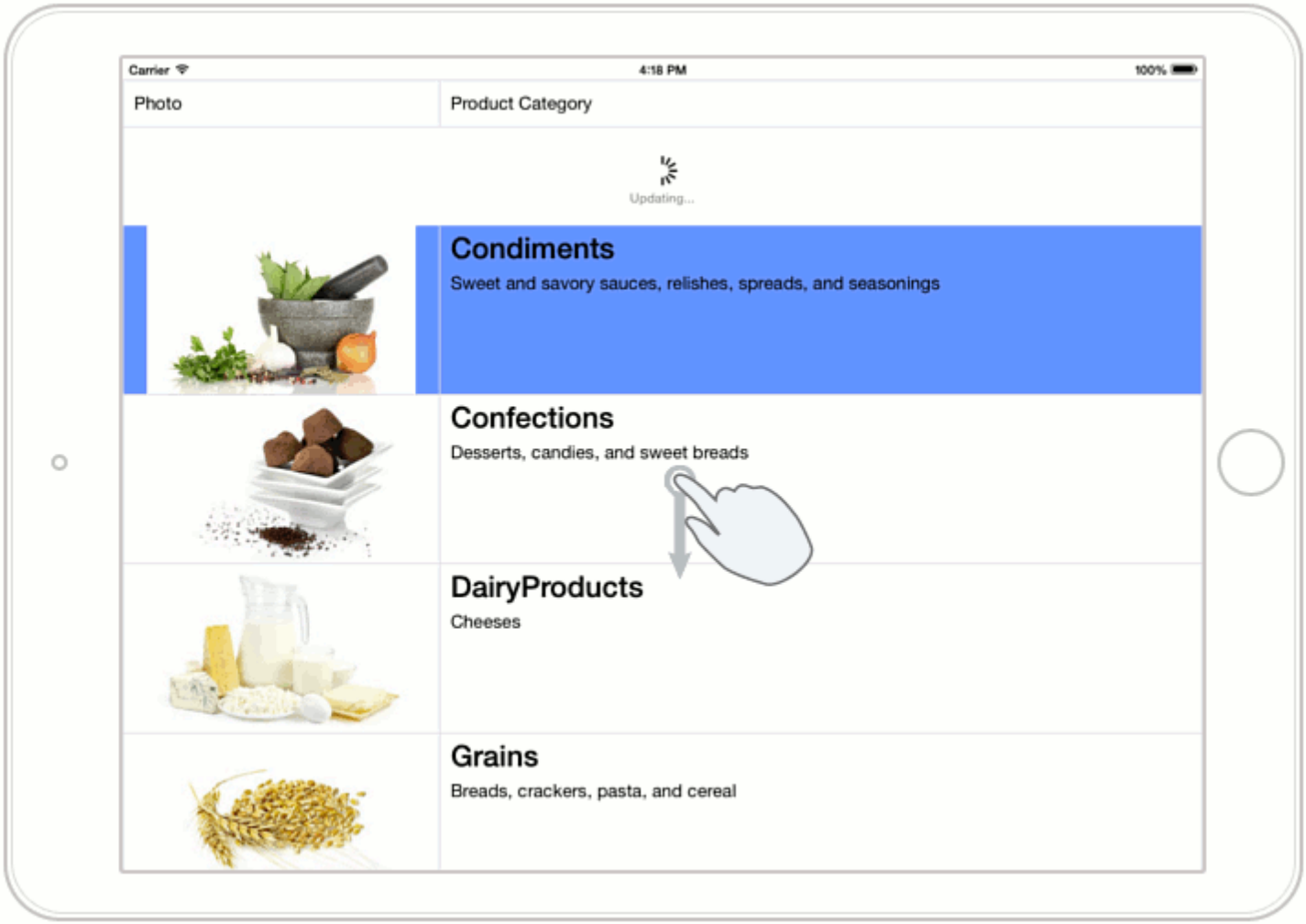
| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T221404 . | |

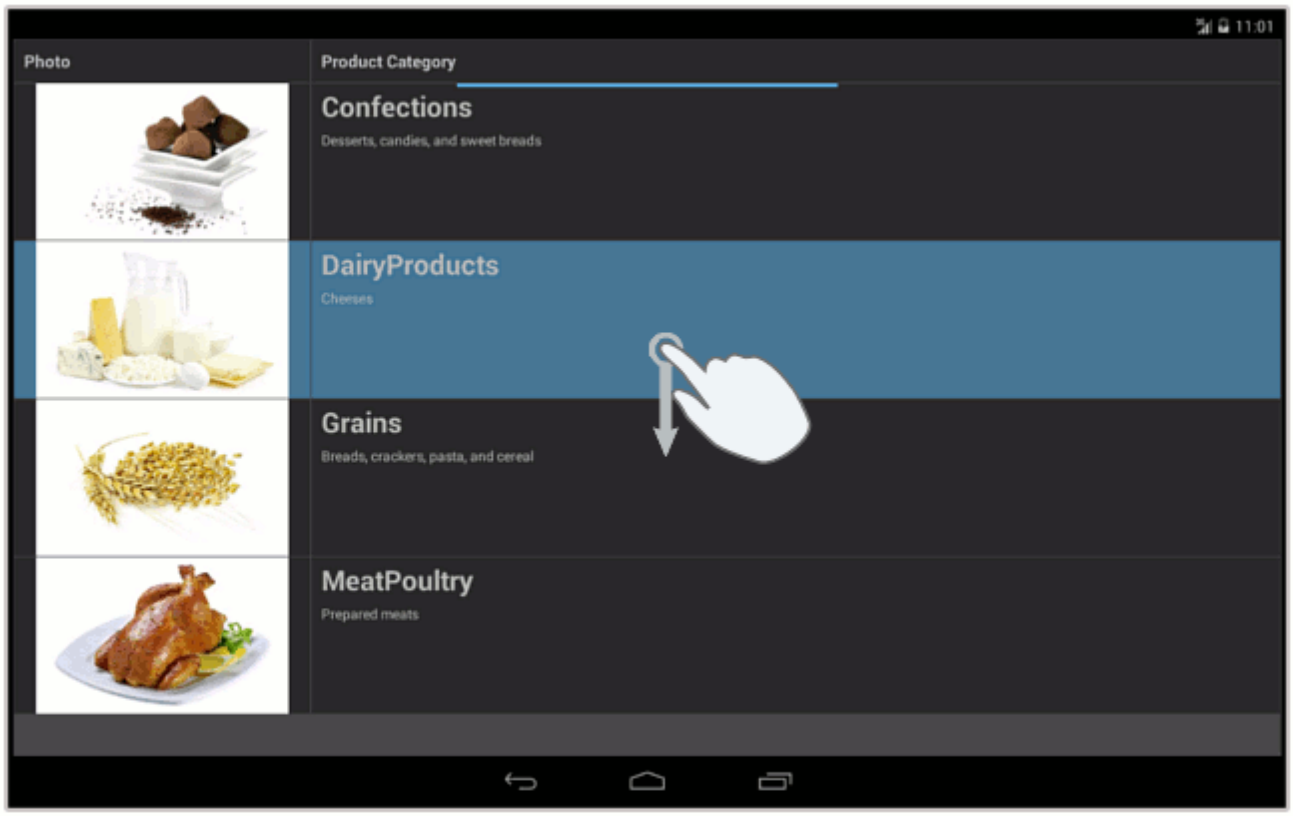
© 2018 Developer Express Inc.

This example shows how to add the pull-to-refresh functionality to a grid to allow your end-users update grid content with the pull-down gesture. To do this, follow the steps below.

1. Set the IsPullToRefreshEnabled property to **true** to enable the grid's pull-to-refresh feature.
2. Create a command to be executed when an end-user pulls the grid down.
3. Bind the created command to the grid using the PullToRefreshCommand property.

© 2018 Developer Express Inc.


Carrier
4:18 PM
100%
Photo
Product Category
Updating...
Condiments
Sweet and savory sauces, relishes, spreads, and seasonings
Confections
Desserts, candies, and sweet breads
DairyProducts
Cheeses
Grains
Breads, crackers, pasta, and cereal


11:01
Photo
Product Category
Confections
Desserts, candies, and sweet breads
DairyProducts
Cheeses
Grains
Breads, crackers, pasta, and cereal
MeatPoultry
Prepared meats

**C#**

```
 (MainPage.xaml.cs)
public partial class MainPage : ContentPage
{
    ProductsRepository repository;
    public static readonly BindableProperty PullToRefreshProperty = BindableProperty.Create<MainPage, ICor
    public static readonly BindableProperty ProductsProperty = BindableProperty.Create<MainPage, Observabl
    public Command PullToRefreshCommand {
        get { return (Command)GetValue(PullToRefreshProperty); }
        set { SetValue(PullToRefreshProperty, value); }
    }
    public ObservableCollection<Product> Products {
        get { return (ObservableCollection<Product>)GetValue(ProductsProperty); }
        set { SetValue(ProductsProperty, value); }
    }
    public MainPage () {
        InitializeComponent ();
        BindingContext = this;
        this.repository = new TestProductsRepository();
        this.Products = repository.Products;
        PullToRefreshCommand = new Command(ExecutePullToRefreshCommand);
    }
    void ExecutePullToRefreshCommand() {
        repository.RefreshProducts();
        Products = repository.Products;
    }
}
```

**C#**

```
 (Product.cs)
public class Product : ModelObject {
    string name;
    public string Name {
        get { return name; }
        set {
            name = value;
            if (Photo == null)
                Photo = ImageSource.FromResource (value);
        }
    }
    public Product(string name) {
        this.Name = name;
    }
    public ImageSource Photo { get; set; }
    public string Description { get; set; }
}
```

**C#**

```
 (ProductsRepository.cs)
public abstract class ProductsRepository {
    public ObservableCollection<Product> Products { get; set; }
    public ProductsRepository() {
        this.Products = new ObservableCollection<Product>();
    }
    protected abstract ObservableCollection<Product> GenerateAvailableProd
    internal void RefreshProducts() {
        int index = new Random().Next (0, 5);
        Products = GenerateAvailableProducts (index);
    }
}
public class TestProductsRepository : ProductsRepository {
    readonly List<Product> products;
    public TestProductsRepository() : base() {
        this.products = new List<Product>();
        GenerateAllProducts();
        int index = new Random().Next (0, 5);
        Products = GenerateAvailableProducts (index);
    }
    protected override ObservableCollection<Product> GenerateAvailableProd
        ObservableCollection<Product> availableProducts = new ObservableCo
        for (int i = 0; i < 4; i++)
            availableProducts.Add(products[number+i]);
        return availableProducts;
    }
    void GenerateAllProducts() {
        products.Add(new Product("Beverages") {Description = "Soft drinks,
        products.Add(new Product("Condiments"){Description = "Sweet and sa
        products.Add(new Product("Confections"){Description = "Desserts, c
        products.Add(new Product("DairyProducts"){Description = "Cheeses"}
        products.Add(new Product("Grains"){Description = "Breads, crackers
        products.Add(new Product("MeatPoultry"){Description = "Prepared me
        products.Add(new Product("Produce"){Description = "Dried fruit and
        products.Add(new Product("Seafood"){Description = "Seaweed and fis
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl ItemsSource="{Binding Products}" RowHeight="160"
                    IsPullToRefreshEnabled="true" PullToRefreshCommand="{B
```

## See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# IsReadOnly Property

Gets or sets whether the grid is read-only. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public bool IsReadOnly { get; set; }
```

### Property Value

**true**, if the grid is read-only; otherwise, **false**.

### Remarks

If the **IsReadOnly** property is set to **true**, end-users are not allowed to change cell values, create new data rows and delete existing rows. To set the read-only status for individual columns, use the column's GridColumn.IsReadOnly property.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# IsRowCellMenuEnabled Property

Gets or sets whether a context menu is shown when an end-user touches and holds a data row cell. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public bool IsRowCellMenuEnabled { get; set; }
```

### Property Value

**true**, to enable the row cell context menu; otherwise, **false**.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# IsTotalSummaryMenuEnabled Property

Gets or sets whether a context menu is shown when an end-user touches and holds a total summary panel. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public bool IsTotalSummaryMenuEnabled { get; set; }
```

### Property Value

**true**, to enable the total summary context menu; otherwise, **false**.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# ItemsSource Property

Gets or sets the grid's data source. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public object ItemsSource { get; set; }
```

### Property Value

An object that implements either the *IList* or *IList<T>* interface and specifies the data source from which the grid retrieves its data. Also, you can use the DataTable as the data source for the grid.

### Remarks

The Bind a Grid to Data lesson of the Getting Started tutorial explains how to use the **ItemsSource** property to provide a data source for a GridControl instance.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# LeftSwipeButtons Property

Provides access to objects containing information on buttons shown on the left side of a data row when an end-user swipes left to right over a data row.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public ObservableCollection<SwipeButtonInfo> LeftSwipeButtons { get; }
```

### Property Value

A collection of objects specifying left swipe buttons added to the grid.

### Example

**Show Me**

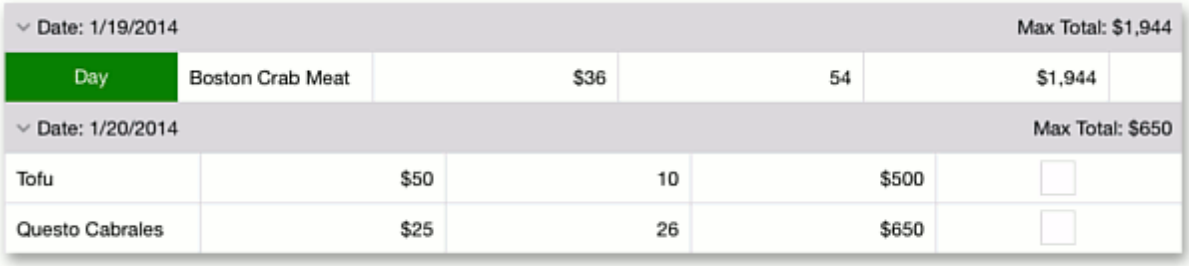
A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T178764 .

This example shows how to extend the grid's UI with additional buttons to be shown when an end-user swipes left to right, or right to left, over a data row. Swipe Buttons. To create a swipe button, add a SwipeButtonInfo object to the LeftSwipeButtons or RightSwipeButtons collection. Use this object's SwipeButtonInfo.Caption, SwipeButtonInfo.Width, SwipeButtonInfo.BackgroundColor and SwipeButtonInfo.TextColor properties to specify button appearance.

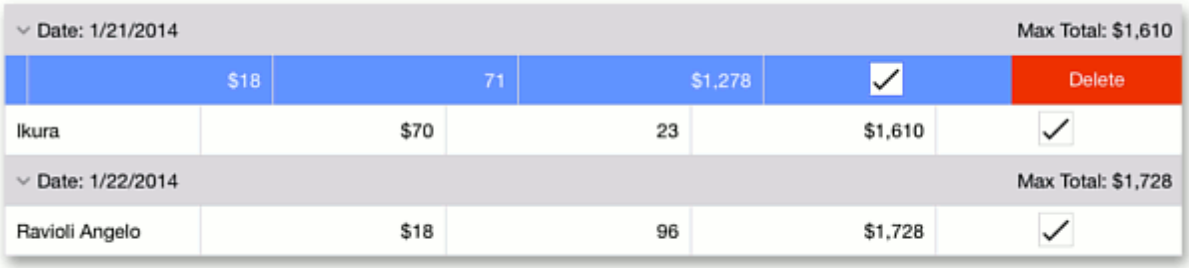
To assign a custom action to a button, handle the SwipeButtonClick event. The SwipeButtonShowing event allows you to specify when a swipe button should be visible.

In the example, two swipe buttons are created.

- When an end-user swipes any data row left to right, the **Day** button is shown on the left side of the row. Tapping this button displays the day of the week when the corresponding order was made.




- When an end-user swipes right to left, a data row which corresponds to an order that is already shipped (the value in the **Shipped** column is **true**), the **Delete** button is shown on the right side of the row. Tapping this button removes the corresponding data row.




**C#**

© 2018 Developer Express Inc.

```
 (MainPage.xaml.cs)
void OnSwipeButtonShowing(object sender, SwipeButtonShowingEventArgs e)
{
    if ((!(Boolean)grid.GetCellValue(e.RowHandle, "Shipped"))
        && (e.ButtonInfo.ButtonName == "RightButton"))
    {
        e.IsVisible = false;
    }
}
void OnSwipeButtonClick(object sender, SwipeButtonEventArgs e)
{
    if (e.ButtonInfo.ButtonName == "LeftButton")
    {
        DateTime orderDate = (DateTime)grid.GetCellValue(e.RowHandle, "Date");
        string orderDateDay = orderDate.ToString("dddd");
        DisplayAlert("Alert from " + e.ButtonInfo.ButtonName, "Day: " + orderDateDay, "OK");
    }
    if (e.ButtonInfo.ButtonName == "RightButton")
    {
        grid.DeleteRow(e.RowHandle);
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl.LeftSwipeButtons>
  <dxGrid:SwipeButtonInfo ButtonName="LeftButton" Caption="Day" BackgroundColor="Green" Width="150"/>
</dxGrid:GridControl.LeftSwipeButtons>
<dxGrid:GridControl.RightSwipeButtons>
  <dxGrid:SwipeButtonInfo ButtonName="RightButton" Caption="Delete" BackgroundColor="Red" Width="150"/>
</dxGrid:GridControl.RightSwipeButtons>
```

## See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# LoadMoreCommand Property

Gets or sets the command executed when an end-user scrolls to the bottom of the grid to load new data items. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public ICommand LoadMoreCommand { get; set; }
```

### Property Value

A command that exposes the System.Windows.Input.ICommand interface.

### Remarks

The GridControl supports dynamic loading of new data records from the bound data source when an end-user scrolls to the bottom of the grid. To enable this functionality, set the IsLoadMoreEnabled property to **true**. Then, either define a command in a view model and bind it to the grid using the **LoadMoreCommand** property, or handle the LoadMore event.

### Example

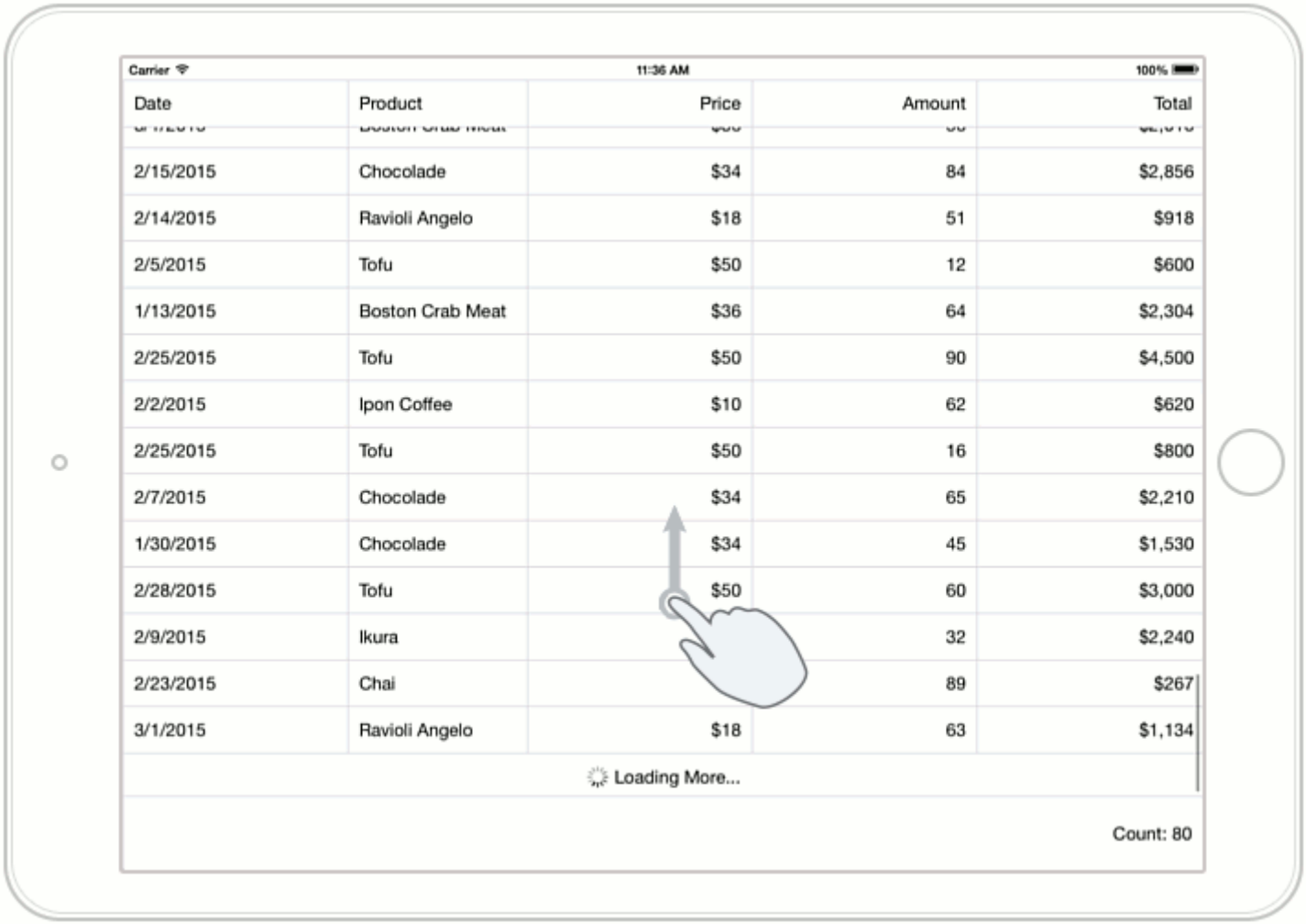
| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T223734. | |

This example shows how to implement the grid's load-more feature - when an end-user scrolls to the bottom of the grid, a set of new data items is added to the end of the grid. To add this functionality, follow the steps below.

1. Set the IsLoadMoreEnabled property to **true** to enable the grid's load-more feature.
2. Create a command to be executed when an end-user scrolls to the bottom of the grid.
3. Bind the created command to the grid using the LoadMoreCommand property.

In this example, data items for each next load (ten new orders) are generated randomly in code. The maximum number of loads an end-user is allowed to perform is **10**. The count of data items currently loaded to the grid is displayed by the total summary that is automatically updated after each load.

© 2018 Developer Express Inc.


Carrier
11:36 AM
100%
Date
Product
Price
Amount
Total
2/15/2015
Chocolade
$34
84
$2,856
2/14/2015
Ravioli Angelo
$18
51
$918
2/5/2015
Tofu
$50
12
$600
1/13/2015
Boston Crab Meat
$36
64
$2,304
2/25/2015
Tofu
$50
90
$4,500
2/2/2015
Ipon Coffee
$10
62
$620
2/25/2015
Tofu
$50
16
$800
2/7/2015
Chocolade
$34
65
$2,210
1/30/2015
Chocolade
$34
45
$1,530
2/28/2015
Tofu
$50
60
$3,000
2/9/2015
Ikura
32
$2,240
2/23/2015
Chai
89
$267
3/1/2015
Ravioli Angelo
$18
63
$1,134
Loading More...
Count: 80

4:42
Date
Product
Price
Amount
Total
2/7/2015
Chocolade
$34
62
$2,108
2/8/2015
Questo Cabrales
$25
87
$2,175
1/2/2015
Chai
$3
49
$147
1/9/2015
Tofu
$50
11
$550
2/2/2015
Questo Cabrales
$25
33
$825
1/17/2015
Ikura
$70
66
$4,620
2/1/2015
Tofu
$50
19
$950
2/26/2015
Ravioli Angelo
$18
67
$1,206
1/13/2015
Chocolade
$34
35
$1,190
2/17/2015
Tofu
$50
56
$2,800
2/27/2015
Ravioli Angelo
1
$18
2/24/2015
Tofu
19
$950
2/16/2015
Questo Cabrales
$25
55
$1,375
Loading More...
Count: 50

C#

```
 (MainPage.xaml.cs)
public partial class MainPage : ContentPage
{
    OrdersRepository repository;
    public static readonly BindableProperty LoadMoreCommandProperty = BindableProperty.Create<MainPage, Lo
    public static readonly BindableProperty OrdersProperty = BindableProperty.Create<MainPage, ObservableC
    public LoadMoreDataCommand LoadMoreCommand {
        get { return (LoadMoreDataCommand)GetValue(LoadMoreCommandProperty); }
        set { SetValue(LoadMoreCommandProperty, value); }
    }
    public ObservableCollection<Order> Orders {
        get { return (ObservableCollection<Order>)GetValue(OrdersProperty); }
        set { SetValue(OrdersProperty, value); }
    }
    public MainPage () {
        InitializeComponent ();
        BindingContext = this;
        this.repository = new TestOrdersRepository();
        this.Orders = repository.Orders;
        LoadMoreCommand = new LoadMoreDataCommand(ExecuteLoadMoreCommand);
    }
    void ExecuteLoadMoreCommand() {
        repository.LoadMoreOrders();
        Orders = repository.Orders;
    }
}
public class LoadMoreDataCommand : ICommand {
    readonly Action execute;
    int numOfLoadMore;
    public event EventHandler CanExecuteChanged;
    public LoadMoreDataCommand(Action execute) {
        this.execute = execute;
    }
    public bool CanExecute(object parameter) {
        if (numOfLoadMore < 10) {
            return true;
        }
        return false;
    }
    public void Execute(object parameter) {
        numOfLoadMore++;
        this.execute();
    }
}
```

C#

© 2018 Developer Express Inc.

```
 (OrdersRepository.cs)
public abstract class OrdersRepository {
    public ObservableCollection<Order> Orders { get; private set; }
    public OrdersRepository() {
        this.Orders = new ObservableCollection<Order>();
    }
    protected abstract Order GenerateOrder(int number);
    internal void LoadMoreOrders() {
        ObservableCollection<Order> newOrders = new ObservableCollection<Order>();
        foreach (var order in Orders)
            newOrders.Add(order);
        for (int i = 0; i < 10; i++)
            newOrders.Add(GenerateOrder(i));
        Orders = newOrders;
    }
}
public class TestOrdersRepository : OrdersRepository {
    const int orderCount = 50;
    readonly List<Product> products;
    readonly Random random;
    public TestOrdersRepository() : base() {
        this.random = new Random((int)DateTime.Now.Ticks);
        this.products = new List<Product>();
        GenerateProducts();
        for (int i = 0; i < orderCount; i++)
            Orders.Add(GenerateOrder(i));
    }
    protected override Order GenerateOrder(int number) {
        Order order = new Order(new DateTime(2015, 1, 1).AddDays(random.Next(0, 60)),
            number % 3 == 0, RandomItem<Product>(products), random.Next(1, 100));
        return order;
    }
    T RandomItem<T>(IList<T> list) {
        int index = (int)(random.NextDouble() * 0.99 * (list.Count));
        return list[index];
    }
    void GenerateProducts() {
        products.Add(new Product("Tofu", 50));
        products.Add(new Product("Chocolade", 34));
        products.Add(new Product("Ikura", 70));
        products.Add(new Product("Chai", 3));
        products.Add(new Product("Boston Crab Meat", 36));
        products.Add(new Product("Ravioli Angelo", 18));
        products.Add(new Product("Ipon Coffee", 10));
        products.Add(new Product("Questo Cabrales", 25));
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl ItemsSource="{Binding Orders}"
                    LoadMoreCommand="{Binding LoadMoreCommand}" IsLoadMore
```

## See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# NewItemRowVisibility Property

Gets or sets whether the **New Item Row** is shown in the grid to allow end-users to enter new data records. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public bool NewItemRowVisibility { get; set; }
```

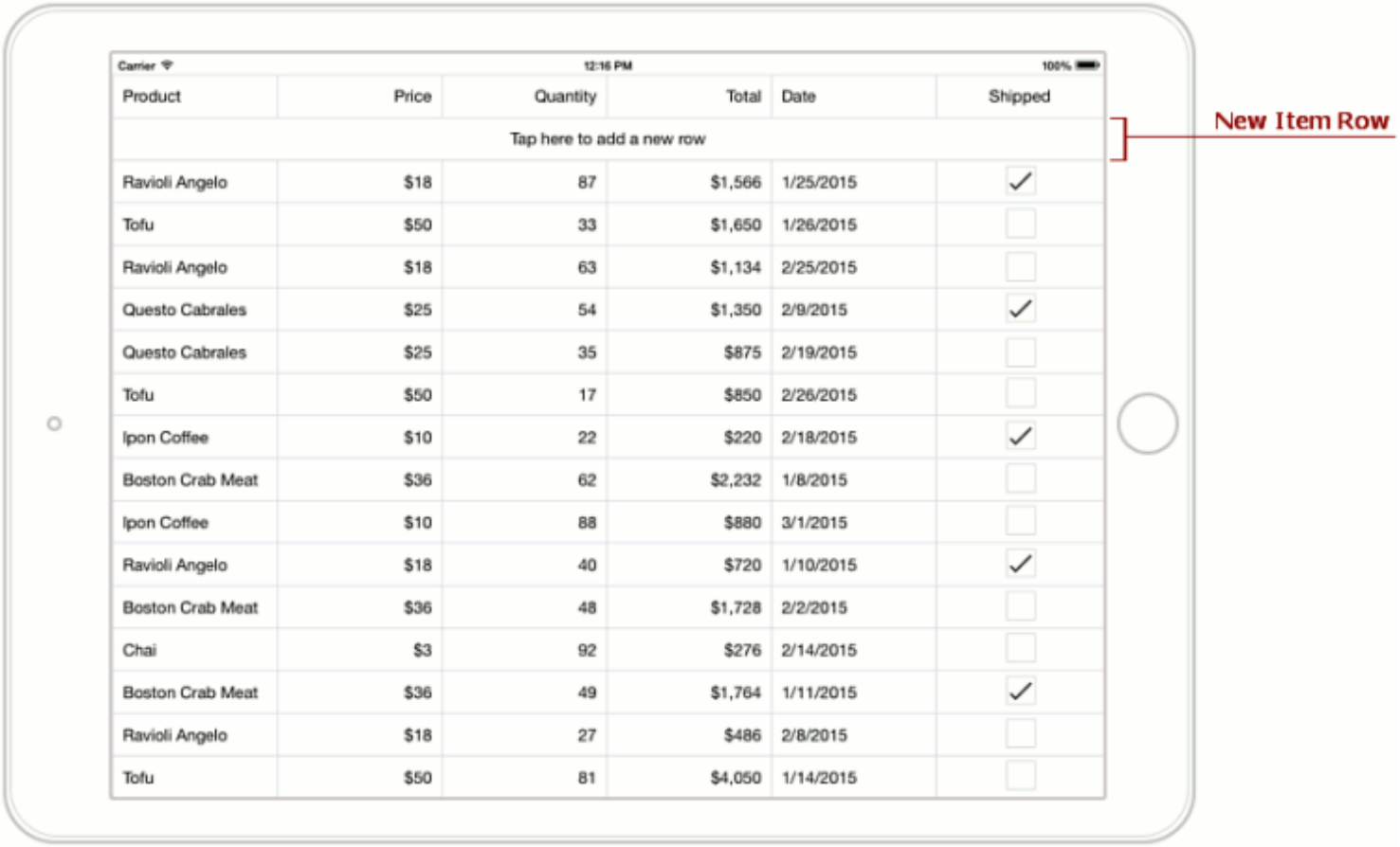
### Property Value

**true**, to display the new item row; otherwise, **false**.

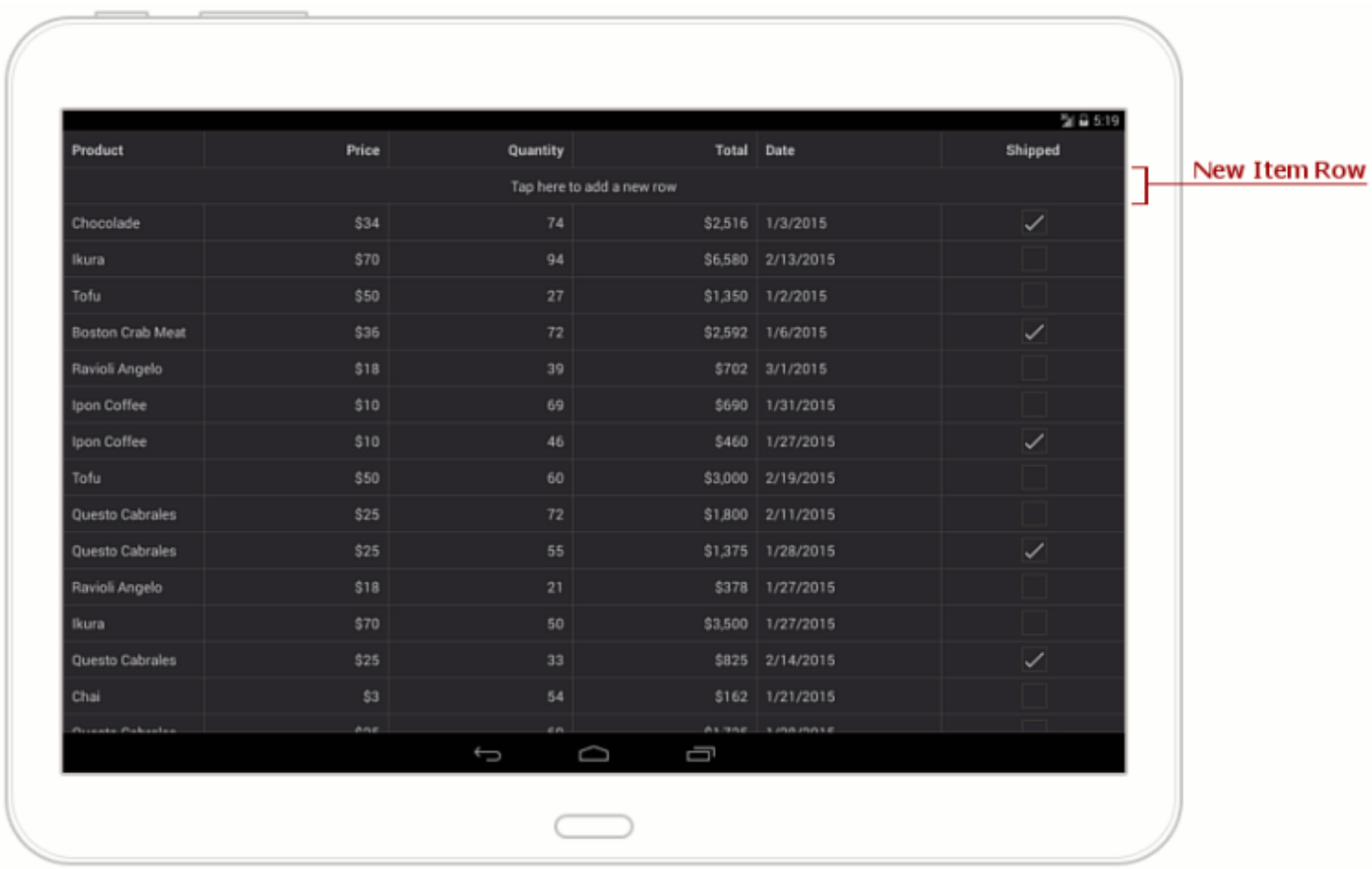
### Remarks

If the **NewItemRowVisibility** property is set to **true**, the **New Item Row** is shown above grid's rows. When an end-user taps it, a row editor (an in-place editor or edit form, depending on the row edit mode set in the grid) is invoked. After an end-user sets required cell values in this editor and taps **Apply** (or **Done**), a new row is added.

To set the default values for a new row, handle the InitNewRow event.




© 2018 Developer Express Inc.




**See Also**
GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
Rows Overview

© 2018 Developer Express Inc.

# OptionsExportCsv Property

Gets or sets how grid data is exported to CSV format.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public CsvExportOptions OptionsExportCsv { get; set; }
```

### Property Value

A CsvExportOptions object that defines CSV-specific export options.

### Remarks

Create an instance of the CsvExportOptions class and assign it to the **OptionsExportCsv** property to apply required settings when exporting data from the GridControl to CSV format.

To export grid data in CSV format, use the ExportToExcel method with the ExportTarget.Csv value passed as the *format* parameter.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
OptionsExportXls
OptionsExportXlsx

© 2018 Developer Express Inc.

# OptionsExportXls Property

Gets or sets how grid data is exported to XLS format.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public XlsExportOptions OptionsExportXls { get; set; }
```

### Property Value

An XlsExportOptions object that defines XLS-specific export options.

### Remarks

Create an instance of the XlsExportOptions class and assign it to the **OptionsExportXls** property to apply required settings when exporting data from the GridControl to XLS format.

To export grid data in XLS format, use the ExportToExcel method with the ExportTarget.Xls value passed as the *format* parameter.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
OptionsExportXlsx
OptionsExportCsv

© 2018 Developer Express Inc.

# OptionsExportXlsx Property

Gets or sets how grid data is exported to XLSX format.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public XlsxExportOptions OptionsExportXlsx { get; set; }
```

### Property Value

An XlsxExportOptions object that defines XLSX-specific export options.

### Remarks

Create an instance of the XlsxExportOptions class and assign it to the **OptionsExportXlsx** property to apply required settings when exporting data from the GridControl to XLSX format.

To export grid data in XLSX format, use the ExportToExcel method with the ExportTarget.Xlsx value passed as the *format* parameter.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
OptionsExportXls
OptionsExportCsv

© 2018 Developer Express Inc.

# PullToRefreshCommand Property

Gets or sets the command executed when an end-user pulls the grid down to refresh data. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

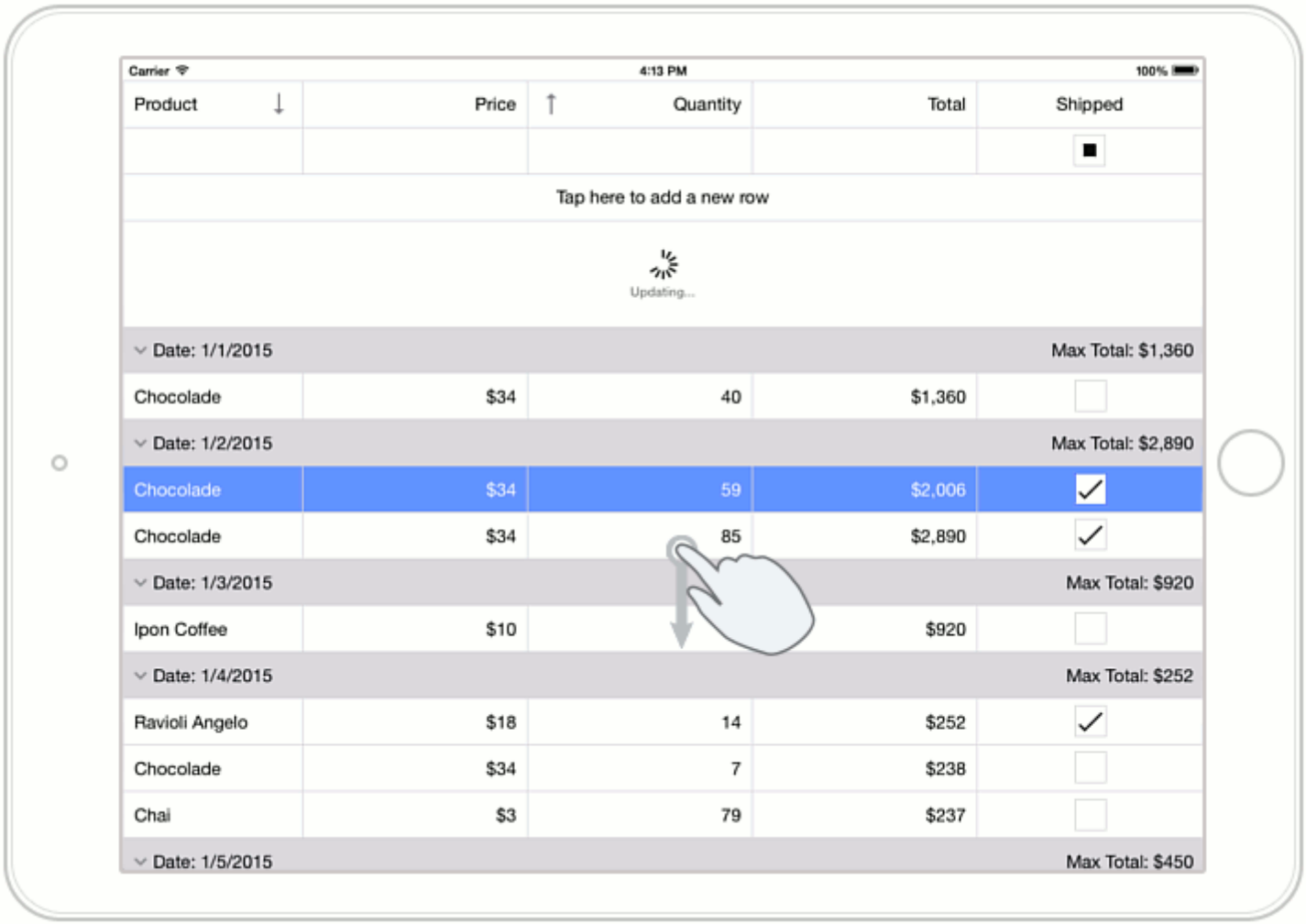
```
public ICommand PullToRefreshCommand { get; set; }
```

### Property Value

A command that exposes the System.Windows.Input.ICommand interface.

### Remarks

The GridControl's pull-to-refresh feature allows end-users to request a grid content update with the pull-down gesture.




To enable this functionality, set the IsPullToRefreshEnabled property to **true**. Then, either define a refresh command in a view model and bind it to the grid using the **PullToRefreshCommand** property, or handle the PullToRefresh event.

### Example

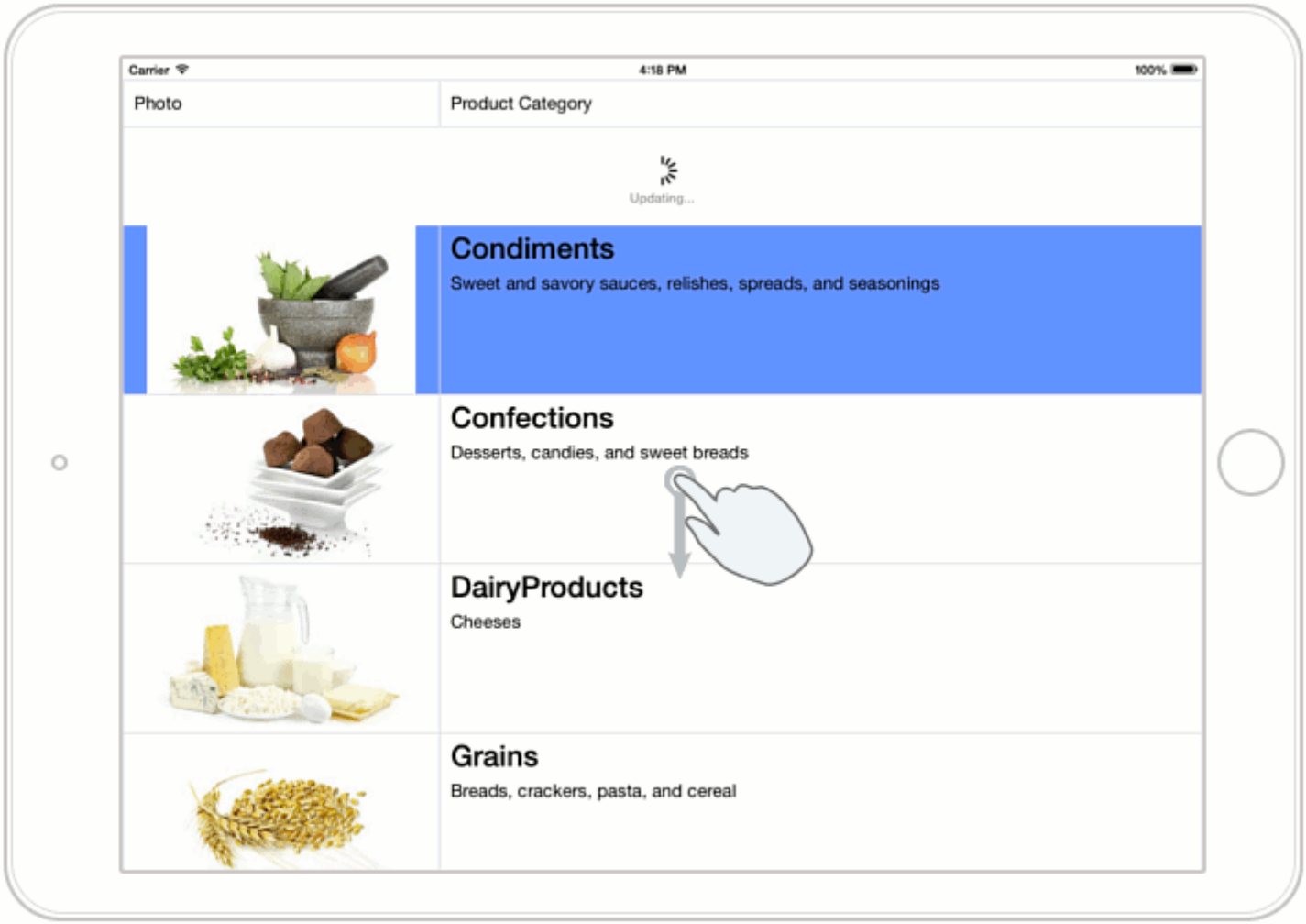
| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T221404 . | |

© 2018 Developer Express Inc.

This example shows how to add the pull-to-refresh functionality to a grid to allow your end-users update grid content with the pull-down gesture. To do this, follow the steps below.

1. Set the IsPullToRefreshEnabled property to **true** to enable the grid's pull-to-refresh feature.
2. Create a command to be executed when an end-user pulls the grid down.
3. Bind the created command to the grid using the PullToRefreshCommand property.

© 2018 Developer Express Inc.


Carrier
4:18 PM
100%
Photo
Product Category
Updating...
Condiments
Sweet and savory sauces, relishes, spreads, and seasonings
Confections
Desserts, candies, and sweet breads
DairyProducts
Cheeses
Grains
Breads, crackers, pasta, and cereal

11:01
Photo
Product Category
Confections
Desserts, candies, and sweet breads
DairyProducts
Cheeses
Grains
Breads, crackers, pasta, and cereal
MeatPoultry
Prepared meats

**C#**

```
 (MainPage.xaml.cs)
public partial class MainPage : ContentPage
{
    ProductsRepository repository;
    public static readonly BindableProperty PullToRefreshProperty = BindableProperty.Create<MainPage, ICor
    public static readonly BindableProperty ProductsProperty = BindableProperty.Create<MainPage, Observabl
    public Command PullToRefreshCommand {
        get { return (Command)GetValue(PullToRefreshProperty); }
        set { SetValue(PullToRefreshProperty, value); }
    }
    public ObservableCollection<Product> Products {
        get { return (ObservableCollection<Product>)GetValue(ProductsProperty); }
        set { SetValue(ProductsProperty, value); }
    }
    public MainPage () {
        InitializeComponent ();
        BindingContext = this;
        this.repository = new TestProductsRepository();
        this.Products = repository.Products;
        PullToRefreshCommand = new Command(ExecutePullToRefreshCommand);
    }
    void ExecutePullToRefreshCommand() {
        repository.RefreshProducts();
        Products = repository.Products;
    }
}
```

**C#**

```
 (Product.cs)
public class Product : ModelObject {
    string name;
    public string Name {
        get { return name; }
        set {
            name = value;
            if (Photo == null)
                Photo = ImageSource.FromResource (value);
        }
    }
    public Product(string name) {
        this.Name = name;
    }
    public ImageSource Photo { get; set; }
    public string Description { get; set; }
}
```

**C#**

```
 (ProductsRepository.cs)
public abstract class ProductsRepository {
    public ObservableCollection<Product> Products { get; set; }
    public ProductsRepository() {
        this.Products = new ObservableCollection<Product>();
    }
    protected abstract ObservableCollection<Product> GenerateAvailableProd
    internal void RefreshProducts() {
        int index = new Random().Next (0, 5);
        Products = GenerateAvailableProducts (index);
    }
}
public class TestProductsRepository : ProductsRepository {
    readonly List<Product> products;
    public TestProductsRepository() : base() {
        this.products = new List<Product>();
        GenerateAllProducts();
        int index = new Random().Next (0, 5);
        Products = GenerateAvailableProducts (index);
    }
    protected override ObservableCollection<Product> GenerateAvailableProd
        ObservableCollection<Product> availableProducts = new ObservableCo
        for (int i = 0; i < 4; i++)
            availableProducts.Add(products[number+i]);
        return availableProducts;
    }
    void GenerateAllProducts() {
        products.Add(new Product("Beverages") {Description = "Soft drinks,
        products.Add(new Product("Condiments"){Description = "Sweet and sa
        products.Add(new Product("Confections"){Description = "Desserts, c
        products.Add(new Product("DairyProducts"){Description = "Cheeses"}
        products.Add(new Product("Grains"){Description = "Breads, crackers
        products.Add(new Product("MeatPoultry"){Description = "Prepared me
        products.Add(new Product("Produce"){Description = "Dried fruit and
        products.Add(new Product("Seafood"){Description = "Seaweed and fis
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl ItemsSource="{Binding Products}" RowHeight="160"
                    IsPullToRefreshEnabled="true" PullToRefreshCommand="{B
```

## See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# RightSwipeButtons Property

Provides access to objects containing information on buttons shown on the right side of a data row when an end-user swipes right to left over a data row.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public ObservableCollection<SwipeButtonInfo> RightSwipeButtons { get; }
```

### Property Value

A collection of objects specifying right swipe buttons added to the grid.

### Example

**Show Me**

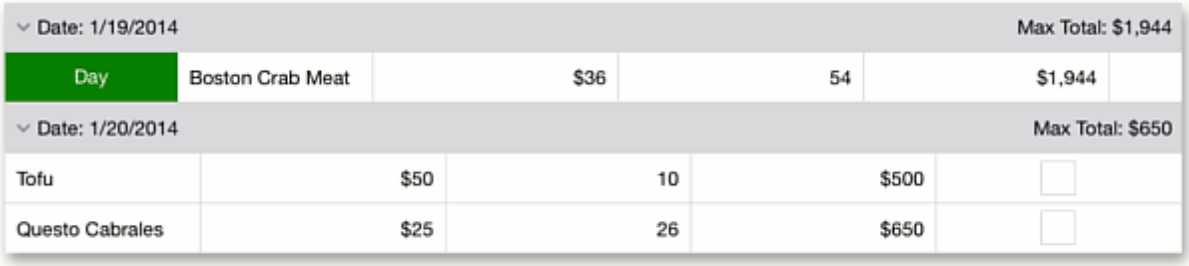
A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T178764 .

This example shows how to extend the grid's UI with additional buttons to be shown when an end-user swipes left to right, or right to left, over a data row. To create a swipe button, add a SwipeButtonInfo object to the LeftSwipeButtons or RightSwipeButtons collection. Use this object's SwipeButtonInfo.Caption, SwipeButtonInfo.Width, SwipeButtonInfo.BackgroundColor and SwipeButtonInfo.TextColor properties to specify button appearance.

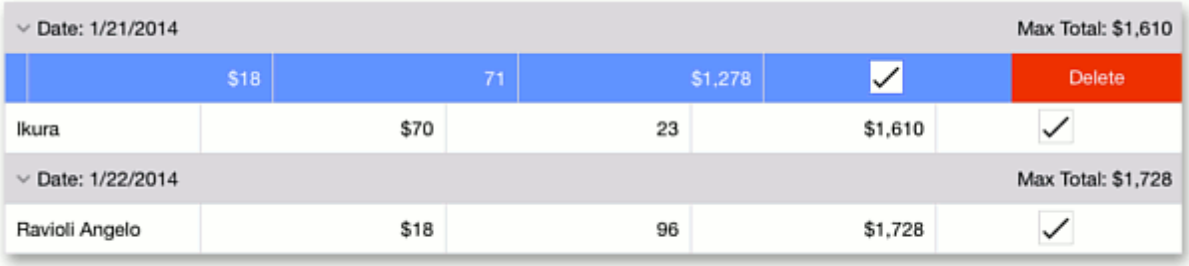
To assign a custom action to a button, handle the SwipeButtonClick event. The SwipeButtonShowing event allows you to specify when a swipe button should be visible.

In the example, two swipe buttons are created.

- When an end-user swipes any data row left to right, the **Day** button is shown on the left side of the row. Tapping this button displays the day of the week when the corresponding order was made.




- When an end-user swipes right to left, a data row which corresponds to an order that is already shipped (the value in the **Shipped** column is **true**), the **Delete** button is shown on the right side of the row. Tapping this button removes the corresponding data row.




**C#**

© 2018 Developer Express Inc.

```
 (MainPage.xaml.cs)
void OnSwipeButtonShowing(object sender, SwipeButtonShowingEventArgs e)
{
    if ((!(Boolean)grid.GetCellValue(e.RowHandle, "Shipped"))
        && (e.ButtonInfo.ButtonName == "RightButton"))
    {
        e.IsVisible = false;
    }
}
void OnSwipeButtonClick(object sender, SwipeButtonEventArgs e)
{
    if (e.ButtonInfo.ButtonName == "LeftButton")
    {
        DateTime orderDate = (DateTime)grid.GetCellValue(e.RowHandle, "Date");
        string orderDateDay = orderDate.ToString("dddd");
        DisplayAlert("Alert from " + e.ButtonInfo.ButtonName, "Day: " + orderDateDay, "OK");
    }
    if (e.ButtonInfo.ButtonName == "RightButton")
    {
        grid.DeleteRow(e.RowHandle);
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl.LeftSwipeButtons>
  <dxGrid:SwipeButtonInfo ButtonName="LeftButton" Caption="Day" BackgroundColor="Green" Width="150"/>
</dxGrid:GridControl.LeftSwipeButtons>
<dxGrid:GridControl.RightSwipeButtons>
  <dxGrid:SwipeButtonInfo ButtonName="RightButton" Caption="Delete" BackgroundColor="Red" Width="150"/>
</dxGrid:GridControl.RightSwipeButtons>
```

## See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# RowCount Property

Gets the number of data rows in the grid.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public int RowCount { get; }
```

### Property Value

An integer value that is the number of rows in the grid.

### Remarks

The **RowCount** property returns the total number of data rows in the grid. All rows are counted including rows that are hidden within collapsed groups.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# RowEditMode Property

Specifies whether the grid invokes an in-place editor or edit form when an end-user starts to edit a data cell or add a new row.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public RowEditMode RowEditMode { get; set; }
```

### Property Value

A RowEditMode enumeration value.

### Remarks

Use the **RowEditMode** property to set one of the available row edit modes for a grid.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# RowHeight Property

Gets or sets the height of rows in the grid. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public Double RowHeight { get; set; }
```

### Property Value

A System.Double value specifying the height of grid rows in density-independent pixels.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# RowTapCommand Property

Gets or sets the command executed when an end-user taps a row within the grid. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public ICommand RowTapCommand { get; set; }
```

### Property Value

An object implementing the System.Windows.Input.ICommand interface, that defines the command.

### Remarks

To define an action to be performed when the grid's row is tapped, you can implement a command and bind it to the grid using the **RowTapCommand** property, or handle the RowTap event.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# SelectedDataObject Property

Returns an object that specifies a data source record to which a data row currently selected in the grid corresponds. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public object SelectedDataObject { get; set; }
```

### Property Value

An object specifying an individual record of a data source bound to the grid.

### Remarks

To obtain a handle of the grid's selected data row, use the SelectedRowHandle property. After the grid's selection is changed, the SelectionChanged event occurs.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# SelectedRowHandle Property

Gets or sets the selected row handle. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public int SelectedRowHandle { get; set; }
```

### Property Value

An integer value that specifies the row handle.

### Remarks

When an end-user taps a data row in the grid, this row becomes selected. Use the **SelectedRowHandle** property to obtain or set a row currently selected in the grid. The SelectedDataObject property returns an object that specifies a data source record to which the row selected in the grid corresponds. After the grid's selection is changed, the SelectionChanged event occurs.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# SortMode Property

Specifies whether the grid allows data sorting by multiple columns or by a single column only. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public GridSortMode SortMode { get; set; }
```

### Property Value

A GridSortMode enumeration value.

### Remarks

Use the **SortMode** property to specify whether grid data sorting is allowed by multiple columns or by a single column only. To control whether the sorting is available for end-users, use the AllowSort or GridColumn.AllowSort property.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
GridColumn.SortOrder
GridColumn.SortIndex
SortBy
Lesson 6 - Sort Data

© 2018 Developer Express Inc.

# SwipeButtonCommand Property

Gets or sets the command executed when a swipe button is tapped. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public ICommand SwipeButtonCommand { get; set; }
```

### Property Value

An object implementing the System.Windows.Input.ICommand interface that defines the command.

### Remarks

Along with the **SwipeButtonCommand** command, GridControl provides you with the SwipeButtonClick event. Both (command and event) rise when an end-user clicks any of the swipe buttons. To determine which swipe button was clicked, the SwipeButtonEventArgs object is passed to the event as its argument, and the command - as its parameter.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
SwipeButtonClick

© 2018 Developer Express Inc.

# TotalSummaries Property

Provides access to a collection of total summary items.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public ObservableCollection<GridColumnSummary> TotalSummaries { get; }
```

### Property Value

A collection of total summary items.

### Remarks

A total summary is an aggregate function value calculated over all rows in the specified column. Total summaries are displayed at the bottom of the grid within the total summary panel, whose visibility is controlled by the TotalSummaryVisibility property. Total summary items are specified by the GridColumnSummary objects, and are stored within the **TotalSummaries** collection. This collection provides methods and properties allowing you to add, remove and access summary items.

Group summaries can be accessed using the GroupSummaries property.

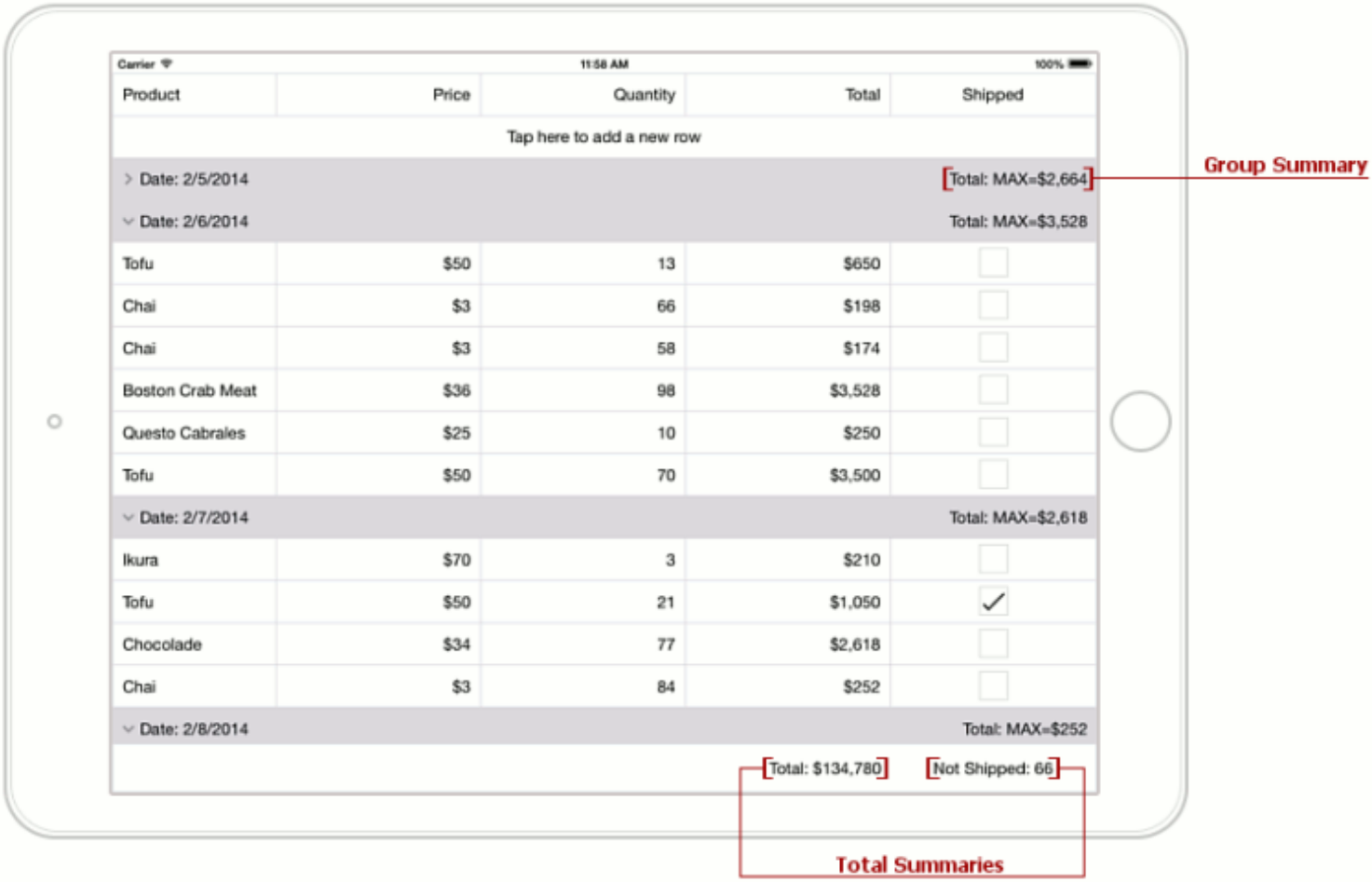
### Example

This example shows how to calculate group and total summaries for grid columns using predefined aggregate functions or a custom rule.

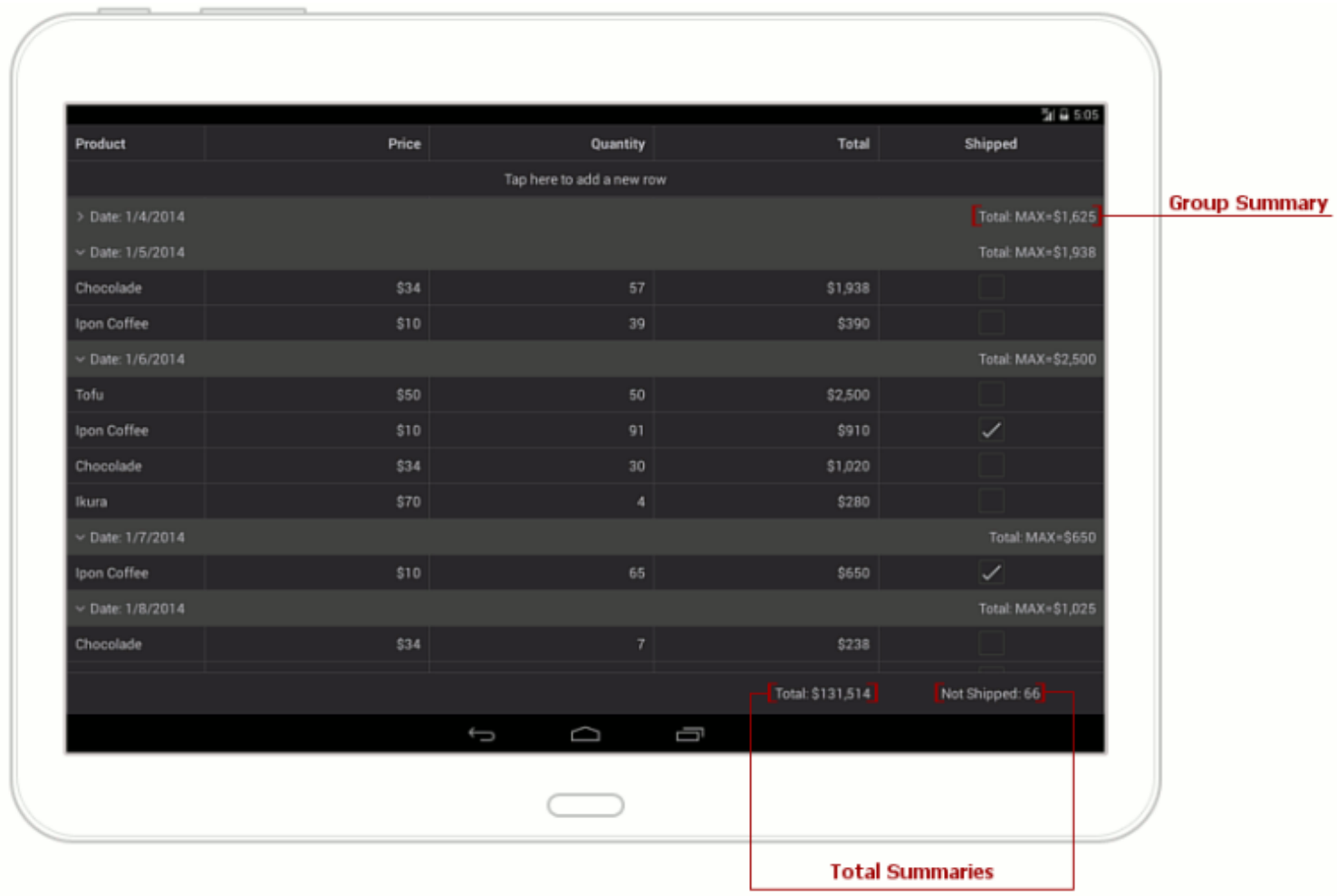
The following data summaries are created.

- Group summary to display the maximum value in a **Total** column for each group of records.
- Total summary to calculate the sum of values by the whole **Total** column.
- Custom total summary to count the number of rows whose value in the **Shipped** column is **false**.

© 2018 Developer Express Inc.




© 2018 Developer Express Inc.




**Xaml**

```
<dxGrid:GridControl x:Name="grid" ItemsSource="{Binding Orders}"
                    NewItemRowVisibility="true"
                    CalculateCustomSummary="OnCalculateCustomSummary">
  <!-- ... -->
  <dxGrid:GridControl.GroupSummaries>
    <dxGrid:GridColumnSummary FieldName="Total" Type="Max"/>
  </dxGrid:GridControl.GroupSummaries>
  <dxGrid:GridControl.TotalSummaries>
    <dxGrid:GridColumnSummary FieldName="Total" Type="Sum" DisplayFormat="Total: {0:C0}"/>
    <dxGrid:GridColumnSummary FieldName="Shipped" Type="Custom" DisplayFormat="Not Shipped: {0}"/>
  </dxGrid:GridControl.TotalSummaries>
  <!-- ... -->
</dxGrid:GridControl>
```

**C#**

```
void OnCalculateCustomSummary(object sender, CustomSummaryEventArgs e) {
    if (e.FieldName.ToString () == "Shipped")
    if (e.IsTotalSummary){
        if (e.SummaryProcess == CustomSummaryProcess.Start) {
            count = 0;
        }
        if (e.SummaryProcess == CustomSummaryProcess.Calculate) {
            if (!(bool)e.FieldValue)
                count++;
            e.TotalValue = count;
        }
    }
}
```

© 2018 Developer Express Inc.

**See Also**
GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
GroupSummaries

© 2018 Developer Express Inc.

# TotalSummaryHeight Property

Gets or sets the total summary panel height. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public Double TotalSummaryHeight { get; set; }
```

### Property Value

A System.Double value specifying the total summary panel height in density-independent pixels.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
TotalSummaryVisibility

© 2018 Developer Express Inc.

# TotalSummaryVisibility Property

Gets or sets whether the total summary panel is displayed. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public VisibilityState TotalSummaryVisibility { get; set; }
```

### Property Value

A VisibilityState enumeration value which specifies when the total summary panel is shown.

### Remarks

By default, the total summary panel is automatically shown at the bottom of the grid after total summary items are added to the TotalSummaries collection. To make this panel always visible or always hidden, use the **TotalSummaryVisibility** property.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# VisibleColumns Property

Gets the collection of visible columns.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public IReadOnlyList<GridColumn> VisibleColumns { get; }
```

### Property Value

A list of columns visible in the grid.

### Remarks

The **VisibleColumns** list contains columns whose GridColumn.IsVisible property is set to **true** and the GridColumn.IsGrouped property is set to **false**. All columns of the grid are accessed using the Columns property.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
Column Chooser

© 2018 Developer Express Inc.

# GridControl Events

The grid control.

The following tables list the members exposed by the GridControl type.

**Public Events**

| | Name | Description |
|---|---|---|
| | AutoGeneratingColumn | Occurs each time an individual column is auto-generated. |
| | BindingContextChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | CalculateCustomSummary | Allows you to calculate summary values manually. |
| | ChildAdded | (Inherited from **Xamarin.Forms.Element**) |
| | ChildRemoved | (Inherited from **Xamarin.Forms.Element**) |
| | ChildrenReordered | (Inherited from **Xamarin.Forms.VisualElement**) |
| | CustomizeCell | Allows you to customize the appearance of cells in the grid. |
| | CustomizeCellDisplayText | Enables custom display text to be provided for any data cell. |
| | CustomUnboundColumnData | Enables you to populate unbound columns with data, and save changes made by end-users in unbound columns. |
| | DescendantAdded | (Inherited from **Xamarin.Forms.Element**) |
| | DescendantRemoved | (Inherited from **Xamarin.Forms.Element**) |
| | EndRowEdit | Occurs after a row editor (an in-place editor or edit form, depending on the edit mode set in the grid) is closed and modified cell values are applied or canceled. |
| | Focused | (Inherited from **Xamarin.Forms.VisualElement**) |
| | GroupRowCollapsed | Occurs after a group row has been collapsed. |
| | GroupRowCollapsing | Occurs before a group row is collapsed. |
| | GroupRowExpanded | Occurs after a group row has been expanded. |
| | GroupRowExpanding | Occurs before a group row is expanded. |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | HorizontalScrollOffsetChanged | Occurs after the HorizontalScrollOffset property has been changed. |
| | InitNewRow | Enables you to initialize newly added rows. |
| | LayoutChanged | (Inherited from **Xamarin.Forms.Layout**) |
| | LoadMore | Occurs when an end-user scrolls to the bottom of the grid to load new data items. |
| | MeasureInvalidated | (Inherited from **Xamarin.Forms.VisualElement**) |
| | PopupMenuCustomization | Enables you to customize the grid's popup menus. |
| | PropertyChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanging | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PullToRefresh | Occurs when an end-user pulls the grid down to refresh its content. |
| | RowTap | Occurs when a data row is tapped. |
| | SelectionChanged | Occurs after the selection has been changed. |
| | SizeChanged | (Inherited from **Xamarin.Forms.VisualElement**) |
| | SwipeButtonClick | Occurs when a swipe button is tapped. |
| | SwipeButtonShowing | Occurs before a swipe button is shown when an end-user swipes left to right, or right to left, over a data row. |
| | Unfocused | (Inherited from **Xamarin.Forms.VisualElement**) |

**See Also**
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# AutoGeneratingColumn Event

Occurs each time an individual column is auto-generated.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public event AutoGeneratingColumnEventHandler AutoGeneratingColumn
```

### Event Data

The event handler receives an argument of type AutoGeneratingColumnEventArgs containing data related to this event.
The following **AutoGeneratingColumnEventArgs** properties provide information specific to this event.

| Property | Description |
|---|---|
| Cancel | |
| Column | Gets the column being auto-generated. |

### Remarks

Use the **AutoGeneratingColumn** event to change or cancel an automatically generated column at the time it is created.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# CalculateCustomSummary Event

Allows you to calculate summary values manually.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public event CustomSummaryEventHandler CalculateCustomSummary
```

### Event Data

The event handler receives an argument of type CustomSummaryEventArgs containing data related to this event.

The following **CustomSummaryEventArgs** properties provide information specific to this event.

| Property | Description |
|---|---|
| FieldName | Gets the processed field name. |
| FieldValue | Gets the processed field value. |
| GroupRowHandle | Gets a value identifying the group row whose child data rows are involved in summary calculation. |
| IsGroupSummary | Gets whether a group summary value is being calculated. |
| IsTotalSummary | Gets whether a total summary value is being calculated. |
| Row | Gets an object that contains information on the currently processed row. |
| SummaryProcess | Gets a value indicating the data summary calculation stage. |
| TotalValue | Gets or sets the total summary value. |
| TotalValueReady | Gets or sets whether the Calculation stage of the custom summary calculation process should be skipped. |

### Remarks

Total and group summaries provide five predefined aggregate functions (Count, Max, Min, Sum and Average). To calculate a summary value using a custom rule, set the summary item's GridColumnSummary.Type property to SummaryType.Custom, and handle the **CalculateCustomSummary** event.

The **CalculateCustomSummary** event fires for each data row involved in summary calculation. When calculating a total summary value, the event is raised for each data row. When calculating a group summary value, this event fires for each data row within a group. To enable you to perform any initialization and finalization, the event is also raised before and after processing rows.

The summary calculation consists of the three stages.

- **Initialization**

The **CalculateCustomSummary** event is raised once and the CustomSummaryEventArgs.SummaryProcess property is set to CustomSummaryProcess.Start. At this stage, you can initialize summary values (e.g., reset internal counters).

- **Calculation**

The **CalculateCustomSummary** event occurs multiple times, once for each data row in the grid or group. The **SummaryProcess** property is set to CustomSummaryProcess.Calculate. At this stage, you should accumulate summaries.

- **Finalization**

The **CalculateCustomSummary** event is raised once and the **SummaryProcess** property is set to CustomSummaryProcess.Finalize. At this point, calculate a final summary value and assign it to the event parameter's CustomSummaryEventArgs.TotalValue property.

To skip the **Calculation** stage and calculate a custom summary at the **Initialization** or **Finalization** stage, set the event parameter's CustomSummaryEventArgs.TotalValueReady property to **true** at the Initialization stage. This automatically skips the Calculation stage and the Finalization stage starts immediately.

© 2018 Developer Express Inc.

**Example**

| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T178764. | |

This example shows how to use the CalculateCustomSummary event to create a custom summary that counts the total number of data rows whose cells in the **Shipped** column (SwitchColumn) contain **false**.

**C#**

```
 (MainPage.xaml.cs)
void OnCalculateCustomSummary(object sender, CustomSummaryEventArgs e){
    if (e.FieldName.ToString () == "Shipped")
    if (e.IsTotalSummary){
        if (e.SummaryProcess == CustomSummaryProcess.Start) {
            count = 0;
        }
        if (e.SummaryProcess == CustomSummaryProcess.Calculate) {
            if (!(bool)e.FieldValue)
                count++;
            e.TotalValue = count;
        }
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl.TotalSummaries>
    <dxGrid:GridColumnSummary FieldName="Total" Type="Sum"
                              DisplayFormat= "Total: {0:C0}"/>
    <dxGrid:GridColumnSummary FieldName="Shipped" Type="Custom"
                              DisplayFormat= "Not Shipped: {0}"/>
</dxGrid:GridControl.TotalSummaries>
```

## See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# CustomizeCell Event

Allows you to customize the appearance of cells in the grid.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public event CustomizeCellEventHandler CustomizeCell
```

### Event Data

The event handler receives an argument of type CustomizeCellEventArgs containing data related to this event.

The following **CustomizeCellEventArgs** properties provide information specific to this event.

| Property | Description |
|---|---|
| BackgroundColor | Gets or sets the cell background color. |
| DisplayText | Gets or sets the text displayed in the cell. |
| FieldName | Gets the field name of the column that contains the cell. |
| ForeColor | Gets or sets the color used to paint the cell value. |
| Handled | Gets or sets whether the CustomizeCell event was handled. If it was handled, the default cell painting is not required. |
| IsSelected | Gets whether the processed cell belongs to a row that is currently selected. |
| RowHandle | Gets the handle of the row that contains the processed cell. |
| Value | Gets the processed cell value. |

### Remarks

The **CustomizeCell** event is raised before a data cell is painted. Use the **FieldName** and **RowHandle** parameters to identify a column and row containing the cell, and **BackgroundColor** and **ForeColor** to set cell colors. Set the **Handled** parameter to **true** to prohibit default cell painting.

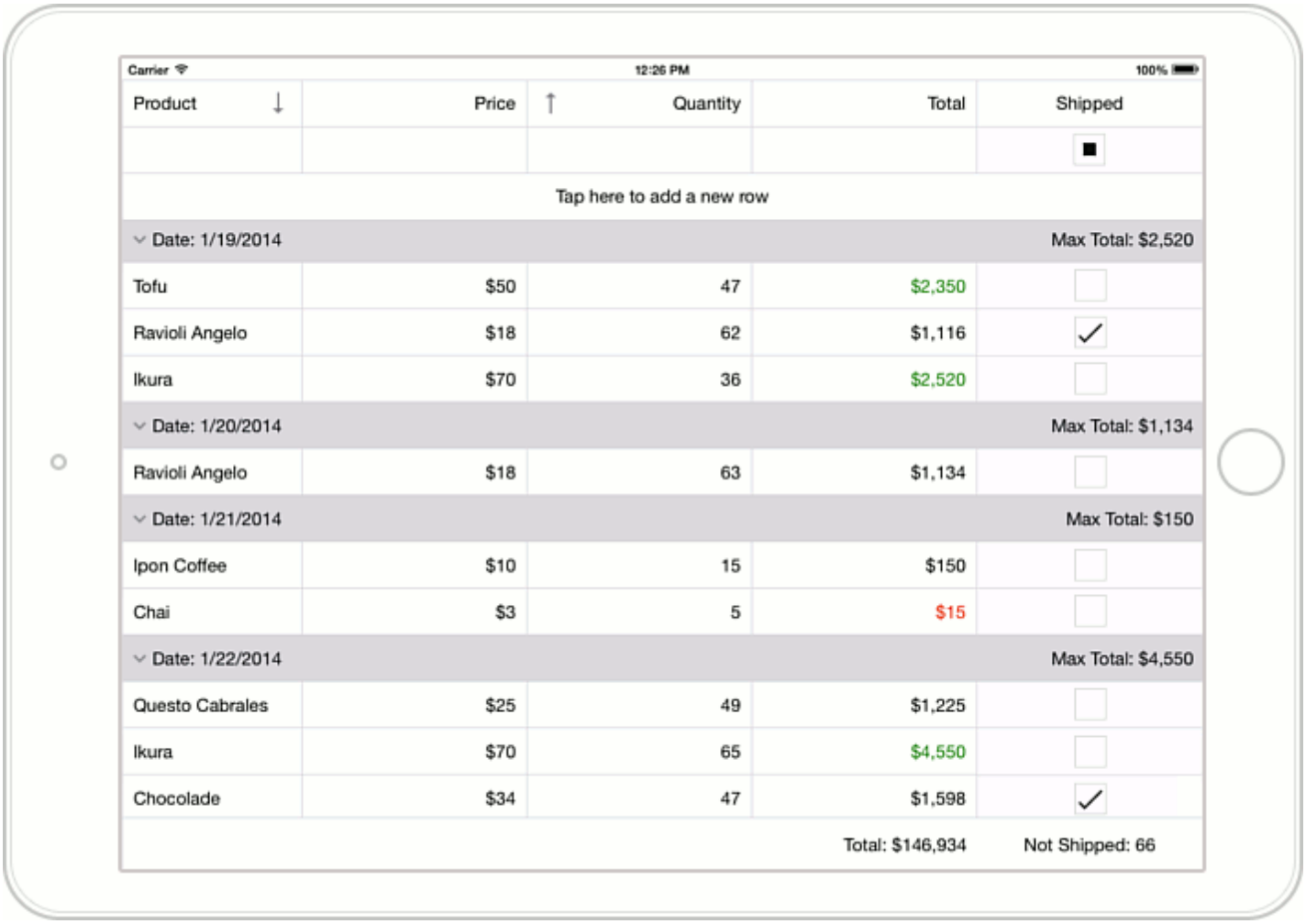
### Example

| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T178764. | |

This example demonstrates how to customize the appearance of cells in the specified grid column depending on cell values by handling the CustomizeCell event.

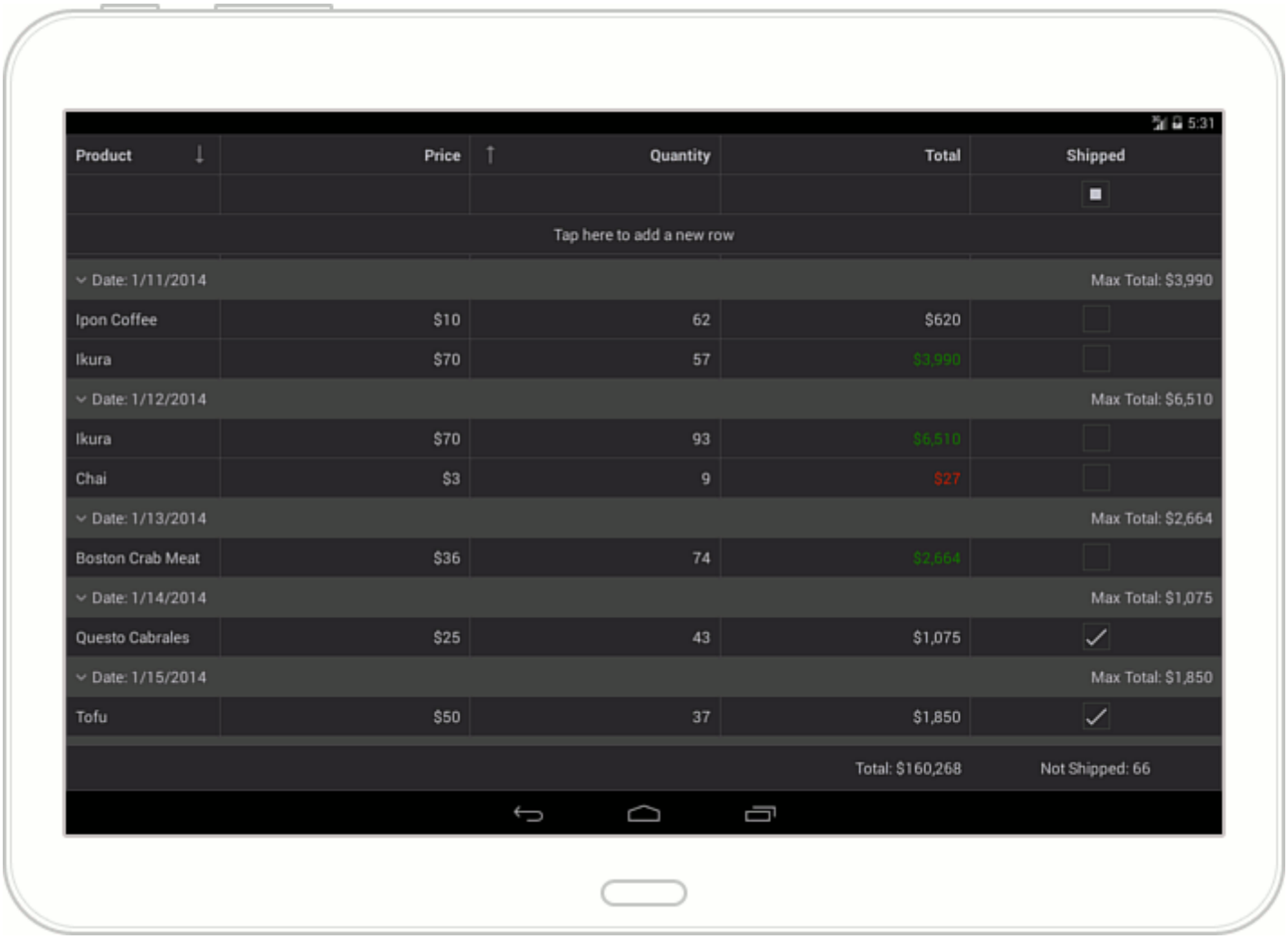
The image below illustrates the result.

© 2018 Developer Express Inc.




| Product | Price | Quantity | Total | Shipped |
|---|---|---|---|---|
| | | | | ■ |
| Tap here to add a new row | | | | |
| Date: 1/19/2014 | | | | Max Total: $2,520 |
| Tofu | $50 | 47 | $2,350 | ☐ |
| Ravioli Angelo | $18 | 62 | $1,116 | ☑ |
| Ikura | $70 | 36 | $2,520 | ☐ |
| Date: 1/20/2014 | | | | Max Total: $1,134 |
| Ravioli Angelo | $18 | 63 | $1,134 | ☐ |
| Date: 1/21/2014 | | | | Max Total: $150 |
| Ipon Coffee | $10 | 15 | $150 | ☐ |
| Chai | $3 | 5 | $15 | ☐ |
| Date: 1/22/2014 | | | | Max Total: $4,550 |
| Questo Cabrales | $25 | 49 | $1,225 | ☐ |
| Ikura | $70 | 65 | $4,550 | ☐ |
| Chocolade | $34 | 47 | $1,598 | ☑ |
| | | | Total: $146,934 | Not Shipped: 66 |

© 2018 Developer Express Inc.




**C#**

```
 (MainPage.xaml.cs)
void OnCustomizeCell(CustomizeCellEventArgs e)
{
    if (e.FieldName == "Total" && !e.IsSelected)
    {
        int total = Convert.ToInt32(e.Value);
        if (total < 50)
            e.ForeColor = Color.Red;
        else if (total > 2000)
            e.ForeColor = Color.Green;
        e.Handled = true;
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl x:Name="grid" ItemsSource="{Binding Orders}"
                    NewItemRowVisibility="true"
                    CalculateCustomSummary="OnCalculateCustomSummary"
                    SortMode="Multiple" AutoFilterPanelVisibility="true"
                    CustomizeCell="OnCustomizeCell"
                    SwipeButtonShowing="OnSwipeButtonShowing"
                    SwipeButtonClick="OnSwipeButtonClick">
```

## See Also

© 2018 Developer Express Inc.

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
Conditional Formatting

© 2018 Developer Express Inc.

# CustomizeCellDisplayText Event

Enables custom display text to be provided for any data cell.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public event CustomizeCellDisplayTextEventHandler CustomizeCellDisplayText
```

### Event Data

The event handler receives an argument of type CustomizeCellDisplayTextEventArgs containing data related to this event.

The following **CustomizeCellDisplayTextEventArgs** properties provide information specific to this event.

| Property | Description |
|---|---|
| DisplayText | Specifies the display text for the cell currently being processed. |
| Index | Gets the processed cell's index. |
| Source | Gets the grid control that raised the event. |
| Value | Gets the processed cell's value. |

### Remarks

The **CustomizeCellDisplayText** event can be used to display custom text within any cell. This event is fired for both bound and unbound columns. The text provided via this event will be used when the grid is printed.

The **CustomizeCellDisplayText** event is raised only when a unique edit value is assigned to a grid cell.

Initially the CustomizeCellDisplayTextEventArgs.DisplayText parameter contains the cell's current display text. To provide custom display text, assign the required string to this property.

The **CustomizeCellDisplayText** event is called in various instances:

- To provide a cell's display text when data is not sorted by a column.
- To provide column display text and compare the supplied strings when data is grouped or sorted by this column.

In most instances the **CustomizeCellDisplayText** event cannot be used to supply custom display text to a column based on values of other cells. A solution to this task is to create an unbound column to provide custom display text.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# CustomUnboundColumnData Event

Enables you to populate unbound columns with data, and save changes made by end-users in unbound columns.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public event GridColumnDataEventHandler CustomUnboundColumnData
```

### Event Data

The event handler receives an argument of type GridColumnDataEventArgs containing data related to this event.

The following **GridColumnDataEventArgs** properties provide information specific to this event.

| Property | Description |
|---|---|
| Column | Gets the unbound column currently being processed. |
| EditableRowData | Gets an object that contains information on the currently processed row and allows you to modify field values. |
| IsGetData | Gets whether you should provide data for the currently processed cell. |
| IsSetData | Gets whether the current cell's value should be saved in a data source. |
| RowData | Gets an object that contains information on the currently processed row. |
| Source | Gets the grid control that raised the event. |
| Value | Gets or sets the value of the cell currently being processed. |

### Remarks

In GridControl, you can create unbound (calculated) columns. These columns are not bound to any field in the underlying data source. Data for these columns can be provided either by creating an expression via the GridColumn.UnboundExpression property or by handling the **CustomUnboundColumnData** event. This event fires in two cases:

- **Provide Data for Unbound Columns**

When the grid is loaded, it raises the **CustomUnboundColumnData** event to populate the unbound columns. The event's **IsGetData** parameter will be set to **true** (consequently the **IsSetData** parameter will be set to **false**). In this case, you need to supply data for the currently processed cell. Assign the required value to the **Value** parameter.

- **Save Changes**

When an unbound column's data is modified using the grid, the **CustomUnboundColumnData** event is fired with the **IsSetData** parameter set to **true** (consequently the **IsGetData** parameter is set to **false**). In this case, you can save the modified data specified by the **Value** property.

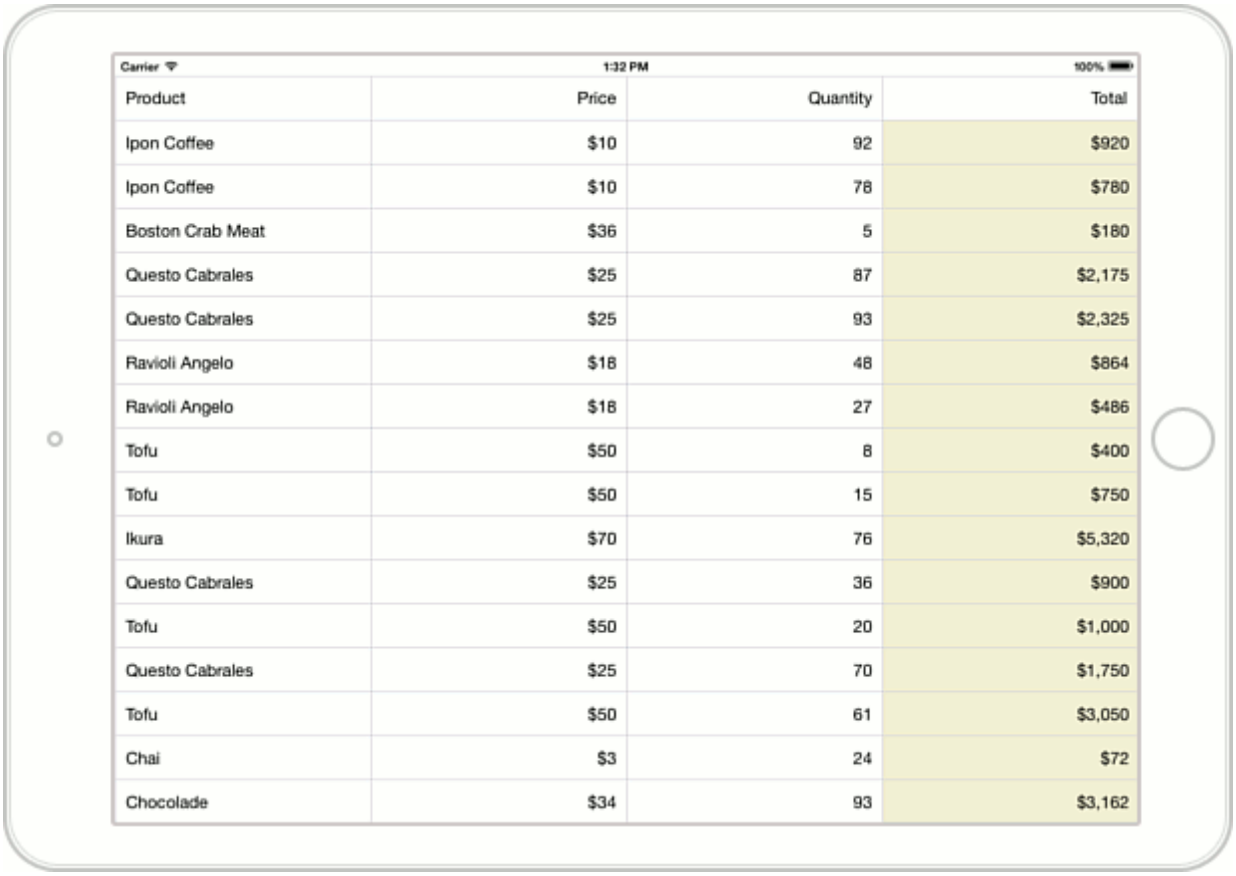
The currently processed cell is identified by the column and row. The GridColumnDataEventArgs.Column parameter refers to the column. To identify the row, use the GridColumnDataEventArgs.RowData event parameter.

### Example

Assume that the GridControl instance is bound to a collection of orders. Each order has the "Product", "UnitPrice"and "Quantity" fields. The example below shows how to add an unbound column to the grid to display the amount of each order according to the expression: UnitPrice*Quantity.

The result is displayed below:

© 2018 Developer Express Inc.



| Product | Price | Quantity | Total |
|---|---|---|---|
| Ipon Coffee | $10 | 92 | $920 |
| Ipon Coffee | $10 | 78 | $780 |
| Boston Crab Meat | $36 | 5 | $180 |
| Questo Cabrales | $25 | 87 | $2,175 |
| Questo Cabrales | $25 | 93 | $2,325 |
| Ravioli Angelo | $18 | 48 | $864 |
| Ravioli Angelo | $18 | 27 | $486 |
| Tofu | $50 | 8 | $400 |
| Tofu | $50 | 15 | $750 |
| Ikura | $70 | 76 | $5,320 |
| Questo Cabrales | $25 | 36 | $900 |
| Tofu | $50 | 20 | $1,000 |
| Questo Cabrales | $25 | 70 | $1,750 |
| Tofu | $50 | 61 | $3,050 |
| Chai | $3 | 24 | $72 |
| Chocolade | $34 | 93 | $3,162 |

**Xaml**

```
<dxGrid:GridControl ItemsSource="{Binding Orders}"
                    CustomUnboundColumnData="OnCustomUnboundColumnData"
                    CustomizeCell="OnCustomizeCell">
    <dxGrid:GridControl.Columns>
         <dxGrid:TextColumn FieldName="Product.Name" Caption="Product"/>
         <dxGrid:NumberColumn FieldName="Product.UnitPrice" Caption="Price" DisplayFormat="C0"/>
         <dxGrid:NumberColumn FieldName="Quantity" />
         <dxGrid:NumberColumn FieldName="Total" IsReadOnly="True"
                              UnboundType="Decimal" DisplayFormat="C0"/>
    </dxGrid:GridControl.Columns>
</dxGrid:GridControl>
```

**C#**

```
// Returns the total for a specific row.
decimal getTotalValue(GridControl grid, int rowHandle) {
    decimal unitPrice = Convert.ToDecimal(grid.GetCellValue(rowHandle, "Product.UnitPrice"));
    decimal quantity = Convert.ToDecimal(grid.GetCellValue(rowHandle, "Quantity"));
    return unitPrice * quantity;
}
// Provides data for the Total column.
void OnCustomUnboundColumnData(object sender, GridColumnDataEventArgs e) {
    if (e.Column.FieldName == "Total" && e.IsGetData)
        e.Value = getTotalValue (e.Source, e.RowData.RowHandle);
}
// Customizes the appearance settings.
void OnCustomizeCell(CustomizeCellEventArgs e) {
if (e.FieldName == "Total") {
        e.BackgroundColor = Color.FromRgb(247, 240, 213);
        e.Handled = true;
    }
}
```

© 2018 Developer Express Inc.

**See Also**
GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# EndRowEdit Event

Occurs after a row editor (an in-place editor or edit form, depending on the edit mode set in the grid) is closed and modified cell values are applied or canceled.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public event RowEditingEventHandler EndRowEdit
```

### Event Data

The event handler receives an argument of type RowEditingEventArgs containing data related to this event.

The following **RowEditingEventArgs** properties provide information specific to this event.

| Property | Description |
|---|---|
| Action | Gets whether an end-user has closed a row editor to apply or cancel modified cell values. |
| RowHandle | Gets the row handle. |
| SourceRowIndex | Gets the index of the record in a data source to which the processed row corresponds. |

### Remarks

You can handle the **EndRowEdit** event to obtain cell values of the row edited by an end-user. To do this, use the GetRow method with the passed row handle that the event's **RowHandle** parameter returns.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
OpenEditor
CloseEditor

© 2018 Developer Express Inc.

# GroupRowCollapsed Event

Occurs after a group row has been collapsed.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public event RowEventHandler GroupRowCollapsed
```

### Event Data

The event handler receives an argument of type RowEventArgs containing data related to this event.

The following **RowEventArgs** properties provide information specific to this event.

| Property | Description |
|---|---|
| RowHandle | Gets the row handle. |

### Remarks

The **GroupRowCollapsed** event is raised after a group row has been collapsed by an end-user or in code. This event serves as a notification and the action cannot be cancelled. To control whether individual group rows can be collapsed, handle the GroupRowCollapsing event. For instance, to prevent a group row from being collapsed, set the **GroupRowCollapsing** event parameter's RowAllowEventArgs.Allow property to **false**.

The CollapseAllGroups method allows you to collapse all group rows. In this instance, the **GroupRowCollapsed** event is raised for each group.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
Lesson 4 - Group Data
GroupRowExpanded
GroupRowExpanding
ExpandAllGroups

© 2018 Developer Express Inc.

# GroupRowCollapsing Event

Occurs before a group row is collapsed.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public event RowAllowEventHandler GroupRowCollapsing
```

### Event Data

The event handler receives an argument of type RowAllowEventArgs containing data related to this event.

The following **RowAllowEventArgs** properties provide information specific to this event.

| Property | Description |
|---|---|
| Allow | Gets or sets a value specifying whether the current operation (group row expanding or collapsing) is allowed. |
| RowHandle | Gets the row handle. |

### Remarks

The **GroupRowCollapsing** event is raised before a group row is collapsed by an end-user or in code, allowing you to cancel this action. To prevent a group row from being collapsed, set the event parameter's **Allow** property to **false**.

The CollapseAllGroups method allows you to collapse all group rows. In this instance, the **GroupRowCollapsing** event is raised for each group.

After the group row has been collapsed, the GroupRowCollapsed event is raised.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
Lesson 4 - Group Data
GroupRowExpanding
ExpandAllGroups

© 2018 Developer Express Inc.

# GroupRowExpanded Event

Occurs after a group row has been expanded.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public event RowEventHandler GroupRowExpanded
```

### Event Data

The event handler receives an argument of type RowEventArgs containing data related to this event.

The following **RowEventArgs** properties provide information specific to this event.

| Property | Description |
|---|---|
| RowHandle | Gets the row handle. |

### Remarks

The **GroupRowExpanded** event is raised after a group row has been expanded by an end-user or in code. This event serves as a notification and the action cannot be canceled. To control whether individual group rows can be expanded, handle the GroupRowExpanding event. For instance, to prevent a group row from being expanded, set the **GroupRowExpanding** event parameter's RowAllowEventArgs.Allow property to **false**.

The ExpandAllGroups method allows you to expand all group rows. In this instance, the **GroupRowExpanded** event is raised for each group.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
Lesson 4 - Group Data
GroupRowCollapsed
CollapseAllGroups

© 2018 Developer Express Inc.

# GroupRowExpanding Event

Occurs before a group row is expanded.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public event RowAllowEventHandler GroupRowExpanding
```

### Event Data

The event handler receives an argument of type RowAllowEventArgs containing data related to this event.

The following **RowAllowEventArgs** properties provide information specific to this event.

| Property | Description |
|---|---|
| Allow | Gets or sets a value specifying whether the current operation (group row expanding or collapsing) is allowed. |
| RowHandle | Gets the row handle. |

### Remarks

The **GroupRowExpanding** event is raised before a group row is expanded by an end-user or in code, allowing you to cancel this action. To prevent a group row from being expanded, set the event parameter's **Allow** property to **false**.

The ExpandAllGroups method allows you to expand all group rows. In this instance, the **GroupRowExpanding** event is raised for each group.

After the group row has been expanded, the GroupRowExpanded event is raised.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
Lesson 4 - Group Data
GroupRowCollapsing
CollapseAllGroups
GroupRowCollapsed

© 2018 Developer Express Inc.

# HorizontalScrollOffsetChanged Event

Occurs after the HorizontalScrollOffset property has been changed.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public event HorizontalScrollOffsetEventHandler HorizontalScrollOffsetChanged
```

### Event Data

The event handler receives an argument of type HorizontalScrollOffsetEventArgs containing data related to this event.
The following **HorizontalScrollOffsetEventArgs** properties provide information specific to this event.

| Property | Description |
|---|---|
| HorizontalOffset | Gets a horizontal scrolling offset. |

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# InitNewRow Event

Enables you to initialize newly added rows.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public event InitNewRowEventHandler InitNewRow
```

### Event Data

The event handler receives an argument of type InitNewRowEventArgs containing data related to this event.

The following **InitNewRowEventArgs** properties provide information specific to this event.

| Property | Description |
|---|---|
| EditableRowData | Gets an object that contains information on the newly added row and allows you to modify cell values in this row. |

### Remarks

You can handle the **InitNewRow** event to initialize newly added rows. To initialize row cells, call the IEditableRowData.SetFieldValue of the IEditableRowData object returned by the event's InitNewRowEventArgs.EditableRowData parameter.

The **InitNewRow** event is raised when an end-users taps New Item Row in the grid.

### Example

The following code shows how to initialize the "Date" field of a new row via the InitNewRow event.

**Xaml**

```
<dxGrid:GridControl ItemsSource="{Binding Orders}"
                    NewItemRowVisibility="true"
                    InitNewRow="OnInitNewRow">
  <!-- ... -->
</dxGrid:GridControl>
```

**C#**

```
using DevExpress.Mobile.DataGrid;
// ...
void OnInitNewRow(object sender, InitNewRowEventArgs e) {
    e.EditableRowData.SetFieldValue ("Date", DateTime.Today);
}
```

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# LoadMore Event

Occurs when an end-user scrolls to the bottom of the grid to load new data items.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public event EventHandler LoadMore
```

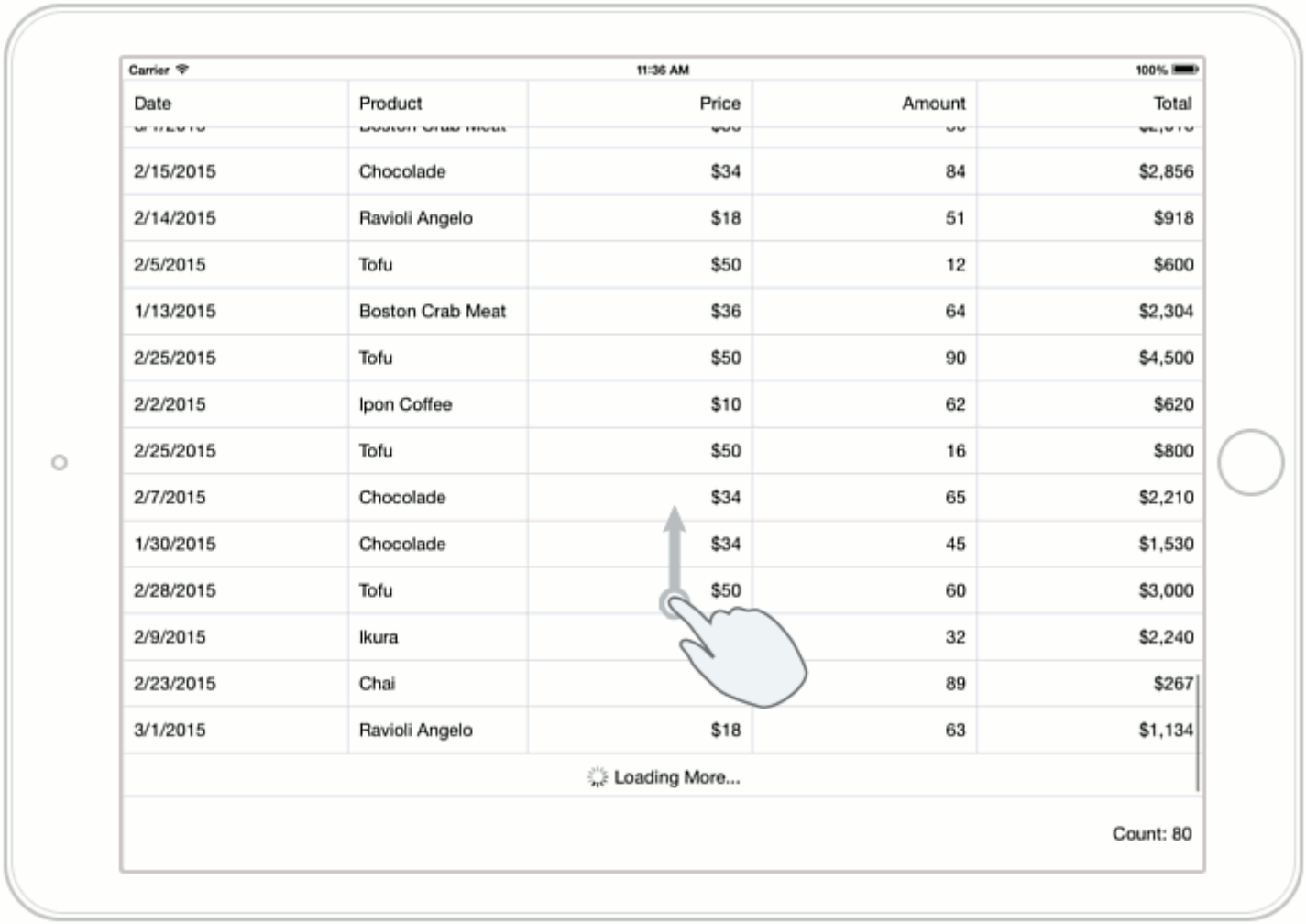
### Event Data

The event handler receives an argument of type System.EventArgs containing data related to this event.

### Remarks

The GridControl supports dynamic loading of new data records from the bound data source when an end-user scrolls to the bottom of the grid.

© 2018 Developer Express Inc.


Carrier
11:36 AM
100%
Date
Product
Price
Amount
Total
2/15/2015
Chocolade
$34
84
$2,856
2/14/2015
Ravioli Angelo
$18
51
$918
2/5/2015
Tofu
$50
12
$600
1/13/2015
Boston Crab Meat
$36
64
$2,304
2/25/2015
Tofu
$50
90
$4,500
2/2/2015
Ipon Coffee
$10
62
$620
2/25/2015
Tofu
$50
16
$800
2/7/2015
Chocolade
$34
65
$2,210
1/30/2015
Chocolade
$34
45
$1,530
2/28/2015
Tofu
$50
60
$3,000
2/9/2015
Ikura
32
$2,240
2/23/2015
Chai
89
$267
3/1/2015
Ravioli Angelo
$18
63
$1,134
Loading More...
Count: 80

4:42
Date
Product
Price
Amount
Total
2/7/2015
Chocolade
$34
62
$2,108
2/8/2015
Questo Cabrales
$25
87
$2,175
1/2/2015
Chai
$3
49
$147
1/9/2015
Tofu
$50
11
$550
2/2/2015
Questo Cabrales
$25
33
$825
1/17/2015
Ikura
$70
66
$4,620
2/1/2015
Tofu
$50
19
$950
2/26/2015
Ravioli Angelo
$18
67
$1,206
1/13/2015
Chocolade
$34
35
$1,190
2/17/2015
Tofu
$50
56
$2,800
2/27/2015
Ravioli Angelo
1
$18
2/24/2015
Tofu
19
$950
2/16/2015
Questo Cabrales
$25
55
$1,375
Loading More...
Count: 50

To enable this functionality, set the IsLoadMoreEnabled property to **true**. Then, either define a command in a view model and bind it to the grid using the LoadMoreCommand property, or handle the **LoadMore** event.

**See Also**
GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# PopupMenuCustomization Event

Enables you to customize the grid's popup menus.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public event PopupMenuEventHandler PopupMenuCustomization
```

### Event Data

The event handler receives an argument of type PopupMenuEventArgs containing data related to this event.

The following **PopupMenuEventArgs** properties provide information specific to this event.

| Property | Description |
|---|---|
| Column | Gets the column for which the popup menu will be shown. |
| Menu | Provides access to the grid's popup menu to be shown. |
| MenuType | Gets the type of the grid's menu to be invoked. |
| RowHandle | Gets the handle of the row for which the popup menu will be shown. |

### Remarks

The GridControl provides a set of built-in popup menus enabling end-users to manage data rows (edit and delete them), sort and group data, show and hide columns, modify total and group summaries. Available menu types are listed by the GridPopupMenuType enumerator.

The **PopupMenuCustomization** event allows you to customize any of these menus. For example, you can remove existing menu items and/or add new custom items.

The following properties allow you to disable a grid's menu:

- IsRowCellMenuEnabled
- IsGroupRowMenuEnabled
- IsColumnMenuEnabled
- IsTotalSummaryMenuEnabled

### Example

The following sample code handles the PopupMenuCustomization event to customize the data row popup menu before it is displayed. The code clears default menu items and adds new ones. One of these custom items applies a specific filter to the grid data and another one clears this filter.

© 2018 Developer Express Inc.

**Xaml**

```
<?xml version="1.0" encoding="UTF-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="HelloGrid.MainPage"
             xmlns:dxg="clr-namespace:DevExpress.Mobile.DataGrid;assembly=DevExpress.Mobile.Grid.v15.1">
    <ContentPage.Padding>
        <OnPlatform x:TypeArguments="Thickness" iOS="0, 20, 0, 0" />
    </ContentPage.Padding>
    <ContentPage.Content>
        <dxg:GridControl x:Name="grid" ItemsSource="{Binding Orders}"
                         PopupMenuCustomization="OnPopupMenuCustomization">
            <dxg:GridControl.Columns>
                <dxg:TextColumn FieldName="Product.Name" Caption="Product"/>
                <dxg:NumberColumn FieldName="Product.UnitPrice" Caption="Price" DisplayFormat="C0"/>
                <dxg:NumberColumn FieldName="Quantity" />
                <dxg:NumberColumn FieldName="Total"
                                  UnboundType="Decimal" UnboundExpression="[Quantity] * [Product.UnitPrice]
                                  IsReadOnly="True" DisplayFormat="C0"/>
            </dxg:GridControl.Columns>
    </dxg:GridControl>
    </ContentPage.Content>
</ContentPage>
```

**C#**

© 2018 Developer Express Inc.

```
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.Core;
// ...
void OnPopupMenuCustomization(object sender, PopupMenuEventArgs e) {
    if (e.MenuType == GridPopupMenuType.DataRow) {
        e.Menu.Items.Clear();
        PopupMenuItem itemFilter = new PopupMenuItem();
        itemFilter.Caption = "Filter by Product";
        itemFilter.Click += ItemFilterClick;
        e.Menu.Items.Insert(0, itemFilter);
        PopupMenuItem itemClearFilter = new PopupMenuItem();
        itemClearFilter.Caption = "Clear Filter";
        itemClearFilter.Click += ItemClearFilterClick;
        e.Menu.Items.Insert(1, itemClearFilter);
    }
}
void ItemFilterClick(object sender, EventArgs e) {
    IRowData rowData = grid.GetRow(grid.SelectedRowHandle);
    Order selectedOrder = rowData.DataObject as Order;
    string productName = selectedOrder.Product.Name;
    grid.Columns ["Product.Name"].AutoFilterValue = productName;
}
void ItemClearFilterClick(object sender, EventArgs e) {
    grid.ClearFilter ();
}
```

**See Also**
GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# PullToRefresh Event

Occurs when an end-user pulls the grid down to refresh its content.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

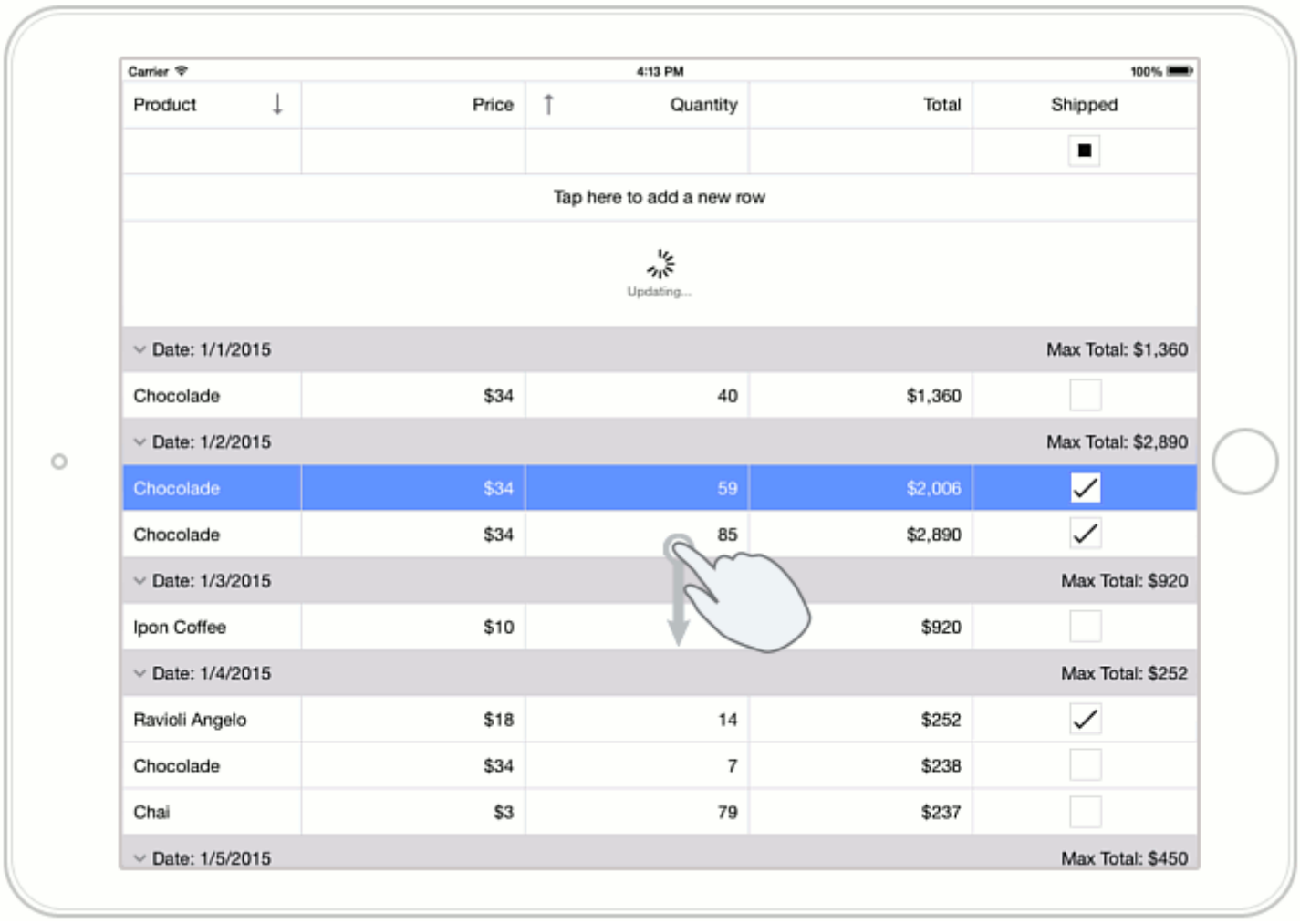
```
public event EventHandler PullToRefresh
```

### Event Data

The event handler receives an argument of type System.EventArgs containing data related to this event.

### Remarks

The GridControl's pull-to-refresh feature allows end-users to request a grid content update with the pull-down gesture.




To enable this functionality, set the IsPullToRefreshEnabled property to **true**. Then, either handle the **PullToRefresh** event, or define a refresh command in a view model and bind it to the grid using the PullToRefreshCommand property.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# RowTap Event

Occurs when a data row is tapped.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public event RowTapEventHandler RowTap
```

### Event Data

The event handler receives an argument of type RowTapEventArgs containing data related to this event.

The following **RowTapEventArgs** properties provide information specific to this event.

| Property | Description |
|---|---|
| FieldName | Gets the field name of a column to which the tapped cell belongs. |
| RowHandle | Gets the row handle. |

### Remarks

The **RowTap** event occurs after an end-user taps any data row in the grid. The event parameter's RowEventArgs.RowHandle property allows you to obtain which row has been tapped.

If an end-user taps a row that is not selected, the SelectionChanged event is also raised.

You can also use the RowTapCommand property to define an action to be performed when the grid's row is tapped.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# SelectionChanged Event

Occurs after the selection has been changed.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public event RowEventHandler SelectionChanged
```

### Event Data

The event handler receives an argument of type RowEventArgs containing data related to this event.

The following **RowEventArgs** properties provide information specific to this event.

| Property | Description |
|---|---|
| RowHandle | Gets the row handle. |

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
SelectedDataObject
SelectedRowHandle

© 2018 Developer Express Inc.

# SwipeButtonClick Event

Occurs when a swipe button is tapped.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public event SwipeButtonEventHandler SwipeButtonClick
```

### Event Data

The event handler receives an argument of type SwipeButtonEventArgs containing data related to this event.

The following **SwipeButtonEventArgs** properties provide information specific to this event.

| Property | Description |
|---|---|
| ButtonInfo | Gets an object that stores information about a swipe button. |
| RowHandle | Gets the row handle. |
| SourceRowIndex | Gets the index of the record in a data source to which the processed row corresponds. |

### Remarks

After you add a swipe button to the grid (by adding a SwipeButtonInfo object to the LeftSwipeButtons or RightSwipeButtons collection), handle the **SwipeButtonClick** event to implement the button functionality.

You can also use the SwipeButtonCommand property to define an action to be performed when a button is tapped.

### Example

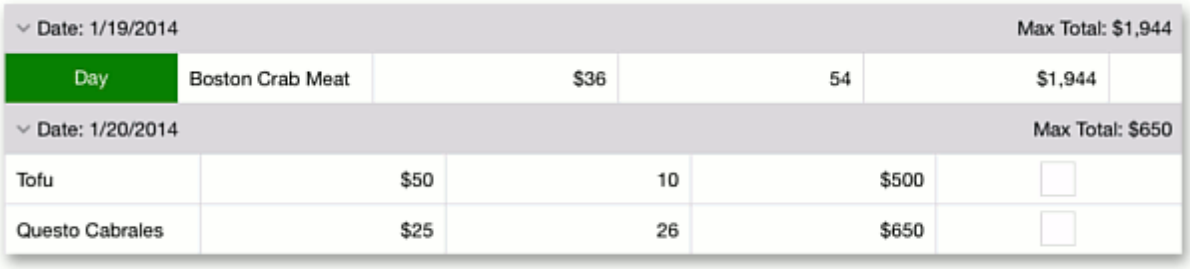
| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T178764 . | |

This example shows how to extend the grid's UI with additional buttons to be shown when an end-user swipes left to right, or right to left, over a data row. Swipe Buttons. To create a swipe button, add a SwipeButtonInfo object to the LeftSwipeButtons or RightSwipeButtons collection. Use this object's SwipeButtonInfo.Caption, SwipeButtonInfo.Width, SwipeButtonInfo.BackgroundColor and SwipeButtonInfo.TextColor properties to specify button appearance.

To assign a custom action to a button, handle the SwipeButtonClick event. The SwipeButtonShowing event allows you to specify when a swipe button should be visible.

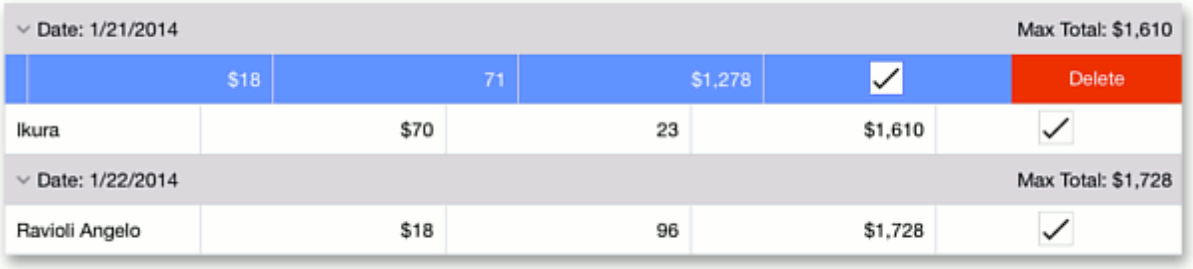
In the example, two swipe buttons are created.

- When an end-user swipes any data row left to right, the **Day** button is shown on the left side of the row. Tapping this button displays the day of the week when the corresponding order was made.




- When an end-user swipes right to left, a data row which corresponds to an order that is already shipped (the value in the **Shipped** column is **true**), the **Delete** button is shown on the right side of the row. Tapping this button removes the corresponding data row.

© 2018 Developer Express Inc.




**C#**

```
 (MainPage.xaml.cs)
void OnSwipeButtonShowing(object sender, SwipeButtonShowingEventArgs e)
{
    if ((!(Boolean)grid.GetCellValue(e.RowHandle, "Shipped"))
        && (e.ButtonInfo.ButtonName == "RightButton"))
    {
        e.IsVisible = false;
    }
}
void OnSwipeButtonClick(object sender, SwipeButtonEventArgs e)
{
    if (e.ButtonInfo.ButtonName == "LeftButton")
    {
        DateTime orderDate = (DateTime)grid.GetCellValue(e.RowHandle, "Date");
        string orderDateDay = orderDate.ToString("dddd");
        DisplayAlert("Alert from " + e.ButtonInfo.ButtonName, "Day: " + orderDateDay, "OK");
    }
    if (e.ButtonInfo.ButtonName == "RightButton")
    {
        grid.DeleteRow(e.RowHandle);
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl.LeftSwipeButtons>
  <dxGrid:SwipeButtonInfo ButtonName="LeftButton" Caption="Day" BackgroundColor="Green" Width="150"/>
</dxGrid:GridControl.LeftSwipeButtons>
<dxGrid:GridControl.RightSwipeButtons>
  <dxGrid:SwipeButtonInfo ButtonName="RightButton" Caption="Delete" BackgroundColor="Red" Width="150"/>
</dxGrid:GridControl.RightSwipeButtons>
```

## See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# SwipeButtonShowing Event

Occurs before a swipe button is shown when an end-user swipes left to right, or right to left, over a data row.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public event SwipeButtonShowingEventHandler SwipeButtonShowing
```

### Event Data

The event handler receives an argument of type SwipeButtonShowingEventArgs containing data related to this event.

The following **SwipeButtonShowingEventArgs** properties provide information specific to this event.

| Property | Description |
|---|---|
| ButtonInfo | Gets an object that stores information about a swipe button. |
| IsVisible | Gets or sets the swipe button visibility. |
| RowHandle | Gets the row handle. |
| SourceRowIndex | Gets the index of the record in a data source to which the processed row corresponds. |

### Remarks

Handle the **SwipeButtonShowing** event and set the **e.IsVisible** property to **false** to restrict a swipe button added to the LeftSwipeButtons or RightSwipeButtons collection from being displayed when an end-user swipes left to right, or right to left, over a data row.

### Example

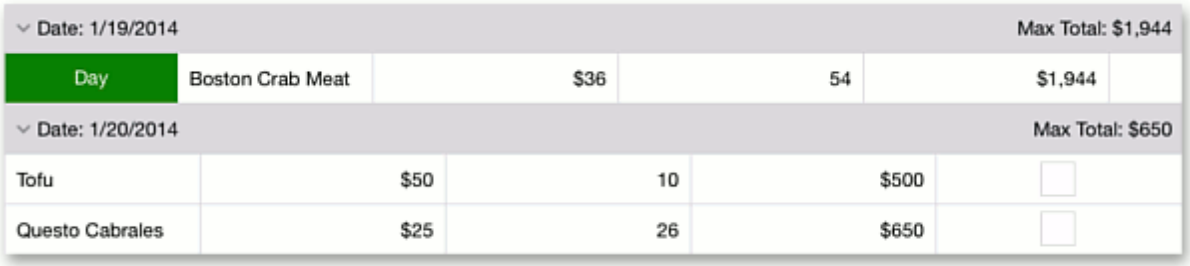
| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T178764 . | |

This example shows how to extend the grid's UI with additional buttons to be shown when an end-user swipes left to right, or right to left, over a data row. To create a swipe button, add a SwipeButtonInfo object to the LeftSwipeButtons or RightSwipeButtons collection. Use this object's SwipeButtonInfo.Caption, SwipeButtonInfo.Width, SwipeButtonInfo.BackgroundColor and SwipeButtonInfo.TextColor properties to specify button appearance.

To assign a custom action to a button, handle the SwipeButtonClick event. The SwipeButtonShowing event allows you to specify when a swipe button should be visible.

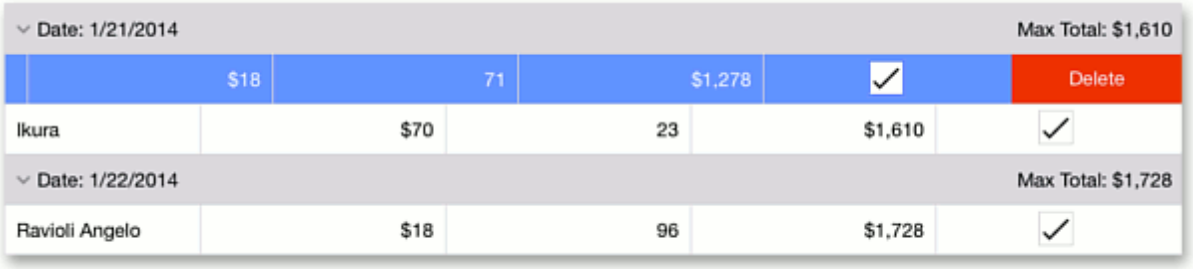
In the example, two swipe buttons are created.

- When an end-user swipes any data row left to right, the **Day** button is shown on the left side of the row. Tapping this button displays the day of the week when the corresponding order was made.




- When an end-user swipes right to left, a data row which corresponds to an order that is already shipped (the value in the **Shipped** column is **true**), the **Delete** button is shown on the right side of the row. Tapping this button removes the corresponding data row.

© 2018 Developer Express Inc.




**C#**

```
 (MainPage.xaml.cs)
void OnSwipeButtonShowing(object sender, SwipeButtonShowingEventArgs e)
{
    if ((!(Boolean)grid.GetCellValue(e.RowHandle, "Shipped"))
        && (e.ButtonInfo.ButtonName == "RightButton"))
    {
        e.IsVisible = false;
    }
}
void OnSwipeButtonClick(object sender, SwipeButtonEventArgs e)
{
    if (e.ButtonInfo.ButtonName == "LeftButton")
    {
        DateTime orderDate = (DateTime)grid.GetCellValue(e.RowHandle, "Date");
        string orderDateDay = orderDate.ToString("dddd");
        DisplayAlert("Alert from " + e.ButtonInfo.ButtonName, "Day: " + orderDateDay, "OK");
    }
    if (e.ButtonInfo.ButtonName == "RightButton")
    {
        grid.DeleteRow(e.RowHandle);
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl.LeftSwipeButtons>
  <dxGrid:SwipeButtonInfo ButtonName="LeftButton" Caption="Day" BackgroundColor="Green" Width="150"/>
</dxGrid:GridControl.LeftSwipeButtons>
<dxGrid:GridControl.RightSwipeButtons>
  <dxGrid:SwipeButtonInfo ButtonName="RightButton" Caption="Delete" BackgroundColor="Red" Width="150"/>
</dxGrid:GridControl.RightSwipeButtons>
```

## See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# GridControl Methods

The grid control.

The following tables list the members exposed by the GridControl type.

**Public Methods**

| | Name | Description |
|---|---|---|
| | BatchBegin | (Inherited from **Xamarin.Forms.VisualElement**) |
| | BatchCommit | (Inherited from **Xamarin.Forms.VisualElement**) |
| | ClearFilter | Clears the filter applied to data in the grid. |
| | ClearGrouping | Ungroups data in the grid. |
| | ClearSorting | Clears the sorting applied to the grid. |
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | CloseEditor | Closes a data row editor (an in-place editor or edit form, depending on the edit mode set in the grid). |
| | CollapseAllGroups | Collapses all group rows. |
| | CollapseGroupRow | Collapses the specified group row. |
| | DeleteRow | Deletes the specified data row. |
| | Dispose | Releases all resources used by the GridControl. |
| | Equals | (Inherited from **System.Object**) |
| S | Equals | (Inherited from **System.Object**) |
| | ExpandAllGroups | Expands all group rows. |
| | ExpandGroupRow | Expands the specified group row. |
| | ExportToExcel | Exports data from the grid to the specified stream in the XLS, XLSX or CSV format. |
| | FindRow | Returns the handle of the first row that matches the specified criteria. |
| | FindRowByValue | Overloaded. Searches for the value in the column and returns the handle of the corresponding row. |
| | Focus | (Inherited from **Xamarin.Forms.VisualElement**) |
| | ForceLayout | (Inherited from **Xamarin.Forms.Layout**) |
| | GetCellDisplayText | Overloaded. Returns the text displayed within the specified cell. |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | GetCellValue | Overloaded. Returns the value of the specified data cell in the grid. |
| S | GetColumn | (Inherited from **Xamarin.Forms.Grid**) |
| S | GetColumnSpan | (Inherited from **Xamarin.Forms.Grid**) |
| | GetGroupInfo | Provides access to information on the specified data group. |
| | GetGroupRowValue | Returns the value of the specified group row. |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetRow | Overloaded. Gets an object that provides access to the data of a row specified by its handle. |
| S | GetRow | (Inherited from **Xamarin.Forms.Grid**) |
| | GetRowHandle | Returns the handle of the row which corresponds to the specified record in the data source. |
| S | GetRowSpan | (Inherited from **Xamarin.Forms.Grid**) |
| | GetService | Gets the service object of the specified type. |
| | GetService<T> | Gets the specified service. |
| | GetSizeRequest | (Inherited from **Xamarin.Forms.Layout**) |
| | GetSourceRowIndex | Returns the index of the data source record to which the specified row handle corresponds. |
| | GetType | (Inherited from **System.Object**) |
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | GroupBy | Overloaded. Groups data in the grid by the specified column. |
| | HideColumnChooser | Overloaded. Hides the column chooser. |
| | IsGroupCollapsed | Indicates whether the specified group row is collapsed. |
| | IsGroupRow | Indicates whether the specified handle corresponds to a group row. |
| | Layout | (Inherited from **Xamarin.Forms.VisualElement**) |
| S | LayoutChildIntoBoundingRegion | (Inherited from **Xamarin.Forms.Layout**) |
| | LoadInVisualTree | For internal use. |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | LowerChild | (Inherited from **Xamarin.Forms.Layout**) |
| | OpenEditor | Invokes an editor (an in-place editor or edit form, depending on the edit mode set in the grid) for a data row containing the specified cell. |
| | RaiseChild | (Inherited from **Xamarin.Forms.Layout**) |
| | Redraw | Forces the control to immediately redraw itself. |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | RefreshData | Updates the grid data. |
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | RemoveDynamicResource | (Inherited from **Xamarin.Forms.Element**) |
| | RestoreLayoutFromStream | Restores a grid's layout from the specified stream. |
| | RestoreLayoutFromXml | Restores a grid's layout from the specified XML string. |
| | SaveLayoutToStream | Saves a grid's layout to the specified stream. |
| | SaveLayoutToXml | Saves a gird's layout to an XML string. |
| | ScrollToRow | Scrolls the grid to make the specified row visible. |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetCellValue | Overloaded. Assigns a value to the specified cell. |
| S | SetColumn | (Inherited from **Xamarin.Forms.Grid**) |
| S | SetColumnSpan | (Inherited from **Xamarin.Forms.Grid**) |
| | SetDynamicResource | (Inherited from **Xamarin.Forms.Element**) |
| S | SetRow | (Inherited from **Xamarin.Forms.Grid**) |
| S | SetRowSpan | (Inherited from **Xamarin.Forms.Grid**) |
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ShowColumnChooser | Invokes the column chooser. |
| | SortBy | Overloaded. Sorts data by the values of the column bound to the specified data field in the ascending order. |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | ToString | (Inherited from **System.Object**) |
| | Unfocus | (Inherited from **Xamarin.Forms.VisualElement**) |

**See Also**
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# ClearFilter Method

Clears the filter applied to data in the grid.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public void ClearFilter()
```

### Remarks

Use the **ClearFilter** method to clear the grid filter, including both column auto filters (GridColumn.AutoFilterValue) and grid filter expression (FilterString, FilterExpression).

To get a string that specifies the grid's current filter, use the ActualFilterString property.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# ClearGrouping Method

Ungroups data in the grid.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public void ClearGrouping()
```

### Remarks

The **ClearGrouping** method cancels grouping and returns a group column to display in the grid. End-users can ungroup data by tapping **Remove Grouping** in a menu that appears when they touch and hold a column header or cell.

To group data by a column, use the GridColumn.IsGrouped property or GroupBy method.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
Lesson 4 - Group Data

© 2018 Developer Express Inc.

# ClearSorting Method

Clears the sorting applied to the grid.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public void ClearSorting()
```

### Remarks

The **ClearSorting** method removes all the sorting applied to any columns. To clear sorting by a particular column, set the column's GridColumn.SortOrder property to ColumnSortOrder.None.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
GridColumn.SortOrder
GridColumn.SortIndex
SortBy
Lesson 6 - Sort Data

© 2018 Developer Express Inc.

# CloseEditor Method

Closes a data row editor (an in-place editor or edit form, depending on the edit mode set in the grid).

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public void CloseEditor(
   bool applyChanges
)
```

### Parameters

*applyChanges*
**true**, to apply cell values modified in a row editor; otherwise, **false**.

### Remarks

When a row editor is closed, the EndRowEdit event occurs.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
OpenEditor

© 2018 Developer Express Inc.

# CollapseAllGroups Method

Collapses all group rows.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public void CollapseAllGroups()
```

### Remarks

To collapse the specified group row, use the CollapseGroupRow method.

Before a group row is collapsed, the GroupRowCollapsing event is raised, allowing you to cancel the action. After a group row has been collapsed, the GroupRowCollapsed event is raised.

To expand group rows, use the ExpandAllGroups or ExpandGroupRow method.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
AllowGroupCollapse

© 2018 Developer Express Inc.

# CollapseGroupRow Method

Collapses the specified group row.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public void CollapseGroupRow(
   int rowHandle
)
```

### Parameters

*rowHandle*
An integer value that specifies the group row handle.

### Remarks

Before a group row is collapsed, the GroupRowCollapsing event is raised, allowing you to cancel the action. After a group row has been collapsed, the GroupRowCollapsed event is raised.

To collapse all group rows, use the CollapseAllGroups method.

To expand group rows, use the ExpandAllGroups or ExpandGroupRow method.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
Lesson 4 - Group Data

© 2018 Developer Express Inc.

# DeleteRow Method

Deletes the specified data row.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public void DeleteRow(
   int rowHandle
)
```

### Parameters

*rowHandle*
An integer value that specifies the handle of the row to be deleted. If the specified handle refers to a group row, no rows are deleted.

### Remarks

End-users can delete a row by tapping **Delete Row** in a menu that appears when they touch and hold a data row.

### Example

| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at | |

© 2018 Developer Express Inc.

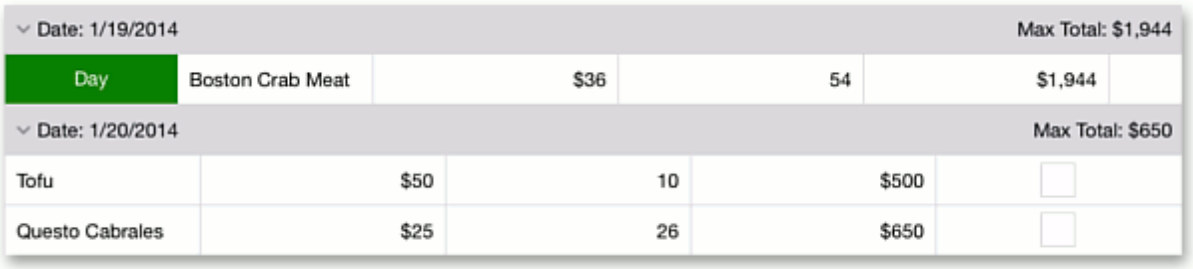
http://www.devexpress.com/example=T178764 .

This example shows how to extend the grid's UI with additional buttons to be shown when an end-user swipes left to right, or right to left, over a data row. Swipe Buttons. To create a swipe button, add a SwipeButtonInfo object to the LeftSwipeButtons or RightSwipeButtons collection. Use this object's SwipeButtonInfo.Caption, SwipeButtonInfo.Width, SwipeButtonInfo.BackgroundColor and SwipeButtonInfo.TextColor properties to specify button appearance.

To assign a custom action to a button, handle the SwipeButtonClick event. The SwipeButtonShowing event allows you to specify when a swipe button should be visible.

In the example, two swipe buttons are created.

- When an end-user swipes any data row left to right, the **Day** button is shown on the left side of the row. Tapping this button displays the day of the week when the corresponding order was made.




- When an end-user swipes right to left, a data row which corresponds to an order that is already shipped (the value in the **Shipped** column is **true**), the **Delete** button is shown on the right side of the row. Tapping this button removes the corresponding data row.

**C#**

```
 (MainPage.xaml.cs)
void OnSwipeButtonShowing(object sender, SwipeButtonShowingEventArgs e)
{
    if ((!(Boolean)grid.GetCellValue(e.RowHandle, "Shipped"))
        && (e.ButtonInfo.ButtonName == "RightButton"))
    {
        e.IsVisible = false;
    }
}
void OnSwipeButtonClick(object sender, SwipeButtonEventArgs e)
{
    if (e.ButtonInfo.ButtonName == "LeftButton")
    {
        DateTime orderDate = (DateTime)grid.GetCellValue(e.RowHandle, "Date");
        string orderDateDay = orderDate.ToString("dddd");
        DisplayAlert("Alert from " + e.ButtonInfo.ButtonName, "Day: " + orderDateDay, "OK");
    }
    if (e.ButtonInfo.ButtonName == "RightButton")
    {
        grid.DeleteRow(e.RowHandle);
    }
}
```

**Xaml**

© 2018 Developer Express Inc.

```
 (MainPage.xaml)
<dxGrid:GridControl.LeftSwipeButtons>
  <dxGrid:SwipeButtonInfo ButtonName="LeftButton" Caption="Day" BackgroundColor="Green" Width="150"/>
</dxGrid:GridControl.LeftSwipeButtons>
<dxGrid:GridControl.RightSwipeButtons>
  <dxGrid:SwipeButtonInfo ButtonName="RightButton" Caption="Delete" BackgroundColor="Red" Width="150"/>
</dxGrid:GridControl.RightSwipeButtons>
```

## See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
AllowDeleteRows

© 2018 Developer Express Inc.

# Dispose Method

Releases all resources used by the GridControl.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**
**C#**

```
public void Dispose()
```

**See Also**
GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# ExpandAllGroups Method

Expands all group rows.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public void ExpandAllGroups()
```

### Remarks

To expand the specified group row, use the ExpandGroupRow method.

Before a group row is expanded, the GroupRowExpanding event is raised, allowing you to cancel the action. After a group row has been expanded, the GroupRowExpanded event is raised.

To collapse group rows, use the CollapseAllGroups or CollapseGroupRow method.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# ExpandGroupRow Method

Expands the specified group row.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public void ExpandGroupRow(
   int rowHandle
)
```

### Parameters

*rowHandle*
An integer value that specifies the group row handle.

### Remarks

Before a group row is expanded, the GroupRowExpanding event is raised, allowing you to cancel the action. After a group row has been expanded, the GroupRowExpanded event is raised.

To expand all group rows, use the ExpandAllGroups method.

To collapse group rows, use the CollapseAllGroups or CollapseGroupRow method.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# ExportToExcel Method

Exports data from the grid to the specified stream in the XLS, XLSX or CSV format.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public bool ExportToExcel(
   Stream stream,
   ExportTarget format
)
```

### Parameters

*stream*
A System.IO.Stream object, to which the created document is exported.
*format*
An ExportTarget value specifying the format to export grid data.

### Return Value

**true**, if export is successfully completed; otherwise, **false**.

### Remarks

To set options for exporting data from the grid to the XLS, XLSX or CSV format, use the OptionsExportXls, OptionsExportXlsx or OptionsExportCsv property respectively.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# FindRow Method

Returns the handle of the first row that matches the specified criteria.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public int FindRow(
   Predicate<IRowData> predicate
)
```

### Parameters

*predicate*
The method that defines a set of criteria used to find a data row.

### Return Value

An integer value that specifies the handle of the first found row.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
Identify and Access Rows

© 2018 Developer Express Inc.

# FindRowByValue Method

Searches for the value in the column and returns the handle of the corresponding row.

**Overload List**

| Name | Description |
|---|---|
| int FindRowByValue(GridColumn column, object value) | Searches for the value in the column and returns the handle of the corresponding row. |
| int FindRowByValue(string fieldName, object value) | Searches for the value in the column and returns the handle of the corresponding row. |

**See Also**
GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
GridControl.FindRowByValue Overload List

Searches for the value in the column and returns the handle of the corresponding row.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**
**C#**
```
public int FindRowByValue(
   GridColumn column,
   object value
)
```

**Parameters**
*column*
The column to be searched.
*value*
An object that is the search value.

**Return Value**

An integer value that is the handle of the corresponding row.

**Remarks**

If the column contains more than one search value, the **FindRowByValue** method returns the handle of the first corresponding row.

If the value is not found, the **FindRowByValue** method returns InvalidRowHandle.

**See Also**
GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
GridControl.FindRowByValue Overload List
Rows Overview
Identify and Access Rows
Columns
Identify and Access Columns

Searches for the value in the column and returns the handle of the corresponding row.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**

© 2018 Developer Express Inc.

**C#**

```
public int FindRowByValue(
   string fieldName,
   object value
)
```

**Parameters**

*fieldName*
A System.String value that specifies the field name of the column to be searched.
*value*
An object that is the search value.

**Return Value**

An integer value that is the handle of the corresponding row.

**Remarks**

If the column contains more than one search value, the **FindRowByValue** method returns the handle of the first corresponding row.

If the value is not found, the **FindRowByValue** method returns InvalidRowHandle.

**See Also**

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
GridControl.FindRowByValue Overload List
Rows Overview
Identify and Access Rows
Columns
Identify and Access Columns

© 2018 Developer Express Inc.

# GetCellDisplayText Method

Returns the text displayed within the specified cell.

**Overload List**

| Name | Description |
|---|---|
| string GetCellDisplayText(int rowHandle, GridColumn column) | Returns the text displayed within the specified cell. |
| string GetCellDisplayText(int rowHandle, string fieldName) | Returns the text displayed within the specified cell. |

**See Also**
GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
GridControl.GetCellDisplayText Overload List

Returns the text displayed within the specified cell.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**
**C#**
```
public string GetCellDisplayText(
   int rowHandle,
   GridColumn column
)
```

**Parameters**
*rowHandle*
An integer value specifying the handle of the row that contains the cell.
*column*
A GridColumn object or descendant specifying the grid column that contains the cell. A **System.ArgumentNullException** exception will be thrown if the grid does not contain the specified column.

**Return Value**

A string displayed within the specified cell.

**Remarks**

Use the **GetCellDisplayText** method to obtain a textual representation of the specified cell value. The returned string is formatted as specified by the GridColumn.DisplayFormat property of the column where the specified cell resides. If you need to obtain the actual cell value, use the GetCellValue method instead.

The **GetCellDisplayText** method returns an empty string if the specified row handle does not point to any of the rows within the grid or points to a group row.

**See Also**
GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
GridControl.GetCellDisplayText Overload List
GetCellValue
Cells

Returns the text displayed within the specified cell.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**

© 2018 Developer Express Inc.

**C#**

```
public string GetCellDisplayText(
   int rowHandle,
   string fieldName
)
```

**Parameters**

*rowHandle*
An integer value specifying the handle of the row that contains the cell.
*fieldName*
A System.String value specifying the column's filed name.

**Return Value**

A string displayed within the specified cell.

**Remarks**

Use the **GetCellDisplayText** method to obtain a textual representation of the specified cell value. The returned string is formatted as specified by the GridColumn.DisplayFormat property of the column where the specified cell resides. If you need to obtain the actual cell value, use the GetCellValue method instead.

The **GetCellDisplayText** method returns an empty string in the following cases:

- the specified row handle does not point to any of the rows within the grid or points to a group row;
- the grid does not contain a column with the specified field name.

**See Also**

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
GridControl.GetCellDisplayText Overload List
Cells

© 2018 Developer Express Inc.

# GetCellValue Method

Returns the value of the specified data cell.

**Overload List**

| Name | Description |
|---|---|
| object GetCellValue(int rowHandle, string fieldName) | Returns the value of the specified data cell. |
| object GetCellValue(int rowHandle, GridColumn column) | Returns the value of the specified data cell in the grid. |

**See Also**
GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
GridControl.GetCellValue Overload List

Returns the value of the specified data cell.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**
**C#**
```
public object GetCellValue(
   int rowHandle,
   string fieldName
)
```

**Parameters**
*rowHandle*
An integer value specifying the handle of the row that contains the cell.
*fieldName*
A string specifying the field name whose value is to be returned. This parameter can refer to any field in the data source, even if the grid does not contain a column referring to this field. A **System.NullReferenceException** exception will be thrown if the underlying data source does not contain the specified field.

**Return Value**

An object that is the specified cell value.

**Example**

| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T178764 . | |

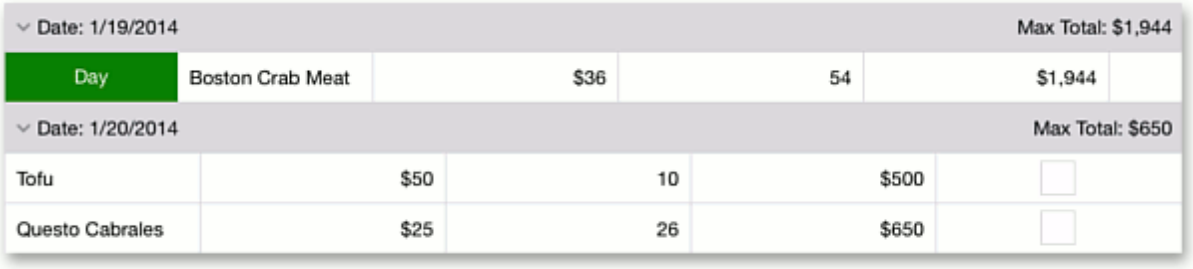
This example shows how to extend the grid's UI with additional buttons to be shown when an end-user swipes left to right, or right to left, over a data row Swipe Buttons. To create a swipe button, add a SwipeButtonInfo object to the LeftSwipeButtons or RightSwipeButtons collection. Use this object's SwipeButtonInfo.Caption, SwipeButtonInfo.Width, SwipeButtonInfo.BackgroundColor and SwipeButtonInfo.TextColor properties to specify button appearance.

To assign a custom action to a button, handle the SwipeButtonClick event. The SwipeButtonShowing event allows you to specify when a swipe button should be visible.

In the example, two swipe buttons are created.

- When an end-user swipes any data row left to right, the **Day** button is shown on the left side of the row. Tapping this button displays the day of the week when the corresponding order was made.

© 2018 Developer Express Inc.




- When an end-user swipes right to left, a data row which corresponds to an order that is already shipped (the value in the **Shipped** column is **true**), the **Delete** button is shown on the right side of the row. Tapping this button removes the corresponding data row.

**C#**

```
 (MainPage.xaml.cs)
void OnSwipeButtonShowing(object sender, SwipeButtonShowingEventArgs e)
{
    if ((!(Boolean)grid.GetCellValue(e.RowHandle, "Shipped"))
        && (e.ButtonInfo.ButtonName == "RightButton"))
    {
        e.IsVisible = false;
    }
}
void OnSwipeButtonClick(object sender, SwipeButtonEventArgs e)
{
    if (e.ButtonInfo.ButtonName == "LeftButton")
    {
        DateTime orderDate = (DateTime)grid.GetCellValue(e.RowHandle, "Date");
        string orderDateDay = orderDate.ToString("dddd");
        DisplayAlert("Alert from " + e.ButtonInfo.ButtonName, "Day: " + orderDateDay, "OK");
    }
    if (e.ButtonInfo.ButtonName == "RightButton")
    {
        grid.DeleteRow(e.RowHandle);
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl.LeftSwipeButtons>
  <dxGrid:SwipeButtonInfo ButtonName="LeftButton" Caption="Day" BackgroundColor="Green" Width="150"/>
</dxGrid:GridControl.LeftSwipeButtons>
<dxGrid:GridControl.RightSwipeButtons>
  <dxGrid:SwipeButtonInfo ButtonName="RightButton" Caption="Delete" BackgroundColor="Red" Width="150"/>
</dxGrid:GridControl.RightSwipeButtons>
```

## See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
GridControl.GetCellValue Overload List

© 2018 Developer Express Inc.

# GetCellDisplayText
# SetCellValue
# Cells

Returns the value of the specified data cell in the grid.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax
**C#**
```
public object GetCellValue(
   int rowHandle,
   GridColumn column
)
```

### Parameters
*rowHandle*
An integer value specifying the handle of the row that contains the cell.
*column*
A GridColumn object or descendant specifying the grid column that contains the cell.

### Return Value

An object that is the specified cell value.

### See Also
GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
GridControl.GetCellValue Overload List
GetCellDisplayText
SetCellValue
Cells

© 2018 Developer Express Inc.

# GetGroupInfo Method

Provides access to information on the specified data group.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public IGroupInfo GetGroupInfo(
   int rowHandle
)
```

### Parameters

*rowHandle*
An integer value that specifies the group row handle.

### Return Value

An object implementing the IGroupInfo interface.

### Remarks

Use the **GetGroupInfo** method to obtain information on a particular data group. For example, you can get group summary values calculated for the specified group (IGroupInfo.Summaries), obtain a value for which the group was created (IGroupInfo.Value), count data rows in the group (IGroupInfo.RowCount), check whether the group is currently collapsed (IsGroupCollapsed), etc.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
Lesson 4 - Group Data

© 2018 Developer Express Inc.

# GetGroupRowValue Method

Returns the value of the specified group row.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public object GetGroupRowValue(
   int rowHandle
)
```

### Parameters

*rowHandle*
An integer value specifying the handle of the required group row.

### Return Value

An object that is the value of the specified group row.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# GetRow Method

Gets an object that provides access to the data of a row specified by its handle.

**Overload List**

| Name | Description |
|---|---|
| static int GetRow(BindableObject bindable) | Inherited from **Xamarin.Forms.Grid** |
| IRowData GetRow(int rowHandle) | Gets an object that provides access to the data of a row specified by its handle. |

**See Also**
GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
GridControl.GetRow Overload List

Gets an object that provides access to the data of a row specified by its handle.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**
**C#**

```
public IRowData GetRow(
    int rowHandle
)
```

**Parameters**
*rowHandle*
An integer value specifying the desired data row handle.

**Return Value**

An object implementing the IRowData interface.

**Remarks**

To get an object representing a record in the grid's underlying data source to which the specified row corresponds, use the IRowData.DataObject property. To get a value in the specified field, use the IRowData.GetFieldValue method.

If the value of the *rowHandle* parameter exceeds the permissible limits (greater than the RowCount - 1), the **GetRow** property returns **null**.

**See Also**
GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
GridControl.GetRow Overload List
Identify and Access Rows

© 2018 Developer Express Inc.

# GetRowHandle Method

Returns the handle of the row which corresponds to the specified record in the data source.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public int GetRowHandle(
   int sourceRowIndex
)
```

### Parameters

*sourceRowIndex*
The zero-based index of a record in the data source.

### Return Value

An integer value specifying the handle of the row which corresponds to the specified record in the data source.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
Rows Overview

© 2018 Developer Express Inc.

# GetService Method

Gets the service object of the specified type.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public object GetService(
   Type serviceType
)
```

### Parameters

*serviceType*
An object that specifies the type of service object to get.

### Return Value

A service object of the specified type, or a null reference if there is no service object of this type.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# GetService<T> Method

Gets the specified service.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public T GetService<T>()
```

### Return Value

A service object of the specified type.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# GetSourceRowIndex Method

Returns the index of the data source record to which the specified row handle corresponds.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public int GetSourceRowIndex(
    int rowHandle
)
```

### Parameters

*rowHandle*
An integer value specifying the row handle.

### Return Value

An integer value specifying the zero-based index of the data record to which the specified row handle corresponds.

### Remarks

To get a row handle by its index in the data source, use the GetRowHandle method.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
Rows Overview

© 2018 Developer Express Inc.

# GroupBy Method

Groups data in the grid by the specified column.

**Overload List**

| Name | Description |
|---|---|
| void GroupBy(GridColumn column) | Groups data in the grid by the specified column. |
| void GroupBy(string fieldName) | Groups data in the grid by the specified column. |

**See Also**
GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
GridControl.GroupBy Overload List

Groups data in the grid by the specified column.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**
**C#**
```
public void GroupBy(
   GridColumn column
)
```

**Parameters**
*column*
A GridColumn object or descendant specifying the grid column by whose values data is grouped.

**Remarks**

Another way to group data by a column is to set this column's GridColumn.IsGrouped property to **true**. End-users can group data by a column using a menu that appears when they touch and hold this column.

To cancel grouping, use the ClearGrouping method.

**See Also**
GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
GridControl.GroupBy Overload List
GridColumn.AllowGroup
Lesson 4 - Group Data

Groups data in the grid by the specified column.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**
**C#**
```
public void GroupBy(
   string fieldName
)
```

**Parameters**
*fieldName*
A System.String value that specifies the column's field name.

**Remarks**

© 2018 Developer Express Inc.

Another way to group data by a column is to set this column's GridColumn.IsGrouped property to **true**. End-users can group data by a column using a menu that appears when they touch and hold this column.

To cancel grouping, use the ClearGrouping method.

**See Also**
GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
GridControl.GroupBy Overload List
GridColumn.AllowGroup
Lesson 4 - Group Data

© 2018 Developer Express Inc.

# HideColumnChooser Method

Hides the column chooser.

**Overload List**

| Name | Description |
|---|---|
| void HideColumnChooser() | Hides the column chooser. |
| void HideColumnChooser(bool ApplyChanges) | Hides the column chooser, applying or discarding its current settings. |

**See Also**
GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
GridControl.HideColumnChooser Overload List

Hides the column chooser.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

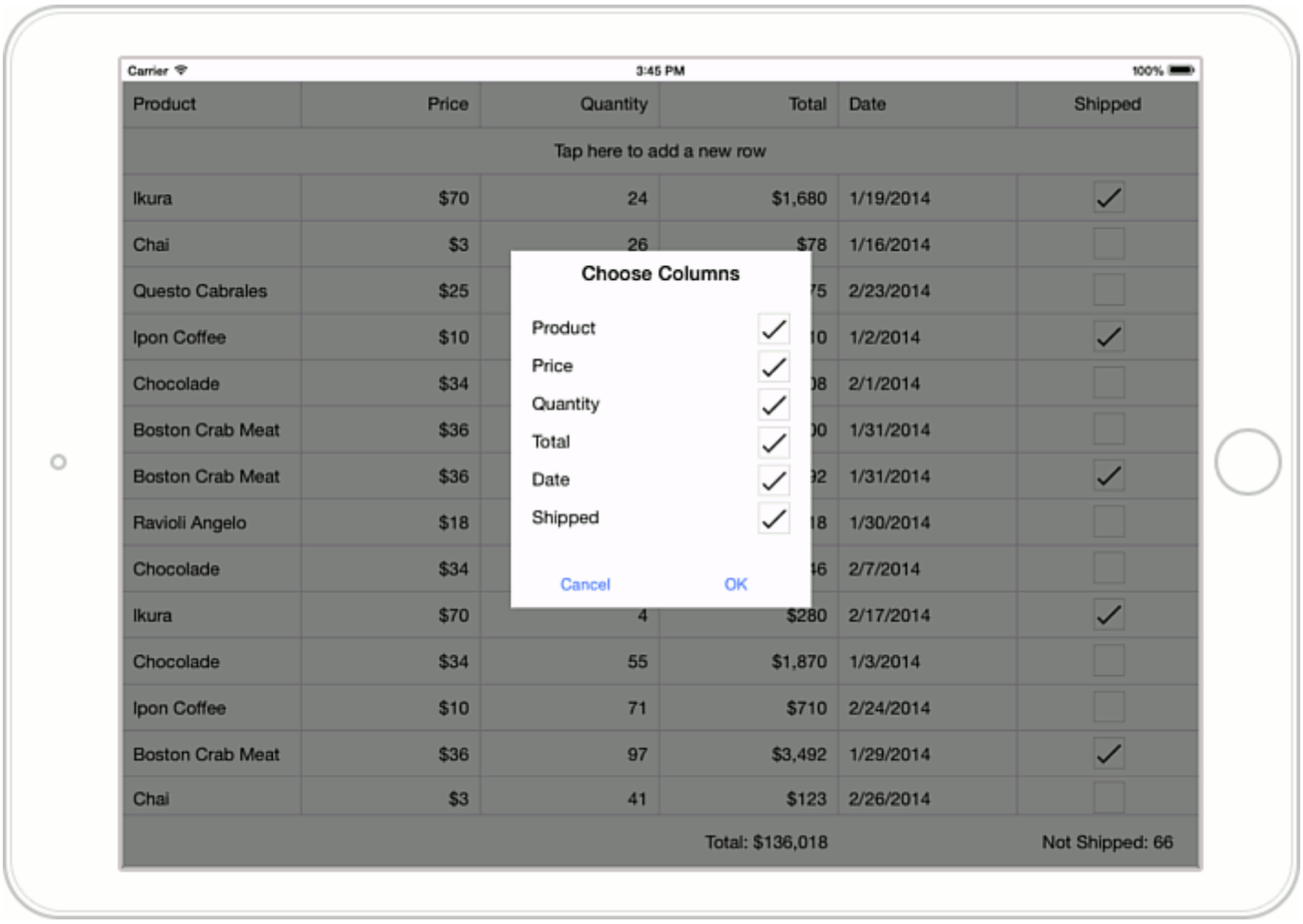
**Syntax**
**C#**
```
public void HideColumnChooser()
```

**Remarks**

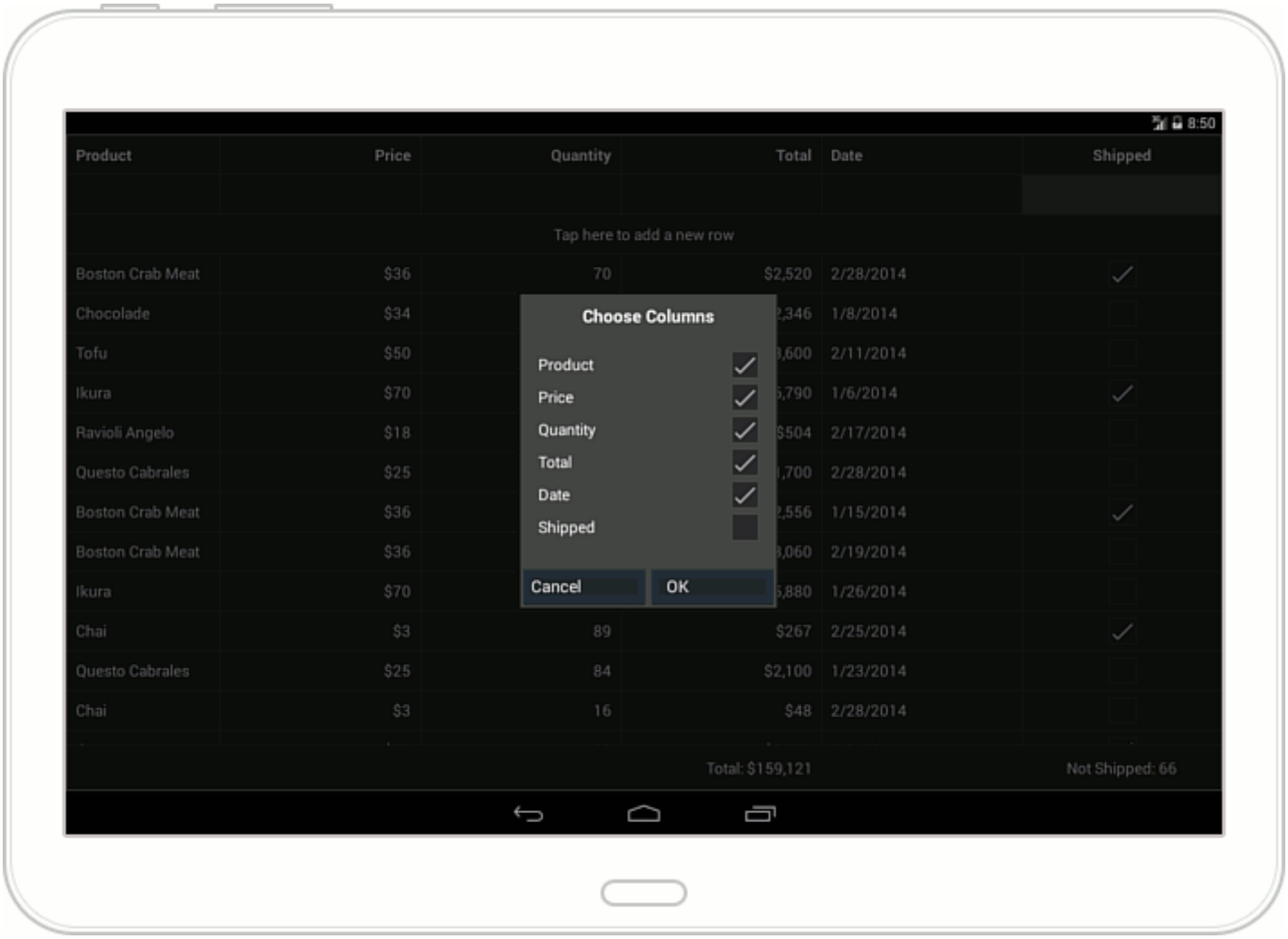
The column chooser enables end-users to show and hide columns from the grid at runtime. The column chooser is shown when an end-user touches and holds a column header and then taps the corresponding item in the invoked menu.

To show and hide the column chooser programmatically, use the ShowColumnChooser and **HideColumnChooser** methods. To disable the column chooser for end-users, set the IsColumnChooserEnabled property to **false**.

© 2018 Developer Express Inc.




© 2018 Developer Express Inc.




**See Also**
GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
GridControl.HideColumnChooser Overload List

Hides the column chooser, applying or discarding its current settings.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**
**C#**
```
public void HideColumnChooser(
   bool ApplyChanges
)
```

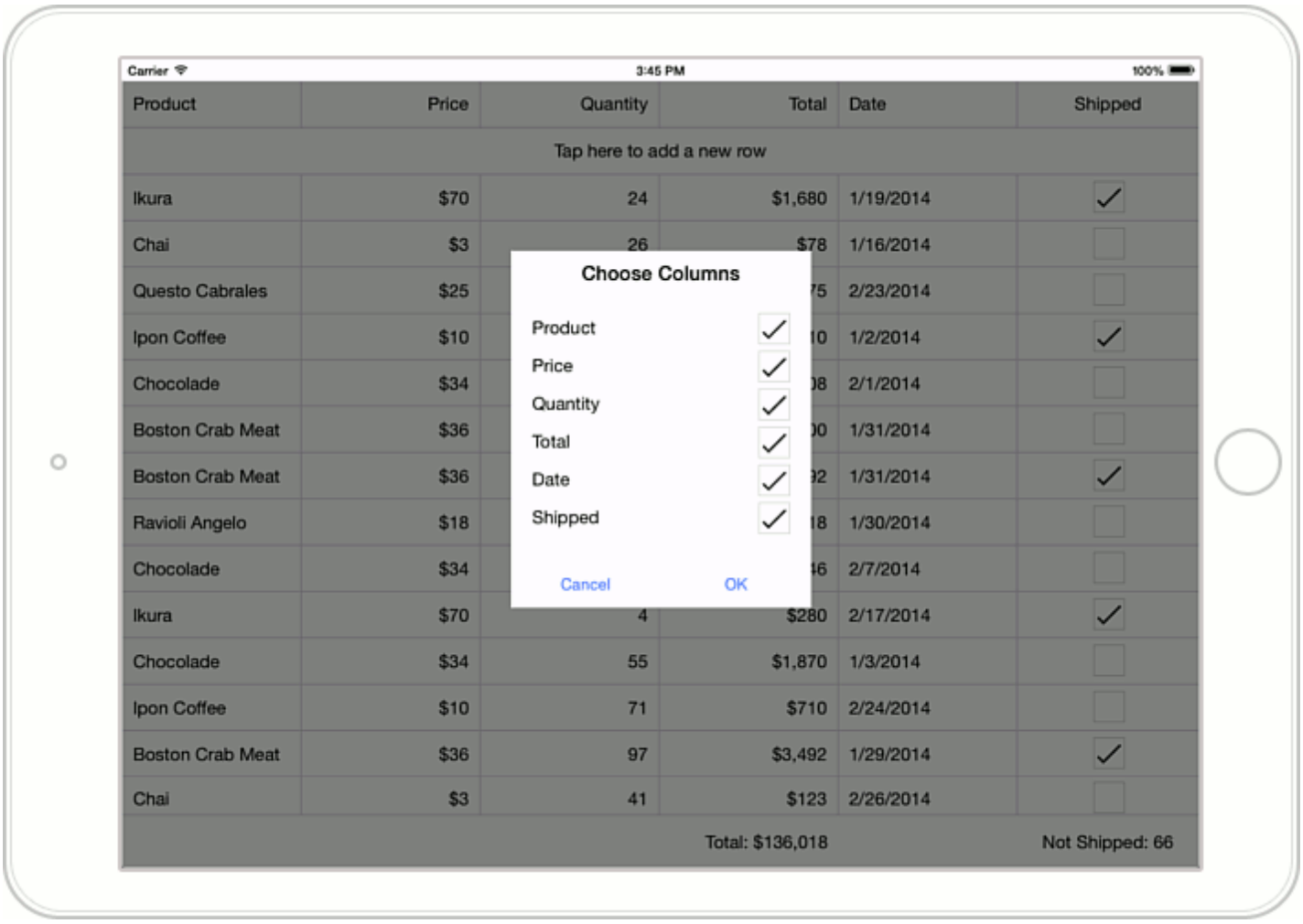
**Parameters**
*ApplyChanges*
**true**, if column visibility settings specified in the column chooser are applied to the grid; otherwise, **false**.

**Remarks**

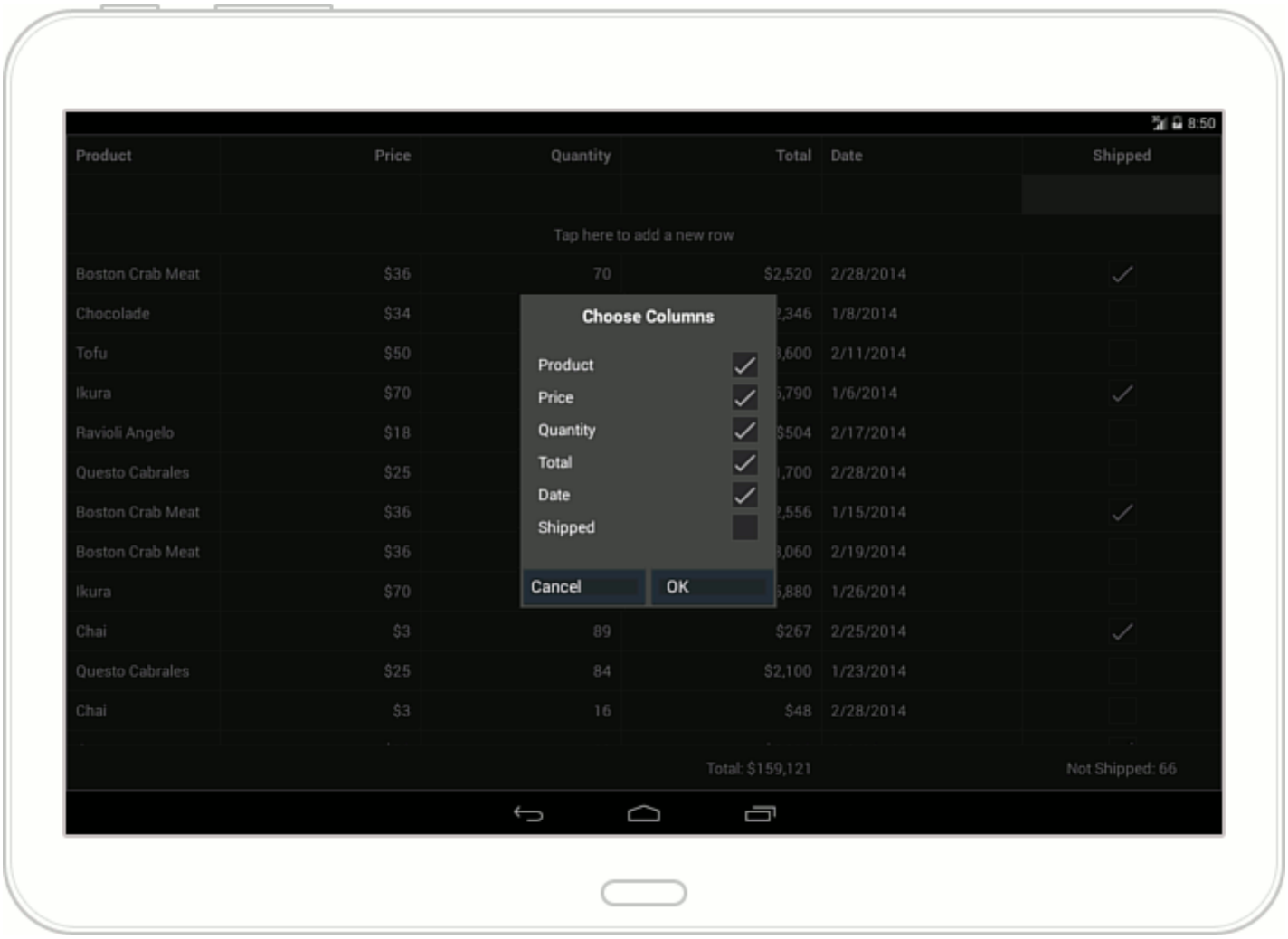
The column chooser enables end-users to show and hide columns from the grid at runtime. The column chooser is shown when an end-user touches and holds a column header and then taps the corresponding item in the invoked menu.

To show and hide the column chooser programmatically, use the ShowColumnChooser and **HideColumnChooser** methods. To disable the column chooser for end-users, set the IsColumnChooserEnabled property to **false**.

© 2018 Developer Express Inc.




© 2018 Developer Express Inc.




**See Also**
GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
GridControl.HideColumnChooser Overload List

© 2018 Developer Express Inc.

# IsGroupCollapsed Method

Indicates whether the specified group row is collapsed.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public bool IsGroupCollapsed(
   int rowHandle
)
```

### Parameters

*rowHandle*
An integer value that specifies the group row handle.

### Return Value

**true**, if the specified group row is collapsed; otherwise, **false**.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
IGroupInfo.IsCollapsed
Lesson 4 - Group Data

© 2018 Developer Express Inc.

# IsGroupRow Method

Indicates whether the specified handle corresponds to a group row.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public bool IsGroupRow(
   int rowHandle
)
```

### Parameters

*rowHandle*
An integer value that is the handle associated with the tested row.

### Return Value

**true**, if the specified handle corresponds to a group row; otherwise, **false**.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
Lesson 4 - Group Data

© 2018 Developer Express Inc.

# LoadInVisualTree Method

For internal use.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**
**C#**

```
public void LoadInVisualTree()
```

**See Also**
GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# OpenEditor Method

Invokes an editor (an in-place editor or edit form, depending on the edit mode set in the grid) for a data row containing the specified cell.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public void OpenEditor(
   CellIndex cellIndex
)
```

### Parameters

*cellIndex*
A CellIndex object that specifies a cell position in the grid.

### Remarks

End-users can switch the grid to edit mode by double-tapping a data cell or tapping **Edit Cell** in a menu that appears when they touch and hold a data row. The AllowEditRows property allows you to prevent end-users from editing cell values via the GridControl's UI. However, it has no effect if a row editor is invoked via the **OpenEditor** method.

To close a row editor, use the CloseEditor method. When a row editor is closed, the EndRowEdit event occurs.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# Redraw Method

Forces the control to immediately redraw itself.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public void Redraw(
   bool forceLayout
)
```

### Parameters

*forceLayout*
**true**, to repaint an entire grid layout ; otherwise, **false**.

### Remarks

The **Redraw** method recalculates the graphical information of the GridControl and then redraws it. It can be useful when you custom paint grid elements and need to redraw them due to custom changes that do not lead to automatic repainting.

If you need to only redraw grid rows, set the *forceLayout* parameter to **false**. To force update of the entry control, set this parameter to **true**.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# RefreshData Method

Updates the grid data.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public void RefreshData()
```

### Remarks

Calling the **RefreshData** method only makes sense when the bound data source does not support the System.Collections.Specialized.INotifyCollectionChanged interface. Otherwise, the control will automatically update itself, since such data sources provide data change notifications.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# RestoreLayoutFromStream Method

Restores a grid's layout from the specified stream.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public void RestoreLayoutFromStream(
   Stream stream
)
```

### Parameters

*stream*
A System.IO.Stream descendant from which grid settings are read.

### Remarks

Use the **RestoreLayoutFromStream** method to load the grid's layout from the stream to which the layout was saved via the SaveLayoutToStream method. If the specified stream is empty or **null**, an exception is raised.

### Example

The following sample code shows how to save the current layout of the grid to a memory stream on a button click and restore it on another button click. The SaveLayoutToStream and RestoreLayoutFromStream methods are used.

**C#**

```
using System.IO;
// ...
MemoryStream layoutStream;
void SaveButton_Click(object sender, EventArgs e) {
    // Save the grid's layout.
    layoutStream = new MemoryStream();
    grid.SaveLayoutToStream(layoutStream);
}
void LoadButton_Click(object sender, EventArgs e) {
    // Restore the previously saved layout.
    layoutStream.Position = 0;
    grid.RestoreLayoutFromStream(layoutStream);
}
```

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
SaveLayoutToXml
RestoreLayoutFromXml

© 2018 Developer Express Inc.

# RestoreLayoutFromXml Method

Restores a grid's layout from the specified XML string.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public void RestoreLayoutFromXml(
   string xmlContent
)
```

### Parameters

*xmlContent*
A string from which grid settings are read.

### Remarks

Use the **RestoreLayoutFromXml** method to load the previously saved grid layout to an XML sting with the SaveLayoutToXml method.

### Example

The following sample code shows how to save the current layout of the grid to an XML string on a button click and restore it on another button click. The SaveLayoutToXml and RestoreLayoutFromXml methods are used.

**C#**

```
string customState = String.Empty;
void SaveButton_Click(object sender, EventArgs e) {
    // Save the grid's layout.
    customState = grid.SaveLayoutToXml();
}
void LoadButton_Click(object sender, EventArgs e) {
    // Restore the previously saved layout.
    grid.RestoreLayoutFromXml(customState);
}
```

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
SaveLayoutToStream
RestoreLayoutFromStream

© 2018 Developer Express Inc.

# SaveLayoutToStream Method

Saves a grid's layout to the specified stream.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public void SaveLayoutToStream(
   Stream stream
)
```

### Parameters

*stream*
A System.IO.Stream descendant to which a control's layout is written.

### Remarks

Use the **SaveLayoutToStream** method to write a GridControl's layout to a stream. You can then read saved settings using the RestoreLayoutFromStream method.

### Example

The following sample code shows how to save the current layout of the grid to a memory stream on a button click and restore it on another button click. The SaveLayoutToStream and RestoreLayoutFromStream methods are used.

**C#**

```
using System.IO;
// ...
MemoryStream layoutStream;
void SaveButton_Click(object sender, EventArgs e) {
    // Save the grid's layout.
    layoutStream = new MemoryStream();
    grid.SaveLayoutToStream(layoutStream);
}
void LoadButton_Click(object sender, EventArgs e) {
    // Restore the previously saved layout.
    layoutStream.Position = 0;
    grid.RestoreLayoutFromStream(layoutStream);
}
```

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
SaveLayoutToXml
RestoreLayoutFromXml

© 2018 Developer Express Inc.

# SaveLayoutToXml Method

Saves a gird's layout to an XML string.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public string SaveLayoutToXml()
```

### Return Value

A string to which a grid's layout settings should be stored.

### Remarks

Use the **SaveLayoutToXml** method to save a GridControl's layout as an XML string. You can then restore saved settings using the RestoreLayoutFromXml method.

### Example

The following sample code shows how to save the current layout of the grid to an XML string on a button click and restore it on another button click. The SaveLayoutToXml and RestoreLayoutFromXml methods are used.

**C#**

```
string customState = String.Empty;
void SaveButton_Click(object sender, EventArgs e) {
    // Save the grid's layout.
    customState = grid.SaveLayoutToXml();
}
void LoadButton_Click(object sender, EventArgs e) {
    // Restore the previously saved layout.
    grid.RestoreLayoutFromXml(customState);
}
```

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
SaveLayoutToStream
RestoreLayoutFromStream

© 2018 Developer Express Inc.

# ScrollToRow Method

Scrolls the grid to make the specified row visible.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public void ScrollToRow(
   int rowHandle
)
```

### Parameters

*rowHandle*
A integer value that specifies the handle of the row to which the grid must be scrolled.

### See Also

GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# SetCellValue Method

Assigns a value to the specified cell.

**Overload List**

| Name | Description |
|---|---|
| void SetCellValue(int rowHandle, GridColumn column, object value) | Assigns a value to the specified cell. |
| void SetCellValue(int rowHandle, string fieldName, object value) | Assigns a value to the specified cell. |

**See Also**
GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
GridControl.SetCellValue Overload List

Assigns a value to the specified cell.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**
**C#**
```
public void SetCellValue(
   int rowHandle,
   GridColumn column,
   object value
)
```

**Parameters**
*rowHandle*
A zero based integer value that specifies the handle of the row which contains the cell.
*column*
A GridColumn object or descendant specifying the column that contains the cell.
*value*
An object that specifies the cell's new value.

**Remarks**

To learn more, see Cells.

**See Also**
GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
GridControl.SetCellValue Overload List
GetCellValue

Assigns a value to the specified cell.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**
**C#**
```
public void SetCellValue(
   int rowHandle,
   string fieldName,
   object value
)
```

© 2018 Developer Express Inc.

**Parameters**
*rowHandle*
A zero-based integer value that specifies the handle of the row that contains the cell.
*fieldName*
A System.String value that specifies the column's field name.
*value*
An object that specifies the cell's new value.

**Remarks**

To learn more, see Cells.

**See Also**
GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
GridControl.SetCellValue Overload List
GetCellValue

© 2018 Developer Express Inc.

# ShowColumnChooser Method

Invokes the column chooser.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

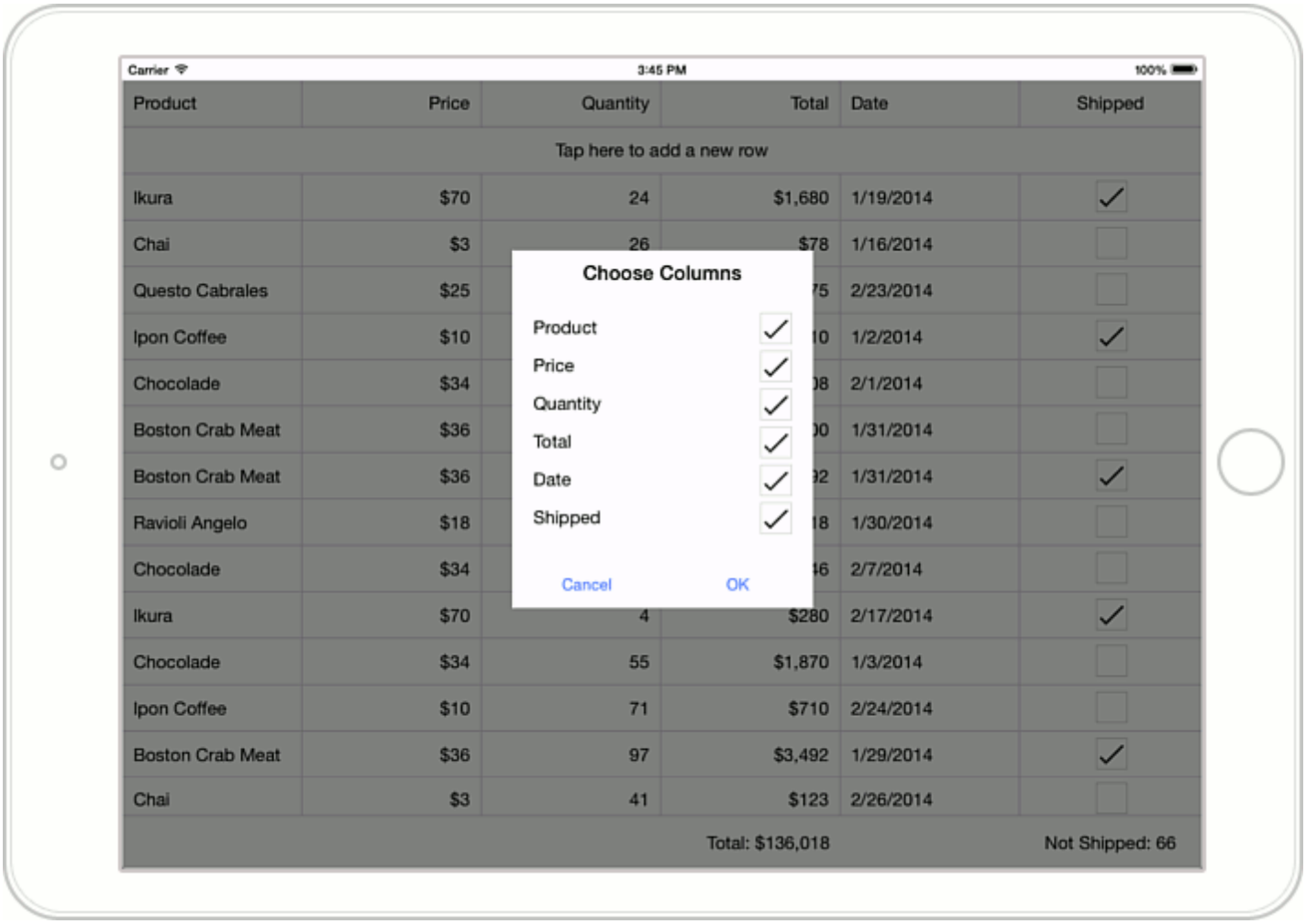
**C#**

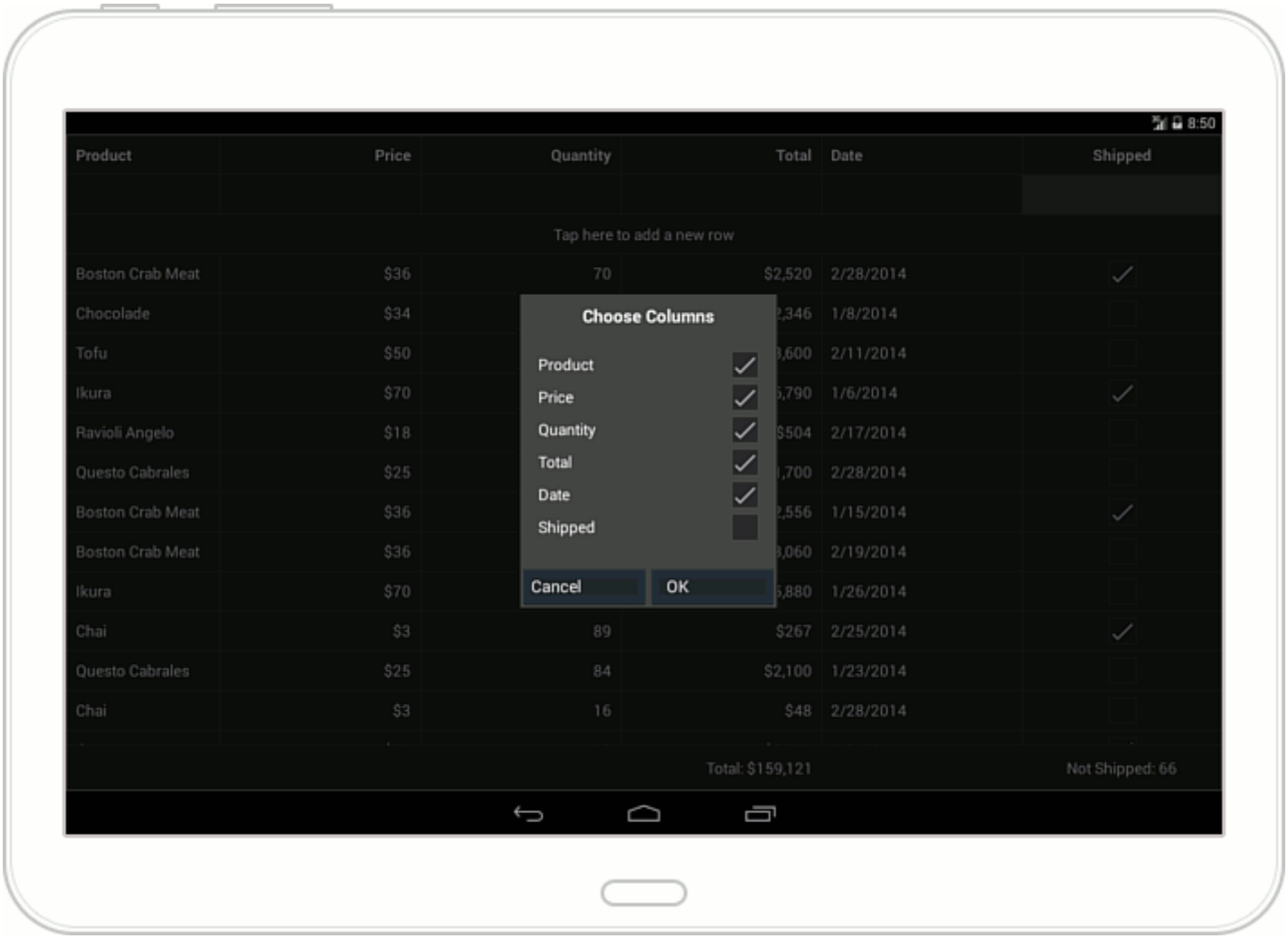
```
public void ShowColumnChooser()
```

### Remarks

The column chooser enables end-users to show and hide columns from the grid at runtime. The column chooser is shown when an end-user touches and holds a column header and then taps the corresponding item in the invoked menu.

To show and hide the column chooser programmatically, use the **ShowColumnChooser** and HideColumnChooser methods. To disable the column chooser for end-users, set the IsColumnChooserEnabled property to **false**.






**See Also**
GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# SortBy Method

Sorts data by the values of the specified column in ascending order.

**Overload List**

| Name | Description |
|---|---|
| void SortBy(GridColumn column) | Sorts data by the values of the specified column in ascending order. |
| void SortBy(string fieldName) | Sorts data by the values of the column bound to the specified data field in the ascending order. |
| void SortBy(GridColumn column, ColumnSortOrder sortOrder) | Sorts data by the values of the specified column in the specified order. |
| void SortBy(GridColumn column, ColumnSortOrder sortOrder, int sortIndex) | Sorts data by the values of the specified column in the specified order, and places the column at the specified position among the sorted columns. |

**See Also**
GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
GridControl.SortBy Overload List

Sorts data by the values of the specified column in ascending order.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**
**C#**

```
public void SortBy(
   GridColumn column
)
```

**Parameters**
*column*
A GridColumn object or descendant that is the column within the grid.

**Remarks**

By default, data in a grid can be sorted by one column at a time (the SortMode property is set to GridSortMode.Single). In this case, applying sorting to a column by calling the **SortBy** method clears sorting previously applied to another column. If sorting data by multiple columns is enabled (the SortMode property is set to GridSortMode.Multiple), any previous sorting is not cleared when calling the **SortBy** method. The new sort order applies an additional sorting condition to the one previously applied. To specify the sequence of applying sorting to columns, assign the GridColumn.SortIndex of these columns.

Another way to sort data in the grid by a column values is to use the column's GridColumn.SortOrder property. End-users can sort data by a column using a menu that appears when they touch and hold this column, or by tapping a column header.

To clear sorting applied in the grid, call the ClearSorting method.

**See Also**
GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
GridControl.SortBy Overload List
GridColumn.SortOrder
GridColumn.SortIndex
SortMode
Lesson 6 - Sort Data

© 2018 Developer Express Inc.

Sorts data by the values of the column bound to the specified data field in the ascending order.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**
**C#**
```
public void SortBy(
   string fieldName
)
```

**Parameters**
*fieldName*
A System.String value that specifies the column's field name.

**Remarks**

By default, data in a grid can be sorted by one column at a time (the SortMode property is set to GridSortMode.Single). In this case, applying sorting to a column by calling the **SortBy** method clears sorting previously applied to another column. If sorting data by multiple columns is enabled (the SortMode property is set to GridSortMode.Multiple), any previous sorting is not cleared when calling the **SortBy** method. The new sort order applies an additional sorting condition to the one previously applied. To specify the sequence of applying sorting to columns, assign the GridColumn.SortIndex of these columns.

Another way to sort data in the grid by column values is to use the column's GridColumn.SortOrder property. End-users can sort data by a column using a menu that appears when they touch and hold this column, or by tapping a column header.

To clear sorting applied in the grid, call the ClearSorting method.

**See Also**
GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
GridControl.SortBy Overload List
GridColumn.SortOrder
GridColumn.SortIndex
SortMode
Lesson 6 - Sort Data

Sorts data by the values of the specified column in the specified order.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**
**C#**
```
public void SortBy(
   GridColumn column,
   ColumnSortOrder sortOrder
)
```

**Parameters**
*column*
A GridColumn object or descendant that is the column within the grid.
*sortOrder*
A ColumnSortOrder enumeration value that specifies the column's sort order.

**Remarks**

By default, data in a grid can be sorted by one column at a time (the SortMode property is set to GridSortMode.Single). In this case, applying sorting to a column by calling the **SortBy** method clears sorting previously applied to another column. If sorting data by multiple columns is enabled (the SortMode property is set to GridSortMode.Multiple), any previous sorting is not cleared when calling the **SortBy** method. The new sort order applies an additional sorting condition to the one previously applied. To specify the sequence of applying sorting to columns, assign the GridColumn.SortIndex of these columns.

Another way to sort data in the grid by column values is to use the column's GridColumn.SortOrder property. End-users can sort data by a column using a menu that appears when they touch and hold this column, or by tapping a column header.

To clear sorting applied in the grid, call the ClearSorting method.

© 2018 Developer Express Inc.

**See Also**
GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
GridControl.SortBy Overload List
GridColumn.SortOrder
GridColumn.SortIndex
SortMode
Lesson 6 - Sort Data

Sorts data by the values of the specified column in the specified order, and places the column at the specified position among the sorted columns.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**
**C#**
```
public void SortBy(
   GridColumn column,
   ColumnSortOrder sortOrder,
   int sortIndex
)
```

**Parameters**
*column*
A GridColumn object or descendant that is the column within the grid.
*sortOrder*
A ColumnSortOrder enumeration value that specifies the column's sort order.
*sortIndex*
An integer value that specifies the zero-based column's index among the sorted columns.

**Remarks**

By default, data in a grid can be sorted by one column at a time (the SortMode property is set to GridSortMode.Single). In this case, applying sorting to a column by calling the **SortBy** method clears sorting previously applied to another column. If sorting data by multiple columns is enabled (the SortMode property is set to GridSortMode.Multiple), any previous sorting is not cleared when calling the **SortBy** method. The new sort order applies an additional sorting condition to the one previously applied.

Another way to sort data in the grid by column values is to use the GridColumn.SortOrder and GridColumn.SortIndex properties. End-users can sort data by a column using a menu that appears when they touch and hold this column, or by tapping a column header.

To clear sorting applied in the grid, call the ClearSorting method.

**See Also**
GridControl Class
GridControl Members
DevExpress.Mobile.DataGrid Namespace
GridControl.SortBy Overload List
GridColumn.SortOrder
GridColumn.SortIndex
SortMode
Lesson 6 - Sort Data

© 2018 Developer Express Inc.

# GridPopupMenuType Enumeration

Lists values specifying GridControl's context menu types.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public enum GridPopupMenuType
```

### Members

| Name | Description |
|---|---|
| DataRow | A context shown when an end-user touches and holds a data row cell. |
| GroupRow | A menu shown when an end-user touches and holds a group row. |
| Header | A menu shown when an end-user touches and holds a column header. |
| TotalSummary | A menu shown when an end-user touches and holds a total summary panel. |

### Remarks

Values listed by the **GridPopupMenuType** enumeration are returned by the PopupMenuEventArgs.MenuType property, which is exposed via the GridControl.PopupMenuCustomization event's parameter.

### See Also

DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# GridSortMode Enumeration

Lists values specifying whether the grid allows data sorting by multiple columns or by a single column only.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public enum GridSortMode
```

### Members

| Name | Description |
|---|---|
| Multiple | The grid allows sorting data by multiple columns. |
| Single | The grid allows sorting data by only a single column at a time. |

### Remarks

The **GridSortMode** enumeration values are used by the GridControl.SortMode property.

### See Also

DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# HorizontalScrollOffsetEventArgs Class

Provides data for the GridControl.HorizontalScrollOffsetChanged event.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public class HorizontalScrollOffsetEventArgs : EventArgs
```

### Remarks

The **HorizontalScrollOffsetEventArgs** objects are automatically created, initialized and passed to the GridControl.HorizontalScrollOffsetChanged event handlers.

### Inheritance Hierarchy

System.Object
  System.EventArgs
    **HorizontalScrollOffsetEventArgs**

### See Also

HorizontalScrollOffsetEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# HorizontalScrollOffsetEventArgs Members

Provides data for the GridControl.HorizontalScrollOffsetChanged event.

The following tables list the members exposed by the HorizontalScrollOffsetEventArgs type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | HorizontalScrollOffsetEventArgs | Initializes a new instance of the HorizontalScrollOffsetEventArgs class. |

### Public Properties

| | Name | Description |
|---|---|---|
| | HorizontalOffset | Gets a horizontal scrolling offset. |

### Public Methods

| | Name | Description |
|---|---|---|
| | Equals | (Inherited from **System.Object**) |
| S | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

### Public Fields

| | Name | Description |
|---|---|---|
| S | Empty | (Inherited from **System.EventArgs**) |

**See Also**
HorizontalScrollOffsetEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# HorizontalScrollOffsetEventArgs Constructor

Initializes a new instance of the HorizontalScrollOffsetEventArgs class.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public HorizontalScrollOffsetEventArgs(
   Double horizontalOffset
)
```

### Parameters

*horizontalOffset*
A double value that specifies the horizontal scrolling offset. This value is assigned to the HorizontalOffset property.

### Remarks

Instances of the HorizontalScrollOffsetEventArgs class are automatically created, initialized and passed to the corresponding event handlers.

### See Also

HorizontalScrollOffsetEventArgs Class
HorizontalScrollOffsetEventArgs Members
DevExpress.Mobile.DataGrid Namespace
GridControl.HorizontalScrollOffset

© 2018 Developer Express Inc.

# HorizontalScrollOffsetEventArgs Properties

Provides data for the GridControl.HorizontalScrollOffsetChanged event.

The following tables list the members exposed by the HorizontalScrollOffsetEventArgs type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | HorizontalScrollOffsetEventArgs | Initializes a new instance of the HorizontalScrollOffsetEventArgs class. |

### Public Properties

| | Name | Description |
|---|---|---|
| | HorizontalOffset | Gets a horizontal scrolling offset. |

### Public Methods

| | Name | Description |
|---|---|---|
| | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

### Public Fields

| | Name | Description |
|---|---|---|
| | Empty | (Inherited from **System.EventArgs**) |

**See Also**
HorizontalScrollOffsetEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# HorizontalOffset Property

Gets a horizontal scrolling offset.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public Double HorizontalOffset { get; }
```

### Property Value

A double value, which specifies the scrolling offset (in density-independent pixels).

### Remarks

To set a horizontal scrolling offset directly from your code, use the GridControl.HorizontalScrollOffset property.

### See Also

HorizontalScrollOffsetEventArgs Class
HorizontalScrollOffsetEventArgs Members
DevExpress.Mobile.DataGrid Namespace
GridControl.HorizontalScrollOffset

© 2018 Developer Express Inc.

# HorizontalScrollOffsetEventHandler Delegate

Represents a method that will handle the GridControl.HorizontalScrollOffsetChanged event.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public delegate void HorizontalScrollOffsetEventHandler(
   object sender,
   HorizontalScrollOffsetEventArgs e
)
```

### Parameters

The declaration of your event handler must have the same parameters as the **HorizontalScrollOffsetEventHandler** delegate declaration.
*sender*
The event sender.
*e*
A HorizontalScrollOffsetEventArgs object that contains event data.

### See Also

DevExpress.Mobile.DataGrid Namespace
GridControl.HorizontalScrollOffset

© 2018 Developer Express Inc.

# IconSetFormatCondition Class

Represents an icon set format condition.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public class IconSetFormatCondition : IndicatorFormatConditionBase
```

### Remarks

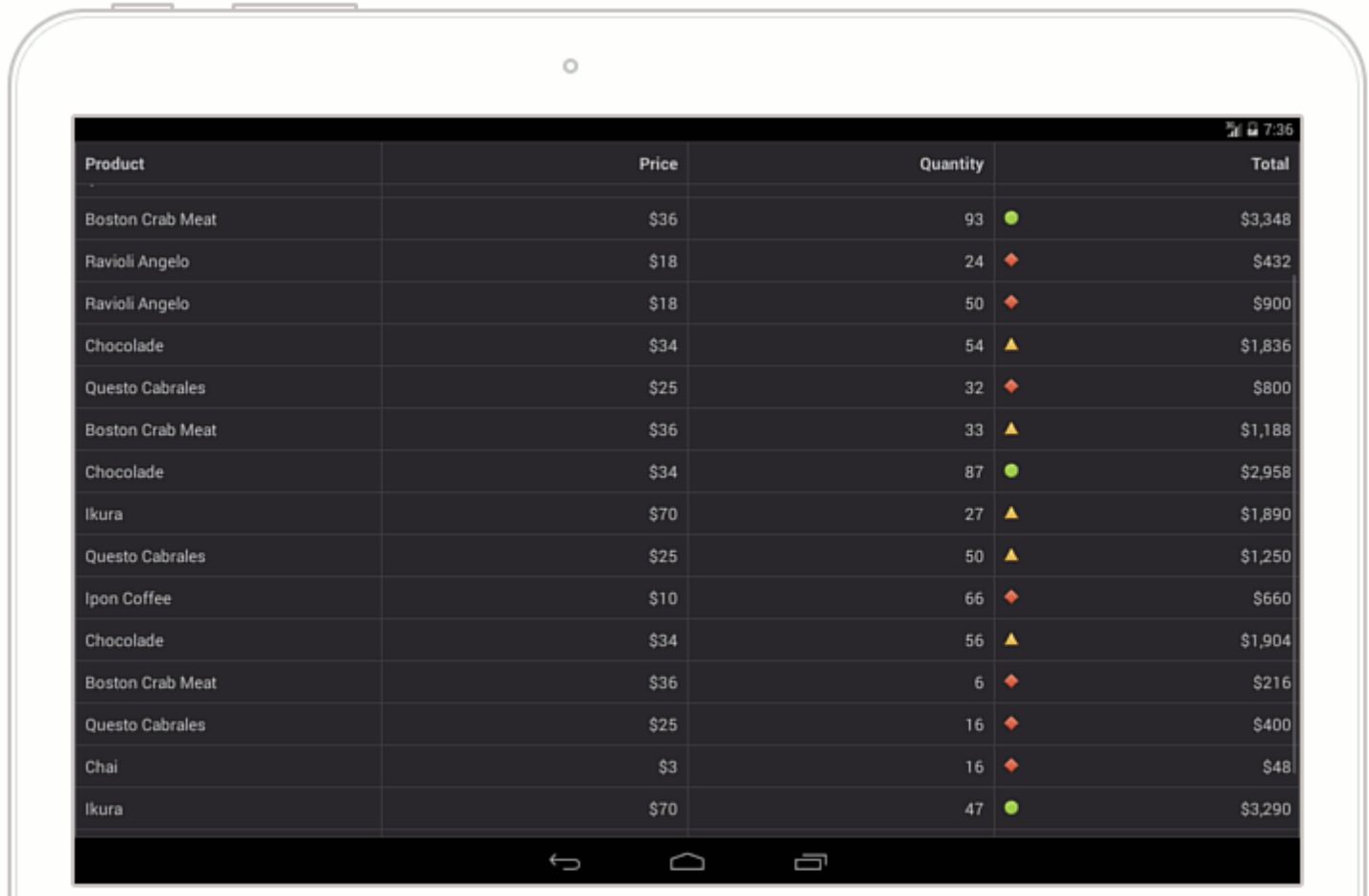
An icon set format allows you to classify column cell values into several ranges, assign an icon to each range, and display a specific icon in a cell based on the cell value.

### Example

This example demonstrates how to apply an icon set conditional formatting rule to the **Total** column in a GridControl.

© 2018 Developer Express Inc.

| Product | Price | Quantity | Total |
|---|---|---|---|
| Ikura | $70 | 9 | $630 |
| Boston Crab Meat | $36 | 81 | $2,916 |
| Questo Cabrales | $25 | 19 | $475 |
| Ravioli Angelo | $18 | 17 | $306 |
| Ravioli Angelo | $18 | 94 | $1,692 |
| Ipon Coffee | $10 | 37 | $370 |
| Ravioli Angelo | $18 | 50 | $900 |
| Boston Crab Meat | $36 | 55 | $1,980 |
| Ikura | $70 | 33 | $2,310 |
| Ravioli Angelo | $18 | 72 | $1,296 |
| Chocolade | $34 | 73 | $2,482 |
| Ikura | $70 | 38 | $2,660 |
| Ikura | $70 | 36 | $2,520 |
| Ikura | $70 | 51 | $3,570 |
| Chocolade | $34 | 80 | $2,720 |
| Chai | $3 | 12 | $36 |




| Product | Price | Quantity | Total |
|---|---|---|---|
| Boston Crab Meat | $36 | 93 | $3,348 |
| Ravioli Angelo | $18 | 24 | $432 |
| Ravioli Angelo | $18 | 50 | $900 |
| Chocolade | $34 | 54 | $1,836 |
| Questo Cabrales | $25 | 32 | $800 |
| Boston Crab Meat | $36 | 33 | $1,188 |
| Chocolade | $34 | 87 | $2,958 |
| Ikura | $70 | 27 | $1,890 |
| Questo Cabrales | $25 | 50 | $1,250 |
| Ipon Coffee | $10 | 66 | $660 |
| Chocolade | $34 | 56 | $1,904 |
| Boston Crab Meat | $36 | 6 | $216 |
| Questo Cabrales | $25 | 16 | $400 |
| Chai | $3 | 16 | $48 |
| Ikura | $70 | 47 | $3,290 |

1. Create a new IconSetFormatCondition class instance.
2. Set this object's FormatConditionBase.FieldName property to **Total**. This column provides values to test against the formatting rule and the specified format will be applied to this column.
3. Assign one of the predefined icon sets to the FormatConditionBase.PredefinedFormatName property. All available format names are listed in the Predefined Format Names document.
4. To apply a conditional formatting rule represented by the created **IconSetFormatCondition** object, add this object to the GridControl.FormatConditions collection.

**C#**

```
using DevExpress.Mobile.DataGrid;
// ...
IconSetFormatCondition condition = new IconSetFormatCondition();
condition.FieldName = "Total";
condition.PredefinedFormatName = "Signs3IconSet";
grid.FormatConditions.Add (condition);
```

### Inheritance Hierarchy

System.Object
  Xamarin.Forms.BindableObject
    FormatConditionBase
      IndicatorFormatConditionBase
        **IconSetFormatCondition**

### See Also

IconSetFormatCondition Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# IconSetFormatCondition Members

Represents an icon set format condition.

The following tables list the members exposed by the IconSetFormatCondition type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | IconSetFormatCondition | Initializes a new instance of the IconSetFormatCondition class with default settings. |

### Public Properties

| | Name | Description |
|---|---|---|
| | BindingContext | (Inherited from **Xamarin.Forms.BindableObject**) |
| | FieldName | Gets or sets the name of the column which provides values to test against the current format rule, and to which the format should be applied. This is a bindable property. (Inherited from **FormatConditionBase**) |
| | Format | Gets or sets the formatting settings applied to cells whose values take part in conditional formatting. This is a bindable property. |
| | MaxValue | Specifies the maximum column value to accept for conditional formatting. This is a bindable property. (Inherited from **IndicatorFormatConditionBase**) |
| | MinValue | Specifies the minimum column value to accept for conditional formatting. This is a bindable property. (Inherited from **IndicatorFormatConditionBase**) |
| | Owner | Gets the format condition's owner. This is a bindable property. (Inherited from **FormatConditionBase**) |
| | PredefinedFormatName | Gets or sets the name of the predefined format to be applied to target cells. This is a bindable property. (Inherited from **FormatConditionBase**) |
| | SelectiveExpression | Specifies a custom rule for applying conditional formatting. This is a bindable property. (Inherited from **IndicatorFormatConditionBase**) |
| | TypeName | Gets or sets the data type name. This property supports the internal infrastructure and is not intended to be used directly from your code. (Inherited from **FormatConditionBase**) |

© 2018 Developer Express Inc.

### Public Methods

| | Name | Description |
|---|---|---|
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Equals | (Inherited from **System.Object**) |
| S | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ToString | (Inherited from **System.Object**) |

### Public Fields

| | Name | Description |
|---|---|---|
| S | BindingContextProperty | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | FieldNameProperty | Identifies the FormatConditionBase.FieldName bindable property. (Inherited from **FormatConditionBase**) |
| S | FormatProperty | Identifies the Format bindable property. |
| S | MaxValueProperty | Identifies the IndicatorFormatConditionBase.MaxValue bindable property. (Inherited from **IndicatorFormatConditionBase**) |
| S | MinValueProperty | Identifies the IndicatorFormatConditionBase.MinValue bindable property. (Inherited from **IndicatorFormatConditionBase**) |
| S | OwnerProperty | Identifies the FormatConditionBase.Owner bindable property. (Inherited from **FormatConditionBase**) |
| S | PredefinedFormatNameProperty | Identifies the FormatConditionBase.PredefinedFormatName bindable property. (Inherited from **FormatConditionBase**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | SelectiveExpressionProperty | Identifies the IndicatorFormatConditionBase.SelectiveExpression bindable property. (Inherited from **IndicatorFormatConditionBase**) |

**Public Events**

| | Name | Description |
|---|---|---|
| | BindingContextChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanging | (Inherited from **Xamarin.Forms.BindableObject**) |

**See Also**
IconSetFormatCondition Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# IconSetFormatCondition Constructor

Initializes a new instance of the IconSetFormatCondition class with default settings.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**
**C#**
```
public IconSetFormatCondition()
```

**See Also**
IconSetFormatCondition Class
IconSetFormatCondition Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# IconSetFormatCondition Properties

Represents an icon set format condition.

The following tables list the members exposed by the IconSetFormatCondition type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | IconSetFormatCondition | Initializes a new instance of the IconSetFormatCondition class with default settings. |

### Public Properties

| | Name | Description |
|---|---|---|
| | BindingContext | (Inherited from **Xamarin.Forms.BindableObject**) |
| | FieldName | Gets or sets the name of the column which provides values to test against the current format rule, and to which the format should be applied. This is a bindable property. (Inherited from **FormatConditionBase**) |
| | Format | Gets or sets the formatting settings applied to cells whose values take part in conditional formatting. This is a bindable property. |
| | MaxValue | Specifies the maximum column value to accept for conditional formatting. This is a bindable property. (Inherited from **IndicatorFormatConditionBase**) |
| | MinValue | Specifies the minimum column value to accept for conditional formatting. This is a bindable property. (Inherited from **IndicatorFormatConditionBase**) |
| | Owner | Gets the format condition's owner. This is a bindable property. (Inherited from **FormatConditionBase**) |
| | PredefinedFormatName | Gets or sets the name of the predefined format to be applied to target cells. This is a bindable property. (Inherited from **FormatConditionBase**) |
| | SelectiveExpression | Specifies a custom rule for applying conditional formatting. This is a bindable property. (Inherited from **IndicatorFormatConditionBase**) |
| | TypeName | Gets or sets the data type name. This property supports the internal infrastructure and is not intended to be used directly from your code. (Inherited from **FormatConditionBase**) |

© 2018 Developer Express Inc.

### Public Methods

| | Name | Description |
|---|---|---|
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Equals | (Inherited from **System.Object**) |
| S | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ToString | (Inherited from **System.Object**) |

### Public Fields

| | Name | Description |
|---|---|---|
| S | BindingContextProperty | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | FieldNameProperty | Identifies the FormatConditionBase.FieldName bindable property. (Inherited from **FormatConditionBase**) |
| S | FormatProperty | Identifies the Format bindable property. |
| S | MaxValueProperty | Identifies the IndicatorFormatConditionBase.MaxValue bindable property. (Inherited from **IndicatorFormatConditionBase**) |
| S | MinValueProperty | Identifies the IndicatorFormatConditionBase.MinValue bindable property. (Inherited from **IndicatorFormatConditionBase**) |
| S | OwnerProperty | Identifies the FormatConditionBase.Owner bindable property. (Inherited from **FormatConditionBase**) |
| S | PredefinedFormatNameProperty | Identifies the FormatConditionBase.PredefinedFormatName bindable property. (Inherited from **FormatConditionBase**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | SelectiveExpressionProperty | Identifies the IndicatorFormatConditionBase.SelectiveExpression bindable property. (Inherited from **IndicatorFormatConditionBase**) |

**Public Events**

| | Name | Description |
|---|---|---|
| | BindingContextChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanging | (Inherited from **Xamarin.Forms.BindableObject**) |

**See Also**
IconSetFormatCondition Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# Format Property

Gets or sets the formatting settings applied to cells whose values take part in conditional formatting. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public IconSetFormat Format { get; set; }
```

### Property Value

An **IconSetFormat** object that provides formatting settings.

### See Also

IconSetFormatCondition Class
IconSetFormatCondition Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# IEditableRowData Interface

Contains information on a grid's row and allows you to modify field values in this row.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public interface IEditableRowData
```

### Remarks

Use the SetFieldValue method to assign a value to the specified field in the currently processed row.

### Inheritance Hierarchy

### See Also

IEditableRowData Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# IEditableRowData Members

Contains information on a grid's row and allows you to modify field values in this row.

The following tables list the members exposed by the IEditableRowData type.

### Public Properties

| | Name | Description |
|---|---|---|
| | DataObject | Gets an object representing a record in the grid's underlying data source. (Inherited from **IRowData**) |
| | RowHandle | Gets the processed row's handle. (Inherited from **IRowData**) |

### Public Methods

| | Name | Description |
|---|---|---|
| | GetFieldValue | Gets a value of the specified field within the processed row. (Inherited from **IRowData**) |
| | GetFieldValueGeneric<T> | Gets a value of the specified field within the processed row. (Inherited from **IRowData**) |
| | SetFieldValue | Sets a value to the specified field in the processed row. |

**See Also**
IEditableRowData Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# IEditableRowData Methods

Contains information on a grid's row and allows you to modify field values in this row.

The following tables list the members exposed by the IEditableRowData type.

**Public Methods**

| | Name | Description |
|---|---|---|
| | GetFieldValue | Gets a value of the specified field within the processed row. (Inherited from **IRowData**) |
| | GetFieldValueGeneric<T> | Gets a value of the specified field within the processed row. (Inherited from **IRowData**) |
| | SetFieldValue | Sets a value to the specified field in the processed row. |

**See Also**
IEditableRowData Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# SetFieldValue Method

Sets a value to the specified field in the processed row.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
void SetFieldValue(
   string fieldName,
   object value
)
```

### Parameters

*fieldName*
A System.String value that specifies the name of the field whose value must be assigned.
*value*
An object representing the value to be assigned.

### Remarks

If the GridControl.CustomUnboundColumnData event is raised when an end-user modifies an unbound column's value in the grid (the GridColumnDataEventArgs.IsSetData parameter is set to **true**), you can use the **SetFieldValue** method to assign a value to the specified field in the processed row.

### See Also

IEditableRowData Interface
IEditableRowData Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# IGroupInfo Interface

Provides properties to obtain information on a data group.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public interface IGroupInfo
```

### Remarks

Call the GridControl.GetGroupInfo method with a group row handle passed as a parameter to obtain an object implementing the **IGroupInfo** interface and holding information on the corresponding data group. This object's properties allow you to get group summary values calculated for the specified group (Summaries), the value for which the group was created (Value), the number of rows in the group (RowCount), etc.

### Inheritance Hierarchy

### See Also

IGroupInfo Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# IGroupInfo Members

Provides properties to obtain information on a data group.

The following tables list the members exposed by the IGroupInfo type.

**Public Properties**

| | Name | Description |
|---|---|---|
| | FieldName | Gets the field name of the column by which data is grouped. |
| | FirstRowHandle | Gets the handle of the first row in the group. |
| | GroupRowHandle | Gets the handle of the current group row. |
| | IsCollapsed | Indicates whether the group is collapsed. |
| | LastRowHandle | Gets the handle of the last row in the group. |
| | RowCount | Gets the number of data rows in the group. |
| | Summaries | Gets the list of objects that contain information on group summaries calculated for the data group and displayed within the group row. |
| | Value | Gets the value for which the group is created (a value which is displayed in a group row). |

**See Also**
IGroupInfo Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# IGroupInfo Properties

Provides properties to obtain information on a data group.

The following tables list the members exposed by the IGroupInfo type.

**Public Properties**

| | Name | Description |
|---|---|---|
| | FieldName | Gets the field name of the column by which data is grouped. |
| | FirstRowHandle | Gets the handle of the first row in the group. |
| | GroupRowHandle | Gets the handle of the current group row. |
| | IsCollapsed | Indicates whether the group is collapsed. |
| | LastRowHandle | Gets the handle of the last row in the group. |
| | RowCount | Gets the number of data rows in the group. |
| | Summaries | Gets the list of objects that contain information on group summaries calculated for the data group and displayed within the group row. |
| | Value | Gets the value for which the group is created (a value which is displayed in a group row). |

**See Also**
IGroupInfo Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# FieldName Property

Gets the field name of the column by which data is grouped.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
string FieldName { get; }
```

### Property Value

A System.String value the specifies the column's field name.

### Remarks

When grid data is grouped by the specified column, data rows that have identical values in this column are arranged into groups. Use the **FieldName** property to obtain the field name of the column by whose values data is grouped. To obtain a column value for which a particular group is created, use the Value property. By default, each group row displays a column caption and grouping value separated with colons.

### See Also

IGroupInfo Interface
IGroupInfo Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# FirstRowHandle Property

Gets the handle of the first row in the group.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
int FirstRowHandle { get; }
```

### Property Value

An integer value that specifies the row handle.

### Remarks

To get the handle of the last row in the group, use the LastRowHandle property. To obtain the number of rows in the group, use the RowCount property.

### See Also

IGroupInfo Interface
IGroupInfo Members
DevExpress.Mobile.DataGrid Namespace
Lesson 4 - Group Data

© 2018 Developer Express Inc.

# GroupRowHandle Property

Gets the handle of the current group row.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
int GroupRowHandle { get; }
```

### Property Value

An integer value that specifies the row handle.

### Remarks

To get handles of the first and last data rows within the group, use the FirstRowHandle and LastRowHandle properties, respectively.

### See Also

IGroupInfo Interface
IGroupInfo Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# IsCollapsed Property

Indicates whether the group is collapsed.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
bool IsCollapsed { get; }
```

### Property Value

**true**, if the group is collapsed; otherwise, **false**.

### See Also

IGroupInfo Interface
IGroupInfo Members
DevExpress.Mobile.DataGrid Namespace
GridControl.IsGroupCollapsed
GridControl.CollapseGroupRow
GridControl.CollapseAllGroups

© 2018 Developer Express Inc.

# LastRowHandle Property

Gets the handle of the last row in the group.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
int LastRowHandle { get; }
```

### Property Value

An integer value that specifies the row handle.

### Remarks

To get the handle of the first row in the group, use the FirstRowHandle property. To obtain the number of rows in the group, use the RowCount property.

### See Also

IGroupInfo Interface
IGroupInfo Members
DevExpress.Mobile.DataGrid Namespace
Lesson 4 - Group Data

© 2018 Developer Express Inc.

# RowCount Property

Gets the number of data rows in the group.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
int RowCount { get; }
```

### Property Value

An integer value that is the number of rows in the group.

### Remarks

To get handles of the first and last data rows within the group, use the FirstRowHandle and LastRowHandle properties, respectively.

### See Also

IGroupInfo Interface
IGroupInfo Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# Summaries Property

Gets the list of objects that contain information on group summaries calculated for the data group and displayed within the group row.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
IReadOnlyList<GridColumnSummary> Summaries { get; }
```

### Property Value

The list of GridColumnSummary objects.

### Remarks

In GridControl, group summaries are specified by the GridColumnSummary objects added to the GridControl.GroupSummaries collection. A group summary is calculated for each data group in the grid, and each group summary value is displayed within the corresponding group row. To obtain information on group summaries calculated for the specified group (for example, to get group summary values displayed within the group row), use the **Summaries** property. This property returns the GridControl.GroupSummaries collection read-only copy, whose elements specify group summaries calculated for a particular data group.

### See Also

IGroupInfo Interface
IGroupInfo Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# Value Property

Gets the value for which the group is created (a value which is displayed in a group row).

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
object Value { get; }
```

### Property Value

An object that specifies the data value.

### Remarks

When grid data is grouped by the specified column, data rows that have identical values in this column are arranged into groups. Use the **Value** property to get a column value for which a particular group is created. A grouping value is displayed in a group row, preceded by a column caption and colons.

### See Also

IGroupInfo Interface
IGroupInfo Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# ImageColumn Class

A column type that allows you to display images.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

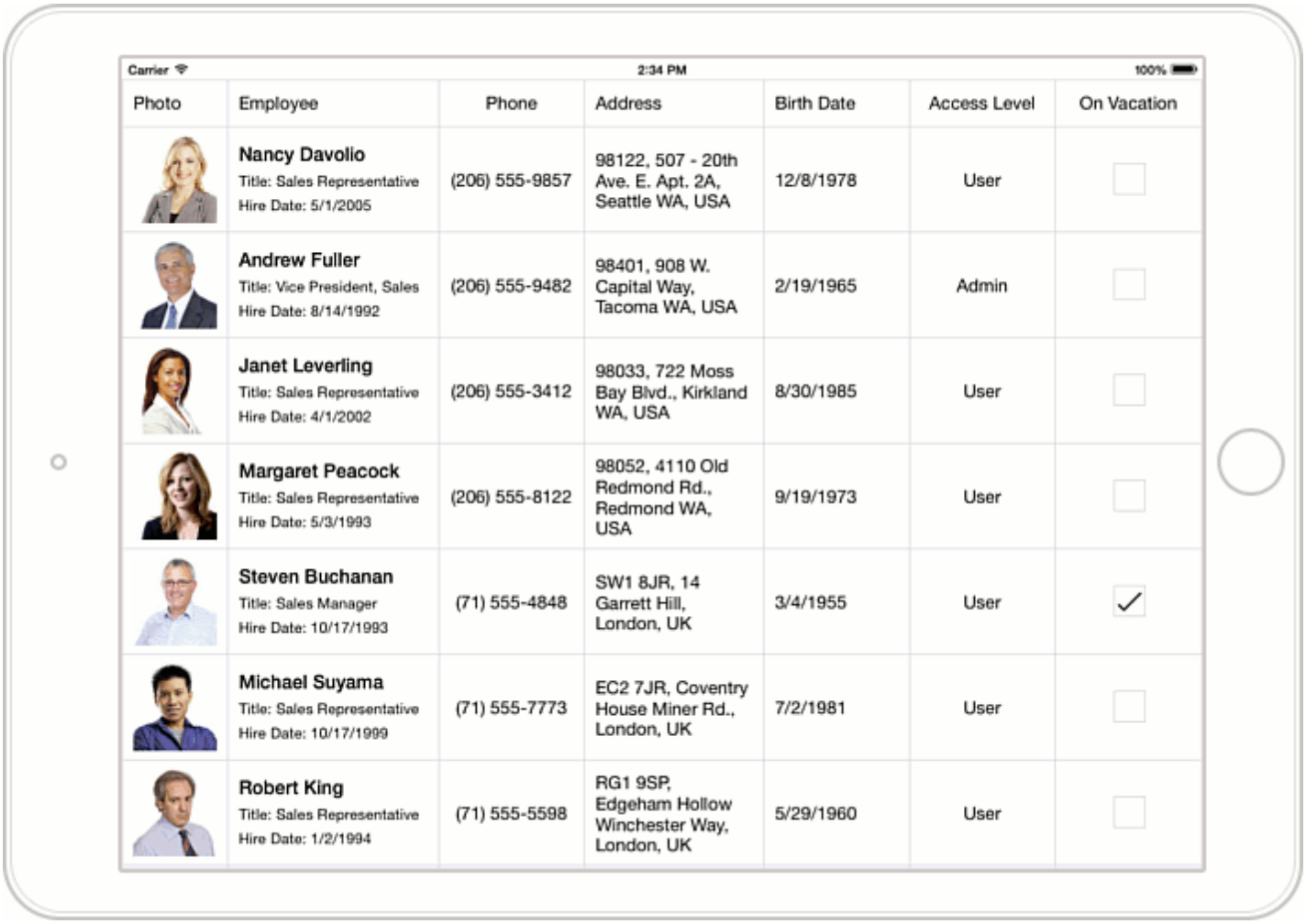
```
public class ImageColumn : GridColumn
```

### Example

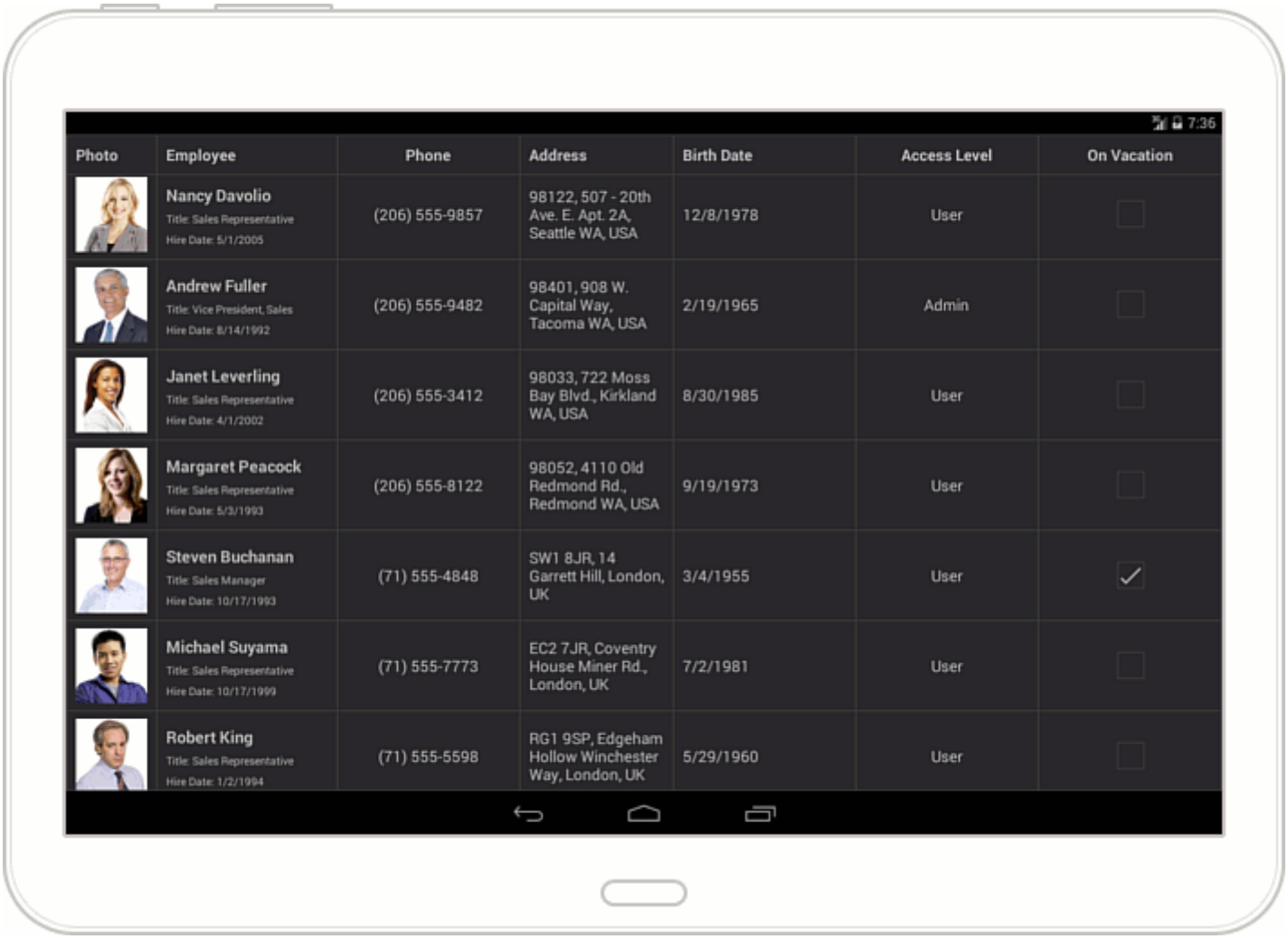
**Show Me**

A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T208848.

This example shows how to add columns to GridControl to display and edit data of different types. The grid is bound to a collection of **Employee** objects. Each **Employee** object contains an employee's photo (image), name, position, phone, address (strings), hire and birth dates (DateTime values), a value that specifies an employee's access level, and a Boolean value indicating whether an employee is on vacation.




© 2018 Developer Express Inc.




Grid columns are stored in the GridControl.Columns collection. An individual column is specified by a GridColumn descendant, which is appropriate to the type of data contained in this column. In this example, to display information on employees and allow end-users to edit it, the following columns are created in the grid.

| Grid Column | Description |
|---|---|
| Photo<br>ImageColumn | This column displays employee photos which are images added to a project as embedded resources. |
| Employee<br>TemplateColumn | A cell in this column displays three employee properties: Name, Position and HireDate. A cell appearance is defined via a template. Each cell contains three Xamarin.Forms.Label elements within a Xamarin.Forms.Grid. Each Label element is bound to a property of the **Employee** class.<br><br>A cell template data context is specified by the CellData object. Its CellData.Value property provides access to a value of a data field assigned to the column's GridColumn.FieldName property. In this example, a column cell displays not only this field value but also values of two more fields. So, it is required to specify the whole data row as a binding source for the template. To do this, set the BindingContext property of the Grid to CellData.Source. Note that BindingContext specified for the Grid is inherited by all its children. This means that all Label elements have the same BindingContext as the Grid, and you can specify |

© 2018 Developer Express Inc.

| | |
|---|---|
| | their simple bindings to properties of that object (Employee).<br><br>Use the GridColumn.AllowSort, GridColumn.AllowGroup and GridColumn.IsReadOnly properties to prevent end-users from sorting and grouping data in the grid by this column and disable data editing. |
| Phone<br>NumberColumn | The employee's **Phone** property is of the string type. The keyboard allowing text input is automatically displayed when an end-user activates this column cell to edit an employee's phone. |
| Address<br>TextColumn | This column is bound to the employee's **Address** string property. The keyboard allowing text input is automatically displayed when an end-user activates this column cell to edit an employee's address. |
| Birth Date<br>DateColumn | This column displays and allows editing employee birth days. The GridColumn.DisplayFormat property specifies the format of displaying dates. |
| Access Level<br>PickerColumn | The employee's **Access** property is of the **AccessLevel** type which is an enumeration with values **Admin** and **User**. To allow editing an employee's access level by selecting one of two enumeration values, this property is bound to the grid's picker column. |
| On Vacation<br>SwitchColumn | The employee's **OnVacation** property is of the Boolean type, so it is bound to the grid's switch column which allows editing a cell value by selecting between two states. |

**C#**

```
 (Employee.cs)
public enum AccessLevel {
    Admin,
    User
}
public class Employee : ModelObject {
    string name;
    public string Name {
        get { return name; }
        set {
            name = value;
            if (Photo == null)
                Photo = ImageSource.FromResource (value.Replace (" ", String.Empty));
        }
    }
    public Employee(string name) {
        this.Name = name;
    }
    public ImageSource Photo { get; set; }
    public DateTime BirthDate { get; set; }
    public DateTime HireDate { get; set; }
    public string Position { get; set; }
    public string Address { get; set; }
    public string Phone { get; set; }
    public AccessLevel Access { get; set; }
    public bool OnVacation { get; set; }
}
```

**C#**

```
 (EmployeesRepository.cs)
public abstract class EmployeesRepository {
    readonly ObservableCollection<Employee> employees;
    public EmployeesRepository() {
        this.employees = new ObservableCollection<Employee>();
    }
    public ObservableCollection<Employee> Employees {
        get { return employees; }
    }
}
public class TestEmployeesRepository : EmployeesRepository {
    public TestEmployeesRepository() : base() {
        GenerateCustomers();
    }
    void GenerateCustomers() {
        Employees.Add(
            new Employee("Nancy Davolio") {
                BirthDate = new DateTime(1978, 12, 8),
                HireDate = new DateTime(2005, 5, 1),
                Position = "Sales Representative",
                Address = "98122, 507 - 20th Ave. E. Apt. 2A, Seattle WA, USA",
                Phone = "(206) 555-9857",
                Access = AccessLevel.User,
                OnVacation = false
            }
        );
        Employees.Add(
            new Employee("Andrew Fuller") {
                BirthDate = new DateTime(1965, 2, 19),
                HireDate = new DateTime(1992, 8, 14),
                Position = "Vice President, Sales",
                Address = "98401, 908 W. Capital Way, Tacoma WA, USA",
                Phone = "(206) 555-9482",
                Access = AccessLevel.Admin,
                OnVacation = false
            }
        );
        Employees.Add(
            new Employee("Janet Leverling") {
                BirthDate = new DateTime(1985, 8, 30),
                HireDate = new DateTime(2002, 4, 1),
                Position = "Sales Representative",
                Address = "98033, 722 Moss Bay Blvd., Kirkland WA, USA",
                Phone = "(206) 555-3412",
                Access = AccessLevel.User,
                OnVacation = false
            }
        );
        Employees.Add(
            new Employee("Margaret Peacock") {
                BirthDate = new DateTime(1973, 9, 19),
                HireDate = new DateTime(1993, 5, 3),
                Position = "Sales Representative",
                Address = "98052, 4110 Old Redmond Rd., Redmond WA, USA",
                Phone = "(206) 555-8122",
                Access = AccessLevel.User,
                OnVacation = false
            }
        );
        Employees.Add(
            new Employee("Steven Buchanan") {
                BirthDate = new DateTime(1955, 3, 4),
                HireDate = new DateTime(1993, 10, 17),
                Position = "Sales Manager",
                Address = "SW1 8JR, 14 Garrett Hill, London, UK",
```

© 2018 Developer Express Inc.

```
                Phone = "(71) 555-4848",
                Access = AccessLevel.User,
                OnVacation = true
            }
        );
        Employees.Add(
            new Employee("Michael Suyama") {
                BirthDate = new DateTime(1981, 7, 2),
                HireDate = new DateTime(1999, 10, 17),
                Position = "Sales Representative",
                Address = "EC2 7JR, Coventry House Miner Rd., London, UK",
                Phone = "(71) 555-7773",
                Access = AccessLevel.User,
                OnVacation = false
            }
        );
        Employees.Add(
            new Employee("Robert King") {
                BirthDate = new DateTime(1960, 5, 29),
                HireDate = new DateTime(1994, 1, 2),
                Position = "Sales Representative",
                Address = "RG1 9SP, Edgeham Hollow Winchester Way, London, UK",
                Phone = "(71) 555-5598",
                Access = AccessLevel.User,
                OnVacation = false
            }
        );
        Employees.Add(
            new Employee("Laura Callahan") {
                BirthDate = new DateTime(1985, 1, 9),
                HireDate = new DateTime(2004, 3, 5),
                Position = "Inside Sales Coordinator",
                Address = "98105, 4726 - 11th Ave. N.E., Seattle WA, USA",
                Phone = "(206) 555-1189",
                Access = AccessLevel.User,
                OnVacation = true
            }
        );
        Employees.Add(
            new Employee("Anne Dodsworth") {
                BirthDate = new DateTime(1980, 1, 27),
                HireDate = new DateTime(2004, 11, 15),
                Position = "Sales Representative",
                Address = "WG2 7LT, 7 Houndstooth Rd., London, UK",
                Phone = "(71) 555-4444",
                Access = AccessLevel.User,
                OnVacation = false
            }
        );
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl ItemsSource="{Binding Employees}" RowHeight="100">
    <dxGrid:GridControl.Columns>
        <dxGrid:ImageColumn FieldName="Photo" Width="100" />
        <dxGrid:TemplateColumn FieldName="Name" Caption="Employee" Width="
                               IsReadOnly="true" AllowGroup="False" AllowS
          <dxGrid:TemplateColumn.DisplayTemplate>
            <DataTemplate>
              <Grid BindingContext="{Binding Source}" Padding="0,15,0,15">
                <Grid.RowDefinitions>
                  <RowDefinition Height="Auto" />
                  <RowDefinition Height="Auto" />
                  <RowDefinition Height="Auto" />
                </Grid.RowDefinitions>
                <Label Text="{Binding Name}" Font="Bold, 18" Grid.Row="0"
                <Label Text="{Binding Position, StringFormat = 'Title: {0}
                <Label Text="{Binding HireDate, StringFormat = 'Hire Date:
              </Grid>
            </DataTemplate>
          </dxGrid:TemplateColumn.DisplayTemplate>
        </dxGrid:TemplateColumn>
        <dxGrid:NumberColumn FieldName="Phone" ContentAlignment="Center"/>
        <dxGrid:TextColumn FieldName="Address" Width="170"/>
        <dxGrid:DateColumn FieldName="BirthDate" DisplayFormat="d"/>
        <dxGrid:PickerColumn FieldName="Access" Caption="Access Level" Con
        <dxGrid:SwitchColumn FieldName="OnVacation"/>
    </dxGrid:GridControl.Columns>
</dxGrid:GridControl>
```

## Inheritance Hierarchy

System.Object
    Xamarin.Forms.BindableObject
        GridColumn
            **ImageColumn**

## See Also

ImageColumn Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# ImageColumn Members

A column type that allows you to display images.

The following tables list the members exposed by the ImageColumn type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | ImageColumn | Initializes a new instance of the ImageColumn class with default settings. |

### Public Properties

| | Name | Description |
|---|---|---|
| | ActualCaption | Gets the column caption as it is displayed in a column header. (Inherited from **GridColumn**) |
| | ActualWidth | Gets the column width. This is a bindable property. (Inherited from **GridColumn**) |
| | AllowAutoFilter | Gets or sets whether an end-user can filter data by the column's values using the auto-filter panel. This is a bindable property. (Inherited from **GridColumn**) |
| | AllowGroup | Gets or sets whether end-users are allowed to group data by the current column. This is a bindable property. (Inherited from **GridColumn**) |
| | AllowSort | Gets or sets whether end-users are allowed to sort data by the column's values. This is a bindable property. (Inherited from **GridColumn**) |
| | AutoFilterCondition | Gets or sets the comparison operator used to create filter conditions for the current column via the auto-filter panel. This is a bindable property. (Inherited from **GridColumn**) |
| | AutoFilterValue | Gets or sets the value in the auto-filter panel for the column. This is a bindable property. (Inherited from **GridColumn**) |
| | BindingContext | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Caption | Gets or sets the column's display caption. This is a bindable property. (Inherited from **GridColumn**) |
| | ColumnFilterMode | Gets or sets how column values are filtered via the auto-filter panel. This is a bindable property. (Inherited from **GridColumn**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | ContentAlignment | Gets or sets the content alignment within column cells. This is a bindable property. (Inherited from **GridColumn**) |
| | DisplayFormat | Gets or sets formatting for values in the column. This is a bindable property. (Inherited from **GridColumn**) |
| | FieldName | Gets or sets the column's unique field name. For bound columns, this property refers to a field in the underlying data source. For unbound columns, this property serves as a column's identifier.<br><br>This is a bindable property.<br><br>(Inherited from **GridColumn**) |
| | FixedStyle | Gets or sets whether a column is fixed. This is a bindable property. (Inherited from **GridColumn**) |
| | GroupInterval | Gets or sets how data rows are grouped when grouping by the current column is applied. This is a bindable property. (Inherited from **GridColumn**) |
| | HeaderTemplate | Gets or sets a template used to define the column headers visual representation. (Inherited from **GridColumn**) |
| | ImmediateUpdateAutoFilter | Gets or sets whether the column's filter condition is updated as soon as an end-user modifies the text within the auto filter panel's cell. This is a bindable property. (Inherited from **GridColumn**) |
| | IsGrouped | Gets or sets whether the grid's data is grouped by the current column. This is a bindable property. (Inherited from **GridColumn**) |
| | IsReadOnly | Gets or sets whether end-users are allowed to change cell values in the column. This is a bindable property. (Inherited from **GridColumn**) |
| | IsUnbound | Gets whether the column is unbound. (Inherited from **GridColumn**) |
| | IsVisible | Gets or sets whether the column is visible in the grid. This is a bindable property. (Inherited from **GridColumn**) |
| | MinWidth | Gets or sets the minimum column width allowed. This is a bindable property. (Inherited from **GridColumn**) |
| | SerializationTypeName | Gets the string containing the column type name. (Inherited from **GridColumn**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | SortIndex | Gets or sets the column's position among sorted columns. This is a bindable property. (Inherited from **GridColumn**) |
| | SortMode | Specifies how column values are sorted and grouped. This is a bindable property. (Inherited from **GridColumn**) |
| | SortOrder | Gets or sets the column's sort order. This is a bindable property. (Inherited from **GridColumn**) |
| | UnboundExpression | Gets or sets an expression used to evaluate values for the current Unbound Columns. This is a bindable property. (Inherited from **GridColumn**) |
| | UnboundType | Gets or sets the data type and binding mode of the column. This is a bindable property. (Inherited from **GridColumn**) |
| | Width | Gets or sets the column width. This is a bindable property. (Inherited from **GridColumn**) |

**Public Methods**

| | Name | Description |
|---|---|---|
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ToString | (Inherited from **System.Object**) |

**Public Fields**

| | Name | Description |
|---|---|---|

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| S | ActualWidthProperty | Identifies the GridColumn.ActualWidth bindable property. (Inherited from **GridColumn**) |
| S | AllowAutoFilterProperty | Identifies the GridColumn.AllowAutoFilter bindable property. (Inherited from **GridColumn**) |
| S | AllowGroupProperty | Identifies the GridColumn.AllowGroup bindable property. (Inherited from **GridColumn**) |
| S | AllowSortProperty | Identifies the GridColumn.AllowSort bindable property. (Inherited from **GridColumn**) |
| S | AutoFilterConditionProperty | Identifies the GridColumn.AutoFilterCondition bindable property. (Inherited from **GridColumn**) |
| S | AutoFilterValueProperty | Identifies the GridColumn.AutoFilterValue bindable property. (Inherited from **GridColumn**) |
| S | BindingContextProperty | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | CaptionProperty | Identifies the GridColumn.Caption bindable property. (Inherited from **GridColumn**) |
| S | ColumnFilterModeProperty | Identifies the GridColumn.ColumnFilterMode bindable property. (Inherited from **GridColumn**) |
| S | ContentAlignmentProperty | Identifies the GridColumn.ContentAlignment bindable property. (Inherited from **GridColumn**) |
| S | DisplayFormatProperty | Identifies the GridColumn.DisplayFormat bindable property. (Inherited from **GridColumn**) |
| S | FieldNameProperty | Identifies the GridColumn.FieldName bindable property. (Inherited from **GridColumn**) |
| S | FixedStyleProperty | Identifies the GridColumn.FixedStyle bindable property. (Inherited from **GridColumn**) |
| S | GroupIntervalProperty | Identifies the GridColumn.GroupInterval bindable property. (Inherited from **GridColumn**) |
| S | HeaderTemplateProperty | Identifies the GridColumn.HeaderTemplate bindable property. (Inherited from **GridColumn**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| S | ImmediateUpdateAutoFilterProperty | Identifies the GridColumn.ImmediateUpdateAutoFilter bindable property. (Inherited from **GridColumn**) |
| S | IsGroupedProperty | Identifies the GridColumn.IsGrouped bindable property. (Inherited from **GridColumn**) |
| S | IsReadOnlyProperty | Identifies the GridColumn.IsReadOnly bindable property. (Inherited from **GridColumn**) |
| S | IsVisibleProperty | Identifies the GridColumn.IsVisible bindable property. (Inherited from **GridColumn**) |
| S | MinWidthProperty | Identifies the GridColumn.MinWidth bindable property. (Inherited from **GridColumn**) |
| S | SortIndexProperty | Identifies the GridColumn.SortIndex bindable property. (Inherited from **GridColumn**) |
| S | SortModeProperty | Identifies the GridColumn.SortMode bindable property. (Inherited from **GridColumn**) |
| S | SortOrderProperty | Identifies the GridColumn.SortOrder bindable property. (Inherited from **GridColumn**) |
| S | UnboundExpressionProperty | Identifies the GridColumn.UnboundExpression bindable property. (Inherited from **GridColumn**) |
| S | UnboundTypeProperty | Identifies the GridColumn.UnboundType bindable property. (Inherited from **GridColumn**) |
| S | WidthProperty | Identifies the GridColumn.Width bindable property. (Inherited from **GridColumn**) |

**Public Events**

| | Name | Description |
|---|---|---|
| ⚡ | BindingContextChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| ⚡ | PropertyChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| ⚡ | PropertyChanging | (Inherited from **Xamarin.Forms.BindableObject**) |

**See Also**
ImageColumn Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# ImageColumn Constructor

Initializes a new instance of the ImageColumn class with default settings.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**
**C#**

```
public ImageColumn()
```

**See Also**
ImageColumn Class
ImageColumn Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# IndicatorFormatConditionBase Class

Serves as a base for classes that represent indicator-type format conditions.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public abstract class IndicatorFormatConditionBase : FormatConditionBase
```

### Inheritance Hierarchy

System.Object
  Xamarin.Forms.BindableObject
    FormatConditionBase
      **IndicatorFormatConditionBase**
        Derived classes

### See Also

IndicatorFormatConditionBase Members
DevExpress.Mobile.DataGrid Namespace
Conditional Formatting

© 2018 Developer Express Inc.

# IndicatorFormatConditionBase Members

Serves as a base for classes that represent indicator-type format conditions.

The following tables list the members exposed by the IndicatorFormatConditionBase type.

### Public Properties

| | Name | Description |
|---|---|---|
| | BindingContext | (Inherited from **Xamarin.Forms.BindableObject**) |
| | FieldName | Gets or sets the name of the column which provides values to test against the current format rule, and to which the format should be applied. This is a bindable property. (Inherited from **FormatConditionBase**) |
| | MaxValue | Specifies the maximum column value to accept for conditional formatting. This is a bindable property. |
| | MinValue | Specifies the minimum column value to accept for conditional formatting. This is a bindable property. |
| | Owner | Gets the format condition's owner. This is a bindable property. (Inherited from **FormatConditionBase**) |
| | PredefinedFormatName | Gets or sets the name of the predefined format to be applied to target cells. This is a bindable property. (Inherited from **FormatConditionBase**) |
| | SelectiveExpression | Specifies a custom rule for applying conditional formatting. This is a bindable property. |
| | TypeName | Gets or sets the data type name. This property supports the internal infrastructure and is not intended to be used directly from your code. (Inherited from **FormatConditionBase**) |

### Public Methods

| | Name | Description |
|---|---|---|
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | ReferenceEquals | (Inherited from **System.Object**) |
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ToString | (Inherited from **System.Object**) |

**Public Fields**

| | Name | Description |
|---|---|---|
| | BindingContextProperty | (Inherited from **Xamarin.Forms.BindableObject**) |
| | FieldNameProperty | Identifies the FormatConditionBase.FieldName bindable property. (Inherited from **FormatConditionBase**) |
| | MaxValueProperty | Identifies the MaxValue bindable property. |
| | MinValueProperty | Identifies the MinValue bindable property. |
| | OwnerProperty | Identifies the FormatConditionBase.Owner bindable property. (Inherited from **FormatConditionBase**) |
| | PredefinedFormatNameProperty | Identifies the FormatConditionBase.PredefinedFormatName bindable property. (Inherited from **FormatConditionBase**) |
| | SelectiveExpressionProperty | Identifies the SelectiveExpression bindable property. |

**Public Events**

| | Name | Description |
|---|---|---|
| | BindingContextChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanging | (Inherited from **Xamarin.Forms.BindableObject**) |

**See Also**
IndicatorFormatConditionBase Members
DevExpress.Mobile.DataGrid Namespace
Conditional Formatting

© 2018 Developer Express Inc.

# IndicatorFormatConditionBase Properties

Serves as a base for classes that represent indicator-type format conditions.

The following tables list the members exposed by the IndicatorFormatConditionBase type.

### Public Properties

| | Name | Description |
|---|---|---|
| | BindingContext | (Inherited from **Xamarin.Forms.BindableObject**) |
| | FieldName | Gets or sets the name of the column which provides values to test against the current format rule, and to which the format should be applied. This is a bindable property. (Inherited from **FormatConditionBase**) |
| | MaxValue | Specifies the maximum column value to accept for conditional formatting. This is a bindable property. |
| | MinValue | Specifies the minimum column value to accept for conditional formatting. This is a bindable property. |
| | Owner | Gets the format condition's owner. This is a bindable property. (Inherited from **FormatConditionBase**) |
| | PredefinedFormatName | Gets or sets the name of the predefined format to be applied to target cells. This is a bindable property. (Inherited from **FormatConditionBase**) |
| | SelectiveExpression | Specifies a custom rule for applying conditional formatting. This is a bindable property. |
| | TypeName | Gets or sets the data type name. This property supports the internal infrastructure and is not intended to be used directly from your code. (Inherited from **FormatConditionBase**) |

### Public Methods

| | Name | Description |
|---|---|---|
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | ReferenceEquals | (Inherited from **System.Object**) |
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ToString | (Inherited from **System.Object**) |

**Public Fields**

| | Name | Description |
|---|---|---|
| | BindingContextProperty | (Inherited from **Xamarin.Forms.BindableObject**) |
| | FieldNameProperty | Identifies the FormatConditionBase.FieldName bindable property. (Inherited from **FormatConditionBase**) |
| | MaxValueProperty | Identifies the MaxValue bindable property. |
| | MinValueProperty | Identifies the MinValue bindable property. |
| | OwnerProperty | Identifies the FormatConditionBase.Owner bindable property. (Inherited from **FormatConditionBase**) |
| | PredefinedFormatNameProperty | Identifies the FormatConditionBase.PredefinedFormatName bindable property. (Inherited from **FormatConditionBase**) |
| | SelectiveExpressionProperty | Identifies the SelectiveExpression bindable property. |

**Public Events**

| | Name | Description |
|---|---|---|
| | BindingContextChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanging | (Inherited from **Xamarin.Forms.BindableObject**) |

**See Also**
IndicatorFormatConditionBase Members
DevExpress.Mobile.DataGrid Namespace
Conditional Formatting

© 2018 Developer Express Inc.

# MaxValue Property

Specifies the maximum column value to accept for conditional formatting. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public object MaxValue { get; set; }
```

### Property Value

An object accepted as the maximum column value.

### Remarks

By default, the **MaxValue** property is set to **null**, and the actual maximum column value is used to format column cells. Use the **MaxValue** property to set a custom maximum column value. This can also improve performance when formatting large data sets.

### See Also

IndicatorFormatConditionBase Class
IndicatorFormatConditionBase Members
DevExpress.Mobile.DataGrid Namespace
MinValue

© 2018 Developer Express Inc.

# MinValue Property

Specifies the minimum column value to accept for conditional formatting. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public object MinValue { get; set; }
```

### Property Value

An object accepted as the minimum column value.

### Remarks

By default, the **MinValue** property is set to **null**, and the actual minimum column value is used to format column cells. Use the **MinValue** property to set a custom minimum column value. This can also improve performance when formatting large data sets.

### See Also

IndicatorFormatConditionBase Class
IndicatorFormatConditionBase Members
DevExpress.Mobile.DataGrid Namespace
MaxValue

© 2018 Developer Express Inc.

# SelectiveExpression Property

Specifies a custom rule for applying conditional formatting. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public string SelectiveExpression { get; set; }
```

### Property Value

A System.String value that is the rule expression.

### Remarks

Use the **SelectiveExpression** property to apply conditional formatting to specific cells based on an expression.

### See Also

IndicatorFormatConditionBase Class
IndicatorFormatConditionBase Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# InitNewRowEventArgs Class

Provides data for the GridControl.InitNewRow event.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public class InitNewRowEventArgs : EventArgs
```

### Remarks

The GridControl.InitNewRow event is raised when an end-user taps New Item Row in the grid. You can handle this event to initialize newly added rows. To initialize row cells, call the IEditableRowData.SetFieldValue of the IEditableRowData object returned by the event's EditableRowData parameter.

### Example

The following code shows how to initialize the "Date" field of a new row via the GridControl.InitNewRow event.

**Xaml**

```
<dxGrid:GridControl ItemsSource="{Binding Orders}"
                    NewItemRowVisibility="true"
                    InitNewRow="OnInitNewRow">
  <!-- ... -->
</dxGrid:GridControl>
```

**C#**

```
using DevExpress.Mobile.DataGrid;
// ...
void OnInitNewRow(object sender, InitNewRowEventArgs e) {
    e.EditableRowData.SetFieldValue ("Date", DateTime.Today);
}
```

### Inheritance Hierarchy

System.Object
  System.EventArgs
    **InitNewRowEventArgs**

### See Also

InitNewRowEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# InitNewRowEventArgs Members

Provides data for the GridControl.InitNewRow event.

The following tables list the members exposed by the InitNewRowEventArgs type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | InitNewRowEventArgs | Initializes a new instance of the InitNewRowEventArgs class. |

### Public Properties

| | Name | Description |
|---|---|---|
| | EditableRowData | Gets an object that contains information on the newly added row and allows you to modify cell values in this row. |

### Public Methods

| | Name | Description |
|---|---|---|
| | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

### Public Fields

| | Name | Description |
|---|---|---|
| | Empty | (Inherited from **System.EventArgs**) |

**See Also**
InitNewRowEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# InitNewRowEventArgs Constructor

Initializes a new instance of the InitNewRowEventArgs class.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public InitNewRowEventArgs(
   IEditableRowData editableRowData
)
```

### Parameters

*editableRowData*
An IEditableRowData object providing access to the added row data. This value is assigned to the EditableRowData property.

### Remarks

Instances of the InitNewRowEventArgs class are automatically created, initialized and passed to the corresponding event handlers.

### See Also

InitNewRowEventArgs Class
InitNewRowEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# InitNewRowEventArgs Properties

Provides data for the GridControl.InitNewRow event.

The following tables list the members exposed by the InitNewRowEventArgs type.

**Public Constructors**

| | Name | Description |
|---|---|---|
| | InitNewRowEventArgs | Initializes a new instance of the InitNewRowEventArgs class. |

**Public Properties**

| | Name | Description |
|---|---|---|
| | EditableRowData | Gets an object that contains information on the newly added row and allows you to modify cell values in this row. |

**Public Methods**

| | Name | Description |
|---|---|---|
| | Equals | (Inherited from **System.Object**) |
| S | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

**Public Fields**

| | Name | Description |
|---|---|---|
| S | Empty | (Inherited from **System.EventArgs**) |

**See Also**
InitNewRowEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# EditableRowData Property

Gets an object that contains information on the newly added row and allows you to modify cell values in this row.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public IEditableRowData EditableRowData { get; }
```

### Property Value

An object implementing the IEditableRowData interface.

### Remarks

Use the IEditableRowData.SetFieldValue method to initialize cells of the newly added row.

### Example

The following code shows how to initialize the "Date" field of a new row via the GridControl.InitNewRow event.

**Xaml**

```
<dxGrid:GridControl ItemsSource="{Binding Orders}"
                    NewItemRowVisibility="true"
                    InitNewRow="OnInitNewRow">
  <!-- ... -->
</dxGrid:GridControl>
```

**C#**

```
using DevExpress.Mobile.DataGrid;
// ...
void OnInitNewRow(object sender, InitNewRowEventArgs e) {
    e.EditableRowData.SetFieldValue ("Date", DateTime.Today);
}
```

### See Also

InitNewRowEventArgs Class
InitNewRowEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# InitNewRowEventHandler Delegate

Represents a method that will handle the GridControl.InitNewRow event.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public delegate void InitNewRowEventHandler(
   object sender,
   InitNewRowEventArgs e
)
```

### Parameters

The declaration of your event handler must have the same parameters as the **InitNewRowEventHandler** delegate declaration.
*sender*
The event sender. Identifies the GridControl object that raised the event.
*e*
A InitNewRowEventArgs object that contains event data.

### See Also

DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# IRowData Interface

Provides access to the data in a grid's row.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public interface IRowData
```

### Remarks

Use the **IRowData** object's DataObject property to get an object representing a data source's record to which the grid's row corresponds. To get a value of the specified field in the row, call the GetFieldValue method.

### Inheritance Hierarchy

### See Also

IRowData Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# IRowData Members

Provides access to the data in a grid's row.

The following tables list the members exposed by the IRowData type.

**Public Properties**

| | Name | Description |
|---|---|---|
| | DataObject | Gets an object representing a record in the grid's underlying data source. |
| | RowHandle | Gets the processed row's handle. |

**Public Methods**

| | Name | Description |
|---|---|---|
| | GetFieldValue | Gets a value of the specified field within the processed row. |
| | GetFieldValueGeneric<T> | Gets a value of the specified field within the processed row. |

**See Also**
IRowData Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# IRowData Properties

Provides access to the data in a grid's row.

The following tables list the members exposed by the IRowData type.

**Public Properties**

| | Name | Description |
|---|---|---|
| | DataObject | Gets an object representing a record in the grid's underlying data source. |
| | RowHandle | Gets the processed row's handle. |

**Public Methods**

| | Name | Description |
|---|---|---|
| | GetFieldValue | Gets a value of the specified field within the processed row. |
| | GetFieldValueGeneric<T> | Gets a value of the specified field within the processed row. |

**See Also**
IRowData Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# DataObject Property

Gets an object representing a record in the grid's underlying data source.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
object DataObject { get; }
```

### Property Value

An object representing a data source record.

### Remarks

The **DataObject** property returns an object that is an individual record in a grid's underlying data source. For example, if a grid's data source is a collection of Order objects, **DataObject** will return an Order object.

### See Also

IRowData Interface
IRowData Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# RowHandle Property

Gets the processed row's handle.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
int RowHandle { get; set; }
```

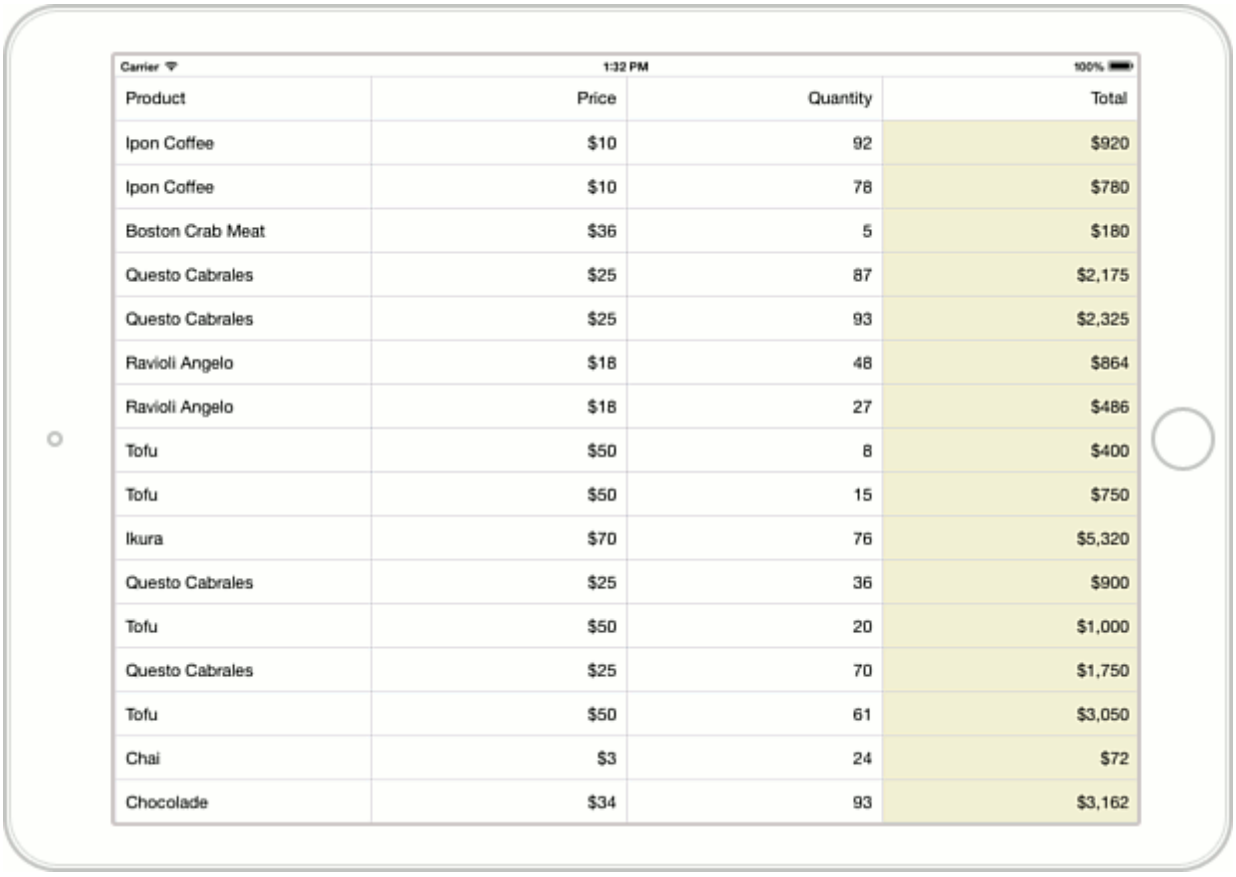
### Property Value

An integer value that specifies the row handle.

### Example

Assume that the GridControl instance is bound to a collection of orders. Each order has the "Product", "UnitPrice"and "Quantity" fields. The example below shows how to add an unbound column to the grid to display the amount of each order according to the expression: UnitPrice*Quantity.

The result is displayed below:




| Product | Price | Quantity | Total |
|---|---|---|---|
| Ipon Coffee | $10 | 92 | $920 |
| Ipon Coffee | $10 | 78 | $780 |
| Boston Crab Meat | $36 | 5 | $180 |
| Questo Cabrales | $25 | 87 | $2,175 |
| Questo Cabrales | $25 | 93 | $2,325 |
| Ravioli Angelo | $18 | 48 | $864 |
| Ravioli Angelo | $18 | 27 | $486 |
| Tofu | $50 | 8 | $400 |
| Tofu | $50 | 15 | $750 |
| Ikura | $70 | 76 | $5,320 |
| Questo Cabrales | $25 | 36 | $900 |
| Tofu | $50 | 20 | $1,000 |
| Questo Cabrales | $25 | 70 | $1,750 |
| Tofu | $50 | 61 | $3,050 |
| Chai | $3 | 24 | $72 |
| Chocolade | $34 | 93 | $3,162 |

**Xaml**

© 2018 Developer Express Inc.

```
<dxGrid:GridControl ItemsSource="{Binding Orders}"
                    CustomUnboundColumnData="OnCustomUnboundColumnData"
                    CustomizeCell="OnCustomizeCell">
    <dxGrid:GridControl.Columns>
         <dxGrid:TextColumn FieldName="Product.Name" Caption="Product"/>
         <dxGrid:NumberColumn FieldName="Product.UnitPrice" Caption="Price" DisplayFormat="C0"/>
         <dxGrid:NumberColumn FieldName="Quantity" />
         <dxGrid:NumberColumn FieldName="Total" IsReadOnly="True"
                              UnboundType="Decimal" DisplayFormat="C0"/>
    </dxGrid:GridControl.Columns>
</dxGrid:GridControl>
```

**C#**

```
// Returns the total for a specific row.
decimal getTotalValue(GridControl grid, int rowHandle) {
    decimal unitPrice = Convert.ToDecimal(grid.GetCellValue(rowHandle, "Product.UnitPrice"));
    decimal quantity = Convert.ToDecimal(grid.GetCellValue(rowHandle, "Quantity"));
    return unitPrice * quantity;
}
// Provides data for the Total column.
void OnCustomUnboundColumnData(object sender, GridColumnDataEventArgs e) {
    if (e.Column.FieldName == "Total" && e.IsGetData)
        e.Value = getTotalValue (e.Source, e.RowData.RowHandle);
}
// Customizes the appearance settings.
void OnCustomizeCell(CustomizeCellEventArgs e) {
if (e.FieldName == "Total") {
        e.BackgroundColor = Color.FromRgb(247, 240, 213);
        e.Handled = true;
    }
}
```

**See Also**

IRowData Interface
IRowData Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# IRowData Methods

Provides access to the data in a grid's row.

The following tables list the members exposed by the IRowData type.

**Public Methods**

| | Name | Description |
|---|---|---|
| | GetFieldValue | Gets a value of the specified field within the processed row. |
| | GetFieldValueGeneric<T> | Gets a value of the specified field within the processed row. |

**See Also**
IRowData Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# GetFieldValue Method

Gets a value of the specified field within the processed row.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
object GetFieldValue(
   string fieldName
)
```

### Parameters

*fieldName*
A System.String value that specifies the name of the field whose value must be returned.

### Return Value

An object representing a field value.

### See Also

IRowData Interface
IRowData Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# GetFieldValueGeneric<T> Method

Gets a value of the specified field within the processed row.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
T GetFieldValueGeneric<T>(
   string fieldName
)
```

### Parameters

*fieldName*
A System.String value that specifies the name of the field whose value must be returned.

### Return Value

An object representing a field value.

### See Also

IRowData Interface
IRowData Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# NumberColumn Class

A grid column to display and edit numeric values.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

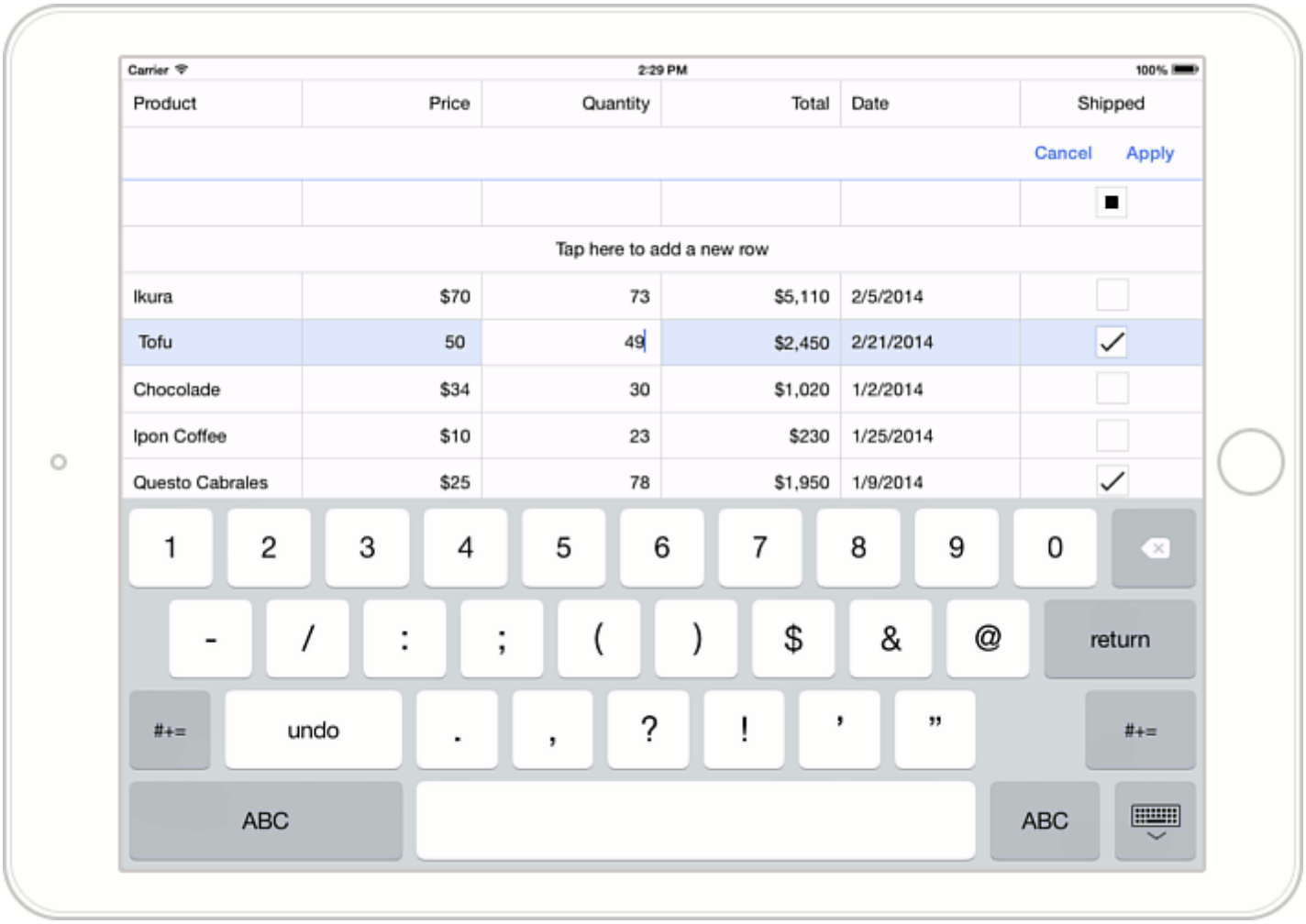
**C#**

```
public class NumberColumn : TextColumn
```

### Remarks

In GridControl, data fields are represented as columns and records are displayed as rows. So, to display data, a grid should contain columns bound to fields of the underlying data source.

An individual column is specified by the **NumberColumn**, TextColumn, DateColumn or SwitchColumn object (a GridColumn descendant) depending on the type of data displayed in this column. Use columns of the **NumberColumn** type to display numbers and allow end-user to edit them.




Column objects provide a set of properties to adjust column settings (for example, GridColumn.DisplayFormat, GridColumn.Width, GridColumn.Caption, GridColumn.IsVisible, etc.).

GridControl stores its columns in the GridControl.Columns collection, and displays them in the order in which they are added to this collection.

### Example

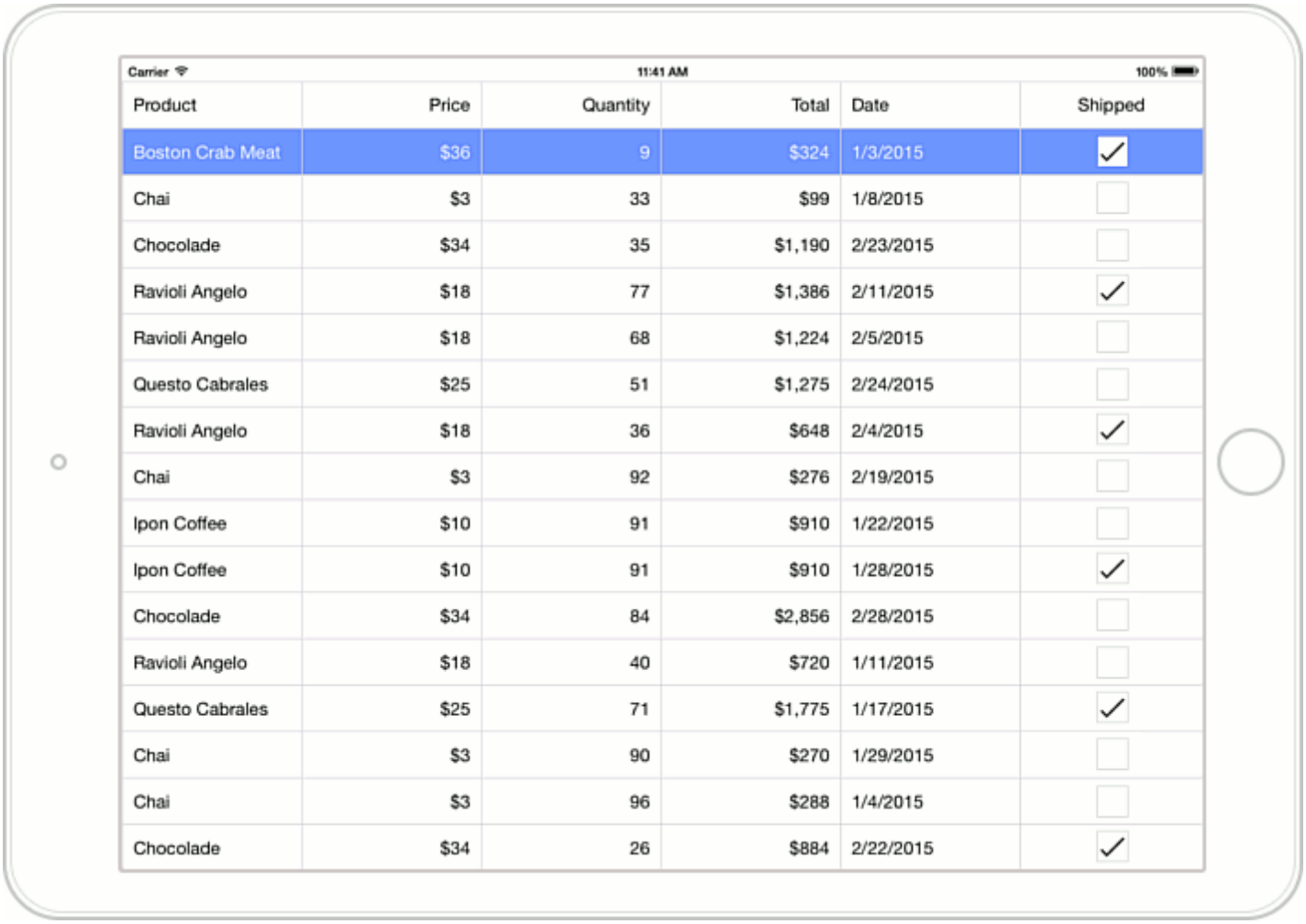
**Show Me**

© 2018 Developer Express Inc.

| | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T178764. | |

This example shows how to create and customize grid columns for displaying and editing data of different types (text, numbers, dates and Boolean values). The specified collection contains columns bound to the data source fields (*Product.Name*, *Product.UnitPrice*, *Quantity*, *Date* and *Shipped*) and one unbound column (*Total*) displaying data values calculated according to a formula based on the values of other columns.

The image below illustrates the result.




| Product | Price | Quantity | Total | Date | Shipped |
|---|---|---|---|---|---|
| Boston Crab Meat | $36 | 9 | $324 | 1/3/2015 | ✓ |
| Chai | $3 | 33 | $99 | 1/8/2015 | |
| Chocolade | $34 | 35 | $1,190 | 2/23/2015 | |
| Ravioli Angelo | $18 | 77 | $1,386 | 2/11/2015 | ✓ |
| Ravioli Angelo | $18 | 68 | $1,224 | 2/5/2015 | |
| Questo Cabrales | $25 | 51 | $1,275 | 2/24/2015 | |
| Ravioli Angelo | $18 | 36 | $648 | 2/4/2015 | ✓ |
| Chai | $3 | 92 | $276 | 2/19/2015 | |
| Ipon Coffee | $10 | 91 | $910 | 1/22/2015 | |
| Ipon Coffee | $10 | 91 | $910 | 1/28/2015 | ✓ |
| Chocolade | $34 | 84 | $2,856 | 2/28/2015 | |
| Ravioli Angelo | $18 | 40 | $720 | 1/11/2015 | |
| Questo Cabrales | $25 | 71 | $1,775 | 1/17/2015 | ✓ |
| Chai | $3 | 90 | $270 | 1/29/2015 | |
| Chai | $3 | 96 | $288 | 1/4/2015 | |
| Chocolade | $34 | 26 | $884 | 2/22/2015 | ✓ |

© 2018 Developer Express Inc.

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl.Columns>
         <dxGrid:TextColumn FieldName="Product.Name" Caption = "Product" Width = "170"
                            SortOrder = "Descending" SortIndex = "0"/>
         <dxGrid:NumberColumn FieldName="Product.UnitPrice" Caption = "Price" DisplayFormat="C0"/>
         <dxGrid:NumberColumn FieldName="Quantity"
                              SortOrder = "Ascending" SortIndex = "1"/>
         <dxGrid:NumberColumn FieldName="Total"
                              UnboundType="Integer" UnboundExpression="[Quantity] * [Product.UnitPrice]"
                              DisplayFormat="C0" IsReadOnly="True"/>
         <dxGrid:DateColumn FieldName="Date" DisplayFormat="d"
                            IsGrouped = "true" GroupInterval = "Date"/>
         <dxGrid:SwitchColumn FieldName="Shipped" AllowSort = "False" AllowAutoFilter="false"/>
     </dxGrid:GridControl.Columns>
```

### Inheritance Hierarchy

System.Object
  Xamarin.Forms.BindableObject
    GridColumn
      TextColumn
        **NumberColumn**

### See Also

NumberColumn Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# NumberColumn Members

A grid column to display and edit numeric values.

The following tables list the members exposed by the NumberColumn type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | NumberColumn | Initializes a new instance of the NumberColumn class with default settings. |

### Public Properties

| | Name | Description |
|---|---|---|
| | ActualCaption | Gets the column caption as it is displayed in a column header. (Inherited from **GridColumn**) |
| | ActualWidth | Gets the column width. This is a bindable property. (Inherited from **GridColumn**) |
| | AllowAutoFilter | Gets or sets whether an end-user can filter data by the column's values using the auto-filter panel. This is a bindable property. (Inherited from **GridColumn**) |
| | AllowGroup | Gets or sets whether end-users are allowed to group data by the current column. This is a bindable property. (Inherited from **GridColumn**) |
| | AllowSort | Gets or sets whether end-users are allowed to sort data by the column's values. This is a bindable property. (Inherited from **GridColumn**) |
| | AutoFilterCondition | Gets or sets the comparison operator used to create filter conditions for the current column via the auto-filter panel. This is a bindable property. (Inherited from **GridColumn**) |
| | AutoFilterValue | Gets or sets the value in the auto-filter panel for the column. This is a bindable property. (Inherited from **GridColumn**) |
| | BindingContext | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Caption | Gets or sets the column's display caption. This is a bindable property. (Inherited from **GridColumn**) |
| | ColumnFilterMode | Gets or sets how column values are filtered via the auto-filter panel. This is a bindable property. (Inherited from **GridColumn**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | ContentAlignment | Gets or sets the content alignment within column cells. This is a bindable property. (Inherited from **GridColumn**) |
| | DisplayFormat | Gets or sets formatting for values in the column. This is a bindable property. (Inherited from **GridColumn**) |
| | FieldName | Gets or sets the column's unique field name. For bound columns, this property refers to a field in the underlying data source. For unbound columns, this property serves as a column's identifier.<br><br>This is a bindable property.<br><br>(Inherited from **GridColumn**) |
| | FixedStyle | Gets or sets whether a column is fixed. This is a bindable property. (Inherited from **GridColumn**) |
| | GroupInterval | Gets or sets how data rows are grouped when grouping by the current column is applied. This is a bindable property. (Inherited from **GridColumn**) |
| | HeaderTemplate | Gets or sets a template used to define the column headers visual representation. (Inherited from **GridColumn**) |
| | ImmediateUpdateAutoFilter | Gets or sets whether the column's filter condition is updated as soon as an end-user modifies the text within the auto filter panel's cell. This is a bindable property. (Inherited from **GridColumn**) |
| | IsGrouped | Gets or sets whether the grid's data is grouped by the current column. This is a bindable property. (Inherited from **GridColumn**) |
| | IsReadOnly | Gets or sets whether end-users are allowed to change cell values in the column. This is a bindable property. (Inherited from **GridColumn**) |
| | IsUnbound | Gets whether the column is unbound. (Inherited from **GridColumn**) |
| | IsVisible | Gets or sets whether the column is visible in the grid. This is a bindable property. (Inherited from **GridColumn**) |
| | MinWidth | Gets or sets the minimum column width allowed. This is a bindable property. (Inherited from **GridColumn**) |
| | SerializationTypeName | Gets the string containing the column type name. (Inherited from **GridColumn**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | SortIndex | Gets or sets the column's position among sorted columns. This is a bindable property. (Inherited from **GridColumn**) |
| | SortMode | Specifies how column values are sorted and grouped. This is a bindable property. (Inherited from **GridColumn**) |
| | SortOrder | Gets or sets the column's sort order. This is a bindable property. (Inherited from **GridColumn**) |
| | UnboundExpression | Gets or sets an expression used to evaluate values for the current Unbound Columns. This is a bindable property. (Inherited from **GridColumn**) |
| | UnboundType | Gets or sets the data type and binding mode of the column. This is a bindable property. (Inherited from **GridColumn**) |
| | Width | Gets or sets the column width. This is a bindable property. (Inherited from **GridColumn**) |

**Public Methods**

| | Name | Description |
|---|---|---|
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ToString | (Inherited from **System.Object**) |

**Public Fields**

| | Name | Description |
|---|---|---|

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| S | ActualWidthProperty | Identifies the GridColumn.ActualWidth bindable property. (Inherited from **GridColumn**) |
| S | AllowAutoFilterProperty | Identifies the GridColumn.AllowAutoFilter bindable property. (Inherited from **GridColumn**) |
| S | AllowGroupProperty | Identifies the GridColumn.AllowGroup bindable property. (Inherited from **GridColumn**) |
| S | AllowSortProperty | Identifies the GridColumn.AllowSort bindable property. (Inherited from **GridColumn**) |
| S | AutoFilterConditionProperty | Identifies the GridColumn.AutoFilterCondition bindable property. (Inherited from **GridColumn**) |
| S | AutoFilterValueProperty | Identifies the GridColumn.AutoFilterValue bindable property. (Inherited from **GridColumn**) |
| S | BindingContextProperty | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | CaptionProperty | Identifies the GridColumn.Caption bindable property. (Inherited from **GridColumn**) |
| S | ColumnFilterModeProperty | Identifies the GridColumn.ColumnFilterMode bindable property. (Inherited from **GridColumn**) |
| S | ContentAlignmentProperty | Identifies the GridColumn.ContentAlignment bindable property. (Inherited from **GridColumn**) |
| S | DisplayFormatProperty | Identifies the GridColumn.DisplayFormat bindable property. (Inherited from **GridColumn**) |
| S | FieldNameProperty | Identifies the GridColumn.FieldName bindable property. (Inherited from **GridColumn**) |
| S | FixedStyleProperty | Identifies the GridColumn.FixedStyle bindable property. (Inherited from **GridColumn**) |
| S | GroupIntervalProperty | Identifies the GridColumn.GroupInterval bindable property. (Inherited from **GridColumn**) |
| S | HeaderTemplateProperty | Identifies the GridColumn.HeaderTemplate bindable property. (Inherited from **GridColumn**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| S | ImmediateUpdateAutoFilterProperty | Identifies the GridColumn.ImmediateUpdateAutoFilter bindable property. (Inherited from **GridColumn**) |
| S | IsGroupedProperty | Identifies the GridColumn.IsGrouped bindable property. (Inherited from **GridColumn**) |
| S | IsReadOnlyProperty | Identifies the GridColumn.IsReadOnly bindable property. (Inherited from **GridColumn**) |
| S | IsVisibleProperty | Identifies the GridColumn.IsVisible bindable property. (Inherited from **GridColumn**) |
| S | MinWidthProperty | Identifies the GridColumn.MinWidth bindable property. (Inherited from **GridColumn**) |
| S | SortIndexProperty | Identifies the GridColumn.SortIndex bindable property. (Inherited from **GridColumn**) |
| S | SortModeProperty | Identifies the GridColumn.SortMode bindable property. (Inherited from **GridColumn**) |
| S | SortOrderProperty | Identifies the GridColumn.SortOrder bindable property. (Inherited from **GridColumn**) |
| S | UnboundExpressionProperty | Identifies the GridColumn.UnboundExpression bindable property. (Inherited from **GridColumn**) |
| S | UnboundTypeProperty | Identifies the GridColumn.UnboundType bindable property. (Inherited from **GridColumn**) |
| S | WidthProperty | Identifies the GridColumn.Width bindable property. (Inherited from **GridColumn**) |

**Public Events**

| | Name | Description |
|---|---|---|
| ⚡ | BindingContextChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| ⚡ | PropertyChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| ⚡ | PropertyChanging | (Inherited from **Xamarin.Forms.BindableObject**) |

**See Also**
NumberColumn Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# NumberColumn Constructor

Initializes a new instance of the NumberColumn class with default settings.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**
**C#**

```
public NumberColumn()
```

**See Also**
NumberColumn Class
NumberColumn Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# PickerColumn Class

A grid column allowing an end-user to edit a cell value by selecting an item from the predefined set.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public class PickerColumn : TextColumn
```

### Example

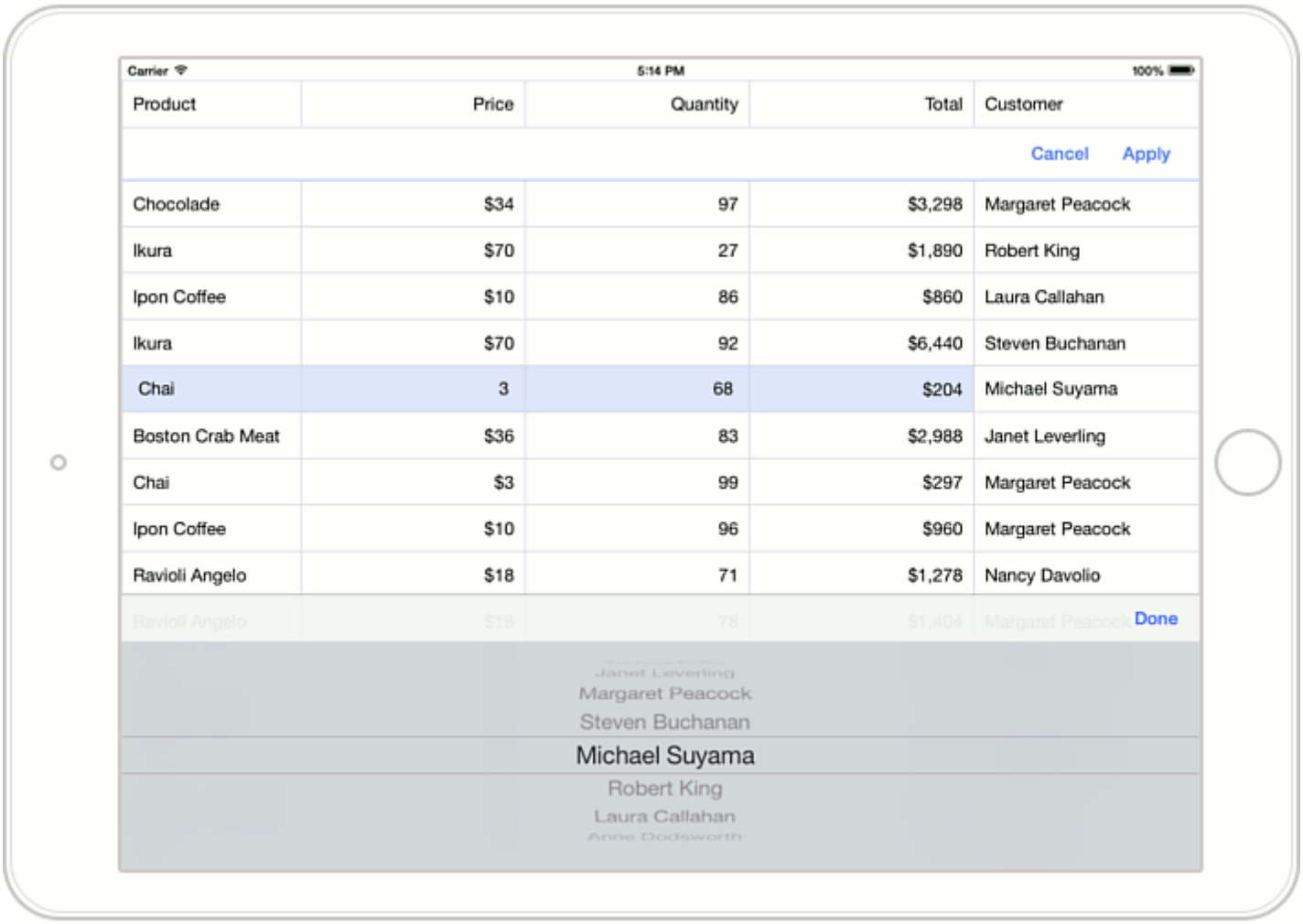
| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T257726. | |

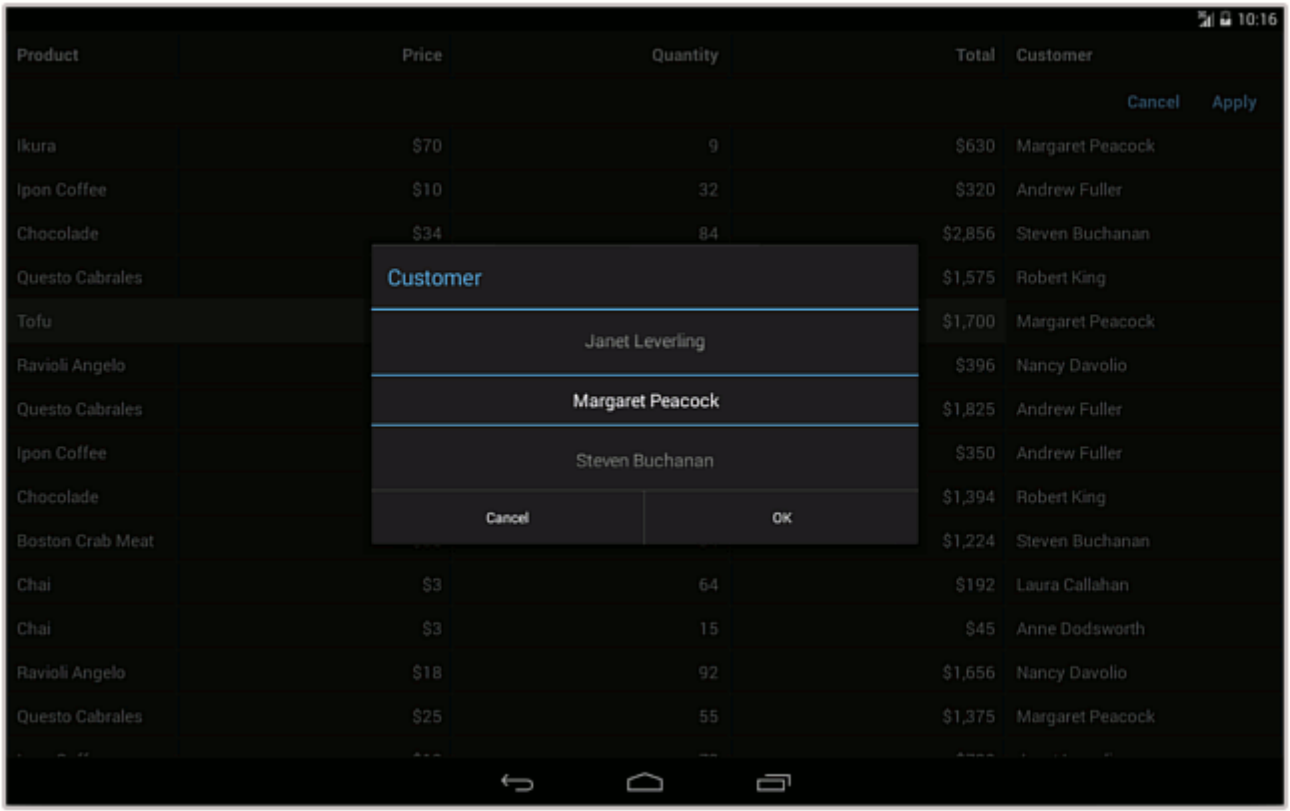
This example demonstrates how to implement the grid's PickerColumn whose items are coming from the grid's underlying data source.

In this example, a grid is bound to a collection of the **Order** objects. Each order contains information about a customer (the **CustomerId** property). A customer is represented by a **Customer** object that has the **Id** and **Name** properties. **Customer** objects are stored in an IList collection. The following properties of the PickerColumn object are set to show customer names for orders in this column within the grid (see the *MainPage.xaml* file):

- GridColumn.FieldName — an order's property to be bound to a column (**CustomerId**).
- ItemsSource — a collection of customers.
- ValueMember — a customer's property used to associate a customer with an order (**Id**).
- DisplayMember — a customer's property to be displayed in a column (**Name**).

© 2018 Developer Express Inc.


Carrier
5:14 PM
100%
Product
Price
Quantity
Total
Customer
Cancel
Apply
Chocolade $34 97 $3,298 Margaret Peacock
Ikura $70 27 $1,890 Robert King
Ipon Coffee $10 86 $860 Laura Callahan
Ikura $70 92 $6,440 Steven Buchanan
Chai 3 68 $204 Michael Suyama
Boston Crab Meat $36 83 $2,988 Janet Leverling
Chai $3 99 $297 Margaret Peacock
Ipon Coffee $10 96 $960 Margaret Peacock
Ravioli Angelo $18 71 $1,278 Nancy Davolio
Done
Janet Leverling
Margaret Peacock
Steven Buchanan
Michael Suyama
Robert King
Laura Callahan


10:16
Product
Price
Quantity
Total
Customer
Cancel
Apply
Ikura $70 9 $630 Margaret Peacock
Ipon Coffee $10 32 $320 Andrew Fuller
Chocolade $34 84 $2,856 Steven Buchanan
Questo Cabrales $1,575 Robert King
Tofu $1,700 Margaret Peacock
Ravioli Angelo $396 Nancy Davolio
Questo Cabrales $1,825 Andrew Fuller
Ipon Coffee $350 Andrew Fuller
Chocolade $1,394 Robert King
Boston Crab Meat $1,224 Steven Buchanan
Chai $3 64 $192 Laura Callahan
Chai $3 15 $45 Anne Dodsworth
Ravioli Angelo $18 92 $1,656 Nancy Davolio
Questo Cabrales $25 55 $1,375 Margaret Peacock
Customer
Janet Leverling
Margaret Peacock
Steven Buchanan
Cancel
OK

C#

```
 (Customer.cs)
using System;
namespace PickerColumnExample
{
    public class Customer : ModelObject {
        string name;
        public Customer(string name, int id) {
            this.name = name;
            this.Id = id;
        }
        public int Id { get; set; }
        public string Name {
            get { return name; }
            set {name = value;}
        }
    }
}
```

C#

```
 (MainPage.xaml.cs)
using System;
using System.Collections.Generic;
using Xamarin.Forms;
namespace PickerColumnExample {
    public partial class MainPage : ContentPage {
        public MainPage () {
            InitializeComponent ();
            TestOrdersRepository model = new TestOrdersRepository ();
            BindingContext = model;
        }
    }
}
```

C#

```
 (ModelObject.cs)
using System;
using System.ComponentModel;
namespace PickerColumnExample
{
    public class ModelObject : INotifyPropertyChanged {
        public event PropertyChangedEventHandler PropertyChanged;
        protected void RaisePropertyChanged(string name) {
            if (PropertyChanged != null)
                PropertyChanged(this, new PropertyChangedEventArgs(name));
        }
    }
}
```

C#

© 2018 Developer Express Inc.

```
 (Order.cs)
using System;
namespace PickerColumnExample
{
    public class Order : ModelObject {
        Product product;
        int quantity;
        int customerId;
        public Order() {
            this.product = new Product ("", 0);
            this.quantity = 0;
        }
        public Order(Product product, int amount, int customerId) {
            this.product = product;
            this.quantity = amount;
            this.customerId = customerId;
        }
        public Product Product {
            get { return product; }
            set { if (product != value) {
                    product = value;
                    RaisePropertyChanged ("Product");}}
        }
        public int Quantity {
            get { return quantity; }
            set { if (quantity != value) {
                    quantity = value;
                    RaisePropertyChanged ("Quantity");}}
        }
        public int CustomerId {
            get { return customerId; }
            set {
                if (customerId != value) {
                    customerId = value;
                    RaisePropertyChanged("CustomerId");
                }
            }
        }
    }
}
```

**C#**

```
 (OrdersRepository.cs)
using System;
using System.Collections.Generic;
using System.Collections.ObjectModel;
namespace PickerColumnExample
{
    public abstract class OrdersRepository {
        readonly ObservableCollection<Order> orders;
        readonly ObservableCollection<Customer> customers;
        public OrdersRepository() {
            this.orders = new ObservableCollection<Order>();
            this.customers = new ObservableCollection<Customer>();
        }
        public ObservableCollection<Order> Orders {
            get { return orders; }
        }
        public IList<Customer> Customers{
            get { return customers; }
        }
    }
    public class TestOrdersRepository : OrdersRepository {
        const int orderCount = 100;
        readonly List<Product> products;
        readonly Random random;
        public TestOrdersRepository() : base() {
            this.random = new Random((int)DateTime.Now.Ticks);
            this.products = new List<Product>();
            GenerateProducts();
            GenerateCustomers();
            for (int i = 0; i < orderCount; i++)
                Orders.Add(GenerateOrder(i));
        }
        Order GenerateOrder(int number) {
            Order order = new Order(RandomItem<Product>(products), random.
            return order;
        }
        T RandomItem<T>(IList<T> list) {
            int index = (int)(random.NextDouble() * 0.99 * (list.Count));
            return list[index];
        }
        void GenerateProducts() {
            products.Add(new Product("Tofu", 50));
            products.Add(new Product("Chocolade", 34));
            products.Add(new Product("Ikura", 70));
            products.Add(new Product("Chai", 3));
            products.Add(new Product("Boston Crab Meat", 36));
            products.Add(new Product("Ravioli Angelo", 18));
            products.Add(new Product("Ipon Coffee", 10));
            products.Add(new Product("Questo Cabrales", 25));
        }
        void GenerateCustomers() {
```

```
            Customers.Add(new Customer("Nancy Davolio", 0));
            Customers.Add(new Customer("Andrew Fuller", 1));
            Customers.Add(new Customer("Janet Leverling", 2));
            Customers.Add(new Customer("Margaret Peacock", 3));
            Customers.Add(new Customer("Steven Buchanan", 4));
            Customers.Add(new Customer("Michael Suyama", 5));
            Customers.Add(new Customer("Robert King", 6));
            Customers.Add(new Customer("Laura Callahan", 7));
            Customers.Add(new Customer("Anne Dodsworth", 8));
        }
    }
}
```

**C#**

```
 (Product.cs)
using System;
namespace PickerColumnExample
{
    public class Product : ModelObject {
        string name;
        int unitPrice;
        public Product(string name, int unitPrice) {
            this.name = name;
            this.unitPrice = unitPrice;
        }
        public string Name {
            get { return name; }
            set { name = value; }
        }
        public int UnitPrice {
            get { return unitPrice; }
            set { unitPrice = value; }
        }
    }
}
```

**Xaml**

© 2018 Developer Express Inc.

```
 (MainPage.xaml)
<?xml version="1.0" encoding="UTF-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="PickerColumnExample.MainPage"
             xmlns:dxg="clr-namespace:DevExpress.Mobile.DataGrid;assembly=
    <ContentPage.Padding>
        <OnPlatform x:TypeArguments="Thickness" iOS="0, 20, 0, 0" />
    </ContentPage.Padding>
    <dxg:GridControl x:Name="grid" ItemsSource="{Binding Orders}">
         <dxg:GridControl.Columns>
             <dxg:TextColumn FieldName="Product.Name" Caption="Product" Wi
             <dxg:NumberColumn FieldName="Product.UnitPrice" Caption="Pric
             <dxg:NumberColumn FieldName="Quantity"/>
             <dxg:NumberColumn FieldName="Total"
                               UnboundType="Integer" UnboundExpression=
                               DisplayFormat="C0" IsReadOnly="True"/>
             <dxg:PickerColumn FieldName="CustomerId" Caption="Customer"
                               ItemsSource="{Binding Customers}" ValueMemb
         </dxg:GridControl.Columns>
    </dxg:GridControl>
</ContentPage>
```

## Inheritance Hierarchy

System.Object
  Xamarin.Forms.BindableObject
    GridColumn
      TextColumn
        **PickerColumn**

## See Also

PickerColumn Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# PickerColumn Members

A grid column allowing an end-user to edit a cell value by selecting an item from the predefined set.

The following tables list the members exposed by the PickerColumn type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | PickerColumn | Initializes a new instance of the PickerColumn class with default settings. |

### Public Properties

| | Name | Description |
|---|---|---|
| | ActualCaption | Gets the column caption as it is displayed in a column header. (Inherited from **GridColumn**) |
| | ActualWidth | Gets the column width. This is a bindable property. (Inherited from **GridColumn**) |
| | AllowAutoFilter | Gets or sets whether an end-user can filter data by the column's values using the auto-filter panel. This is a bindable property. (Inherited from **GridColumn**) |
| | AllowGroup | Gets or sets whether end-users are allowed to group data by the current column. This is a bindable property. (Inherited from **GridColumn**) |
| | AllowSort | Gets or sets whether end-users are allowed to sort data by the column's values. This is a bindable property. (Inherited from **GridColumn**) |
| | AutoFilterCondition | Gets or sets the comparison operator used to create filter conditions for the current column via the auto-filter panel. This is a bindable property. (Inherited from **GridColumn**) |
| | AutoFilterValue | Gets or sets the value in the auto-filter panel for the column. This is a bindable property. (Inherited from **GridColumn**) |
| | BindingContext | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Caption | Gets or sets the column's display caption. This is a bindable property. (Inherited from **GridColumn**) |
| | ColumnFilterMode | Gets or sets how column values are filtered via the auto-filter panel. This is a bindable property. (Inherited from **GridColumn**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | ContentAlignment | Gets or sets the content alignment within column cells. This is a bindable property. (Inherited from **GridColumn**) |
| | DisplayFormat | Gets or sets formatting for values in the column. This is a bindable property. (Inherited from **GridColumn**) |
| | DisplayMember | Gets or sets a field name in the bound data source whose contents are to be displayed by the column's picker control. This is a bindable property. |
| | FieldName | Gets or sets the column's unique field name. For bound columns, this property refers to a field in the underlying data source. For unbound columns, this property serves as a column's identifier.<br>This is a bindable property.<br>(Inherited from **GridColumn**) |
| | FixedStyle | Gets or sets whether a column is fixed. This is a bindable property. (Inherited from **GridColumn**) |
| | GroupInterval | Gets or sets how data rows are grouped when grouping by the current column is applied. This is a bindable property. (Inherited from **GridColumn**) |
| | HeaderTemplate | Gets or sets a template used to define the column headers visual representation. (Inherited from **GridColumn**) |
| | ImmediateUpdateAutoFilter | Gets or sets whether the column's filter condition is updated as soon as an end-user modifies the text within the auto filter panel's cell. This is a bindable property. (Inherited from **GridColumn**) |
| | IsGrouped | Gets or sets whether the grid's data is grouped by the current column. This is a bindable property. (Inherited from **GridColumn**) |
| | IsReadOnly | Gets or sets whether end-users are allowed to change cell values in the column. This is a bindable property. (Inherited from **GridColumn**) |
| | IsUnbound | Gets whether the column is unbound. (Inherited from **GridColumn**) |
| | IsVisible | Gets or sets whether the column is visible in the grid. This is a bindable property. (Inherited from **GridColumn**) |
| | ItemsSource | Gets or sets a data source for a picker control that appears when a column |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | | cell is in the edit mode. This is a bindable property. |
| | MinWidth | Gets or sets the minimum column width allowed. This is a bindable property. (Inherited from **GridColumn**) |
| | SerializationTypeName | Gets the string containing the column type name. (Inherited from **GridColumn**) |
| | SortIndex | Gets or sets the column's position among sorted columns. This is a bindable property. (Inherited from **GridColumn**) |
| | SortMode | Specifies how column values are sorted and grouped. This is a bindable property. (Inherited from **GridColumn**) |
| | SortOrder | Gets or sets the column's sort order. This is a bindable property. (Inherited from **GridColumn**) |
| | UnboundExpression | Gets or sets an expression used to evaluate values for the current Unbound Columns. This is a bindable property. (Inherited from **GridColumn**) |
| | UnboundType | Gets or sets the data type and binding mode of the column. This is a bindable property. (Inherited from **GridColumn**) |
| | ValueMember | Gets or sets the field name in the bound data source whose contents are assigned item values of the column's picker control. This is a bindable property. |
| | Width | Gets or sets the column width. This is a bindable property. (Inherited from **GridColumn**) |

**Public Methods**

| | Name | Description |
|---|---|---|
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ReferenceEquals | (Inherited from **System.Object**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ToString | (Inherited from **System.Object**) |

**Public Fields**

| | Name | Description |
|---|---|---|
| S | ActualWidthProperty | Identifies the GridColumn.ActualWidth bindable property. (Inherited from **GridColumn**) |
| S | AllowAutoFilterProperty | Identifies the GridColumn.AllowAutoFilter bindable property. (Inherited from **GridColumn**) |
| S | AllowGroupProperty | Identifies the GridColumn.AllowGroup bindable property. (Inherited from **GridColumn**) |
| S | AllowSortProperty | Identifies the GridColumn.AllowSort bindable property. (Inherited from **GridColumn**) |
| S | AutoFilterConditionProperty | Identifies the GridColumn.AutoFilterCondition bindable property. (Inherited from **GridColumn**) |
| S | AutoFilterValueProperty | Identifies the GridColumn.AutoFilterValue bindable property. (Inherited from **GridColumn**) |
| S | BindingContextProperty | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | CaptionProperty | Identifies the GridColumn.Caption bindable property. (Inherited from **GridColumn**) |
| S | ColumnFilterModeProperty | Identifies the GridColumn.ColumnFilterMode bindable property. (Inherited from **GridColumn**) |
| S | ContentAlignmentProperty | Identifies the GridColumn.ContentAlignment bindable property. (Inherited from **GridColumn**) |
| S | DisplayFormatProperty | Identifies the GridColumn.DisplayFormat bindable property. (Inherited from **GridColumn**) |
| S | DisplayMemberProperty | Identifies the DisplayMember bindable property. |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| S | FieldNameProperty | Identifies the GridColumn.FieldName bindable property. (Inherited from **GridColumn**) |
| S | FixedStyleProperty | Identifies the GridColumn.FixedStyle bindable property. (Inherited from **GridColumn**) |
| S | GroupIntervalProperty | Identifies the GridColumn.GroupInterval bindable property. (Inherited from **GridColumn**) |
| S | HeaderTemplateProperty | Identifies the GridColumn.HeaderTemplate bindable property. (Inherited from **GridColumn**) |
| S | ImmediateUpdateAutoFilterProperty | Identifies the GridColumn.ImmediateUpdateAutoFilter bindable property. (Inherited from **GridColumn**) |
| S | IsGroupedProperty | Identifies the GridColumn.IsGrouped bindable property. (Inherited from **GridColumn**) |
| S | IsReadOnlyProperty | Identifies the GridColumn.IsReadOnly bindable property. (Inherited from **GridColumn**) |
| S | IsVisibleProperty | Identifies the GridColumn.IsVisible bindable property. (Inherited from **GridColumn**) |
| S | ItemsSourceProperty | Identifies the ItemsSource bindable property. |
| S | MinWidthProperty | Identifies the GridColumn.MinWidth bindable property. (Inherited from **GridColumn**) |
| S | SortIndexProperty | Identifies the GridColumn.SortIndex bindable property. (Inherited from **GridColumn**) |
| S | SortModeProperty | Identifies the GridColumn.SortMode bindable property. (Inherited from **GridColumn**) |
| S | SortOrderProperty | Identifies the GridColumn.SortOrder bindable property. (Inherited from **GridColumn**) |
| S | UnboundExpressionProperty | Identifies the GridColumn.UnboundExpression bindable property. (Inherited from **GridColumn**) |
| S | UnboundTypeProperty | Identifies the GridColumn.UnboundType bindable property. (Inherited from **GridColumn**) |
| S | ValueMemberProperty | Identifies the ValueMember bindable property. |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | WidthProperty | Identifies the GridColumn.Width bindable property. (Inherited from **GridColumn**) |

**Public Events**

| | Name | Description |
|---|---|---|
| | BindingContextChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanging | (Inherited from **Xamarin.Forms.BindableObject**) |

**See Also**
PickerColumn Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# PickerColumn Constructor

Initializes a new instance of the PickerColumn class with default settings.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**
**C#**

```
public PickerColumn()
```

**See Also**
PickerColumn Class
PickerColumn Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# PickerColumn Properties

A grid column allowing an end-user to edit a cell value by selecting an item from the predefined set.

The following tables list the members exposed by the PickerColumn type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | PickerColumn | Initializes a new instance of the PickerColumn class with default settings. |

### Public Properties

| | Name | Description |
|---|---|---|
| | ActualCaption | Gets the column caption as it is displayed in a column header. (Inherited from **GridColumn**) |
| | ActualWidth | Gets the column width. This is a bindable property. (Inherited from **GridColumn**) |
| | AllowAutoFilter | Gets or sets whether an end-user can filter data by the column's values using the auto-filter panel. This is a bindable property. (Inherited from **GridColumn**) |
| | AllowGroup | Gets or sets whether end-users are allowed to group data by the current column. This is a bindable property. (Inherited from **GridColumn**) |
| | AllowSort | Gets or sets whether end-users are allowed to sort data by the column's values. This is a bindable property. (Inherited from **GridColumn**) |
| | AutoFilterCondition | Gets or sets the comparison operator used to create filter conditions for the current column via the auto-filter panel. This is a bindable property. (Inherited from **GridColumn**) |
| | AutoFilterValue | Gets or sets the value in the auto-filter panel for the column. This is a bindable property. (Inherited from **GridColumn**) |
| | BindingContext | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Caption | Gets or sets the column's display caption. This is a bindable property. (Inherited from **GridColumn**) |
| | ColumnFilterMode | Gets or sets how column values are filtered via the auto-filter panel. This is a bindable property. (Inherited from **GridColumn**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | ContentAlignment | Gets or sets the content alignment within column cells. This is a bindable property. (Inherited from **GridColumn**) |
| | DisplayFormat | Gets or sets formatting for values in the column. This is a bindable property. (Inherited from **GridColumn**) |
| | DisplayMember | Gets or sets a field name in the bound data source whose contents are to be displayed by the column's picker control. This is a bindable property. |
| | FieldName | Gets or sets the column's unique field name. For bound columns, this property refers to a field in the underlying data source. For unbound columns, this property serves as a column's identifier.<br>This is a bindable property.<br>(Inherited from **GridColumn**) |
| | FixedStyle | Gets or sets whether a column is fixed. This is a bindable property. (Inherited from **GridColumn**) |
| | GroupInterval | Gets or sets how data rows are grouped when grouping by the current column is applied. This is a bindable property. (Inherited from **GridColumn**) |
| | HeaderTemplate | Gets or sets a template used to define the column headers visual representation. (Inherited from **GridColumn**) |
| | ImmediateUpdateAutoFilter | Gets or sets whether the column's filter condition is updated as soon as an end-user modifies the text within the auto filter panel's cell. This is a bindable property. (Inherited from **GridColumn**) |
| | IsGrouped | Gets or sets whether the grid's data is grouped by the current column. This is a bindable property. (Inherited from **GridColumn**) |
| | IsReadOnly | Gets or sets whether end-users are allowed to change cell values in the column. This is a bindable property. (Inherited from **GridColumn**) |
| | IsUnbound | Gets whether the column is unbound. (Inherited from **GridColumn**) |
| | IsVisible | Gets or sets whether the column is visible in the grid. This is a bindable property. (Inherited from **GridColumn**) |
| | ItemsSource | Gets or sets a data source for a picker control that appears when a column |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | | cell is in the edit mode. This is a bindable property. |
| | MinWidth | Gets or sets the minimum column width allowed. This is a bindable property. (Inherited from **GridColumn**) |
| | SerializationTypeName | Gets the string containing the column type name. (Inherited from **GridColumn**) |
| | SortIndex | Gets or sets the column's position among sorted columns. This is a bindable property. (Inherited from **GridColumn**) |
| | SortMode | Specifies how column values are sorted and grouped. This is a bindable property. (Inherited from **GridColumn**) |
| | SortOrder | Gets or sets the column's sort order. This is a bindable property. (Inherited from **GridColumn**) |
| | UnboundExpression | Gets or sets an expression used to evaluate values for the current Unbound Columns. This is a bindable property. (Inherited from **GridColumn**) |
| | UnboundType | Gets or sets the data type and binding mode of the column. This is a bindable property. (Inherited from **GridColumn**) |
| | ValueMember | Gets or sets the field name in the bound data source whose contents are assigned item values of the column's picker control. This is a bindable property. |
| | Width | Gets or sets the column width. This is a bindable property. (Inherited from **GridColumn**) |

**Public Methods**

| | Name | Description |
|---|---|---|
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ToString | (Inherited from **System.Object**) |

**Public Fields**

| | Name | Description |
|---|---|---|
| S | ActualWidthProperty | Identifies the GridColumn.ActualWidth bindable property. (Inherited from **GridColumn**) |
| S | AllowAutoFilterProperty | Identifies the GridColumn.AllowAutoFilter bindable property. (Inherited from **GridColumn**) |
| S | AllowGroupProperty | Identifies the GridColumn.AllowGroup bindable property. (Inherited from **GridColumn**) |
| S | AllowSortProperty | Identifies the GridColumn.AllowSort bindable property. (Inherited from **GridColumn**) |
| S | AutoFilterConditionProperty | Identifies the GridColumn.AutoFilterCondition bindable property. (Inherited from **GridColumn**) |
| S | AutoFilterValueProperty | Identifies the GridColumn.AutoFilterValue bindable property. (Inherited from **GridColumn**) |
| S | BindingContextProperty | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | CaptionProperty | Identifies the GridColumn.Caption bindable property. (Inherited from **GridColumn**) |
| S | ColumnFilterModeProperty | Identifies the GridColumn.ColumnFilterMode bindable property. (Inherited from **GridColumn**) |
| S | ContentAlignmentProperty | Identifies the GridColumn.ContentAlignment bindable property. (Inherited from **GridColumn**) |
| S | DisplayFormatProperty | Identifies the GridColumn.DisplayFormat bindable property. (Inherited from **GridColumn**) |
| S | DisplayMemberProperty | Identifies the DisplayMember bindable property. |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| S | FieldNameProperty | Identifies the GridColumn.FieldName bindable property. (Inherited from **GridColumn**) |
| S | FixedStyleProperty | Identifies the GridColumn.FixedStyle bindable property. (Inherited from **GridColumn**) |
| S | GroupIntervalProperty | Identifies the GridColumn.GroupInterval bindable property. (Inherited from **GridColumn**) |
| S | HeaderTemplateProperty | Identifies the GridColumn.HeaderTemplate bindable property. (Inherited from **GridColumn**) |
| S | ImmediateUpdateAutoFilterProperty | Identifies the GridColumn.ImmediateUpdateAutoFilter bindable property. (Inherited from **GridColumn**) |
| S | IsGroupedProperty | Identifies the GridColumn.IsGrouped bindable property. (Inherited from **GridColumn**) |
| S | IsReadOnlyProperty | Identifies the GridColumn.IsReadOnly bindable property. (Inherited from **GridColumn**) |
| S | IsVisibleProperty | Identifies the GridColumn.IsVisible bindable property. (Inherited from **GridColumn**) |
| S | ItemsSourceProperty | Identifies the ItemsSource bindable property. |
| S | MinWidthProperty | Identifies the GridColumn.MinWidth bindable property. (Inherited from **GridColumn**) |
| S | SortIndexProperty | Identifies the GridColumn.SortIndex bindable property. (Inherited from **GridColumn**) |
| S | SortModeProperty | Identifies the GridColumn.SortMode bindable property. (Inherited from **GridColumn**) |
| S | SortOrderProperty | Identifies the GridColumn.SortOrder bindable property. (Inherited from **GridColumn**) |
| S | UnboundExpressionProperty | Identifies the GridColumn.UnboundExpression bindable property. (Inherited from **GridColumn**) |
| S | UnboundTypeProperty | Identifies the GridColumn.UnboundType bindable property. (Inherited from **GridColumn**) |
| S | ValueMemberProperty | Identifies the ValueMember bindable property. |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | WidthProperty | Identifies the GridColumn.Width bindable property. (Inherited from **GridColumn**) |

**Public Events**

| | Name | Description |
|---|---|---|
| | BindingContextChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanging | (Inherited from **Xamarin.Forms.BindableObject**) |

**See Also**
PickerColumn Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# DisplayMember Property

Gets or sets a field name in the bound data source whose contents are to be displayed by the column's picker control. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public string DisplayMember { get; set; }
```

### Property Value

A System.String value specifying a field name in a data source.

### Remarks

A picker control which appears when a column cell is in the edit mode can be populated with items from a data source. To bind the picker to a data source, use the ItemsSource property. The **DisplayMember** and ValueMember properties specify the field names in the bound data source that supply display strings and item values, respectively.

### Example

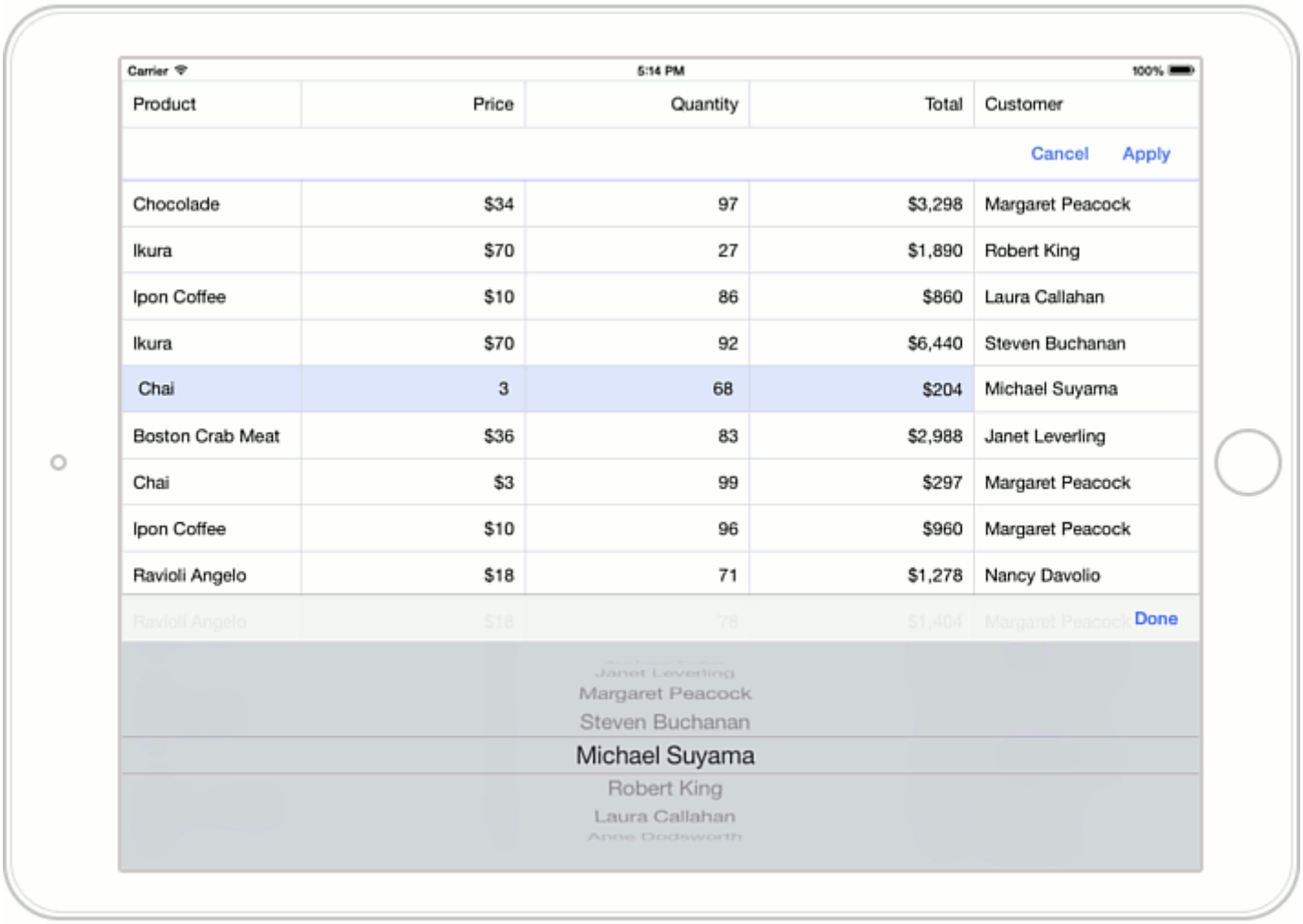
| **Show Me** | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T257726. | |

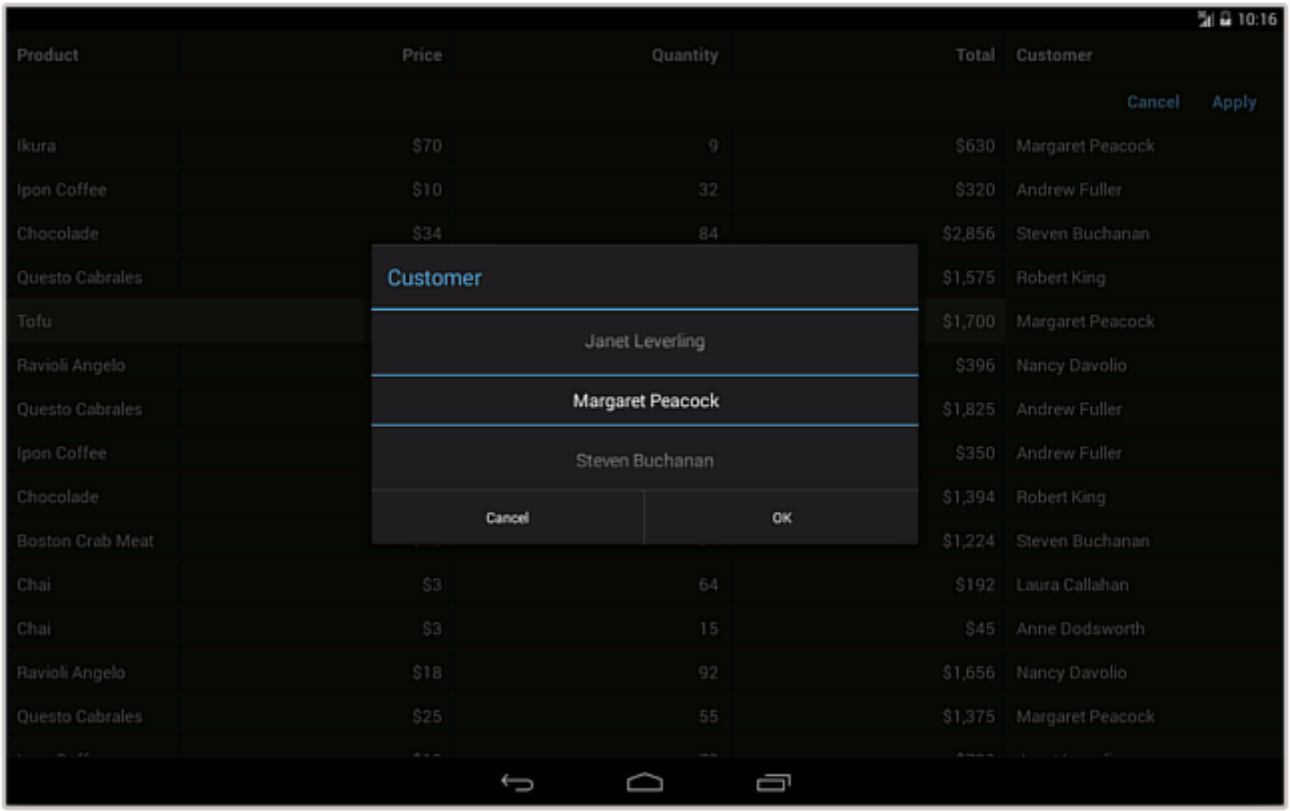
This example demonstrates how to implement the grid's PickerColumn whose items are coming from the grid's underlying data source.

In this example, a grid is bound to a collection of the **Order** objects. Each order contains information about a customer (the **CustomerId** property). A customer is represented by a **Customer** object that has the **Id** and **Name** properties. **Customer** objects are stored in an IList collection. The following properties of the PickerColumn object are set to show customer names for orders in this column within the grid (see the *MainPage.xaml* file):

- GridColumn.FieldName — an order's property to be bound to a column (**CustomerId**).
- ItemsSource — a collection of customers.
- ValueMember — a customer's property used to associate a customer with an order (**Id**).
- DisplayMember — a customer's property to be displayed in a column (**Name**).

© 2018 Developer Express Inc.


Carrier
5:14 PM
100%
Product
Price
Quantity
Total
Customer
Cancel
Apply
Chocolade $34 97 $3,298 Margaret Peacock
Ikura $70 27 $1,890 Robert King
Ipon Coffee $10 86 $860 Laura Callahan
Ikura $70 92 $6,440 Steven Buchanan
Chai 3 68 $204 Michael Suyama
Boston Crab Meat $36 83 $2,988 Janet Leverling
Chai $3 99 $297 Margaret Peacock
Ipon Coffee $10 96 $960 Margaret Peacock
Ravioli Angelo $18 71 $1,278 Nancy Davolio
Done
Janet Leverling
Margaret Peacock
Steven Buchanan
Michael Suyama
Robert King
Laura Callahan


10:16
Product
Price
Quantity
Total
Customer
Cancel
Apply
Ikura $70 9 $630 Margaret Peacock
Ipon Coffee $10 32 $320 Andrew Fuller
Chocolade $34 84 $2,856 Steven Buchanan
Questo Cabrales $1,575 Robert King
Tofu $1,700 Margaret Peacock
Ravioli Angelo $396 Nancy Davolio
Questo Cabrales $1,825 Andrew Fuller
Ipon Coffee $350 Andrew Fuller
Chocolade $1,394 Robert King
Boston Crab Meat $1,224 Steven Buchanan
Chai $3 64 $192 Laura Callahan
Chai $3 15 $45 Anne Dodsworth
Ravioli Angelo $18 92 $1,656 Nancy Davolio
Questo Cabrales $25 55 $1,375 Margaret Peacock
Customer
Janet Leverling
Margaret Peacock
Steven Buchanan
Cancel
OK

C#

```
 (Customer.cs)
using System;
namespace PickerColumnExample
{
    public class Customer : ModelObject {
        string name;
        public Customer(string name, int id) {
            this.name = name;
            this.Id = id;
        }
        public int Id { get; set; }
        public string Name {
            get { return name; }
            set {name = value;}
        }
    }
}
```

C#

```
 (MainPage.xaml.cs)
using System;
using System.Collections.Generic;
using Xamarin.Forms;
namespace PickerColumnExample {
    public partial class MainPage : ContentPage {
        public MainPage () {
            InitializeComponent ();
            TestOrdersRepository model = new TestOrdersRepository ();
            BindingContext = model;
        }
    }
}
```

C#

```
 (ModelObject.cs)
using System;
using System.ComponentModel;
namespace PickerColumnExample
{
    public class ModelObject : INotifyPropertyChanged {
        public event PropertyChangedEventHandler PropertyChanged;
        protected void RaisePropertyChanged(string name) {
            if (PropertyChanged != null)
                PropertyChanged(this, new PropertyChangedEventArgs(name));
        }
    }
}
```

C#

© 2018 Developer Express Inc.

```
 (Order.cs)
using System;
namespace PickerColumnExample
{
    public class Order : ModelObject {
        Product product;
        int quantity;
        int customerId;
        public Order() {
            this.product = new Product ("", 0);
            this.quantity = 0;
        }
        public Order(Product product, int amount, int customerId) {
            this.product = product;
            this.quantity = amount;
            this.customerId = customerId;
        }
        public Product Product {
            get { return product; }
            set { if (product != value) {
                    product = value;
                    RaisePropertyChanged ("Product");}}
        }
        public int Quantity {
            get { return quantity; }
            set { if (quantity != value) {
                    quantity = value;
                    RaisePropertyChanged ("Quantity");}}
        }
        public int CustomerId {
            get { return customerId; }
            set {
                if (customerId != value) {
                    customerId = value;
                    RaisePropertyChanged("CustomerId");
                }
            }
        }
    }
}
```

**C#**

© 2018 Developer Express Inc.

```
 (OrdersRepository.cs)
using System;
using System.Collections.Generic;
using System.Collections.ObjectModel;
namespace PickerColumnExample
{
    public abstract class OrdersRepository {
        readonly ObservableCollection<Order> orders;
        readonly ObservableCollection<Customer> customers;
        public OrdersRepository() {
            this.orders = new ObservableCollection<Order>();
            this.customers = new ObservableCollection<Customer>();
        }
        public ObservableCollection<Order> Orders {
            get { return orders; }
        }
        public IList<Customer> Customers{
            get { return customers; }
        }
    }
    public class TestOrdersRepository : OrdersRepository {
        const int orderCount = 100;
        readonly List<Product> products;
        readonly Random random;
        public TestOrdersRepository() : base() {
            this.random = new Random((int)DateTime.Now.Ticks);
            this.products = new List<Product>();
            GenerateProducts();
            GenerateCustomers();
            for (int i = 0; i < orderCount; i++)
                Orders.Add(GenerateOrder(i));
        }
        Order GenerateOrder(int number) {
            Order order = new Order(RandomItem<Product>(products), random.
            return order;
        }
        T RandomItem<T>(IList<T> list) {
            int index = (int)(random.NextDouble() * 0.99 * (list.Count));
            return list[index];
        }
        void GenerateProducts() {
            products.Add(new Product("Tofu", 50));
            products.Add(new Product("Chocolade", 34));
            products.Add(new Product("Ikura", 70));
            products.Add(new Product("Chai", 3));
            products.Add(new Product("Boston Crab Meat", 36));
            products.Add(new Product("Ravioli Angelo", 18));
            products.Add(new Product("Ipon Coffee", 10));
            products.Add(new Product("Questo Cabrales", 25));
        }
        void GenerateCustomers() {
```

© 2018 Developer Express Inc.

```
            Customers.Add(new Customer("Nancy Davolio", 0));
            Customers.Add(new Customer("Andrew Fuller", 1));
            Customers.Add(new Customer("Janet Leverling", 2));
            Customers.Add(new Customer("Margaret Peacock", 3));
            Customers.Add(new Customer("Steven Buchanan", 4));
            Customers.Add(new Customer("Michael Suyama", 5));
            Customers.Add(new Customer("Robert King", 6));
            Customers.Add(new Customer("Laura Callahan", 7));
            Customers.Add(new Customer("Anne Dodsworth", 8));
        }
    }
}
```

**C#**

```
 (Product.cs)
using System;
namespace PickerColumnExample
{
    public class Product : ModelObject {
        string name;
        int unitPrice;
        public Product(string name, int unitPrice) {
            this.name = name;
            this.unitPrice = unitPrice;
        }
        public string Name {
            get { return name; }
            set { name = value; }
        }
        public int UnitPrice {
            get { return unitPrice; }
            set { unitPrice = value; }
        }
    }
}
```

**Xaml**

© 2018 Developer Express Inc.

```
 (MainPage.xaml)
<?xml version="1.0" encoding="UTF-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="PickerColumnExample.MainPage"
             xmlns:dxg="clr-namespace:DevExpress.Mobile.DataGrid;assembly=
    <ContentPage.Padding>
        <OnPlatform x:TypeArguments="Thickness" iOS="0, 20, 0, 0" />
    </ContentPage.Padding>
    <dxg:GridControl x:Name="grid" ItemsSource="{Binding Orders}">
         <dxg:GridControl.Columns>
             <dxg:TextColumn FieldName="Product.Name" Caption="Product" Wi
             <dxg:NumberColumn FieldName="Product.UnitPrice" Caption="Pric
             <dxg:NumberColumn FieldName="Quantity"/>
             <dxg:NumberColumn FieldName="Total"
                               UnboundType="Integer" UnboundExpression=
                               DisplayFormat="C0" IsReadOnly="True"/>
             <dxg:PickerColumn FieldName="CustomerId" Caption="Customer"
                               ItemsSource="{Binding Customers}" ValueMemb
         </dxg:GridControl.Columns>
    </dxg:GridControl>
</ContentPage>
```

## See Also

PickerColumn Class
PickerColumn Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# ItemsSource Property

Gets or sets a data source for a picker control that appears when a column cell is in the edit mode. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public object ItemsSource { get; set; }
```

### Property Value

An object that specifies the data source from which a picker control retrieves its data.

### Remarks

The **ItemsSource** property specifies a data source to populate the column's picker control with items. The DisplayMember and ValueMember properties specify the field names in the bound data source that supply display strings and item values, respectively.

### Example

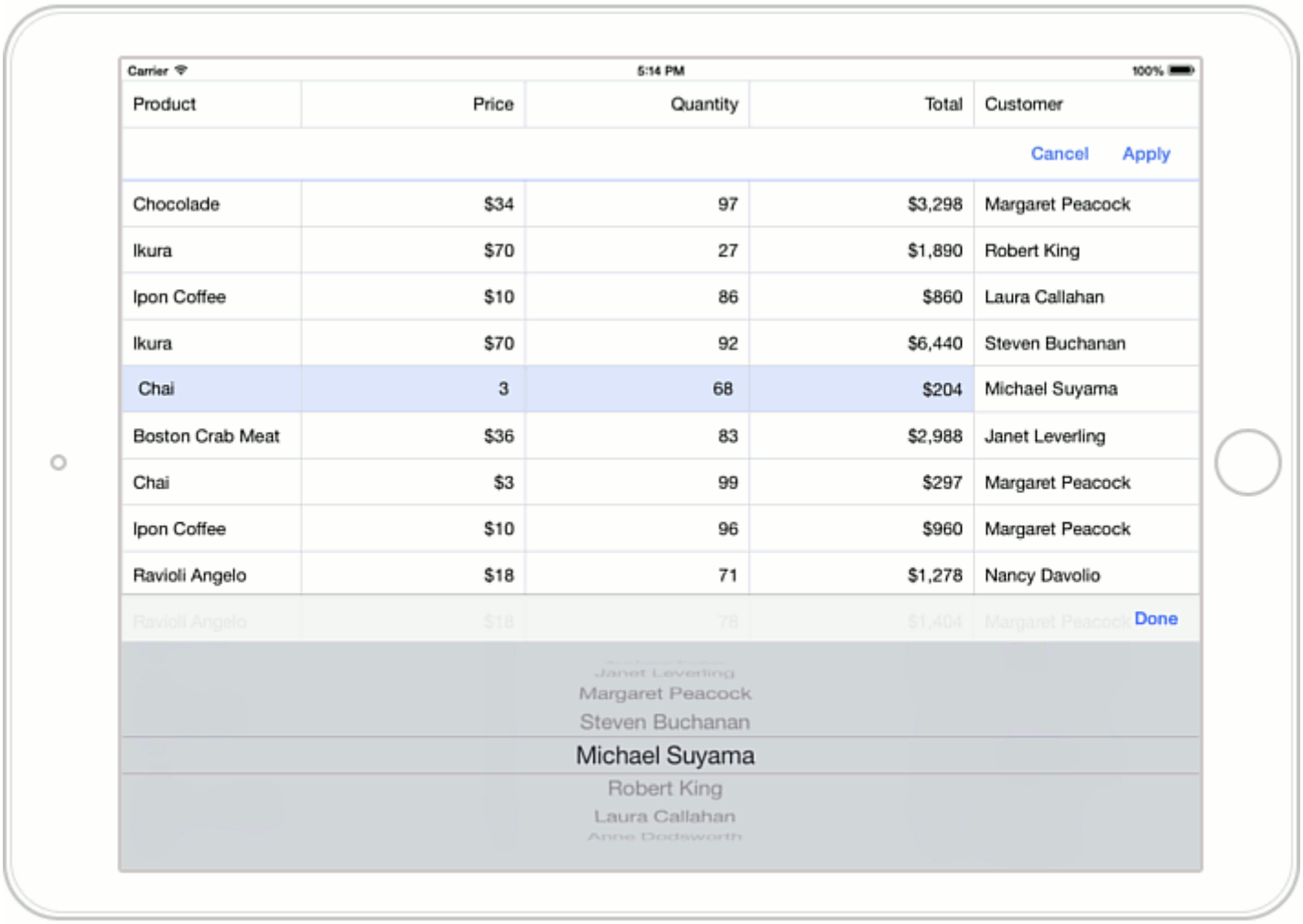
| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T257726. | |

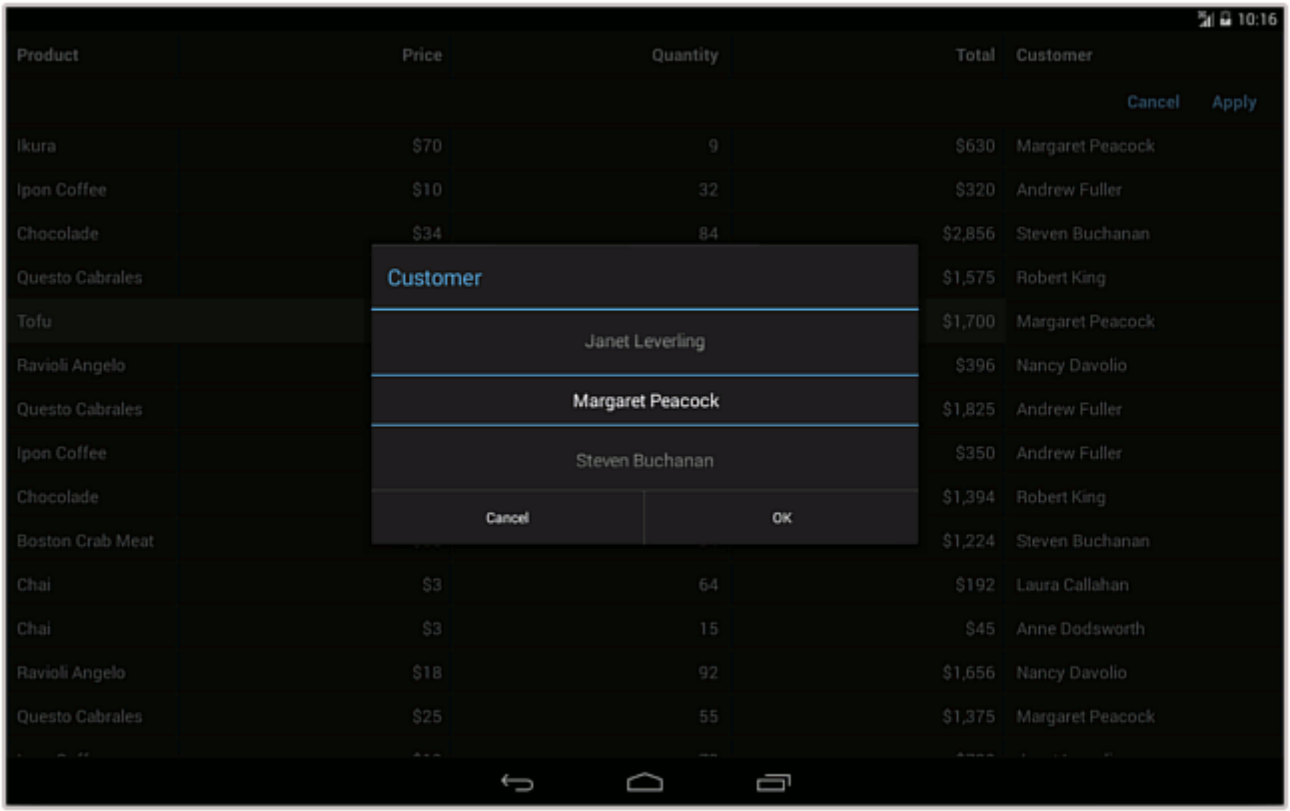
This example demonstrates how to implement the grid's PickerColumn whose items are coming from the grid's underlying data source.

In this example, a grid is bound to a collection of the **Order** objects. Each order contains information about a customer (the **CustomerId** property). A customer is represented by a **Customer** object that has the **Id** and **Name** properties. **Customer** objects are stored in an IList collection. The following properties of the PickerColumn object are set to show customer names for orders in this column within the grid (see the *MainPage.xaml* file):

- GridColumn.FieldName — an order's property to be bound to a column (**CustomerId**).
- ItemsSource — a collection of customers.
- ValueMember — a customer's property used to associate a customer with an order (**Id**).
- DisplayMember — a customer's property to be displayed in a column (**Name**).

© 2018 Developer Express Inc.


Carrier
5:14 PM
100%
Product
Price
Quantity
Total
Customer
Cancel
Apply
Chocolade $34 97 $3,298 Margaret Peacock
Ikura $70 27 $1,890 Robert King
Ipon Coffee $10 86 $860 Laura Callahan
Ikura $70 92 $6,440 Steven Buchanan
Chai 3 68 $204 Michael Suyama
Boston Crab Meat $36 83 $2,988 Janet Leverling
Chai $3 99 $297 Margaret Peacock
Ipon Coffee $10 96 $960 Margaret Peacock
Ravioli Angelo $18 71 $1,278 Nancy Davolio
Done
Janet Leverling
Margaret Peacock
Steven Buchanan
Michael Suyama
Robert King
Laura Callahan


10:16
Product
Price
Quantity
Total
Customer
Cancel
Apply
Ikura $70 9 $630 Margaret Peacock
Ipon Coffee $10 32 $320 Andrew Fuller
Chocolade $34 84 $2,856 Steven Buchanan
Questo Cabrales $1,575 Robert King
Tofu $1,700 Margaret Peacock
Ravioli Angelo $396 Nancy Davolio
Questo Cabrales $1,825 Andrew Fuller
Ipon Coffee $350 Andrew Fuller
Chocolade $1,394 Robert King
Boston Crab Meat $1,224 Steven Buchanan
Chai $3 64 $192 Laura Callahan
Chai $3 15 $45 Anne Dodsworth
Ravioli Angelo $18 92 $1,656 Nancy Davolio
Questo Cabrales $25 55 $1,375 Margaret Peacock
Customer
Janet Leverling
Margaret Peacock
Steven Buchanan
Cancel
OK

C#

```
 (Customer.cs)
using System;
namespace PickerColumnExample
{
    public class Customer : ModelObject {
        string name;
        public Customer(string name, int id) {
            this.name = name;
            this.Id = id;
        }
        public int Id { get; set; }
        public string Name {
            get { return name; }
            set {name = value;}
        }
    }
}
```

C#

```
 (MainPage.xaml.cs)
using System;
using System.Collections.Generic;
using Xamarin.Forms;
namespace PickerColumnExample {
    public partial class MainPage : ContentPage {
        public MainPage () {
            InitializeComponent ();
            TestOrdersRepository model = new TestOrdersRepository ();
            BindingContext = model;
        }
    }
}
```

C#

```
 (ModelObject.cs)
using System;
using System.ComponentModel;
namespace PickerColumnExample
{
    public class ModelObject : INotifyPropertyChanged {
        public event PropertyChangedEventHandler PropertyChanged;
        protected void RaisePropertyChanged(string name) {
            if (PropertyChanged != null)
                PropertyChanged(this, new PropertyChangedEventArgs(name));
        }
    }
}
```

C#

© 2018 Developer Express Inc.

```
 (Order.cs)
using System;
namespace PickerColumnExample
{
    public class Order : ModelObject {
        Product product;
        int quantity;
        int customerId;
        public Order() {
            this.product = new Product ("", 0);
            this.quantity = 0;
        }
        public Order(Product product, int amount, int customerId) {
            this.product = product;
            this.quantity = amount;
            this.customerId = customerId;
        }
        public Product Product {
            get { return product; }
            set { if (product != value) {
                    product = value;
                    RaisePropertyChanged ("Product");}}
        }
        public int Quantity {
            get { return quantity; }
            set { if (quantity != value) {
                    quantity = value;
                    RaisePropertyChanged ("Quantity");}}
        }
        public int CustomerId {
            get { return customerId; }
            set {
                if (customerId != value) {
                    customerId = value;
                    RaisePropertyChanged("CustomerId");
                }
            }
        }
    }
}
```

| C# | |
|---|---|

© 2018 Developer Express Inc.

```
 (OrdersRepository.cs)
using System;
using System.Collections.Generic;
using System.Collections.ObjectModel;
namespace PickerColumnExample
{
    public abstract class OrdersRepository {
        readonly ObservableCollection<Order> orders;
        readonly ObservableCollection<Customer> customers;
        public OrdersRepository() {
            this.orders = new ObservableCollection<Order>();
            this.customers = new ObservableCollection<Customer>();
        }
        public ObservableCollection<Order> Orders {
            get { return orders; }
        }
        public IList<Customer> Customers{
            get { return customers; }
        }
    }
    public class TestOrdersRepository : OrdersRepository {
        const int orderCount = 100;
        readonly List<Product> products;
        readonly Random random;
        public TestOrdersRepository() : base() {
            this.random = new Random((int)DateTime.Now.Ticks);
            this.products = new List<Product>();
            GenerateProducts();
            GenerateCustomers();
            for (int i = 0; i < orderCount; i++)
                Orders.Add(GenerateOrder(i));
        }
        Order GenerateOrder(int number) {
            Order order = new Order(RandomItem<Product>(products), random.
            return order;
        }
        T RandomItem<T>(IList<T> list) {
            int index = (int)(random.NextDouble() * 0.99 * (list.Count));
            return list[index];
        }
        void GenerateProducts() {
            products.Add(new Product("Tofu", 50));
            products.Add(new Product("Chocolade", 34));
            products.Add(new Product("Ikura", 70));
            products.Add(new Product("Chai", 3));
            products.Add(new Product("Boston Crab Meat", 36));
            products.Add(new Product("Ravioli Angelo", 18));
            products.Add(new Product("Ipon Coffee", 10));
            products.Add(new Product("Questo Cabrales", 25));
        }
        void GenerateCustomers() {
```

```
            Customers.Add(new Customer("Nancy Davolio", 0));
            Customers.Add(new Customer("Andrew Fuller", 1));
            Customers.Add(new Customer("Janet Leverling", 2));
            Customers.Add(new Customer("Margaret Peacock", 3));
            Customers.Add(new Customer("Steven Buchanan", 4));
            Customers.Add(new Customer("Michael Suyama", 5));
            Customers.Add(new Customer("Robert King", 6));
            Customers.Add(new Customer("Laura Callahan", 7));
            Customers.Add(new Customer("Anne Dodsworth", 8));
        }
    }
}
```

**C#**

```
 (Product.cs)
using System;
namespace PickerColumnExample
{
    public class Product : ModelObject {
        string name;
        int unitPrice;
        public Product(string name, int unitPrice) {
            this.name = name;
            this.unitPrice = unitPrice;
        }
        public string Name {
            get { return name; }
            set { name = value; }
        }
        public int UnitPrice {
            get { return unitPrice; }
            set { unitPrice = value; }
        }
    }
}
```

**Xaml**

© 2018 Developer Express Inc.

```
 (MainPage.xaml)
<?xml version="1.0" encoding="UTF-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="PickerColumnExample.MainPage"
             xmlns:dxg="clr-namespace:DevExpress.Mobile.DataGrid;assembly=
    <ContentPage.Padding>
        <OnPlatform x:TypeArguments="Thickness" iOS="0, 20, 0, 0" />
    </ContentPage.Padding>
    <dxg:GridControl x:Name="grid" ItemsSource="{Binding Orders}">
         <dxg:GridControl.Columns>
             <dxg:TextColumn FieldName="Product.Name" Caption="Product" Wi
             <dxg:NumberColumn FieldName="Product.UnitPrice" Caption="Pric
             <dxg:NumberColumn FieldName="Quantity"/>
             <dxg:NumberColumn FieldName="Total"
                               UnboundType="Integer" UnboundExpression=
                               DisplayFormat="C0" IsReadOnly="True"/>
             <dxg:PickerColumn FieldName="CustomerId" Caption="Customer"
                               ItemsSource="{Binding Customers}" ValueMemb
         </dxg:GridControl.Columns>
    </dxg:GridControl>
</ContentPage>
```

## See Also

PickerColumn Class
PickerColumn Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# ValueMember Property

Gets or sets the field name in the bound data source whose contents are assigned item values of the column's picker control. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public string ValueMember { get; set; }
```

### Property Value

A System.String value that specifies the field name in a bound data source.

### Remarks

A picker control which appears when a column cell is in the edit mode can be populated with items from a data source. To bind the picker to a data source, use the ItemsSource property. The DisplayMember and **ValueMember** properties specify the field names in the bound data source that supply display strings and item values, respectively.

### Example

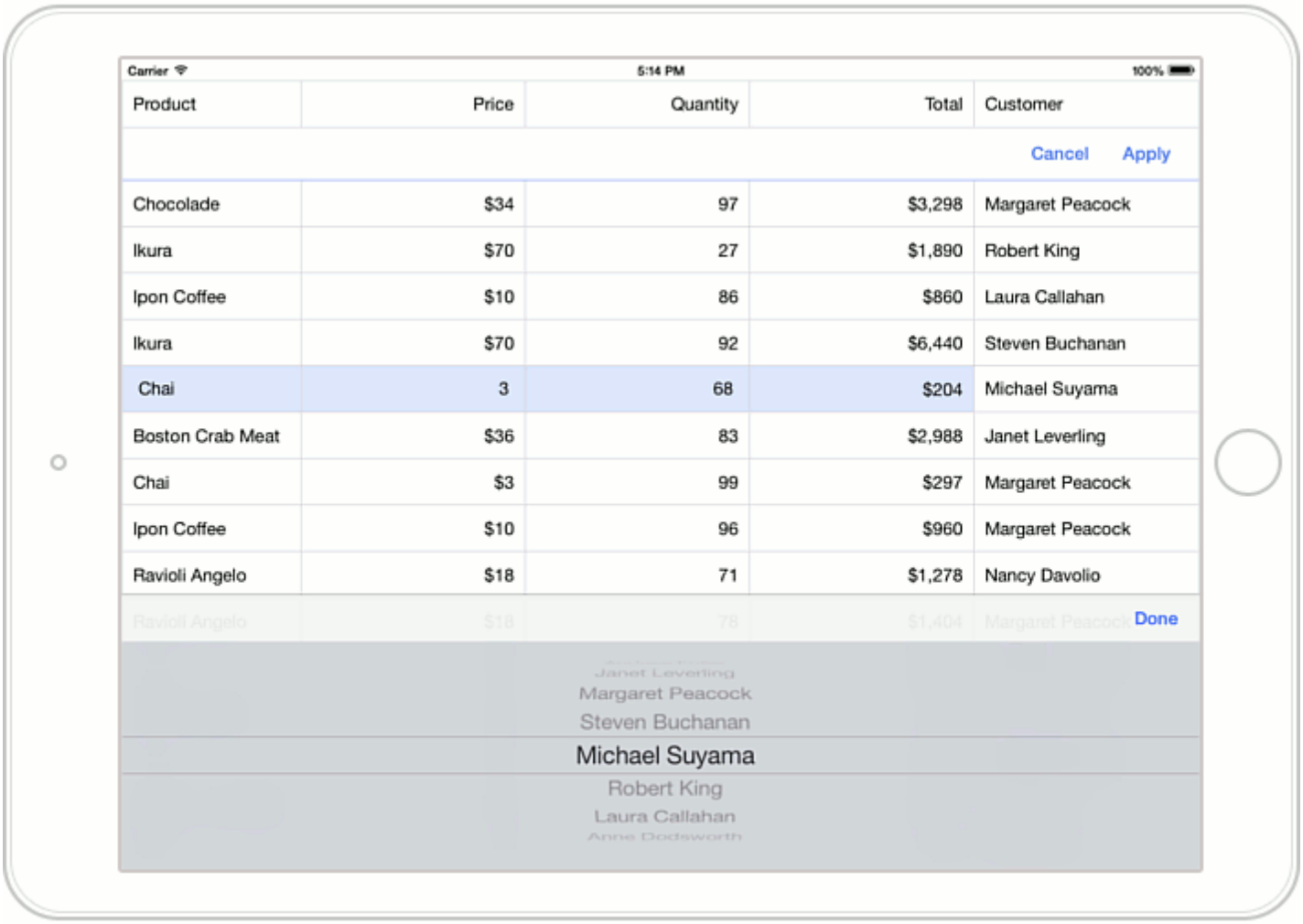
| **Show Me** | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T257726. | |

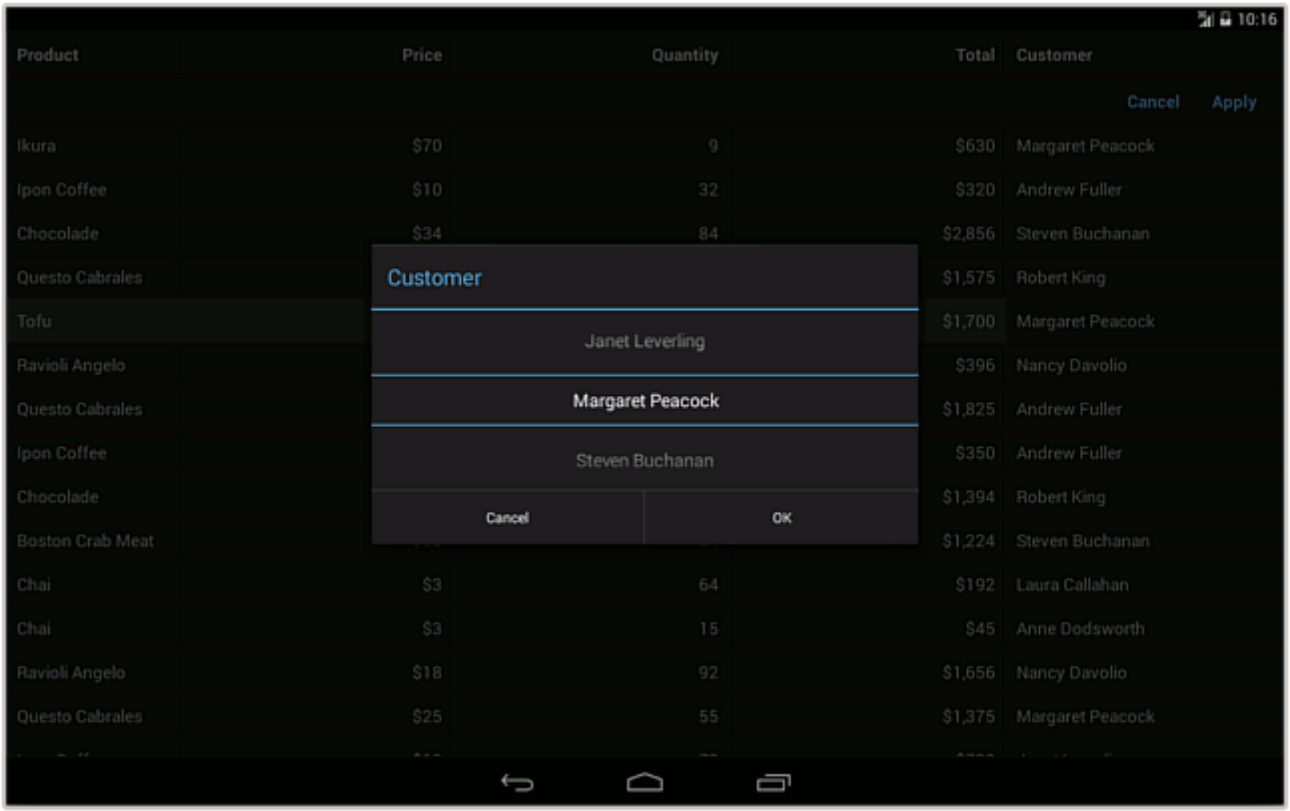
This example demonstrates how to implement the grid's PickerColumn whose items are coming from the grid's underlying data source.

In this example, a grid is bound to a collection of the **Order** objects. Each order contains information about a customer (the **CustomerId** property). A customer is represented by a **Customer** object that has the **Id** and **Name** properties. **Customer** objects are stored in an IList collection. The following properties of the PickerColumn object are set to show customer names for orders in this column within the grid (see the *MainPage.xaml* file):

- GridColumn.FieldName — an order's property to be bound to a column (**CustomerId**).
- ItemsSource — a collection of customers.
- ValueMember — a customer's property used to associate a customer with an order (**Id**).
- DisplayMember — a customer's property to be displayed in a column (**Name**).

© 2018 Developer Express Inc.


Carrier
5:14 PM
100%
Product
Price
Quantity
Total
Customer
Cancel
Apply
Chocolade $34 97 $3,298 Margaret Peacock
Ikura $70 27 $1,890 Robert King
Ipon Coffee $10 86 $860 Laura Callahan
Ikura $70 92 $6,440 Steven Buchanan
Chai 3 68 $204 Michael Suyama
Boston Crab Meat $36 83 $2,988 Janet Leverling
Chai $3 99 $297 Margaret Peacock
Ipon Coffee $10 96 $960 Margaret Peacock
Ravioli Angelo $18 71 $1,278 Nancy Davolio
Done
Janet Leverling
Margaret Peacock
Steven Buchanan
Michael Suyama
Robert King
Laura Callahan


10:16
Product
Price
Quantity
Total
Customer
Cancel
Apply
Ikura $70 9 $630 Margaret Peacock
Ipon Coffee $10 32 $320 Andrew Fuller
Chocolade $34 84 $2,856 Steven Buchanan
Questo Cabrales $1,575 Robert King
Tofu $1,700 Margaret Peacock
Ravioli Angelo $396 Nancy Davolio
Questo Cabrales $1,825 Andrew Fuller
Ipon Coffee $350 Andrew Fuller
Chocolade $1,394 Robert King
Boston Crab Meat $1,224 Steven Buchanan
Chai $3 64 $192 Laura Callahan
Chai $3 15 $45 Anne Dodsworth
Ravioli Angelo $18 92 $1,656 Nancy Davolio
Questo Cabrales $25 55 $1,375 Margaret Peacock
Customer
Janet Leverling
Margaret Peacock
Steven Buchanan
Cancel
OK

C#

```
 (Customer.cs)
using System;
namespace PickerColumnExample
{
    public class Customer : ModelObject {
        string name;
        public Customer(string name, int id) {
            this.name = name;
            this.Id = id;
        }
        public int Id { get; set; }
        public string Name {
            get { return name; }
            set {name = value;}
        }
    }
}
```

C#

```
 (MainPage.xaml.cs)
using System;
using System.Collections.Generic;
using Xamarin.Forms;
namespace PickerColumnExample {
    public partial class MainPage : ContentPage {
        public MainPage () {
            InitializeComponent ();
            TestOrdersRepository model = new TestOrdersRepository ();
            BindingContext = model;
        }
    }
}
```

C#

```
 (ModelObject.cs)
using System;
using System.ComponentModel;
namespace PickerColumnExample
{
    public class ModelObject : INotifyPropertyChanged {
        public event PropertyChangedEventHandler PropertyChanged;
        protected void RaisePropertyChanged(string name) {
            if (PropertyChanged != null)
                PropertyChanged(this, new PropertyChangedEventArgs(name));
        }
    }
}
```

C#

© 2018 Developer Express Inc.

```
 (Order.cs)
using System;
namespace PickerColumnExample
{
    public class Order : ModelObject {
        Product product;
        int quantity;
        int customerId;
        public Order() {
            this.product = new Product ("", 0);
            this.quantity = 0;
        }
        public Order(Product product, int amount, int customerId) {
            this.product = product;
            this.quantity = amount;
            this.customerId = customerId;
        }
        public Product Product {
            get { return product; }
            set { if (product != value) {
                    product = value;
                    RaisePropertyChanged ("Product");}}
        }
        public int Quantity {
            get { return quantity; }
            set { if (quantity != value) {
                    quantity = value;
                    RaisePropertyChanged ("Quantity");}}
        }
        public int CustomerId {
            get { return customerId; }
            set {
                if (customerId != value) {
                    customerId = value;
                    RaisePropertyChanged("CustomerId");
                }
            }
        }
    }
}
```

**C#**

© 2018 Developer Express Inc.

```
 (OrdersRepository.cs)
using System;
using System.Collections.Generic;
using System.Collections.ObjectModel;
namespace PickerColumnExample
{
    public abstract class OrdersRepository {
        readonly ObservableCollection<Order> orders;
        readonly ObservableCollection<Customer> customers;
        public OrdersRepository() {
            this.orders = new ObservableCollection<Order>();
            this.customers = new ObservableCollection<Customer>();
        }
        public ObservableCollection<Order> Orders {
            get { return orders; }
        }
        public IList<Customer> Customers{
            get { return customers; }
        }
    }
    public class TestOrdersRepository : OrdersRepository {
        const int orderCount = 100;
        readonly List<Product> products;
        readonly Random random;
        public TestOrdersRepository() : base() {
            this.random = new Random((int)DateTime.Now.Ticks);
            this.products = new List<Product>();
            GenerateProducts();
            GenerateCustomers();
            for (int i = 0; i < orderCount; i++)
                Orders.Add(GenerateOrder(i));
        }
        Order GenerateOrder(int number) {
            Order order = new Order(RandomItem<Product>(products), random.
            return order;
        }
        T RandomItem<T>(IList<T> list) {
            int index = (int)(random.NextDouble() * 0.99 * (list.Count));
            return list[index];
        }
        void GenerateProducts() {
            products.Add(new Product("Tofu", 50));
            products.Add(new Product("Chocolade", 34));
            products.Add(new Product("Ikura", 70));
            products.Add(new Product("Chai", 3));
            products.Add(new Product("Boston Crab Meat", 36));
            products.Add(new Product("Ravioli Angelo", 18));
            products.Add(new Product("Ipon Coffee", 10));
            products.Add(new Product("Questo Cabrales", 25));
        }
        void GenerateCustomers() {
```

© 2018 Developer Express Inc.

```
            Customers.Add(new Customer("Nancy Davolio", 0));
            Customers.Add(new Customer("Andrew Fuller", 1));
            Customers.Add(new Customer("Janet Leverling", 2));
            Customers.Add(new Customer("Margaret Peacock", 3));
            Customers.Add(new Customer("Steven Buchanan", 4));
            Customers.Add(new Customer("Michael Suyama", 5));
            Customers.Add(new Customer("Robert King", 6));
            Customers.Add(new Customer("Laura Callahan", 7));
            Customers.Add(new Customer("Anne Dodsworth", 8));
        }
    }
}
```

**C#**

```
 (Product.cs)
using System;
namespace PickerColumnExample
{
    public class Product : ModelObject {
        string name;
        int unitPrice;
        public Product(string name, int unitPrice) {
            this.name = name;
            this.unitPrice = unitPrice;
        }
        public string Name {
            get { return name; }
            set { name = value; }
        }
        public int UnitPrice {
            get { return unitPrice; }
            set { unitPrice = value; }
        }
    }
}
```

**Xaml**

© 2018 Developer Express Inc.

```
 (MainPage.xaml)
<?xml version="1.0" encoding="UTF-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="PickerColumnExample.MainPage"
             xmlns:dxg="clr-namespace:DevExpress.Mobile.DataGrid;assembly=
    <ContentPage.Padding>
        <OnPlatform x:TypeArguments="Thickness" iOS="0, 20, 0, 0" />
    </ContentPage.Padding>
    <dxg:GridControl x:Name="grid" ItemsSource="{Binding Orders}">
         <dxg:GridControl.Columns>
             <dxg:TextColumn FieldName="Product.Name" Caption="Product" Wi
             <dxg:NumberColumn FieldName="Product.UnitPrice" Caption="Pric
             <dxg:NumberColumn FieldName="Quantity"/>
             <dxg:NumberColumn FieldName="Total"
                               UnboundType="Integer" UnboundExpression=
                               DisplayFormat="C0" IsReadOnly="True"/>
             <dxg:PickerColumn FieldName="CustomerId" Caption="Customer"
                               ItemsSource="{Binding Customers}" ValueMemb
         </dxg:GridControl.Columns>
    </dxg:GridControl>
</ContentPage>
```

## See Also

PickerColumn Class
PickerColumn Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# PopupMenuEventArgs Class

Provides data for the GridControl.PopupMenuCustomization event.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public class PopupMenuEventArgs : EventArgs
```

### Remarks

The **PopupMenuEventArgs** objects are automatically created, initialized and passed to the GridControl.PopupMenuCustomization event handlers.

### Example

The following sample code handles the GridControl.PopupMenuCustomization event to customize the data row popup menu before it is displayed. The code clears default menu items and adds new ones. One of these custom items applies a specific filter to the grid data and another one clears this filter.

| Xaml | |
|---|---|

```
<?xml version="1.0" encoding="UTF-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="HelloGrid.MainPage"
             xmlns:dxg="clr-namespace:DevExpress.Mobile.DataGrid;assembly=DevExpress.Mobile.Grid.v15.1">
    <ContentPage.Padding>
        <OnPlatform x:TypeArguments="Thickness" iOS="0, 20, 0, 0" />
    </ContentPage.Padding>
    <ContentPage.Content>
        <dxg:GridControl x:Name="grid" ItemsSource="{Binding Orders}"
                         PopupMenuCustomization="OnPopupMenuCustomization">
            <dxg:GridControl.Columns>
                <dxg:TextColumn FieldName="Product.Name" Caption="Product"/>
                <dxg:NumberColumn FieldName="Product.UnitPrice" Caption="Price" DisplayFormat="C0"/>
                <dxg:NumberColumn FieldName="Quantity" />
                <dxg:NumberColumn FieldName="Total"
                                  UnboundType="Decimal" UnboundExpression="[Quantity] * [Product.UnitPrice]
                                  IsReadOnly="True" DisplayFormat="C0"/>
            </dxg:GridControl.Columns>
    </dxg:GridControl>
    </ContentPage.Content>
</ContentPage>
```

**C#**

```
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.Core;
// ...
void OnPopupMenuCustomization(object sender, PopupMenuEventArgs e) {
    if (e.MenuType == GridPopupMenuType.DataRow) {
        e.Menu.Items.Clear();
        PopupMenuItem itemFilter = new PopupMenuItem();
        itemFilter.Caption = "Filter by Product";
        itemFilter.Click += ItemFilterClick;
        e.Menu.Items.Insert(0, itemFilter);
        PopupMenuItem itemClearFilter = new PopupMenuItem();
        itemClearFilter.Caption = "Clear Filter";
        itemClearFilter.Click += ItemClearFilterClick;
        e.Menu.Items.Insert(1, itemClearFilter);
    }
}
void ItemFilterClick(object sender, EventArgs e) {
    IRowData rowData = grid.GetRow(grid.SelectedRowHandle);
    Order selectedOrder = rowData.DataObject as Order;
    string productName = selectedOrder.Product.Name;
    grid.Columns ["Product.Name"].AutoFilterValue = productName;
}
void ItemClearFilterClick(object sender, EventArgs e) {
    grid.ClearFilter ();
}
```

### Inheritance Hierarchy

System.Object
  System.EventArgs
    **PopupMenuEventArgs**

### See Also

PopupMenuEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# PopupMenuEventArgs Members

Provides data for the GridControl.PopupMenuCustomization event.

The following tables list the members exposed by the PopupMenuEventArgs type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | PopupMenuEventArgs | Initializes a new instance of the PopupMenuEventArgs class with the specified settings. |

### Public Properties

| | Name | Description |
|---|---|---|
| | Column | Gets the column for which the popup menu will be shown. |
| | Menu | Provides access to the grid's popup menu to be shown. |
| | MenuType | Gets the type of the grid's menu to be invoked. |
| | RowHandle | Gets the handle of the row for which the popup menu will be shown. |

### Public Methods

| | Name | Description |
|---|---|---|
| S | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

### Public Fields

| | Name | Description |
|---|---|---|
| S | Empty | (Inherited from **System.EventArgs**) |

**See Also**
PopupMenuEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# PopupMenuEventArgs Constructor

Initializes a new instance of the PopupMenuEventArgs class with the specified settings.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public PopupMenuEventArgs(
   GridPopupMenuView menu,
   GridPopupMenuType menuType,
   int rowHandle,
   GridColumn column
)
```

### Parameters

*menu*
A DevExpress.Mobile.DataGrid.GridPopupMenuView object which represents the grid's popup menu. This value is assigned to the Menu property.
*menuType*
A GridPopupMenuType enumeration value specifying the type of the invoked grid's popup menu. This value is assigned to the MenuType property.
*rowHandle*
An integer value specifying the handle of the row for which an end-user invokes a popup menu. This value is assigned to the RowHandle property.
*column*
A GridColumn object or descendant specifying the column for which an end-user invokes a popup menu. This value is assigned to the Column property.

### Remarks

Instances of the PopupMenuEventArgs class are automatically created, initialized and passed to the corresponding event handlers.

### See Also

PopupMenuEventArgs Class
PopupMenuEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# PopupMenuEventArgs Properties

Provides data for the GridControl.PopupMenuCustomization event.

The following tables list the members exposed by the PopupMenuEventArgs type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | PopupMenuEventArgs | Initializes a new instance of the PopupMenuEventArgs class with the specified settings. |

### Public Properties

| | Name | Description |
|---|---|---|
| | Column | Gets the column for which the popup menu will be shown. |
| | Menu | Provides access to the grid's popup menu to be shown. |
| | MenuType | Gets the type of the grid's menu to be invoked. |
| | RowHandle | Gets the handle of the row for which the popup menu will be shown. |

### Public Methods

| | Name | Description |
|---|---|---|
| S | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

### Public Fields

| | Name | Description |
|---|---|---|
| S | Empty | (Inherited from **System.EventArgs**) |

### See Also

PopupMenuEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# Column Property

Gets the column for which the popup menu will be shown.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public GridColumn Column { get; }
```

### Property Value

A GridColumn object or descendant specifying the grid's column.

### Remarks

When an end-user touches and holds a grid's element (a data cell, a column header, a group row or total panel), a popup menu of the corresponding type (MenuType) is shown. To obtain for which column and row an end-user invokes a popup menu, use the **Column** and RowHandle properties.

### See Also

PopupMenuEventArgs Class
PopupMenuEventArgs Members
DevExpress.Mobile.DataGrid Namespace
How to: Customize Popup Menu

© 2018 Developer Express Inc.

# Menu Property

Provides access to the grid's popup menu to be shown.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public GridPopupMenuView Menu { get; }
```

### Property Value

A GridPopupMenuView object that is the menu that will be shown.

### Remarks

The **Menu** property allows you to add or remove items from the grid's popup menu. This can be accomplished by using methods accessible via the **Menu.Items** collection.

### Example

The following sample code handles the GridControl.PopupMenuCustomization event to customize the data row popup menu before it is displayed. The code clears default menu items and adds new ones. One of these custom items applies a specific filter to the grid data and another one clears this filter.

**Xaml**

```
<?xml version="1.0" encoding="UTF-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="HelloGrid.MainPage"
             xmlns:dxg="clr-namespace:DevExpress.Mobile.DataGrid;assembly=DevExpress.Mobile.Grid.v15.1">
    <ContentPage.Padding>
        <OnPlatform x:TypeArguments="Thickness" iOS="0, 20, 0, 0" />
    </ContentPage.Padding>
    <ContentPage.Content>
        <dxg:GridControl x:Name="grid" ItemsSource="{Binding Orders}"
                         PopupMenuCustomization="OnPopupMenuCustomization">
           <dxg:GridControl.Columns>
               <dxg:TextColumn FieldName="Product.Name" Caption="Product"/>
               <dxg:NumberColumn FieldName="Product.UnitPrice" Caption="Price" DisplayFormat="C0"/>
               <dxg:NumberColumn FieldName="Quantity" />
               <dxg:NumberColumn FieldName="Total"
                                 UnboundType="Decimal" UnboundExpression="[Quantity] * [Product.UnitPrice]
                                 IsReadOnly="True" DisplayFormat="C0"/>
           </dxg:GridControl.Columns>
    </dxg:GridControl>
    </ContentPage.Content>
</ContentPage>
```

**C#**

```
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.Core;
// ...
void OnPopupMenuCustomization(object sender, PopupMenuEventArgs e) {
    if (e.MenuType == GridPopupMenuType.DataRow) {
        e.Menu.Items.Clear();
        PopupMenuItem itemFilter = new PopupMenuItem();
        itemFilter.Caption = "Filter by Product";
        itemFilter.Click += ItemFilterClick;
        e.Menu.Items.Insert(0, itemFilter);
        PopupMenuItem itemClearFilter = new PopupMenuItem();
        itemClearFilter.Caption = "Clear Filter";
        itemClearFilter.Click += ItemClearFilterClick;
        e.Menu.Items.Insert(1, itemClearFilter);
    }
}
void ItemFilterClick(object sender, EventArgs e) {
    IRowData rowData = grid.GetRow(grid.SelectedRowHandle);
    Order selectedOrder = rowData.DataObject as Order;
    string productName = selectedOrder.Product.Name;
    grid.Columns ["Product.Name"].AutoFilterValue = productName;
}
void ItemClearFilterClick(object sender, EventArgs e) {
    grid.ClearFilter ();
}
```

**See Also**

PopupMenuEventArgs Class
PopupMenuEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# MenuType Property

Gets the type of the grid's menu to be invoked.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public GridPopupMenuType MenuType { get; }
```

### Property Value

A GridPopupMenuType enumeration value that identifies the grid's popup menu to be invoked.

### Example

The following sample code handles the GridControl.PopupMenuCustomization event to customize the data row popup menu before it is displayed. The code clears default menu items and adds new ones. One of these custom items applies a specific filter to the grid data and another one clears this filter.

| Xaml | |
|---|---|

```
<?xml version="1.0" encoding="UTF-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="HelloGrid.MainPage"
             xmlns:dxg="clr-namespace:DevExpress.Mobile.DataGrid;assembly=DevExpress.Mobile.Grid.v15.1">
    <ContentPage.Padding>
        <OnPlatform x:TypeArguments="Thickness" iOS="0, 20, 0, 0" />
    </ContentPage.Padding>
    <ContentPage.Content>
        <dxg:GridControl x:Name="grid" ItemsSource="{Binding Orders}"
                         PopupMenuCustomization="OnPopupMenuCustomization">
            <dxg:GridControl.Columns>
                <dxg:TextColumn FieldName="Product.Name" Caption="Product"/>
                <dxg:NumberColumn FieldName="Product.UnitPrice" Caption="Price" DisplayFormat="C0"/>
                <dxg:NumberColumn FieldName="Quantity" />
                <dxg:NumberColumn FieldName="Total"
                                  UnboundType="Decimal" UnboundExpression="[Quantity] * [Product.UnitPrice]
                                  IsReadOnly="True" DisplayFormat="C0"/>
            </dxg:GridControl.Columns>
    </dxg:GridControl>
    </ContentPage.Content>
</ContentPage>
```

**C#**

```
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.Core;
// ...
void OnPopupMenuCustomization(object sender, PopupMenuEventArgs e) {
    if (e.MenuType == GridPopupMenuType.DataRow) {
        e.Menu.Items.Clear();
        PopupMenuItem itemFilter = new PopupMenuItem();
        itemFilter.Caption = "Filter by Product";
        itemFilter.Click += ItemFilterClick;
        e.Menu.Items.Insert(0, itemFilter);
        PopupMenuItem itemClearFilter = new PopupMenuItem();
        itemClearFilter.Caption = "Clear Filter";
        itemClearFilter.Click += ItemClearFilterClick;
        e.Menu.Items.Insert(1, itemClearFilter);
    }
}
void ItemFilterClick(object sender, EventArgs e) {
    IRowData rowData = grid.GetRow(grid.SelectedRowHandle);
    Order selectedOrder = rowData.DataObject as Order;
    string productName = selectedOrder.Product.Name;
    grid.Columns ["Product.Name"].AutoFilterValue = productName;
}
void ItemClearFilterClick(object sender, EventArgs e) {
    grid.ClearFilter ();
}
```

**See Also**

PopupMenuEventArgs Class
PopupMenuEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# RowHandle Property

Gets the handle of the row for which the popup menu will be shown.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public int RowHandle { get; }
```

### Property Value

An integer value that specifies the row handle.

### Remarks

When an end-user touches and holds a grid's element (a data cell, a column header, a group row or total panel), a popup menu of the corresponding type (MenuType) is shown. To obtain for which row and column an end-user invokes a popup menu, use the **RowHandle** and Column properties.

### See Also

PopupMenuEventArgs Class
PopupMenuEventArgs Members
DevExpress.Mobile.DataGrid Namespace
How to: Customize Popup Menu

© 2018 Developer Express Inc.

# PopupMenuEventHandler Delegate

Represents a method that will handle the GridControl.PopupMenuCustomization event.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public delegate void PopupMenuEventHandler(
   object sender,
   PopupMenuEventArgs e
)
```

### Parameters

The declaration of your event handler must have the same parameters as the **PopupMenuEventHandler** delegate declaration.
*sender*
The event source. Identifies the GridControl object that raised the event.
*e*
A PopupMenuEventArgs object that contains event data.

### See Also

DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# RowAllowEventArgs Class

Provides data for the GridControl.GroupRowExpanding and GridControl.GroupRowCollapsing events.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public class RowAllowEventArgs : RowEventArgs
```

### Remarks

The GridControl.GroupRowExpanding and GridControl.GroupRowCollapsing events fire when a group row is about to be expanded or collapsed respectively. They enable you to prohibit changing the row's expansion state. The RowAllowEventArgs class introduces the Allow property for this purpose. The inherited RowEventArgs.RowHandle property allows you to identify the row whose expansion state is about to be changed. RowEventArgs objects are automatically created, initialized and passed to GridControl.GroupRowExpanding and GridControl.GroupRowCollapsing event handlers.

### Inheritance Hierarchy

System.Object
  System.EventArgs
    RowEventArgs
      **RowAllowEventArgs**

### See Also

RowAllowEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# RowAllowEventArgs Members

Provides data for the GridControl.GroupRowExpanding and GridControl.GroupRowCollapsing events.

The following tables list the members exposed by the RowAllowEventArgs type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | RowAllowEventArgs | Initializes a new instance of the RowAllowEventArgs class. |

### Public Properties

| | Name | Description |
|---|---|---|
| | Allow | Gets or sets a value specifying whether the current operation (group row expanding or collapsing) is allowed. |
| | RowHandle | Gets the row handle. (Inherited from **RowEventArgs**) |

### Public Methods

| | Name | Description |
|---|---|---|
| S | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

### Public Fields

| | Name | Description |
|---|---|---|
| S | Empty | (Inherited from **System.EventArgs**) |

**See Also**
RowAllowEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# RowAllowEventArgs Constructor

Initializes a new instance of the RowAllowEventArgs class.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public RowAllowEventArgs(
   int rowHandle
)
```

### Parameters

*rowHandle*
An integer value that specifies the processed row's handle. This value is assigned to the RowEventArgs.RowHandle property.

### See Also

RowAllowEventArgs Class
RowAllowEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# RowAllowEventArgs Properties

Provides data for the GridControl.GroupRowExpanding and GridControl.GroupRowCollapsing events.

The following tables list the members exposed by the RowAllowEventArgs type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | RowAllowEventArgs | Initializes a new instance of the RowAllowEventArgs class. |

### Public Properties

| | Name | Description |
|---|---|---|
| | Allow | Gets or sets a value specifying whether the current operation (group row expanding or collapsing) is allowed. |
| | RowHandle | Gets the row handle. (Inherited from **RowEventArgs**) |

### Public Methods

| | Name | Description |
|---|---|---|
| | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

### Public Fields

| | Name | Description |
|---|---|---|
| | Empty | (Inherited from **System.EventArgs**) |

**See Also**
RowAllowEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# Allow Property

Gets or sets a value specifying whether the current operation (group row expanding or collapsing) is allowed.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public bool Allow { get; set; }
```

### Property Value

**true**, if the operation is allowed; otherwise, **false**.

### Remarks

By default, the operation is allowed (so, the property value is set to **true**). To disable group row expanding or collapsing, set the **Allow** property to **false** in the GridControl.GroupRowExpanding or GridControl.GroupRowCollapsing event handler respectively.

### See Also

RowAllowEventArgs Class
RowAllowEventArgs Members
DevExpress.Mobile.DataGrid Namespace
GridControl.CollapseAllGroups
GridControl.CollapseGroupRow
GridControl.ExpandAllGroups
GridControl.ExpandGroupRow

© 2018 Developer Express Inc.

# RowAllowEventHandler Delegate

Represents a method that will handle the GridControl.GroupRowExpanding and GridControl.GroupRowCollapsed events.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public delegate void RowAllowEventHandler(
   object sender,
   RowAllowEventArgs e
)
```

### Parameters

The declaration of your event handler must have the same parameters as the **RowAllowEventHandler** delegate declaration.
*sender*
The event source. Identifies the View that raised the event.
*e*
A RowAllowEventArgs object that contains event data.

### Remarks

When you create a **RowAllowEventHandler** delegate, you identify the method that will handle the event. To associate the event with your event handler, add an instance of the delegate to the event. The event handler is called whenever the event occurs, unless you remove the delegate. For more information about event handler delegates, see **Events and Delegates** in **MSDN**.

### See Also

DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# RowEditingEventArgs Class

Provides data for the GridControl.EndRowEdit event.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public class RowEditingEventArgs : SourceRowEventArgs
```

### Remarks

The **RowEditingEventArgs** objects are automatically created, initialized and passed to the GridControl.EndRowEdit event handlers.

### Inheritance Hierarchy

System.Object
  System.EventArgs
    RowEventArgs
      SourceRowEventArgs
        **RowEditingEventArgs**

### See Also

RowEditingEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# RowEditingEventArgs Members

Provides data for the GridControl.EndRowEdit event.

The following tables list the members exposed by the RowEditingEventArgs type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | RowEditingEventArgs | Initializes a new instance of the RowEditingEventArgs class. |

### Public Properties

| | Name | Description |
|---|---|---|
| | Action | Gets whether an end-user has closed a row editor to apply or cancel modified cell values. |
| | RowHandle | Gets the row handle. (Inherited from **RowEventArgs**) |
| | SourceRowIndex | Gets the index of the record in a data source to which the processed row corresponds. (Inherited from **SourceRowEventArgs**) |

### Public Methods

| | Name | Description |
|---|---|---|
| | Equals | (Inherited from **System.Object**) |
| S | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

### Public Fields

| | Name | Description |
|---|---|---|
| S | Empty | (Inherited from **System.EventArgs**) |

**See Also**
RowEditingEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# RowEditingEventArgs Constructor

Initializes a new instance of the RowEditingEventArgs class.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public RowEditingEventArgs(
   int rowHandle,
   int sourceRowIndex,
   EditingRowAction action
)
```

### Parameters

*rowHandle*
An integer value that specifies the row handle. This value is assigned to the **RowHandle** property.
*sourceRowIndex*
An integer value that specifies the row index in the data source. This value is assigned to the **SourceRowIndex** property.
*action*
An EditingRowAction enumeration value specifying how the data row editor has been closed. This value is assigned to the Action property.

### Remarks

Instances of the RowEditingEventArgs class are automatically created, initialized and passed to the corresponding event handlers.

### See Also

RowEditingEventArgs Class
RowEditingEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# RowEditingEventArgs Properties

Provides data for the GridControl.EndRowEdit event.

The following tables list the members exposed by the RowEditingEventArgs type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | RowEditingEventArgs | Initializes a new instance of the RowEditingEventArgs class. |

### Public Properties

| | Name | Description |
|---|---|---|
| | Action | Gets whether an end-user has closed a row editor to apply or cancel modified cell values. |
| | RowHandle | Gets the row handle. (Inherited from **RowEventArgs**) |
| | SourceRowIndex | Gets the index of the record in a data source to which the processed row corresponds. (Inherited from **SourceRowEventArgs**) |

### Public Methods

| | Name | Description |
|---|---|---|
| | Equals | (Inherited from **System.Object**) |
| S | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

### Public Fields

| | Name | Description |
|---|---|---|
| S | Empty | (Inherited from **System.EventArgs**) |

**See Also**
RowEditingEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# Action Property

Gets whether an end-user has closed a row editor to apply or cancel modified cell values.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public EditingRowAction Action { get; }
```

### Property Value

An EditingRowAction enumeration value that specifies whether an end-user saved or discarded changed cell values.

### See Also

RowEditingEventArgs Class
RowEditingEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# RowEditingEventHandler Delegate

Represents a method that will handle the GridControl.EndRowEdit event.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public delegate void RowEditingEventHandler(
   object sender,
   RowEditingEventArgs e
)
```

### Parameters

The declaration of your event handler must have the same parameters as the **RowEditingEventHandler** delegate declaration.
*sender*
The event source. Identifies the GridControl object that raised the event.
*e*
An EditingRowAction object that contains event data.

### See Also

DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# RowEditMode Enumeration

Lists values that specify the GridControl's editing mode.

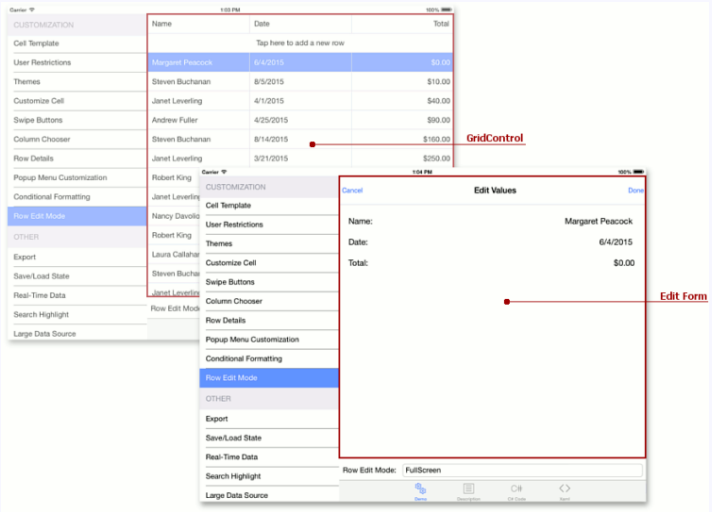
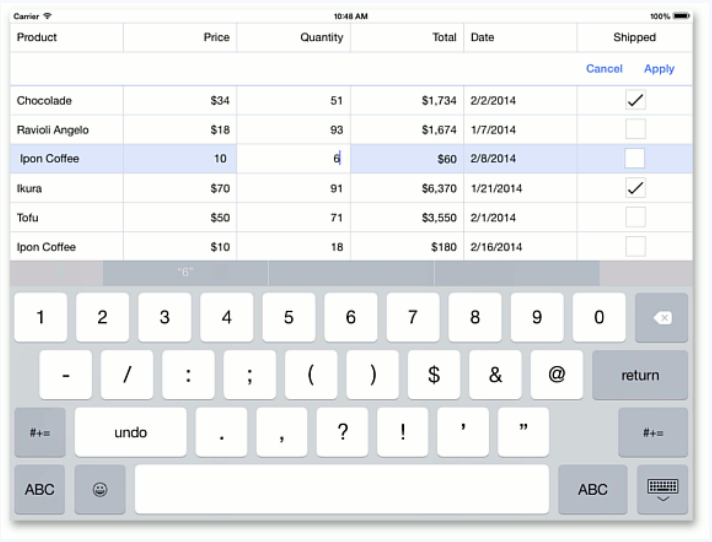
**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

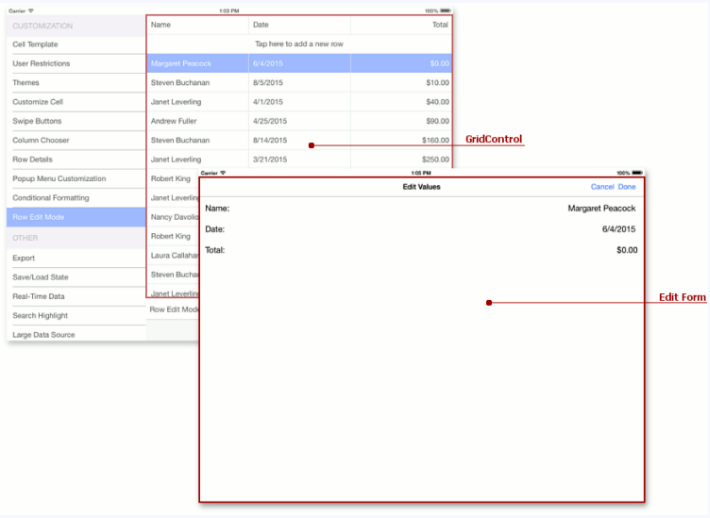
### Syntax

**C#**

```
public enum RowEditMode
```

### Members

| Name | Description |
|---|---|
| GridArea | Cell values are edited via an edit form displayed over the entire grid area.<br>  |
| Inplace | Cell values are edited within the in-line edit row.<br>  |
| Popup | Cell values are edited via the popup edit form. |

| | |
|---|---|
| | |
| ScreenArea | Cell values are edited via an edit form placed on a separate NavigationPage and displayed in the full screen mode with the navigation bar at the top.<br>  |

**Remarks**

Values listed by this enumeration are used to set the GridControl.RowEditMode property value.

**See Also**

DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# RowEventArgs Class

Provides data for the GridControl.RowTap event.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public class RowEventArgs : EventArgs
```

### Remarks

The **RowEventArgs** objects are automatically created, initialized and passed to the GridControl.RowTap event handlers.

### Inheritance Hierarchy

System.Object
  System.EventArgs
    **RowEventArgs**
      Derived classes

### See Also

RowEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# RowEventArgs Members

Provides data for the GridControl.RowTap event.

The following tables list the members exposed by the RowEventArgs type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | RowEventArgs | Initializes a new instance of the RowEventArgs class. |

### Public Properties

| | Name | Description |
|---|---|---|
| | RowHandle | Gets the row handle. |

### Public Methods

| | Name | Description |
|---|---|---|
| | Equals | (Inherited from **System.Object**) |
| S | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

### Public Fields

| | Name | Description |
|---|---|---|
| S | Empty | (Inherited from **System.EventArgs**) |

**See Also**
RowEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# RowEventArgs Constructor

Initializes a new instance of the RowEventArgs class.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public RowEventArgs(
   int rowHandle
)
```

### Parameters

*rowHandle*
An integer value that specifies the row handle. This value is assigned to the RowHandle property.

### Remarks

Instances of the RowEventArgs class are automatically created, initialized and passed to the corresponding event handlers.

### See Also

RowEventArgs Class
RowEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# RowEventArgs Properties

Provides data for the GridControl.RowTap event.

The following tables list the members exposed by the RowEventArgs type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | RowEventArgs | Initializes a new instance of the RowEventArgs class. |

### Public Properties

| | Name | Description |
|---|---|---|
| | RowHandle | Gets the row handle. |

### Public Methods

| | Name | Description |
|---|---|---|
| | Equals | (Inherited from **System.Object**) |
| S | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

### Public Fields

| | Name | Description |
|---|---|---|
| S | Empty | (Inherited from **System.EventArgs**) |

**See Also**
RowEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# RowHandle Property

Gets the row handle.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public int RowHandle { get; }
```

### Property Value

An integer value that specifies the row handle.

### Example

**Show Me**

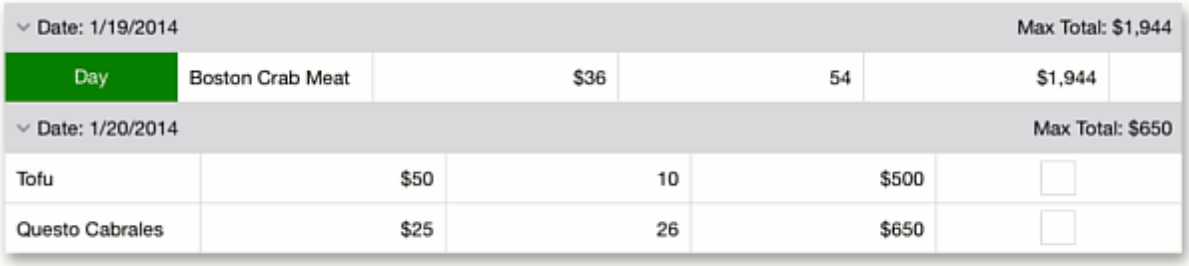
A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T178764 .

This example shows how to extend the grid's UI with additional buttons to be shown when an end-user swipes left to right, or right to left, over a data row. To create a swipe button, add a SwipeButtonInfo object to the GridControl.LeftSwipeButtons or GridControl.RightSwipeButtons collection. Use this object's SwipeButtonInfo.Caption, SwipeButtonInfo.Width, SwipeButtonInfo.BackgroundColor and SwipeButtonInfo.TextColor properties to specify button appearance.

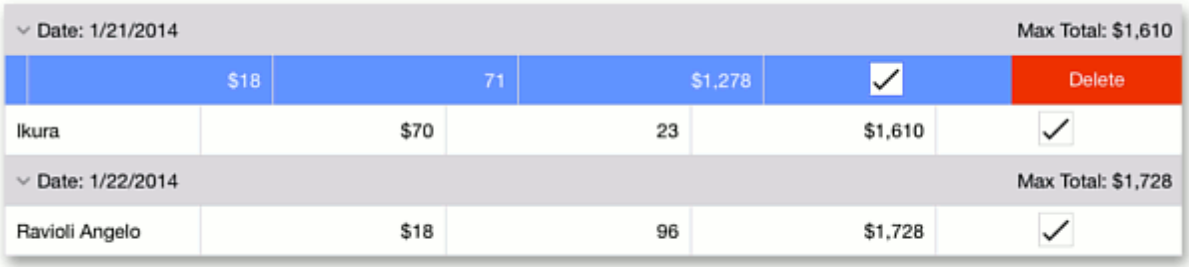
To assign a custom action to a button, handle the GridControl.SwipeButtonClick event. The GridControl.SwipeButtonShowing event allows you to specify when a swipe button should be visible.

In the example, two swipe buttons are created.

- When an end-user swipes any data row left to right, the **Day** button is shown on the left side of the row. Tapping this button displays the day of the week when the corresponding order was made.




- When an end-user swipes right to left, a data row which corresponds to an order that is already shipped (the value in the **Shipped** column is **true**), the **Delete** button is shown on the right side of the row. Tapping this button removes the corresponding data row.




**C#**

```
 (MainPage.xaml.cs)
void OnSwipeButtonShowing(object sender, SwipeButtonShowingEventArgs e)
{
    if ((!(Boolean)grid.GetCellValue(e.RowHandle, "Shipped"))
        && (e.ButtonInfo.ButtonName == "RightButton"))
    {
        e.IsVisible = false;
    }
}
void OnSwipeButtonClick(object sender, SwipeButtonEventArgs e)
{
    if (e.ButtonInfo.ButtonName == "LeftButton")
    {
        DateTime orderDate = (DateTime)grid.GetCellValue(e.RowHandle, "Date");
        string orderDateDay = orderDate.ToString("dddd");
        DisplayAlert("Alert from " + e.ButtonInfo.ButtonName, "Day: " + orderDateDay, "OK");
    }
    if (e.ButtonInfo.ButtonName == "RightButton")
    {
        grid.DeleteRow(e.RowHandle);
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl.LeftSwipeButtons>
  <dxGrid:SwipeButtonInfo ButtonName="LeftButton" Caption="Day" BackgroundColor="Green" Width="150"/>
</dxGrid:GridControl.LeftSwipeButtons>
<dxGrid:GridControl.RightSwipeButtons>
  <dxGrid:SwipeButtonInfo ButtonName="RightButton" Caption="Delete" BackgroundColor="Red" Width="150"/>
</dxGrid:GridControl.RightSwipeButtons>
```

## See Also

RowEventArgs Class
RowEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# RowEventHandler Delegate

Represents a method that will handle the GridControl.RowTap event.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public delegate void RowEventHandler(
   object sender,
   RowEventArgs e
)
```

### Parameters

The declaration of your event handler must have the same parameters as the **RowEventHandler** delegate declaration.
*sender*
The event source. Identifies the GridControl object that raised the event.
*e*
A RowEventArgs class instance that contains event data.

### See Also

DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# RowTapEventArgs Class

Provides data for the GridControl.RowTap event.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public class RowTapEventArgs : RowEventArgs
```

### Remarks

The **RowTapEventArgs** objects are automatically created, initialized and passed to the GridControl.RowTap event handlers.

### Inheritance Hierarchy

System.Object
  System.EventArgs
    RowEventArgs
      **RowTapEventArgs**

### See Also

RowTapEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# RowTapEventArgs Members

Provides data for the GridControl.RowTap event.

The following tables list the members exposed by the RowTapEventArgs type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | RowTapEventArgs | Initializes a new instance of the RowTapEventArgs class. |

### Public Properties

| | Name | Description |
|---|---|---|
| | FieldName | Gets the field name of a column to which the tapped cell belongs. |
| | RowHandle | Gets the row handle. (Inherited from **RowEventArgs**) |

### Public Methods

| | Name | Description |
|---|---|---|
| S | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

### Public Fields

| | Name | Description |
|---|---|---|
| S | Empty | (Inherited from **System.EventArgs**) |

**See Also**
RowTapEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# RowTapEventArgs Constructor

Initializes a new instance of the RowTapEventArgs class.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public RowTapEventArgs(
   int rowHandle,
   string fieldName
)
```

### Parameters

*rowHandle*
An integer value that specifies the row handle. This value is assigned to the RowEventArgs.RowHandle property.
*fieldName*
A string value that specifies the column name. This value is assigned to the FieldName property.

### Remarks

Instances of the RowTapEventArgs class are automatically created, initialized and passed to the corresponding event handlers.

### See Also

RowTapEventArgs Class
RowTapEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# RowTapEventArgs Properties

Provides data for the GridControl.RowTap event.

The following tables list the members exposed by the RowTapEventArgs type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | RowTapEventArgs | Initializes a new instance of the RowTapEventArgs class. |

### Public Properties

| | Name | Description |
|---|---|---|
| | FieldName | Gets the field name of a column to which the tapped cell belongs. |
| | RowHandle | Gets the row handle. (Inherited from **RowEventArgs**) |

### Public Methods

| | Name | Description |
|---|---|---|
| S | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

### Public Fields

| | Name | Description |
|---|---|---|
| S | Empty | (Inherited from **System.EventArgs**) |

**See Also**
RowTapEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# FieldName Property

Gets the field name of a column to which the tapped cell belongs.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public string FieldName { get; }
```

### Property Value

A System.String value that specifies a column's field name.

### See Also

RowTapEventArgs Class
RowTapEventArgs Members
DevExpress.Mobile.DataGrid Namespace
GridColumn.FieldName

© 2018 Developer Express Inc.

# RowTapEventHandler Delegate

Represents a method that will handle the GridControl.RowTap event.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public delegate void RowTapEventHandler(
   object sender,
   RowTapEventArgs e
)
```

### Parameters

The declaration of your event handler must have the same parameters as the **RowTapEventHandler** delegate declaration.

*sender*
The event source. Identifies the GridControl object that raised the event.

*e*
A RowTapEventArgs class instance that contains event data.

### See Also

DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# SourceRowEventArgs Class

Provides data for the GridControl.EndRowEdit event.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public class SourceRowEventArgs : RowEventArgs
```

### Remarks

The **SourceRowEventArgs** objects are automatically created, initialized and passed to the GridControl.EndRowEdit event handlers.

### Inheritance Hierarchy

System.Object
  System.EventArgs
    RowEventArgs
      **SourceRowEventArgs**
        Derived classes

### See Also

SourceRowEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# SourceRowEventArgs Members

Provides data for the GridControl.EndRowEdit event.

The following tables list the members exposed by the SourceRowEventArgs type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | SourceRowEventArgs | Initializes a new instance of the SourceRowEventArgs class. |

### Public Properties

| | Name | Description |
|---|---|---|
| | RowHandle | Gets the row handle. (Inherited from **RowEventArgs**) |
| | SourceRowIndex | Gets the index of the record in a data source to which the processed row corresponds. |

### Public Methods

| | Name | Description |
|---|---|---|
| S | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

### Public Fields

| | Name | Description |
|---|---|---|
| S | Empty | (Inherited from **System.EventArgs**) |

**See Also**
SourceRowEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# SourceRowEventArgs Constructor

Initializes a new instance of the SourceRowEventArgs class.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public SourceRowEventArgs(
   int rowHandle,
   int sourceRowIndex
)
```

### Parameters

*rowHandle*
An integer value that specifies the row handle. This value is assigned to the **RowHandle** property.
*sourceRowIndex*
An integer value that specifies the row index in the data source. This value is assigned to the SourceRowIndex property.

### Remarks

Instances of the SourceRowEventArgs class are automatically created, initialized and passed to the corresponding event handlers.

### See Also

SourceRowEventArgs Class
SourceRowEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# SourceRowEventArgs Properties

Provides data for the GridControl.EndRowEdit event.

The following tables list the members exposed by the SourceRowEventArgs type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | SourceRowEventArgs | Initializes a new instance of the SourceRowEventArgs class. |

### Public Properties

| | Name | Description |
|---|---|---|
| | RowHandle | Gets the row handle. (Inherited from **RowEventArgs**) |
| | SourceRowIndex | Gets the index of the record in a data source to which the processed row corresponds. |

### Public Methods

| | Name | Description |
|---|---|---|
| S | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

### Public Fields

| | Name | Description |
|---|---|---|
| S | Empty | (Inherited from **System.EventArgs**) |

**See Also**
SourceRowEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# SourceRowIndex Property

Gets the index of the record in a data source to which the processed row corresponds.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public int SourceRowIndex { get; }
```

### Property Value

A zero-based integer value that specifies the index of the record in a data source to which the processed row corresponds.

### See Also

SourceRowEventArgs Class
SourceRowEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# SourceRowEventHandler Delegate

Represents a method that will handle the GridControl.EndRowEdit event.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public delegate void SourceRowEventHandler(
   object sender,
   SourceRowEventArgs e
)
```

### Parameters

The declaration of your event handler must have the same parameters as the **SourceRowEventHandler** delegate declaration.
*sender*
The event source. Identifies the GridControl object that raised the event.
*e*
A SourceRowEventArgs object that contains event data.

### See Also

DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# SummaryType Enumeration

Lists values specifying the aggregate function types.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public enum SummaryType
```

### Members

| Name | Description |
|---|---|
| Average | The average value of a column. |
| Count | The count of records. |
| Custom | Specifies whether calculations should be performed manually using a specially designed event. |
| Max | The maximum value in a column. |
| Min | The minimum value in a column. |
| None | Disables summary value calculation. |
| Sum | The sum of all values in a column. |

### Remarks

The **SummaryType** enumeration values are used to specify the aggregate function to calculate summary values. These values are assigned to the GridColumnSummary.Type property.

### See Also

DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# SwipeButtonEventArgs Class

Provides data for the GridControl.SwipeButtonClick event.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public class SwipeButtonEventArgs : SourceRowEventArgs
```

### Remarks

The GridControl.SwipeButtonClick event is raised when an end-user taps a swipe button in the grid. You can handle this event to define an action to be performed when a button is tapped. The ButtonInfo property allows you to obtain information on the currently processed button. The **RowHandle** parameter returns a row handle for which a swipe button is shown.

The **SwipeButtonEventArgs** objects are automatically created, initialized and passed to the GridControl.SwipeButtonClick event handlers.

### Example

**Show Me**

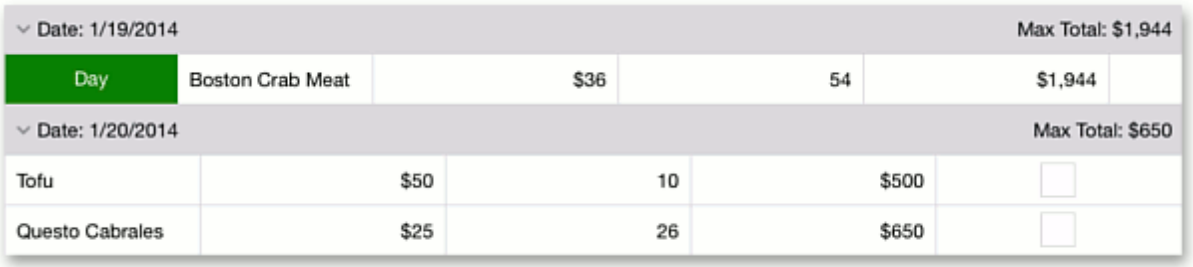
A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T178764 .

This example shows how to extend the grid's UI with additional buttons to be shown when an end-user swipes left to right, or right to left, over a data row. Swipe Buttons. To create a swipe button, add a SwipeButtonInfo object to the GridControl.LeftSwipeButtons or GridControl.RightSwipeButtons collection. Use this object's SwipeButtonInfo.Caption, SwipeButtonInfo.Width, SwipeButtonInfo.BackgroundColor and SwipeButtonInfo.TextColor properties to specify button appearance.

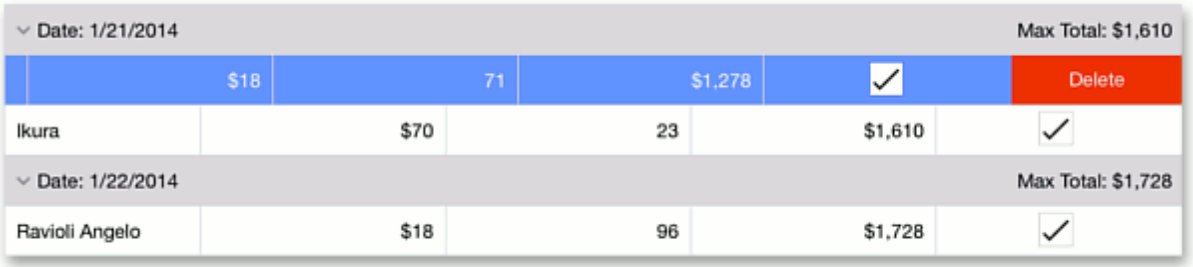
To assign a custom action to a button, handle the GridControl.SwipeButtonClick event. The GridControl.SwipeButtonShowing event allows you to specify when a swipe button should be visible.

In the example, two swipe buttons are created.

- When an end-user swipes any data row left to right, the **Day** button is shown on the left side of the row. Tapping this button displays the day of the week when the corresponding order was made.




- When an end-user swipes right to left, a data row which corresponds to an order that is already shipped (the value in the **Shipped** column is **true**), the **Delete** button is shown on the right side of the row. Tapping this button removes the corresponding data row.




**C#**

© 2018 Developer Express Inc.

```
 (MainPage.xaml.cs)
void OnSwipeButtonShowing(object sender, SwipeButtonShowingEventArgs e)
{
    if ((!(Boolean)grid.GetCellValue(e.RowHandle, "Shipped"))
        && (e.ButtonInfo.ButtonName == "RightButton"))
    {
        e.IsVisible = false;
    }
}
void OnSwipeButtonClick(object sender, SwipeButtonEventArgs e)
{
    if (e.ButtonInfo.ButtonName == "LeftButton")
    {
        DateTime orderDate = (DateTime)grid.GetCellValue(e.RowHandle, "Date");
        string orderDateDay = orderDate.ToString("dddd");
        DisplayAlert("Alert from " + e.ButtonInfo.ButtonName, "Day: " + orderDateDay, "OK");
    }
    if (e.ButtonInfo.ButtonName == "RightButton")
    {
        grid.DeleteRow(e.RowHandle);
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl.LeftSwipeButtons>
  <dxGrid:SwipeButtonInfo ButtonName="LeftButton" Caption="Day" BackgroundColor="Green" Width="150"/>
</dxGrid:GridControl.LeftSwipeButtons>
<dxGrid:GridControl.RightSwipeButtons>
  <dxGrid:SwipeButtonInfo ButtonName="RightButton" Caption="Delete" BackgroundColor="Red" Width="150"/>
</dxGrid:GridControl.RightSwipeButtons>
```

## Inheritance Hierarchy

System.Object
    System.EventArgs
        RowEventArgs
            SourceRowEventArgs
                **SwipeButtonEventArgs**
                    Derived classes

## See Also

SwipeButtonEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# SwipeButtonEventArgs Members

Provides data for the GridControl.SwipeButtonClick event.

The following tables list the members exposed by the SwipeButtonEventArgs type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | SwipeButtonEventArgs | Initializes a new instance of the SwipeButtonEventArgs class. |

### Public Properties

| | Name | Description |
|---|---|---|
| | ButtonInfo | Gets an object that stores information about a swipe button. |
| | RowHandle | Gets the row handle. (Inherited from **RowEventArgs**) |
| | SourceRowIndex | Gets the index of the record in a data source to which the processed row corresponds. (Inherited from **SourceRowEventArgs**) |

### Public Methods

| | Name | Description |
|---|---|---|
| | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

### Public Fields

| | Name | Description |
|---|---|---|
| | Empty | (Inherited from **System.EventArgs**) |

**See Also**
SwipeButtonEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# SwipeButtonEventArgs Constructor

Initializes a new instance of the SwipeButtonEventArgs class.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public SwipeButtonEventArgs(
   SwipeButtonInfo buttonInfo,
   int rowHandle,
   int sourceRowIndex
)
```

### Parameters

*buttonInfo*
A SwipeButtonInfo object that contains information on the swipe button. This value is assigned to the ButtonInfo property.
*rowHandle*
An integer value that specifies the row handle. This value is assigned to the **RowHandle** property.
*sourceRowIndex*
An integer value that specifies the row index in the data source. This value is assigned to the **SourceRowIndex** property.

### Remarks

Instances of the SwipeButtonEventArgs class are automatically created, initialized and passed to the corresponding event handlers.

### See Also

SwipeButtonEventArgs Class
SwipeButtonEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# SwipeButtonEventArgs Properties

Provides data for the GridControl.SwipeButtonClick event.

The following tables list the members exposed by the SwipeButtonEventArgs type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | SwipeButtonEventArgs | Initializes a new instance of the SwipeButtonEventArgs class. |

### Public Properties

| | Name | Description |
|---|---|---|
| | ButtonInfo | Gets an object that stores information about a swipe button. |
| | RowHandle | Gets the row handle. (Inherited from **RowEventArgs**) |
| | SourceRowIndex | Gets the index of the record in a data source to which the processed row corresponds. (Inherited from **SourceRowEventArgs**) |

### Public Methods

| | Name | Description |
|---|---|---|
| | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

### Public Fields

| | Name | Description |
|---|---|---|
| | Empty | (Inherited from **System.EventArgs**) |

**See Also**
SwipeButtonEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# ButtonInfo Property

Gets an object that stores information about a swipe button.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public SwipeButtonInfo ButtonInfo { get; }
```

### Property Value

A SwipeButtonInfo object.

### Remarks

When handling the GridControl.SwipeButtonClick or GridControl.SwipeButtonShowing event, you can use properties of the SwipeButtonInfo object to obtain the button name (SwipeButtonInfo.ButtonName), or button appearance settings (SwipeButtonInfo.Caption, SwipeButtonInfo.BackgroundColor, SwipeButtonInfo.TextColor or SwipeButtonInfo.Width).

### Example

**Show Me**

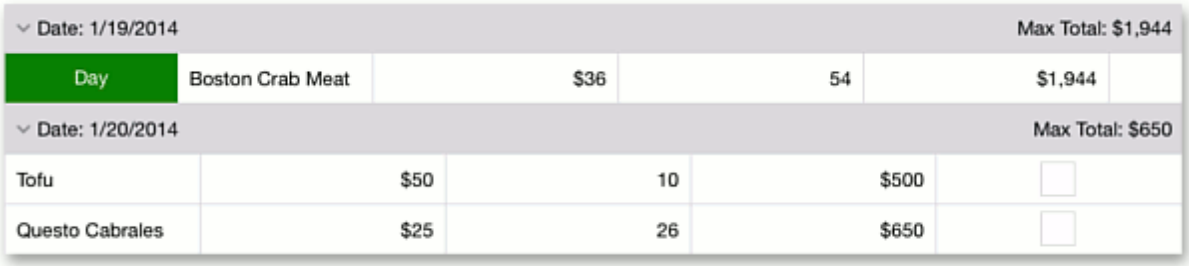
A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T178764 .

This example shows how to extend the grid's UI with additional buttons to be shown when an end-user swipes left to right, or right to left, over a data row. Swipe Buttons. To create a swipe button, add a SwipeButtonInfo object to the GridControl.LeftSwipeButtons or GridControl.RightSwipeButtons collection. Use this object's SwipeButtonInfo.Caption, SwipeButtonInfo.Width, SwipeButtonInfo.BackgroundColor and SwipeButtonInfo.TextColor properties to specify button appearance.

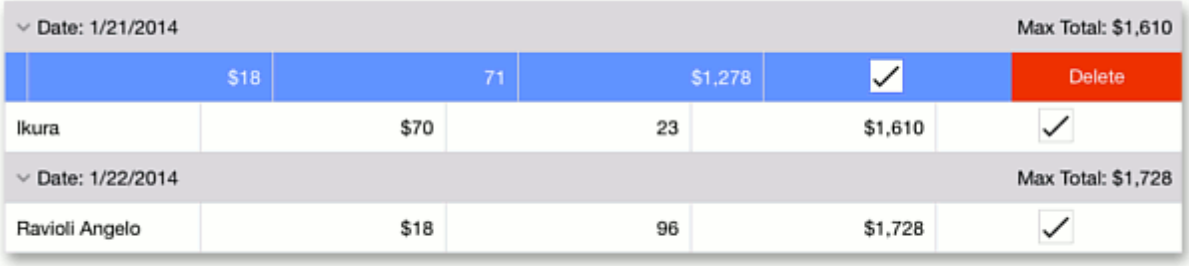
To assign a custom action to a button, handle the GridControl.SwipeButtonClick event. The GridControl.SwipeButtonShowing event allows you to specify when a swipe button should be visible.

In the example, two swipe buttons are created.

- When an end-user swipes any data row left to right, the **Day** button is shown on the left side of the row. Tapping this button displays the day of the week when the corresponding order was made.




- When an end-user swipes right to left, a data row which corresponds to an order that is already shipped (the value in the **Shipped** column is **true**), the **Delete** button is shown on the right side of the row. Tapping this button removes the corresponding data row.




© 2018 Developer Express Inc.

**C#**

```
 (MainPage.xaml.cs)
void OnSwipeButtonShowing(object sender, SwipeButtonShowingEventArgs e)
{
    if ((!(Boolean)grid.GetCellValue(e.RowHandle, "Shipped"))
        && (e.ButtonInfo.ButtonName == "RightButton"))
    {
        e.IsVisible = false;
    }
}
void OnSwipeButtonClick(object sender, SwipeButtonEventArgs e)
{
    if (e.ButtonInfo.ButtonName == "LeftButton")
    {
        DateTime orderDate = (DateTime)grid.GetCellValue(e.RowHandle, "Date");
        string orderDateDay = orderDate.ToString("dddd");
        DisplayAlert("Alert from " + e.ButtonInfo.ButtonName, "Day: " + orderDateDay, "OK");
    }
    if (e.ButtonInfo.ButtonName == "RightButton")
    {
        grid.DeleteRow(e.RowHandle);
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl.LeftSwipeButtons>
  <dxGrid:SwipeButtonInfo ButtonName="LeftButton" Caption="Day" BackgroundColor="Green" Width="150"/>
</dxGrid:GridControl.LeftSwipeButtons>
<dxGrid:GridControl.RightSwipeButtons>
  <dxGrid:SwipeButtonInfo ButtonName="RightButton" Caption="Delete" BackgroundColor="Red" Width="150"/>
</dxGrid:GridControl.RightSwipeButtons>
```

## See Also

SwipeButtonEventArgs Class
SwipeButtonEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# SwipeButtonEventHandler Delegate

Represents a method that will handle the GridControl.SwipeButtonClick event.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public delegate void SwipeButtonEventHandler(
   object sender,
   SwipeButtonEventArgs e
)
```

### Parameters

The declaration of your event handler must have the same parameters as the **SwipeButtonEventHandler** delegate declaration.
*sender*
The event sender.
*e*
A SwipeButtonEventArgs object that contains event data.

### See Also

DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# SwipeButtonInfo Class

Contains information on a button shown when an end-user swipes left to right, or right to left, over a data row.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public class SwipeButtonInfo : object
```

### Remarks

Create **SwipeButtonInfo** objects and add them to the GridControl.LeftSwipeButtons and GridControl.RightSwipeButtons collections to create swipe buttons in the grid.

### Example

**Show Me**

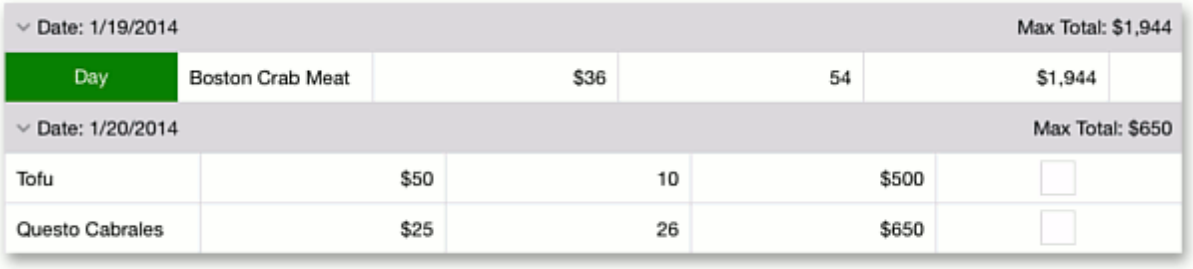
A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T178764 .

This example shows how to extend the grid's UI with additional buttons to be shown when an end-user swipes left to right, or right to left, over a data row. Swipe Buttons. To create a swipe button, add a SwipeButtonInfo object to the GridControl.LeftSwipeButtons or GridControl.RightSwipeButtons collection. Use this object's Caption, Width, BackgroundColor and TextColor properties to specify button appearance.

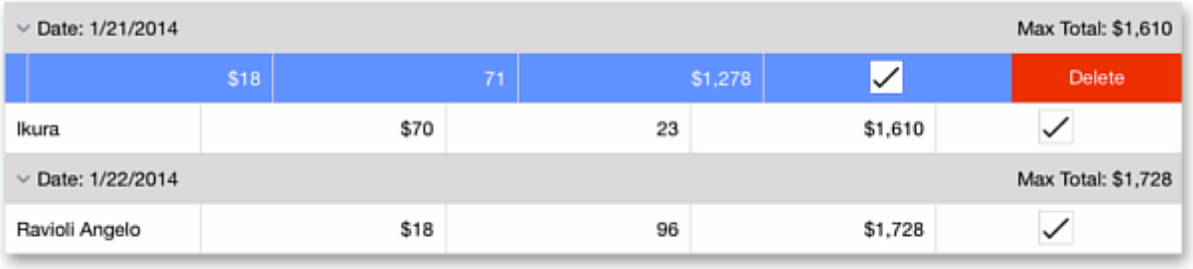
To assign a custom action to a button, handle the GridControl.SwipeButtonClick event. The GridControl.SwipeButtonShowing event allows you to specify when a swipe button should be visible.

In the example, two swipe buttons are created.

- When an end-user swipes any data row left to right, the **Day** button is shown on the left side of the row. Tapping this button displays the day of the week when the corresponding order was made.




- When an end-user swipes right to left, a data row which corresponds to an order that is already shipped (the value in the **Shipped** column is **true**), the **Delete** button is shown on the right side of the row. Tapping this button removes the corresponding data row.




**C#**

```
 (MainPage.xaml.cs)
void OnSwipeButtonShowing(object sender, SwipeButtonShowingEventArgs e)
{
    if ((!(Boolean)grid.GetCellValue(e.RowHandle, "Shipped"))
        && (e.ButtonInfo.ButtonName == "RightButton"))
    {
        e.IsVisible = false;
    }
}
void OnSwipeButtonClick(object sender, SwipeButtonEventArgs e)
{
    if (e.ButtonInfo.ButtonName == "LeftButton")
    {
        DateTime orderDate = (DateTime)grid.GetCellValue(e.RowHandle, "Date");
        string orderDateDay = orderDate.ToString("dddd");
        DisplayAlert("Alert from " + e.ButtonInfo.ButtonName, "Day: " + orderDateDay, "OK");
    }
    if (e.ButtonInfo.ButtonName == "RightButton")
    {
        grid.DeleteRow(e.RowHandle);
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl.LeftSwipeButtons>
  <dxGrid:SwipeButtonInfo ButtonName="LeftButton" Caption="Day" BackgroundColor="Green" Width="150"/>
</dxGrid:GridControl.LeftSwipeButtons>
<dxGrid:GridControl.RightSwipeButtons>
  <dxGrid:SwipeButtonInfo ButtonName="RightButton" Caption="Delete" BackgroundColor="Red" Width="150"/>
</dxGrid:GridControl.RightSwipeButtons>
```

## Inheritance Hierarchy

System.Object
    **SwipeButtonInfo**

## See Also

SwipeButtonInfo Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# SwipeButtonInfo Members

Contains information on a button shown when an end-user swipes left to right, or right to left, over a data row.

The following tables list the members exposed by the SwipeButtonInfo type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | SwipeButtonInfo | Initializes a new instance of the SwipeButtonInfo class with default settings. |

### Public Properties

| | Name | Description |
|---|---|---|
| | AutoCloseOnTap | Gets or sets whether a swipe button should be hidden immediately after an end-user taps it. |
| | BackgroundColor | Gets or sets the button background color. |
| | ButtonName | Gets or sets the button name. |
| | Caption | Gets or sets the text displayed within the button. |
| | TextColor | Gets or sets the color used to paint the text within the button. |
| | Width | Gets or sets the button width. |

### Public Methods

| | Name | Description |
|---|---|---|
| | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

**See Also**
SwipeButtonInfo Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# SwipeButtonInfo Constructor

Initializes a new instance of the SwipeButtonInfo class with default settings.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**
**C#**

```
public SwipeButtonInfo()
```

**See Also**
SwipeButtonInfo Class
SwipeButtonInfo Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# SwipeButtonInfo Properties

Contains information on a button shown when an end-user swipes left to right, or right to left, over a data row.

The following tables list the members exposed by the SwipeButtonInfo type.

**Public Constructors**

| | Name | Description |
|---|---|---|
| | SwipeButtonInfo | Initializes a new instance of the SwipeButtonInfo class with default settings. |

**Public Properties**

| | Name | Description |
|---|---|---|
| | AutoCloseOnTap | Gets or sets whether a swipe button should be hidden immediately after an end-user taps it. |
| | BackgroundColor | Gets or sets the button background color. |
| | ButtonName | Gets or sets the button name. |
| | Caption | Gets or sets the text displayed within the button. |
| | TextColor | Gets or sets the color used to paint the text within the button. |
| | Width | Gets or sets the button width. |

**Public Methods**

| | Name | Description |
|---|---|---|
| | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

**See Also**
SwipeButtonInfo Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# AutoCloseOnTap Property

Gets or sets whether a swipe button should be hidden immediately after an end-user taps it.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public bool AutoCloseOnTap { get; set; }
```

### Property Value

**true**, to automatically hide a swipe button; otherwise, **false**.

### See Also

SwipeButtonInfo Class
SwipeButtonInfo Members
DevExpress.Mobile.DataGrid Namespace
How to: Implement Swipe Buttons for Data Rows

© 2018 Developer Express Inc.

# BackgroundColor Property

Gets or sets the button background color.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public Color BackgroundColor { get; set; }
```

### Property Value

A Xamarin.Forms.Color object specifying the button background color.

### Example

**Show Me**

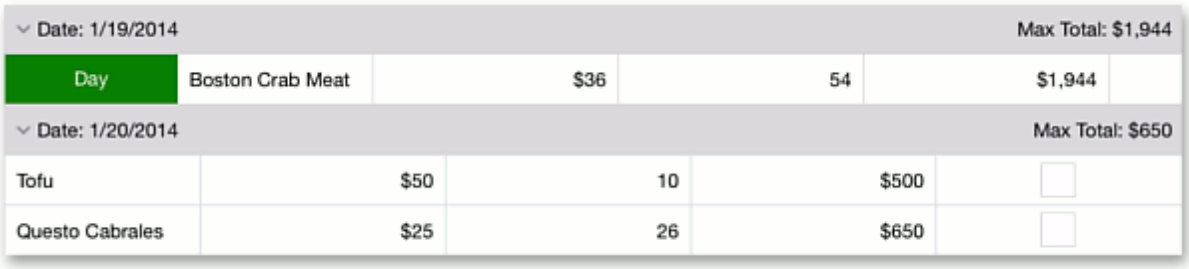
A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T178764 .

This example shows how to extend the grid's UI with additional buttons to be shown when an end-user swipes left to right, or right to left, over a data row. Swipe Buttons. To create a swipe button, add a SwipeButtonInfo object to the GridControl.LeftSwipeButtons or GridControl.RightSwipeButtons collection. Use this object's Caption, Width, BackgroundColor and TextColor properties to specify button appearance.

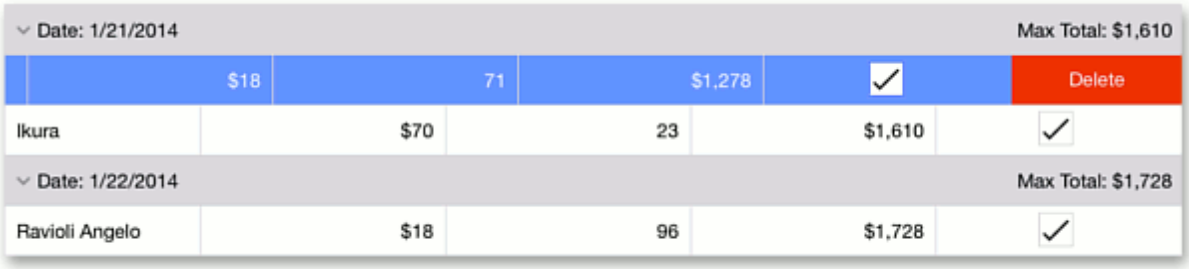
To assign a custom action to a button, handle the GridControl.SwipeButtonClick event. The GridControl.SwipeButtonShowing event allows you to specify when a swipe button should be visible.

In the example, two swipe buttons are created.

- When an end-user swipes any data row left to right, the **Day** button is shown on the left side of the row. Tapping this button displays the day of the week when the corresponding order was made.




- When an end-user swipes right to left, a data row which corresponds to an order that is already shipped (the value in the **Shipped** column is **true**), the **Delete** button is shown on the right side of the row. Tapping this button removes the corresponding data row.




**C#**

```
 (MainPage.xaml.cs)
void OnSwipeButtonShowing(object sender, SwipeButtonShowingEventArgs e)
{
    if ((!(Boolean)grid.GetCellValue(e.RowHandle, "Shipped"))
        && (e.ButtonInfo.ButtonName == "RightButton"))
    {
        e.IsVisible = false;
    }
}
void OnSwipeButtonClick(object sender, SwipeButtonEventArgs e)
{
    if (e.ButtonInfo.ButtonName == "LeftButton")
    {
        DateTime orderDate = (DateTime)grid.GetCellValue(e.RowHandle, "Date");
        string orderDateDay = orderDate.ToString("dddd");
        DisplayAlert("Alert from " + e.ButtonInfo.ButtonName, "Day: " + orderDateDay, "OK");
    }
    if (e.ButtonInfo.ButtonName == "RightButton")
    {
        grid.DeleteRow(e.RowHandle);
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl.LeftSwipeButtons>
  <dxGrid:SwipeButtonInfo ButtonName="LeftButton" Caption="Day" BackgroundColor="Green" Width="150"/>
</dxGrid:GridControl.LeftSwipeButtons>
<dxGrid:GridControl.RightSwipeButtons>
  <dxGrid:SwipeButtonInfo ButtonName="RightButton" Caption="Delete" BackgroundColor="Red" Width="150"/>
</dxGrid:GridControl.RightSwipeButtons>
```

## See Also

SwipeButtonInfo Class
SwipeButtonInfo Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# ButtonName Property

Gets or sets the button name.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public string ButtonName { get; set; }
```

### Property Value

A string that specifies the button name.

### Example

**Show Me**

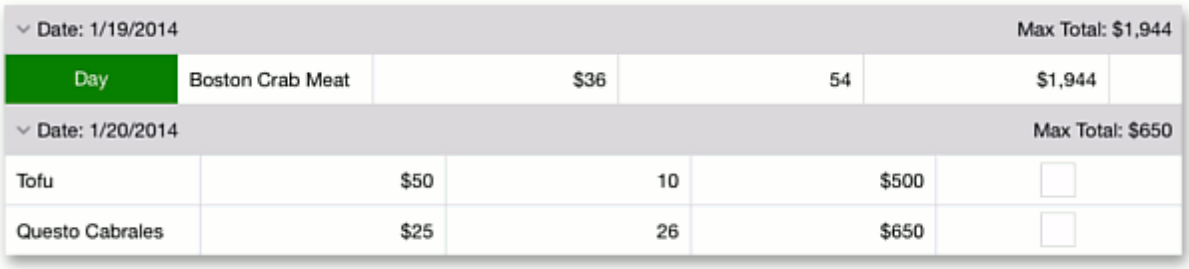
A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T178764 .

This example shows how to extend the grid's UI with additional buttons to be shown when an end-user swipes left to right, or right to left, over a data row. To create a swipe button, add a SwipeButtonInfo object to the GridControl.LeftSwipeButtons or GridControl.RightSwipeButtons collection. Use this object's Caption, Width, BackgroundColor and TextColor properties to specify button appearance.

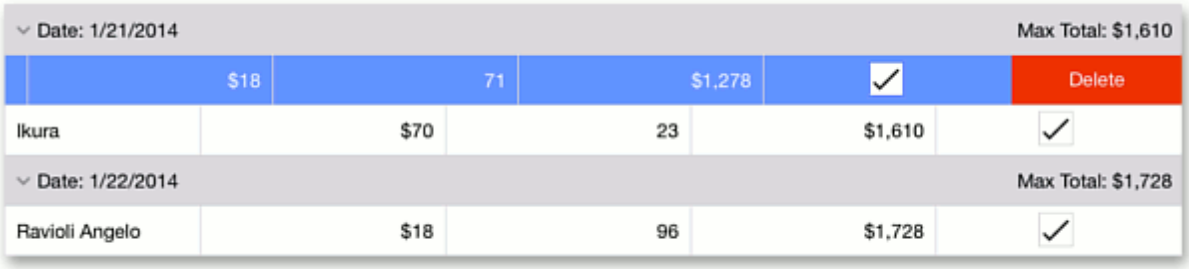
To assign a custom action to a button, handle the GridControl.SwipeButtonClick event. The GridControl.SwipeButtonShowing event allows you to specify when a swipe button should be visible.

In the example, two swipe buttons are created.

- When an end-user swipes any data row left to right, the **Day** button is shown on the left side of the row. Tapping this button displays the day of the week when the corresponding order was made.




- When an end-user swipes right to left, a data row which corresponds to an order that is already shipped (the value in the **Shipped** column is **true**), the **Delete** button is shown on the right side of the row. Tapping this button removes the corresponding data row.




**C#**

```
 (MainPage.xaml.cs)
void OnSwipeButtonShowing(object sender, SwipeButtonShowingEventArgs e)
{
    if ((!(Boolean)grid.GetCellValue(e.RowHandle, "Shipped"))
        && (e.ButtonInfo.ButtonName == "RightButton"))
    {
        e.IsVisible = false;
    }
}
void OnSwipeButtonClick(object sender, SwipeButtonEventArgs e)
{
    if (e.ButtonInfo.ButtonName == "LeftButton")
    {
        DateTime orderDate = (DateTime)grid.GetCellValue(e.RowHandle, "Date");
        string orderDateDay = orderDate.ToString("dddd");
        DisplayAlert("Alert from " + e.ButtonInfo.ButtonName, "Day: " + orderDateDay, "OK");
    }
    if (e.ButtonInfo.ButtonName == "RightButton")
    {
        grid.DeleteRow(e.RowHandle);
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl.LeftSwipeButtons>
  <dxGrid:SwipeButtonInfo ButtonName="LeftButton" Caption="Day" BackgroundColor="Green" Width="150"/>
</dxGrid:GridControl.LeftSwipeButtons>
<dxGrid:GridControl.RightSwipeButtons>
  <dxGrid:SwipeButtonInfo ButtonName="RightButton" Caption="Delete" BackgroundColor="Red" Width="150"/>
</dxGrid:GridControl.RightSwipeButtons>
```

## See Also

SwipeButtonInfo Class
SwipeButtonInfo Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# Caption Property

Gets or sets the text displayed within the button.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public string Caption { get; set; }
```

### Property Value

A string that specifies the button's caption.

### Example

**Show Me**

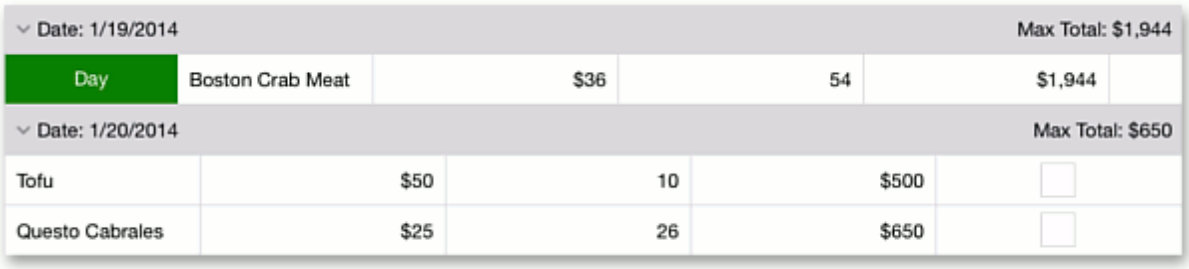
A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T178764 .

This example shows how to extend the grid's UI with additional buttons to be shown when an end-user swipes left to right, or right to left, over a data row. To create a swipe button, add a SwipeButtonInfo object to the GridControl.LeftSwipeButtons or GridControl.RightSwipeButtons collection. Use this object's Caption, Width, BackgroundColor and TextColor properties to specify button appearance.

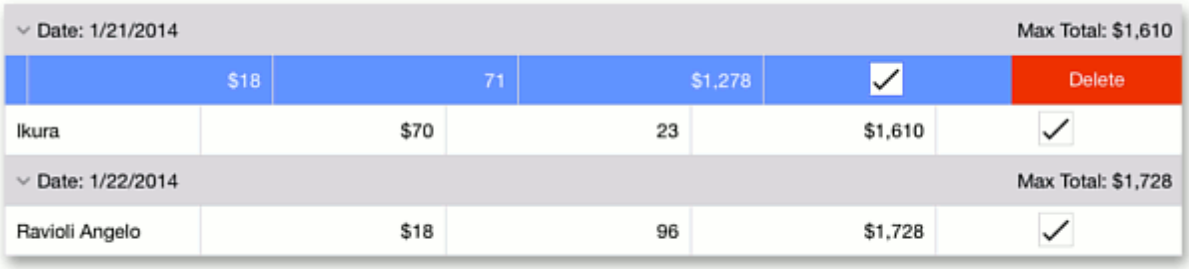
To assign a custom action to a button, handle the GridControl.SwipeButtonClick event. The GridControl.SwipeButtonShowing event allows you to specify when a swipe button should be visible.

In the example, two swipe buttons are created.

- When an end-user swipes any data row left to right, the **Day** button is shown on the left side of the row. Tapping this button displays the day of the week when the corresponding order was made.




- When an end-user swipes right to left, a data row which corresponds to an order that is already shipped (the value in the **Shipped** column is **true**), the **Delete** button is shown on the right side of the row. Tapping this button removes the corresponding data row.




**C#**

```
 (MainPage.xaml.cs)
void OnSwipeButtonShowing(object sender, SwipeButtonShowingEventArgs e)
{
    if ((!(Boolean)grid.GetCellValue(e.RowHandle, "Shipped"))
        && (e.ButtonInfo.ButtonName == "RightButton"))
    {
        e.IsVisible = false;
    }
}
void OnSwipeButtonClick(object sender, SwipeButtonEventArgs e)
{
    if (e.ButtonInfo.ButtonName == "LeftButton")
    {
        DateTime orderDate = (DateTime)grid.GetCellValue(e.RowHandle, "Date");
        string orderDateDay = orderDate.ToString("dddd");
        DisplayAlert("Alert from " + e.ButtonInfo.ButtonName, "Day: " + orderDateDay, "OK");
    }
    if (e.ButtonInfo.ButtonName == "RightButton")
    {
        grid.DeleteRow(e.RowHandle);
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl.LeftSwipeButtons>
  <dxGrid:SwipeButtonInfo ButtonName="LeftButton" Caption="Day" BackgroundColor="Green" Width="150"/>
</dxGrid:GridControl.LeftSwipeButtons>
<dxGrid:GridControl.RightSwipeButtons>
  <dxGrid:SwipeButtonInfo ButtonName="RightButton" Caption="Delete" BackgroundColor="Red" Width="150"/>
</dxGrid:GridControl.RightSwipeButtons>
```

## See Also

SwipeButtonInfo Class
SwipeButtonInfo Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# TextColor Property

Gets or sets the color used to paint the text within the button.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public Color TextColor { get; set; }
```

### Property Value

A Xamarin.Forms.Color object specifying the button foreground color.

### See Also

SwipeButtonInfo Class
SwipeButtonInfo Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# Width Property

Gets or sets the button width.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public Double Width { get; set; }
```

### Property Value

A System.Double value that specifies the button width in density-independent pixels.

### Example

**Show Me**

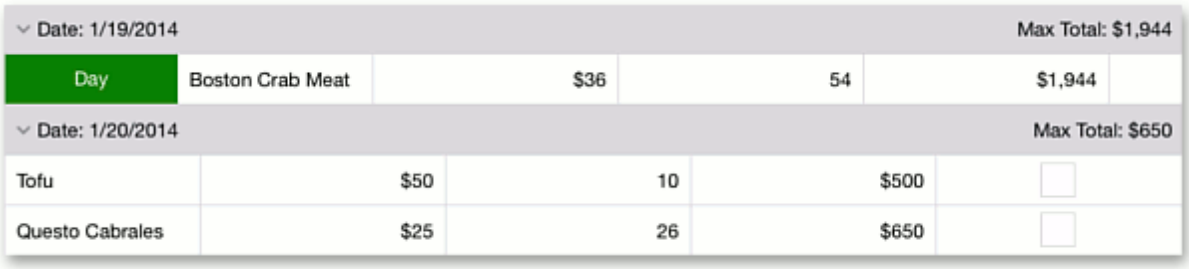
A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T178764 .

This example shows how to extend the grid's UI with additional buttons to be shown when an end-user swipes left to right, or right to left, over a data row. To create a swipe button, add a SwipeButtonInfo object to the GridControl.LeftSwipeButtons or GridControl.RightSwipeButtons collection. Use this object's Caption, Width, BackgroundColor and TextColor properties to specify button appearance.

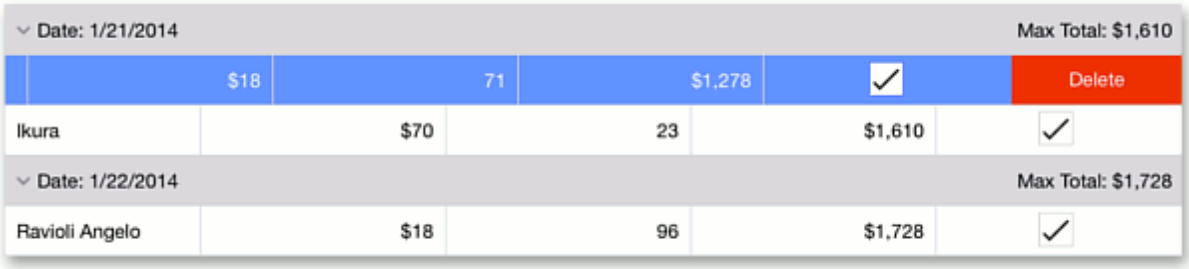
To assign a custom action to a button, handle the GridControl.SwipeButtonClick event. The GridControl.SwipeButtonShowing event allows you to specify when a swipe button should be visible.

In the example, two swipe buttons are created.

- When an end-user swipes any data row left to right, the **Day** button is shown on the left side of the row. Tapping this button displays the day of the week when the corresponding order was made.




- When an end-user swipes right to left, a data row which corresponds to an order that is already shipped (the value in the **Shipped** column is **true**), the **Delete** button is shown on the right side of the row. Tapping this button removes the corresponding data row.




**C#**

```
 (MainPage.xaml.cs)
void OnSwipeButtonShowing(object sender, SwipeButtonShowingEventArgs e)
{
    if ((!(Boolean)grid.GetCellValue(e.RowHandle, "Shipped"))
        && (e.ButtonInfo.ButtonName == "RightButton"))
    {
        e.IsVisible = false;
    }
}
void OnSwipeButtonClick(object sender, SwipeButtonEventArgs e)
{
    if (e.ButtonInfo.ButtonName == "LeftButton")
    {
        DateTime orderDate = (DateTime)grid.GetCellValue(e.RowHandle, "Date");
        string orderDateDay = orderDate.ToString("dddd");
        DisplayAlert("Alert from " + e.ButtonInfo.ButtonName, "Day: " + orderDateDay, "OK");
    }
    if (e.ButtonInfo.ButtonName == "RightButton")
    {
        grid.DeleteRow(e.RowHandle);
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl.LeftSwipeButtons>
  <dxGrid:SwipeButtonInfo ButtonName="LeftButton" Caption="Day" BackgroundColor="Green" Width="150"/>
</dxGrid:GridControl.LeftSwipeButtons>
<dxGrid:GridControl.RightSwipeButtons>
  <dxGrid:SwipeButtonInfo ButtonName="RightButton" Caption="Delete" BackgroundColor="Red" Width="150"/>
</dxGrid:GridControl.RightSwipeButtons>
```

## See Also

SwipeButtonInfo Class
SwipeButtonInfo Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# SwipeButtonShowingEventArgs Class

Provides data for the GridControl.SwipeButtonShowing event.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public class SwipeButtonShowingEventArgs : SwipeButtonEventArgs
```

### Remarks

The GridControl.SwipeButtonShowing event occurs before a swipe button is shown when an end-user swipes left to right, or right to left, over a data row. You can handle this event and set the IsVisible property to **false** to prevent a swipe button from being displayed.

The **SwipeButtonShowingEventArgs** objects are automatically created, initialized and passed to the GridControl.SwipeButtonShowing event handlers.

### Example

**Show Me**

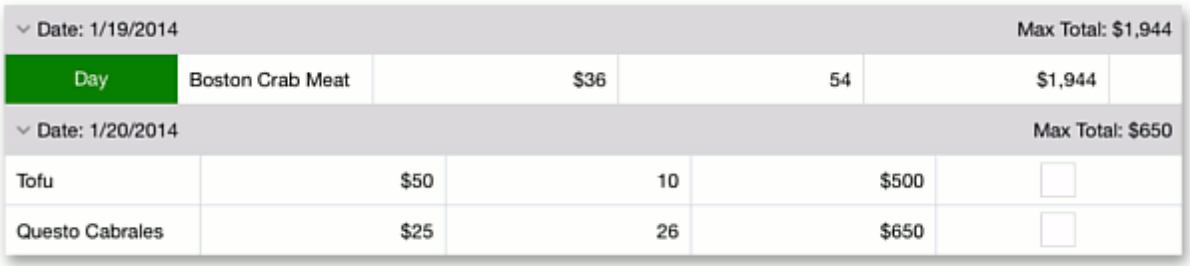
A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T178764 .

This example shows how to extend the grid's UI with additional buttons to be shown when an end-user swipes left to right, or right to left, over a data row Swipe Buttons. To create a swipe button, add a SwipeButtonInfo object to the GridControl.LeftSwipeButtons or GridControl.RightSwipeButtons collection. Use this object's SwipeButtonInfo.Caption, SwipeButtonInfo.Width, SwipeButtonInfo.BackgroundColor and SwipeButtonInfo.TextColor properties to specify button appearance.

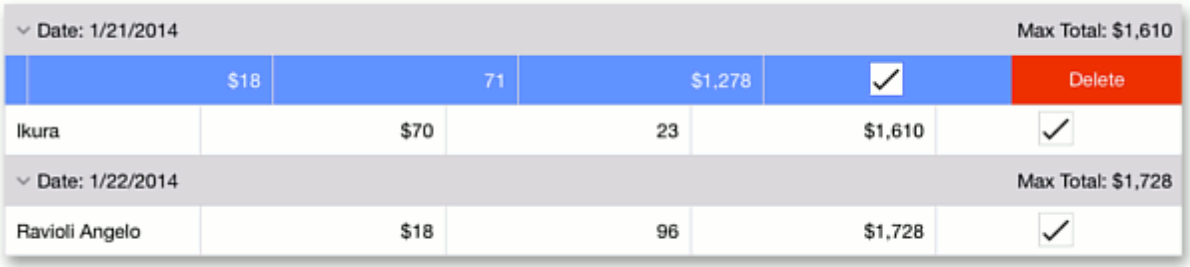
To assign a custom action to a button, handle the GridControl.SwipeButtonClick event. The GridControl.SwipeButtonShowing event allows you to specify when a swipe button should be visible.

In the example, two swipe buttons are created.

- When an end-user swipes any data row left to right, the **Day** button is shown on the left side of the row. Tapping this button displays the day of the week when the corresponding order was made.




- When an end-user swipes right to left, a data row which corresponds to an order that is already shipped (the value in the **Shipped** column is **true**), the **Delete** button is shown on the right side of the row. Tapping this button removes the corresponding data row.




**C#**

© 2018 Developer Express Inc.

```
 (MainPage.xaml.cs)
void OnSwipeButtonShowing(object sender, SwipeButtonShowingEventArgs e)
{
    if ((!(Boolean)grid.GetCellValue(e.RowHandle, "Shipped"))
        && (e.ButtonInfo.ButtonName == "RightButton"))
    {
        e.IsVisible = false;
    }
}
void OnSwipeButtonClick(object sender, SwipeButtonEventArgs e)
{
    if (e.ButtonInfo.ButtonName == "LeftButton")
    {
        DateTime orderDate = (DateTime)grid.GetCellValue(e.RowHandle, "Date");
        string orderDateDay = orderDate.ToString("dddd");
        DisplayAlert("Alert from " + e.ButtonInfo.ButtonName, "Day: " + orderDateDay, "OK");
    }
    if (e.ButtonInfo.ButtonName == "RightButton")
    {
        grid.DeleteRow(e.RowHandle);
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl.LeftSwipeButtons>
  <dxGrid:SwipeButtonInfo ButtonName="LeftButton" Caption="Day" BackgroundColor="Green" Width="150"/>
</dxGrid:GridControl.LeftSwipeButtons>
<dxGrid:GridControl.RightSwipeButtons>
  <dxGrid:SwipeButtonInfo ButtonName="RightButton" Caption="Delete" BackgroundColor="Red" Width="150"/>
</dxGrid:GridControl.RightSwipeButtons>
```

## Inheritance Hierarchy

System.Object
    System.EventArgs
        RowEventArgs
            SourceRowEventArgs
                SwipeButtonEventArgs
                    **SwipeButtonShowingEventArgs**

## See Also

SwipeButtonShowingEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# SwipeButtonShowingEventArgs Members

Provides data for the GridControl.SwipeButtonShowing event.

The following tables list the members exposed by the SwipeButtonShowingEventArgs type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | SwipeButtonShowingEventArgs | Initializes a new instance of the SwipeButtonShowingEventArgs class. |

### Public Properties

| | Name | Description |
|---|---|---|
| | ButtonInfo | Gets an object that stores information about a swipe button. (Inherited from **SwipeButtonEventArgs**) |
| | IsVisible | Gets or sets the swipe button visibility. |
| | RowHandle | Gets the row handle. (Inherited from **RowEventArgs**) |
| | SourceRowIndex | Gets the index of the record in a data source to which the processed row corresponds. (Inherited from **SourceRowEventArgs**) |

### Public Methods

| | Name | Description |
|---|---|---|
| S | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

### Public Fields

| | Name | Description |
|---|---|---|
| S | Empty | (Inherited from **System.EventArgs**) |

### See Also

SwipeButtonShowingEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# SwipeButtonShowingEventArgs Constructor

Initializes a new instance of the SwipeButtonShowingEventArgs class.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public SwipeButtonShowingEventArgs(
   SwipeButtonInfo buttonInfo,
   int rowHandle,
   int sourceRowIndex
)
```

### Parameters

*buttonInfo*
A SwipeButtonInfo object that contains information on the swipe button. This value is assigned to the SwipeButtonEventArgs.ButtonInfo property.
*rowHandle*
An integer value that specifies the row handle. This value is assigned to the **RowHandle** property.
*sourceRowIndex*
An integer value that specifies the row index in the data source. This value is assigned to the **SourceRowIndex** property.

### Remarks

Instances of the SwipeButtonShowingEventArgs class are automatically created, initialized and passed to the corresponding event handlers.

### See Also

SwipeButtonShowingEventArgs Class
SwipeButtonShowingEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# SwipeButtonShowingEventArgs Properties

Provides data for the GridControl.SwipeButtonShowing event.

The following tables list the members exposed by the SwipeButtonShowingEventArgs type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | SwipeButtonShowingEventArgs | Initializes a new instance of the SwipeButtonShowingEventArgs class. |

### Public Properties

| | Name | Description |
|---|---|---|
| | ButtonInfo | Gets an object that stores information about a swipe button. (Inherited from **SwipeButtonEventArgs**) |
| | IsVisible | Gets or sets the swipe button visibility. |
| | RowHandle | Gets the row handle. (Inherited from **RowEventArgs**) |
| | SourceRowIndex | Gets the index of the record in a data source to which the processed row corresponds. (Inherited from **SourceRowEventArgs**) |

### Public Methods

| | Name | Description |
|---|---|---|
| S | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

### Public Fields

| | Name | Description |
|---|---|---|
| S | Empty | (Inherited from **System.EventArgs**) |

### See Also

SwipeButtonShowingEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# IsVisible Property

Gets or sets the swipe button visibility.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public bool IsVisible { get; set; }
```

### Property Value

**true**, to show the swipe button; otherwise, **false**.

### Example

**Show Me**

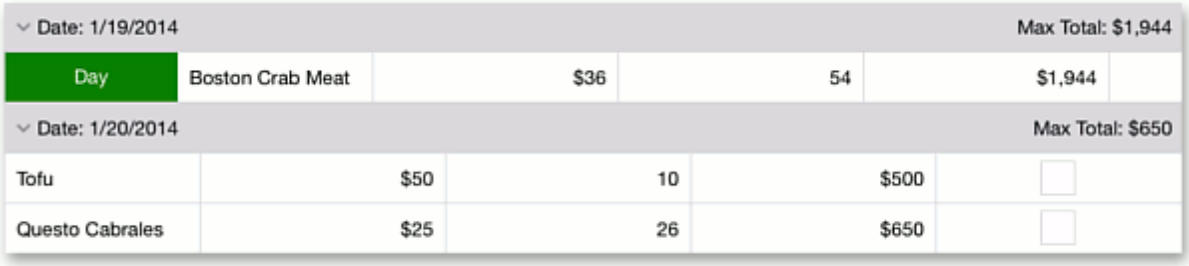
A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T178764 .

This example shows how to extend the grid's UI with additional buttons to be shown when an end-user swipes left to right, or right to left, over a data row. To create a swipe button, add a SwipeButtonInfo object to the GridControl.LeftSwipeButtons or GridControl.RightSwipeButtons collection. Use this object's SwipeButtonInfo.Caption, SwipeButtonInfo.Width, SwipeButtonInfo.BackgroundColor and SwipeButtonInfo.TextColor properties to specify button appearance.

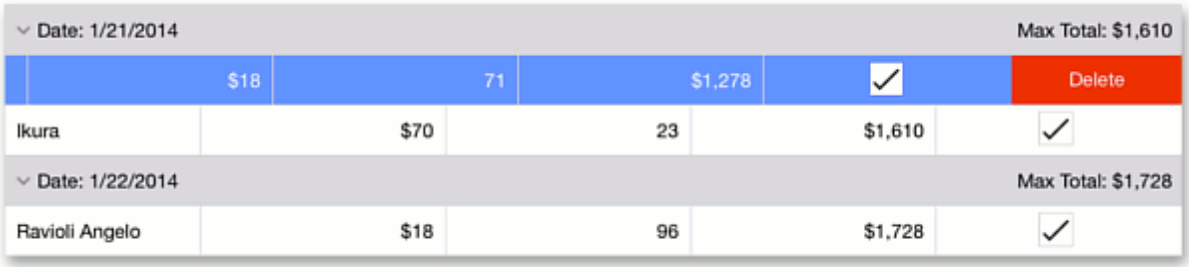
To assign a custom action to a button, handle the GridControl.SwipeButtonClick event. The GridControl.SwipeButtonShowing event allows you to specify when a swipe button should be visible.

In the example, two swipe buttons are created.

- When an end-user swipes any data row left to right, the **Day** button is shown on the left side of the row. Tapping this button displays the day of the week when the corresponding order was made.




- When an end-user swipes right to left, a data row which corresponds to an order that is already shipped (the value in the **Shipped** column is **true**), the **Delete** button is shown on the right side of the row. Tapping this button removes the corresponding data row.




**C#**

```
 (MainPage.xaml.cs)
void OnSwipeButtonShowing(object sender, SwipeButtonShowingEventArgs e)
{
    if ((!(Boolean)grid.GetCellValue(e.RowHandle, "Shipped"))
        && (e.ButtonInfo.ButtonName == "RightButton"))
    {
        e.IsVisible = false;
    }
}
void OnSwipeButtonClick(object sender, SwipeButtonEventArgs e)
{
    if (e.ButtonInfo.ButtonName == "LeftButton")
    {
        DateTime orderDate = (DateTime)grid.GetCellValue(e.RowHandle, "Date");
        string orderDateDay = orderDate.ToString("dddd");
        DisplayAlert("Alert from " + e.ButtonInfo.ButtonName, "Day: " + orderDateDay, "OK");
    }
    if (e.ButtonInfo.ButtonName == "RightButton")
    {
        grid.DeleteRow(e.RowHandle);
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl.LeftSwipeButtons>
  <dxGrid:SwipeButtonInfo ButtonName="LeftButton" Caption="Day" BackgroundColor="Green" Width="150"/>
</dxGrid:GridControl.LeftSwipeButtons>
<dxGrid:GridControl.RightSwipeButtons>
  <dxGrid:SwipeButtonInfo ButtonName="RightButton" Caption="Delete" BackgroundColor="Red" Width="150"/>
</dxGrid:GridControl.RightSwipeButtons>
```

## See Also

SwipeButtonShowingEventArgs Class
SwipeButtonShowingEventArgs Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# SwipeButtonShowingEventHandler Delegate

Represents a method that will handle the GridControl.SwipeButtonShowing event.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public delegate void SwipeButtonShowingEventHandler(
   object sender,
   SwipeButtonShowingEventArgs e
)
```

### Parameters

The declaration of your event handler must have the same parameters as the **SwipeButtonShowingEventHandler** delegate declaration.

*sender*
The event sender.

*e*
A SwipeButtonShowingEventArgs object that contains event data.

### See Also

DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# SwitchColumn Class

A grid column to display Boolean values and select between two states.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

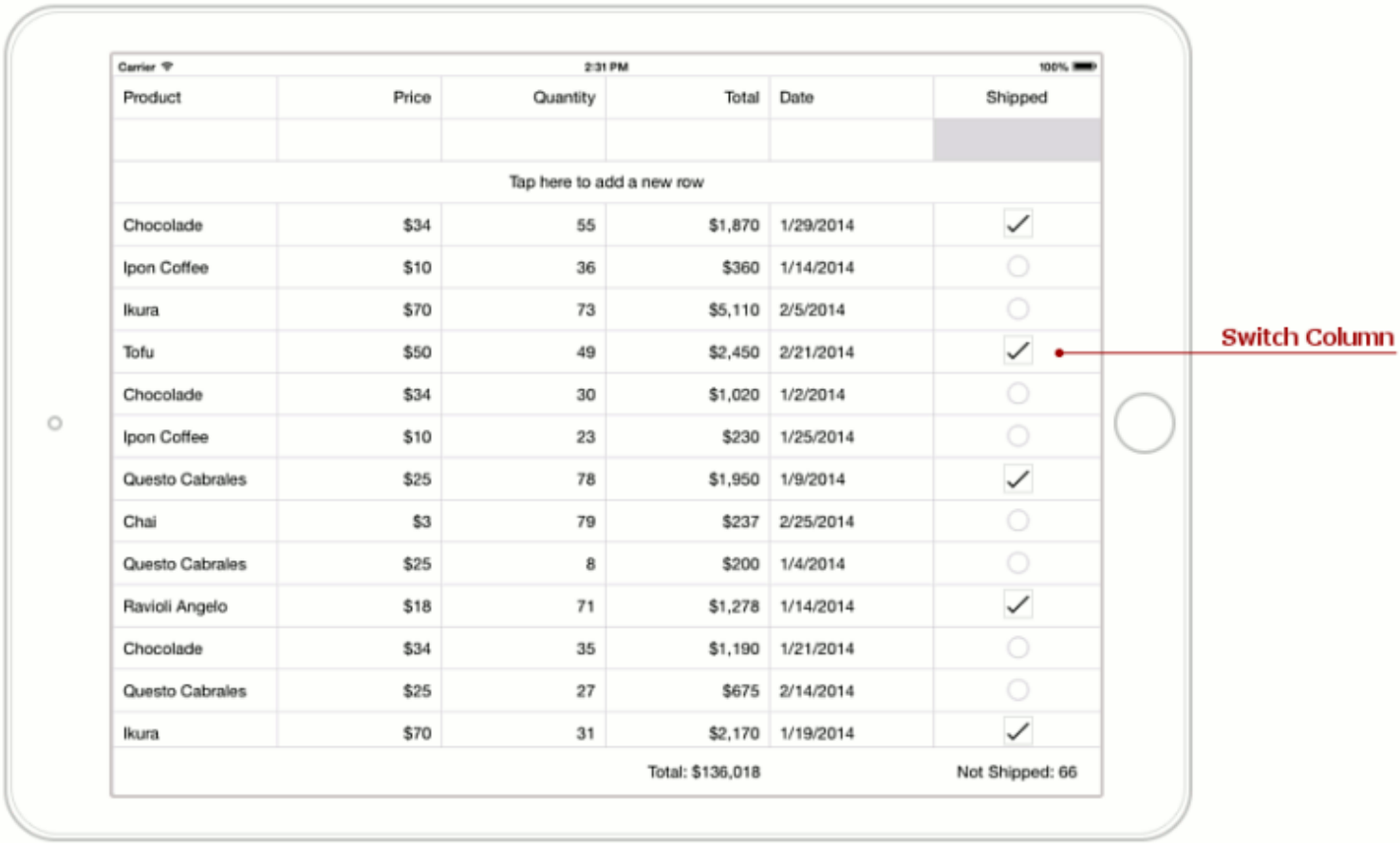
**C#**

```
public class SwitchColumn : GridColumn
```

### Remarks

In GridControl, data fields are represented as columns and records are displayed as rows. So, to display data, a grid should contain columns bound to fields of the underlying data source.

An individual column is specified by the **SwitchColumn**, NumberColumn, TextColumn or DateColumn object (a GridColumn descendant) depending on the type of data displayed in this column. Use columns of the **SwitchColumn** type that contain two-state switches within cells to allow end-users to change Boolean values.




Column objects provide a set of properties to adjust column settings (for example, GridColumn.DisplayFormat, GridColumn.Width, GridColumn.Caption, GridColumn.IsVisible, etc.).

GridControl stores its columns in the GridControl.Columns collection, and displays them in the order in which they are added to this collection.

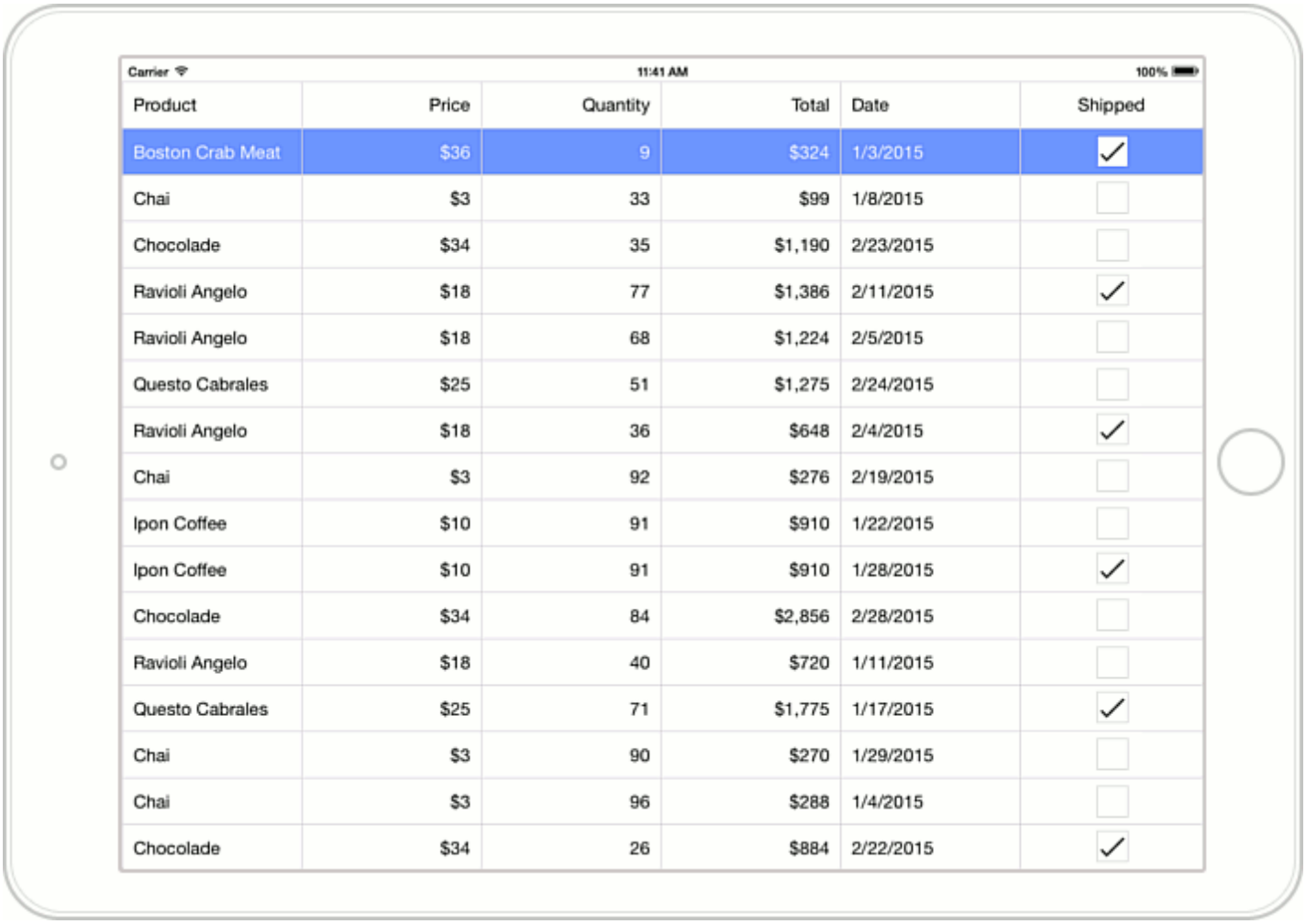
### Example

| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T178764. | |

© 2018 Developer Express Inc.

This example shows how to create and customize grid columns for displaying and editing data of different types (text, numbers, dates and Boolean values). The specified collection contains columns bound to the data source fields (*Product.Name*, *Product.UnitPrice*, *Quantity*, *Date* and *Shipped*) and one unbound column (*Total*) displaying data values calculated according to a formula based on the values of other columns.

The image below illustrates the result.




© 2018 Developer Express Inc.

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl.Columns>
         <dxGrid:TextColumn FieldName="Product.Name" Caption = "Product" Width = "170"
                            SortOrder = "Descending" SortIndex = "0"/>
         <dxGrid:NumberColumn FieldName="Product.UnitPrice" Caption = "Price" DisplayFormat="C0"/>
         <dxGrid:NumberColumn FieldName="Quantity"
                              SortOrder = "Ascending" SortIndex = "1"/>
         <dxGrid:NumberColumn FieldName="Total"
                              UnboundType="Integer" UnboundExpression="[Quantity] * [Product.UnitPrice]"
                              DisplayFormat="C0" IsReadOnly="True"/>
         <dxGrid:DateColumn FieldName="Date" DisplayFormat="d"
                            IsGrouped = "true" GroupInterval = "Date"/>
         <dxGrid:SwitchColumn FieldName="Shipped" AllowSort = "False" AllowAutoFilter="false"/>
     </dxGrid:GridControl.Columns>
```

### Inheritance Hierarchy

System.Object
  Xamarin.Forms.BindableObject
    GridColumn
      **SwitchColumn**

### See Also

SwitchColumn Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# SwitchColumn Members

A grid column to display Boolean values and select between two states.

The following tables list the members exposed by the SwitchColumn type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | SwitchColumn | Initializes a new instance of the SwitchColumn class with default settings. |

### Public Properties

| | Name | Description |
|---|---|---|
| | ActualCaption | Gets the column caption as it is displayed in a column header. (Inherited from **GridColumn**) |
| | ActualWidth | Gets the column width. This is a bindable property. (Inherited from **GridColumn**) |
| | AllowAutoFilter | Gets or sets whether an end-user can filter data by the column's values using the auto-filter panel. This is a bindable property. (Inherited from **GridColumn**) |
| | AllowGroup | Gets or sets whether end-users are allowed to group data by the current column. This is a bindable property. (Inherited from **GridColumn**) |
| | AllowSort | Gets or sets whether end-users are allowed to sort data by the column's values. This is a bindable property. (Inherited from **GridColumn**) |
| | AutoFilterCondition | Gets or sets the comparison operator used to create filter conditions for the current column via the auto-filter panel. This is a bindable property. (Inherited from **GridColumn**) |
| | AutoFilterValue | Gets or sets the value in the auto-filter panel for the column. This is a bindable property. (Inherited from **GridColumn**) |
| | BindingContext | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Caption | Gets or sets the column's display caption. This is a bindable property. (Inherited from **GridColumn**) |
| | ColumnFilterMode | Gets or sets how column values are filtered via the auto-filter panel. This is a bindable property. (Inherited from **GridColumn**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | ContentAlignment | Gets or sets the content alignment within column cells. This is a bindable property. (Inherited from **GridColumn**) |
| | DisplayFormat | Gets or sets formatting for values in the column. This is a bindable property. (Inherited from **GridColumn**) |
| | FieldName | Gets or sets the column's unique field name. For bound columns, this property refers to a field in the underlying data source. For unbound columns, this property serves as a column's identifier.<br><br>This is a bindable property.<br><br>(Inherited from **GridColumn**) |
| | FixedStyle | Gets or sets whether a column is fixed. This is a bindable property. (Inherited from **GridColumn**) |
| | GroupInterval | Gets or sets how data rows are grouped when grouping by the current column is applied. This is a bindable property. (Inherited from **GridColumn**) |
| | HeaderTemplate | Gets or sets a template used to define the column headers visual representation. (Inherited from **GridColumn**) |
| | ImmediateUpdateAutoFilter | Gets or sets whether the column's filter condition is updated as soon as an end-user modifies the text within the auto filter panel's cell. This is a bindable property. (Inherited from **GridColumn**) |
| | IsGrouped | Gets or sets whether the grid's data is grouped by the current column. This is a bindable property. (Inherited from **GridColumn**) |
| | IsReadOnly | Gets or sets whether end-users are allowed to change cell values in the column. This is a bindable property. (Inherited from **GridColumn**) |
| | IsUnbound | Gets whether the column is unbound. (Inherited from **GridColumn**) |
| | IsVisible | Gets or sets whether the column is visible in the grid. This is a bindable property. (Inherited from **GridColumn**) |
| | MinWidth | Gets or sets the minimum column width allowed. This is a bindable property. (Inherited from **GridColumn**) |
| | SerializationTypeName | Gets the string containing the column type name. (Inherited from **GridColumn**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | SortIndex | Gets or sets the column's position among sorted columns. This is a bindable property. (Inherited from **GridColumn**) |
| | SortMode | Specifies how column values are sorted and grouped. This is a bindable property. (Inherited from **GridColumn**) |
| | SortOrder | Gets or sets the column's sort order. This is a bindable property. (Inherited from **GridColumn**) |
| | TextOff | Gets or sets the text that indicates the OFF state of switches in data cells (Android) and of a segmented button in an auto-filter panel's cell (both iOS and Android) of the grid's SwitchColumn. |
| | TextOn | Gets or sets the text that indicates the ON state of switches in data cells (Android) and of a segmented button in an auto-filter panel's cell (both iOS and Android) of the grid's SwitchColumn. |
| | TextUndefined | Gets or sets the text that indicates an undefined state of the segmented button displayed in an auto-filter panel's cell of the grid's SwitchColumn. |
| | UnboundExpression | Gets or sets an expression used to evaluate values for the current Unbound Columns. This is a bindable property. (Inherited from **GridColumn**) |
| | UnboundType | Gets or sets the data type and binding mode of the column. This is a bindable property. (Inherited from **GridColumn**) |
| | Width | Gets or sets the column width. This is a bindable property. (Inherited from **GridColumn**) |

**Public Methods**

| | Name | Description |
|---|---|---|
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ToString | (Inherited from **System.Object**) |

**Public Fields**

| | Name | Description |
|---|---|---|
| S | ActualWidthProperty | Identifies the GridColumn.ActualWidth bindable property. (Inherited from **GridColumn**) |
| S | AllowAutoFilterProperty | Identifies the GridColumn.AllowAutoFilter bindable property. (Inherited from **GridColumn**) |
| S | AllowGroupProperty | Identifies the GridColumn.AllowGroup bindable property. (Inherited from **GridColumn**) |
| S | AllowSortProperty | Identifies the GridColumn.AllowSort bindable property. (Inherited from **GridColumn**) |
| S | AutoFilterConditionProperty | Identifies the GridColumn.AutoFilterCondition bindable property. (Inherited from **GridColumn**) |
| S | AutoFilterValueProperty | Identifies the GridColumn.AutoFilterValue bindable property. (Inherited from **GridColumn**) |
| S | BindingContextProperty | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | CaptionProperty | Identifies the GridColumn.Caption bindable property. (Inherited from **GridColumn**) |
| S | ColumnFilterModeProperty | Identifies the GridColumn.ColumnFilterMode bindable property. (Inherited from **GridColumn**) |
| S | ContentAlignmentProperty | Identifies the GridColumn.ContentAlignment bindable property. (Inherited from **GridColumn**) |
| S | DisplayFormatProperty | Identifies the GridColumn.DisplayFormat bindable property. (Inherited from **GridColumn**) |
| S | FieldNameProperty | Identifies the GridColumn.FieldName bindable property. (Inherited from **GridColumn**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| S | FixedStyleProperty | Identifies the GridColumn.FixedStyle bindable property. (Inherited from **GridColumn**) |
| S | GroupIntervalProperty | Identifies the GridColumn.GroupInterval bindable property. (Inherited from **GridColumn**) |
| S | HeaderTemplateProperty | Identifies the GridColumn.HeaderTemplate bindable property. (Inherited from **GridColumn**) |
| S | ImmediateUpdateAutoFilterProperty | Identifies the GridColumn.ImmediateUpdateAutoFilter bindable property. (Inherited from **GridColumn**) |
| S | IsGroupedProperty | Identifies the GridColumn.IsGrouped bindable property. (Inherited from **GridColumn**) |
| S | IsReadOnlyProperty | Identifies the GridColumn.IsReadOnly bindable property. (Inherited from **GridColumn**) |
| S | IsVisibleProperty | Identifies the GridColumn.IsVisible bindable property. (Inherited from **GridColumn**) |
| S | MinWidthProperty | Identifies the GridColumn.MinWidth bindable property. (Inherited from **GridColumn**) |
| S | SortIndexProperty | Identifies the GridColumn.SortIndex bindable property. (Inherited from **GridColumn**) |
| S | SortModeProperty | Identifies the GridColumn.SortMode bindable property. (Inherited from **GridColumn**) |
| S | SortOrderProperty | Identifies the GridColumn.SortOrder bindable property. (Inherited from **GridColumn**) |
| S | UnboundExpressionProperty | Identifies the GridColumn.UnboundExpression bindable property. (Inherited from **GridColumn**) |
| S | UnboundTypeProperty | Identifies the GridColumn.UnboundType bindable property. (Inherited from **GridColumn**) |
| S | WidthProperty | Identifies the GridColumn.Width bindable property. (Inherited from **GridColumn**) |

**Public Events**

| | Name | Description |
|---|---|---|
| | BindingContextChanged | (Inherited from **Xamarin.Forms.BindableObject**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | PropertyChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanging | (Inherited from **Xamarin.Forms.BindableObject**) |

**See Also**
SwitchColumn Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# SwitchColumn Constructor

Initializes a new instance of the SwitchColumn class with default settings.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**
**C#**
```
public SwitchColumn()
```

**See Also**
SwitchColumn Class
SwitchColumn Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# SwitchColumn Properties

A grid column to display Boolean values and select between two states.

The following tables list the members exposed by the SwitchColumn type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | SwitchColumn | Initializes a new instance of the SwitchColumn class with default settings. |

### Public Properties

| | Name | Description |
|---|---|---|
| | ActualCaption | Gets the column caption as it is displayed in a column header. (Inherited from **GridColumn**) |
| | ActualWidth | Gets the column width. This is a bindable property. (Inherited from **GridColumn**) |
| | AllowAutoFilter | Gets or sets whether an end-user can filter data by the column's values using the auto-filter panel. This is a bindable property. (Inherited from **GridColumn**) |
| | AllowGroup | Gets or sets whether end-users are allowed to group data by the current column. This is a bindable property. (Inherited from **GridColumn**) |
| | AllowSort | Gets or sets whether end-users are allowed to sort data by the column's values. This is a bindable property. (Inherited from **GridColumn**) |
| | AutoFilterCondition | Gets or sets the comparison operator used to create filter conditions for the current column via the auto-filter panel. This is a bindable property. (Inherited from **GridColumn**) |
| | AutoFilterValue | Gets or sets the value in the auto-filter panel for the column. This is a bindable property. (Inherited from **GridColumn**) |
| | BindingContext | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Caption | Gets or sets the column's display caption. This is a bindable property. (Inherited from **GridColumn**) |
| | ColumnFilterMode | Gets or sets how column values are filtered via the auto-filter panel. This is a bindable property. (Inherited from **GridColumn**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | ContentAlignment | Gets or sets the content alignment within column cells. This is a bindable property. (Inherited from **GridColumn**) |
| | DisplayFormat | Gets or sets formatting for values in the column. This is a bindable property. (Inherited from **GridColumn**) |
| | FieldName | Gets or sets the column's unique field name. For bound columns, this property refers to a field in the underlying data source. For unbound columns, this property serves as a column's identifier.<br><br>This is a bindable property.<br><br>(Inherited from **GridColumn**) |
| | FixedStyle | Gets or sets whether a column is fixed. This is a bindable property. (Inherited from **GridColumn**) |
| | GroupInterval | Gets or sets how data rows are grouped when grouping by the current column is applied. This is a bindable property. (Inherited from **GridColumn**) |
| | HeaderTemplate | Gets or sets a template used to define the column headers visual representation. (Inherited from **GridColumn**) |
| | ImmediateUpdateAutoFilter | Gets or sets whether the column's filter condition is updated as soon as an end-user modifies the text within the auto filter panel's cell. This is a bindable property. (Inherited from **GridColumn**) |
| | IsGrouped | Gets or sets whether the grid's data is grouped by the current column. This is a bindable property. (Inherited from **GridColumn**) |
| | IsReadOnly | Gets or sets whether end-users are allowed to change cell values in the column. This is a bindable property. (Inherited from **GridColumn**) |
| | IsUnbound | Gets whether the column is unbound. (Inherited from **GridColumn**) |
| | IsVisible | Gets or sets whether the column is visible in the grid. This is a bindable property. (Inherited from **GridColumn**) |
| | MinWidth | Gets or sets the minimum column width allowed. This is a bindable property. (Inherited from **GridColumn**) |
| | SerializationTypeName | Gets the string containing the column type name. (Inherited from **GridColumn**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | SortIndex | Gets or sets the column's position among sorted columns. This is a bindable property. (Inherited from **GridColumn**) |
| | SortMode | Specifies how column values are sorted and grouped. This is a bindable property. (Inherited from **GridColumn**) |
| | SortOrder | Gets or sets the column's sort order. This is a bindable property. (Inherited from **GridColumn**) |
| | TextOff | Gets or sets the text that indicates the OFF state of switches in data cells (Android) and of a segmented button in an auto-filter panel's cell (both iOS and Android) of the grid's SwitchColumn. |
| | TextOn | Gets or sets the text that indicates the ON state of switches in data cells (Android) and of a segmented button in an auto-filter panel's cell (both iOS and Android) of the grid's SwitchColumn. |
| | TextUndefined | Gets or sets the text that indicates an undefined state of the segmented button displayed in an auto-filter panel's cell of the grid's SwitchColumn. |
| | UnboundExpression | Gets or sets an expression used to evaluate values for the current Unbound Columns. This is a bindable property. (Inherited from **GridColumn**) |
| | UnboundType | Gets or sets the data type and binding mode of the column. This is a bindable property. (Inherited from **GridColumn**) |
| | Width | Gets or sets the column width. This is a bindable property. (Inherited from **GridColumn**) |

**Public Methods**

| | Name | Description |
|---|---|---|
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ToString | (Inherited from **System.Object**) |

**Public Fields**

| | Name | Description |
|---|---|---|
| S | ActualWidthProperty | Identifies the GridColumn.ActualWidth bindable property. (Inherited from **GridColumn**) |
| S | AllowAutoFilterProperty | Identifies the GridColumn.AllowAutoFilter bindable property. (Inherited from **GridColumn**) |
| S | AllowGroupProperty | Identifies the GridColumn.AllowGroup bindable property. (Inherited from **GridColumn**) |
| S | AllowSortProperty | Identifies the GridColumn.AllowSort bindable property. (Inherited from **GridColumn**) |
| S | AutoFilterConditionProperty | Identifies the GridColumn.AutoFilterCondition bindable property. (Inherited from **GridColumn**) |
| S | AutoFilterValueProperty | Identifies the GridColumn.AutoFilterValue bindable property. (Inherited from **GridColumn**) |
| S | BindingContextProperty | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | CaptionProperty | Identifies the GridColumn.Caption bindable property. (Inherited from **GridColumn**) |
| S | ColumnFilterModeProperty | Identifies the GridColumn.ColumnFilterMode bindable property. (Inherited from **GridColumn**) |
| S | ContentAlignmentProperty | Identifies the GridColumn.ContentAlignment bindable property. (Inherited from **GridColumn**) |
| S | DisplayFormatProperty | Identifies the GridColumn.DisplayFormat bindable property. (Inherited from **GridColumn**) |
| S | FieldNameProperty | Identifies the GridColumn.FieldName bindable property. (Inherited from **GridColumn**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| S | FixedStyleProperty | Identifies the GridColumn.FixedStyle bindable property. (Inherited from **GridColumn**) |
| S | GroupIntervalProperty | Identifies the GridColumn.GroupInterval bindable property. (Inherited from **GridColumn**) |
| S | HeaderTemplateProperty | Identifies the GridColumn.HeaderTemplate bindable property. (Inherited from **GridColumn**) |
| S | ImmediateUpdateAutoFilterProperty | Identifies the GridColumn.ImmediateUpdateAutoFilter bindable property. (Inherited from **GridColumn**) |
| S | IsGroupedProperty | Identifies the GridColumn.IsGrouped bindable property. (Inherited from **GridColumn**) |
| S | IsReadOnlyProperty | Identifies the GridColumn.IsReadOnly bindable property. (Inherited from **GridColumn**) |
| S | IsVisibleProperty | Identifies the GridColumn.IsVisible bindable property. (Inherited from **GridColumn**) |
| S | MinWidthProperty | Identifies the GridColumn.MinWidth bindable property. (Inherited from **GridColumn**) |
| S | SortIndexProperty | Identifies the GridColumn.SortIndex bindable property. (Inherited from **GridColumn**) |
| S | SortModeProperty | Identifies the GridColumn.SortMode bindable property. (Inherited from **GridColumn**) |
| S | SortOrderProperty | Identifies the GridColumn.SortOrder bindable property. (Inherited from **GridColumn**) |
| S | UnboundExpressionProperty | Identifies the GridColumn.UnboundExpression bindable property. (Inherited from **GridColumn**) |
| S | UnboundTypeProperty | Identifies the GridColumn.UnboundType bindable property. (Inherited from **GridColumn**) |
| S | WidthProperty | Identifies the GridColumn.Width bindable property. (Inherited from **GridColumn**) |

**Public Events**

| | Name | Description |
|---|---|---|
| | BindingContextChanged | (Inherited from **Xamarin.Forms.BindableObject**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | PropertyChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanging | (Inherited from **Xamarin.Forms.BindableObject**) |

**See Also**
SwitchColumn Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# TextOff Property

Gets or sets the text that indicates the OFF state of switches in data cells (Android) and of a segmented button in an auto-filter panel's cell (both iOS and Android) of the grid's SwitchColumn.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public string TextOff { get; set; }
```

### Property Value

A string value that specifies the text displayed on a switch's OFF element.

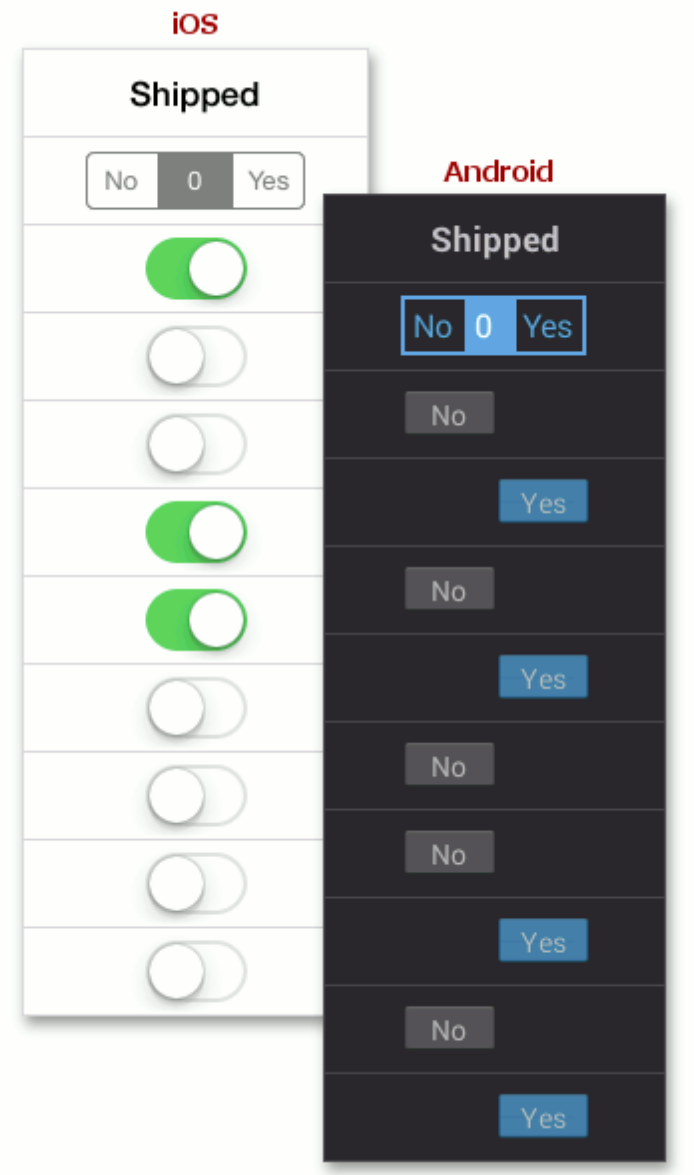
### Remarks

Use the **TextOff**, TextOn and TextUndefined properties to customize the text displayed within switches in data cells and auto-filter panel's cell in the grid's SwitchColumn. For example, the code below displays NO/YES instead of the default OFF/ON.

**Xaml**

```
<dxGrid:SwitchColumn FieldName="Shipped" TextOn="Yes" TextOff="No"/>
```

© 2018 Developer Express Inc.




**See Also**
SwitchColumn Class
SwitchColumn Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# TextOn Property

Gets or sets the text that indicates the ON state of switches in data cells (Android) and of a segmented button in an auto-filter panel's cell (both iOS and Android) of the grid's SwitchColumn.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public string TextOn { get; set; }
```

### Property Value

A string value that specifies the text displayed on a switch's ON element.

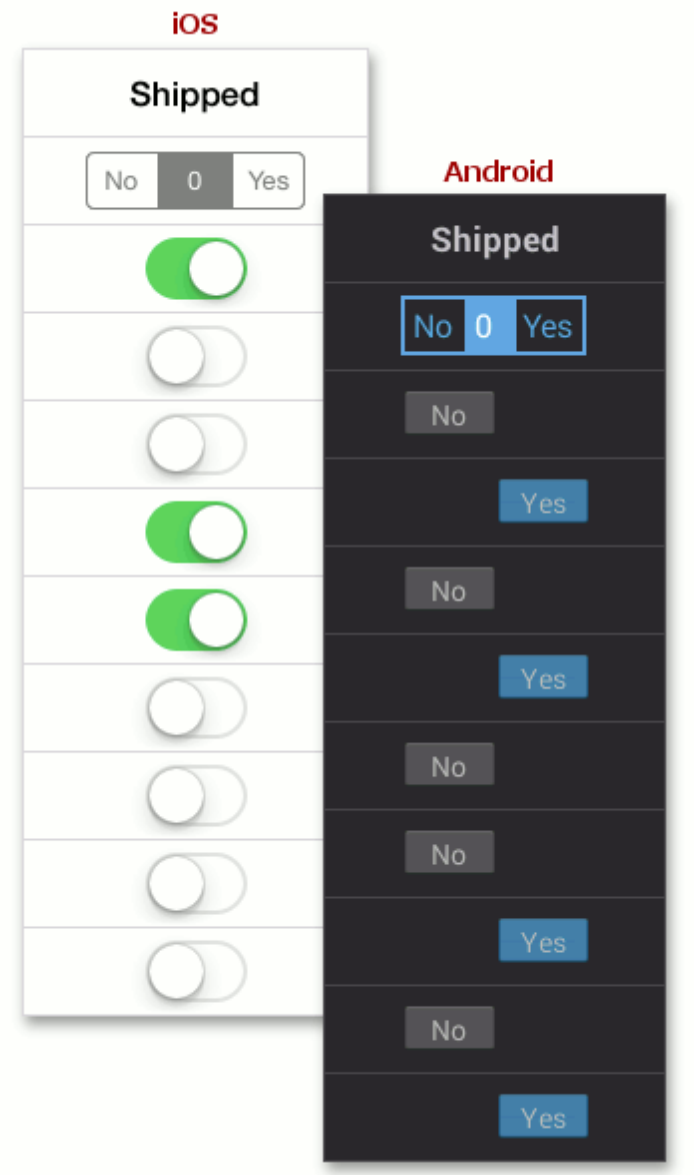
### Remarks

Use the **TextOn**, TextOff and TextUndefined properties to customize the text displayed within switches in data cells and auto-filter panel's cell in the grid's SwitchColumn. For example, the code below displays NO/YES instead of the default OFF/ON.

**Xaml**

```
<dxGrid:SwitchColumn FieldName="Shipped" TextOn="Yes" TextOff="No"/>
```

© 2018 Developer Express Inc.




**See Also**
SwitchColumn Class
SwitchColumn Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# TextUndefined Property

Gets or sets the text that indicates an undefined state of the segmented button displayed in an auto-filter panel's cell of the grid's SwitchColumn.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public string TextUndefined { get; set; }
```

### Property Value

A string value that specifies the text displayed on the segmented button's middle element that indicates an undefined state.

### Remarks

Use the TextOn, TextOff and **TextUndefined** properties to customize the text displayed within switches in data cells (iOS) and the auto-filter panel's cell (both iOS and Android) in the grid's SwitchColumn.

### See Also

SwitchColumn Class
SwitchColumn Members
DevExpress.Mobile.DataGrid Namespace
TextOn
TextOff

© 2018 Developer Express Inc.

# TemplateColumn Class

A column type that allows you to define a custom template for column cells.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public class TemplateColumn : GridColumn
```

### Remarks

GridControl provides various ways to customize the appearance of its elements. For simple customizations (such as modifying colors, fonts, etc.), use appropriate customizer objects. If you need to completely modify the appearance of a particular grid element, you can define a custom template for it.

The **TemplateColumn** class allows you to create templates for the column cells. Create a column of the **TemplateColumn** type, assign the required template to the column's DisplayTemplate property and add the templated column to the GridControl.Columns collection.

To define a template for the column values editor, use the templated column's EditTemplate property.

| Tip |
| --- |
| Also, you can define a data template for a column header. For this purpose, use the GridColumn.HeaderTemplate property. Column header templates are available for columns of any type. |

### Example

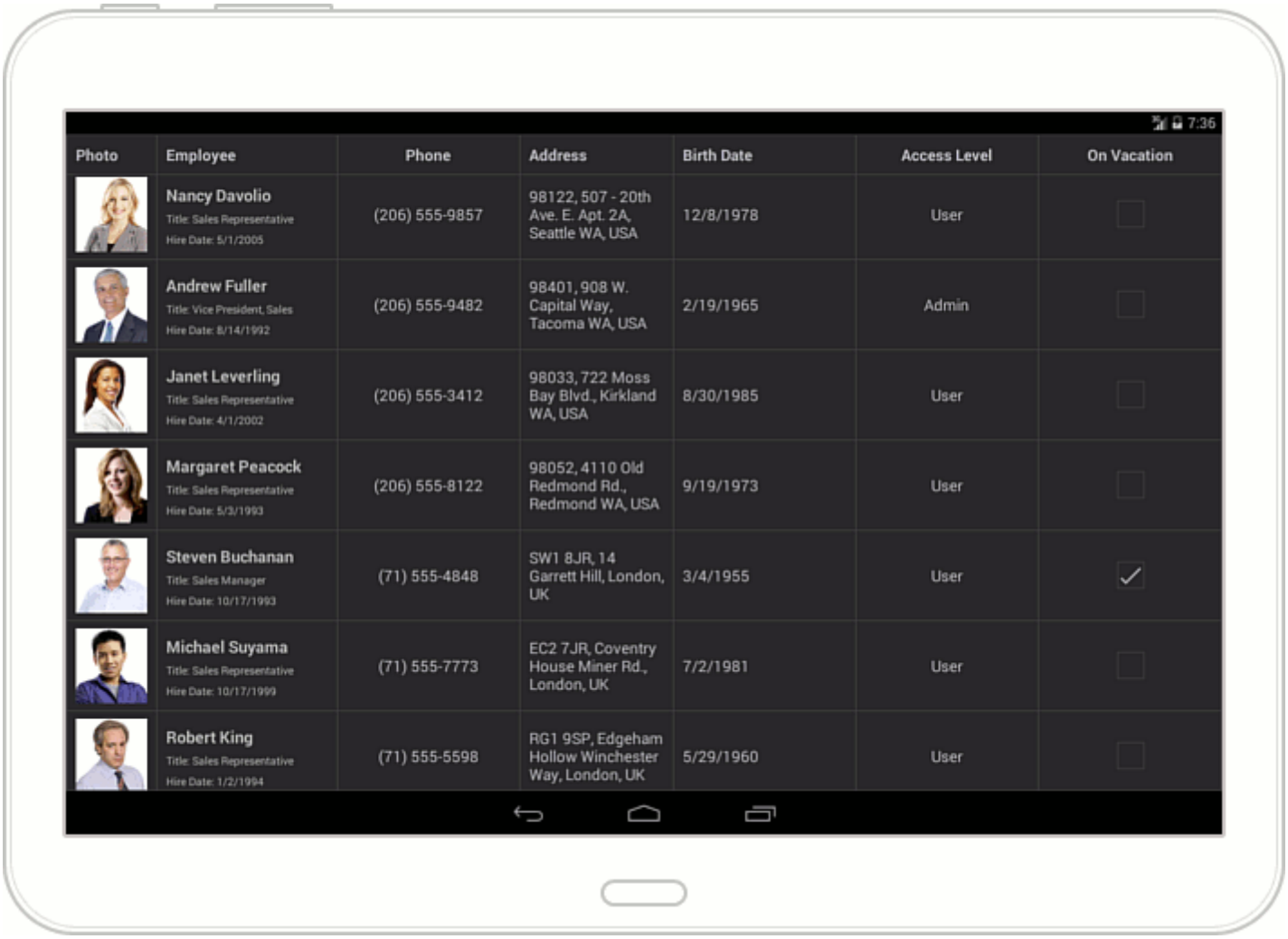
| Show Me | |
| --- | --- |
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T208848. | |

This example shows how to add columns to GridControl to display and edit data of different types. The grid is bound to a collection of **Employee** objects. Each **Employee** object contains an employee's photo (image), name, position, phone, address (strings), hire and birth dates (DateTime values), a value that specifies an employee's access level, and a Boolean value indicating whether an employee is on vacation.

© 2018 Developer Express Inc.

| Photo | Employee | Phone | Address | Birth Date | Access Level | On Vacation |
|---|---|---|---|---|---|---|
| | **Nancy Davolio**<br>Title: Sales Representative<br>Hire Date: 5/1/2005 | (206) 555-9857 | 98122, 507 - 20th Ave. E. Apt. 2A, Seattle WA, USA | 12/8/1978 | User | ☐ |
| | **Andrew Fuller**<br>Title: Vice President, Sales<br>Hire Date: 8/14/1992 | (206) 555-9482 | 98401, 908 W. Capital Way, Tacoma WA, USA | 2/19/1965 | Admin | ☐ |
| | **Janet Leverling**<br>Title: Sales Representative<br>Hire Date: 4/1/2002 | (206) 555-3412 | 98033, 722 Moss Bay Blvd., Kirkland WA, USA | 8/30/1985 | User | ☐ |
| | **Margaret Peacock**<br>Title: Sales Representative<br>Hire Date: 5/3/1993 | (206) 555-8122 | 98052, 4110 Old Redmond Rd., Redmond WA, USA | 9/19/1973 | User | ☐ |
| | **Steven Buchanan**<br>Title: Sales Manager<br>Hire Date: 10/17/1993 | (71) 555-4848 | SW1 8JR, 14 Garrett Hill, London, UK | 3/4/1955 | User | ☑ |
| | **Michael Suyama**<br>Title: Sales Representative<br>Hire Date: 10/17/1999 | (71) 555-7773 | EC2 7JR, Coventry House Miner Rd., London, UK | 7/2/1981 | User | ☐ |
| | **Robert King**<br>Title: Sales Representative<br>Hire Date: 1/2/1994 | (71) 555-5598 | RG1 9SP, Edgeham Hollow Winchester Way, London, UK | 5/29/1960 | User | ☐ |

© 2018 Developer Express Inc.




Grid columns are stored in the GridControl.Columns collection. An individual column is specified by a GridColumn descendant, which is appropriate to the type of data contained in this column. In this example, to display information on employees and allow end-users to edit it, the following columns are created in the grid.

| Grid Column | Description |
|---|---|
| Photo<br>ImageColumn | This column displays employee photos which are images added to a project as embedded resources. |
| Employee<br>TemplateColumn | A cell in this column displays three employee properties: Name, Position and HireDate. A cell appearance is defined via a template. Each cell contains three Xamarin.Forms.Label elements within a Xamarin.Forms.Grid. Each Label element is bound to a property of the **Employee** class.<br><br>A cell template data context is specified by the CellData object. Its CellData.Value property provides access to a value of a data field assigned to the column's GridColumn.FieldName property. In this example, a column cell displays not only this field value but also values of two more fields. So, it is required to specify the whole data row as a binding source for the template. To do this, set the BindingContext property of the Grid to CellData.Source. Note that BindingContext specified for the Grid is inherited by all its children. This means that all Label elements have the same BindingContext as the Grid, and you can specify |

© 2018 Developer Express Inc.

| | |
|---|---|
| | their simple bindings to properties of that object (Employee).<br><br>Use the GridColumn.AllowSort, GridColumn.AllowGroup and GridColumn.IsReadOnly properties to prevent end-users from sorting and grouping data in the grid by this column and disable data editing. |
| Phone<br>NumberColumn | The employee's **Phone** property is of the string type. The keyboard allowing text input is automatically displayed when an end-user activates this column cell to edit an employee's phone. |
| Address<br>TextColumn | This column is bound to the employee's **Address** string property. The keyboard allowing text input is automatically displayed when an end-user activates this column cell to edit an employee's address. |
| Birth Date<br>DateColumn | This column displays and allows editing employee birth days. The GridColumn.DisplayFormat property specifies the format of displaying dates. |
| Access Level<br>PickerColumn | The employee's **Access** property is of the **AccessLevel** type which is an enumeration with values **Admin** and **User**. To allow editing an employee's access level by selecting one of two enumeration values, this property is bound to the grid's picker column. |
| On Vacation<br>SwitchColumn | The employee's **OnVacation** property is of the Boolean type, so it is bound to the grid's switch column which allows editing a cell value by selecting between two states. |

**C#**

```
 (Employee.cs)
public enum AccessLevel {
    Admin,
    User
}
public class Employee : ModelObject {
    string name;
    public string Name {
        get { return name; }
        set {
            name = value;
            if (Photo == null)
                Photo = ImageSource.FromResource (value.Replace (" ", String.Empty));
        }
    }
    public Employee(string name) {
        this.Name = name;
    }
    public ImageSource Photo { get; set; }
    public DateTime BirthDate { get; set; }
    public DateTime HireDate { get; set; }
    public string Position { get; set; }
    public string Address { get; set; }
    public string Phone { get; set; }
    public AccessLevel Access { get; set; }
    public bool OnVacation { get; set; }
}
```

**C#**

```
 (EmployeesRepository.cs)
public abstract class EmployeesRepository {
    readonly ObservableCollection<Employee> employees;
    public EmployeesRepository() {
        this.employees = new ObservableCollection<Employee>();
    }
    public ObservableCollection<Employee> Employees {
        get { return employees; }
    }
}
public class TestEmployeesRepository : EmployeesRepository {
    public TestEmployeesRepository() : base() {
        GenerateCustomers();
    }
    void GenerateCustomers() {
        Employees.Add(
            new Employee("Nancy Davolio") {
                BirthDate = new DateTime(1978, 12, 8),
                HireDate = new DateTime(2005, 5, 1),
                Position = "Sales Representative",
                Address = "98122, 507 - 20th Ave. E. Apt. 2A, Seattle WA, USA",
                Phone = "(206) 555-9857",
                Access = AccessLevel.User,
                OnVacation = false
            }
        );
        Employees.Add(
            new Employee("Andrew Fuller") {
                BirthDate = new DateTime(1965, 2, 19),
                HireDate = new DateTime(1992, 8, 14),
                Position = "Vice President, Sales",
                Address = "98401, 908 W. Capital Way, Tacoma WA, USA",
                Phone = "(206) 555-9482",
                Access = AccessLevel.Admin,
                OnVacation = false
            }
        );
        Employees.Add(
            new Employee("Janet Leverling") {
                BirthDate = new DateTime(1985, 8, 30),
                HireDate = new DateTime(2002, 4, 1),
                Position = "Sales Representative",
                Address = "98033, 722 Moss Bay Blvd., Kirkland WA, USA",
                Phone = "(206) 555-3412",
                Access = AccessLevel.User,
                OnVacation = false
            }
        );
        Employees.Add(
            new Employee("Margaret Peacock") {
                BirthDate = new DateTime(1973, 9, 19),
                HireDate = new DateTime(1993, 5, 3),
                Position = "Sales Representative",
                Address = "98052, 4110 Old Redmond Rd., Redmond WA, USA",
                Phone = "(206) 555-8122",
                Access = AccessLevel.User,
                OnVacation = false
            }
        );
        Employees.Add(
            new Employee("Steven Buchanan") {
                BirthDate = new DateTime(1955, 3, 4),
                HireDate = new DateTime(1993, 10, 17),
                Position = "Sales Manager",
                Address = "SW1 8JR, 14 Garrett Hill, London, UK",
```

© 2018 Developer Express Inc.

```
                Phone = "(71) 555-4848",
                Access = AccessLevel.User,
                OnVacation = true
            }
        );
        Employees.Add(
            new Employee("Michael Suyama") {
                BirthDate = new DateTime(1981, 7, 2),
                HireDate = new DateTime(1999, 10, 17),
                Position = "Sales Representative",
                Address = "EC2 7JR, Coventry House Miner Rd., London, UK",
                Phone = "(71) 555-7773",
                Access = AccessLevel.User,
                OnVacation = false
            }
        );
        Employees.Add(
            new Employee("Robert King") {
                BirthDate = new DateTime(1960, 5, 29),
                HireDate = new DateTime(1994, 1, 2),
                Position = "Sales Representative",
                Address = "RG1 9SP, Edgeham Hollow Winchester Way, London, UK",
                Phone = "(71) 555-5598",
                Access = AccessLevel.User,
                OnVacation = false
            }
        );
        Employees.Add(
            new Employee("Laura Callahan") {
                BirthDate = new DateTime(1985, 1, 9),
                HireDate = new DateTime(2004, 3, 5),
                Position = "Inside Sales Coordinator",
                Address = "98105, 4726 - 11th Ave. N.E., Seattle WA, USA",
                Phone = "(206) 555-1189",
                Access = AccessLevel.User,
                OnVacation = true
            }
        );
        Employees.Add(
            new Employee("Anne Dodsworth") {
                BirthDate = new DateTime(1980, 1, 27),
                HireDate = new DateTime(2004, 11, 15),
                Position = "Sales Representative",
                Address = "WG2 7LT, 7 Houndstooth Rd., London, UK",
                Phone = "(71) 555-4444",
                Access = AccessLevel.User,
                OnVacation = false
            }
        );
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl ItemsSource="{Binding Employees}" RowHeight="100">
    <dxGrid:GridControl.Columns>
        <dxGrid:ImageColumn FieldName="Photo" Width="100" />
        <dxGrid:TemplateColumn FieldName="Name" Caption="Employee" Width="
                               IsReadOnly="true" AllowGroup="False" AllowS
          <dxGrid:TemplateColumn.DisplayTemplate>
            <DataTemplate>
              <Grid BindingContext="{Binding Source}" Padding="0,15,0,15">
                <Grid.RowDefinitions>
                  <RowDefinition Height="Auto" />
                  <RowDefinition Height="Auto" />
                  <RowDefinition Height="Auto" />
                </Grid.RowDefinitions>
                <Label Text="{Binding Name}" Font="Bold, 18" Grid.Row="0"
                <Label Text="{Binding Position, StringFormat = 'Title: {0}
                <Label Text="{Binding HireDate, StringFormat = 'Hire Date:
              </Grid>
            </DataTemplate>
          </dxGrid:TemplateColumn.DisplayTemplate>
        </dxGrid:TemplateColumn>
        <dxGrid:NumberColumn FieldName="Phone" ContentAlignment="Center"/>
        <dxGrid:TextColumn FieldName="Address" Width="170"/>
        <dxGrid:DateColumn FieldName="BirthDate" DisplayFormat="d"/>
        <dxGrid:PickerColumn FieldName="Access" Caption="Access Level" Con
        <dxGrid:SwitchColumn FieldName="OnVacation"/>
    </dxGrid:GridControl.Columns>
</dxGrid:GridControl>
```

## Inheritance Hierarchy

System.Object
  Xamarin.Forms.BindableObject
    GridColumn
      **TemplateColumn**

## See Also

TemplateColumn Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# TemplateColumn Members

A column type that allows you to define a custom template for column cells.

The following tables list the members exposed by the TemplateColumn type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | TemplateColumn | Initializes a new instance of the TemplateColumn class with default settings. |

### Public Properties

| | Name | Description |
|---|---|---|
| | ActualCaption | Gets the column caption as it is displayed in a column header. (Inherited from **GridColumn**) |
| | ActualWidth | Gets the column width. This is a bindable property. (Inherited from **GridColumn**) |
| | AllowAutoFilter | Gets or sets whether an end-user can filter data by the column's values using the auto-filter panel. This is a bindable property. (Inherited from **GridColumn**) |
| | AllowGroup | Gets or sets whether end-users are allowed to group data by the current column. This is a bindable property. (Inherited from **GridColumn**) |
| | AllowSort | Gets or sets whether end-users are allowed to sort data by the column's values. This is a bindable property. (Inherited from **GridColumn**) |
| | AutoFilterCondition | Gets or sets the comparison operator used to create filter conditions for the current column via the auto-filter panel. This is a bindable property. (Inherited from **GridColumn**) |
| | AutoFilterValue | Gets or sets the value in the auto-filter panel for the column. This is a bindable property. (Inherited from **GridColumn**) |
| | BindingContext | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Caption | Gets or sets the column's display caption. This is a bindable property. (Inherited from **GridColumn**) |
| | ColumnFilterMode | Gets or sets how column values are filtered via the auto-filter panel. This is a bindable property. (Inherited from **GridColumn**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | ContentAlignment | Gets or sets the content alignment within column cells. This is a bindable property. (Inherited from **GridColumn**) |
| | DisplayFormat | Gets or sets formatting for values in the column. This is a bindable property. (Inherited from **GridColumn**) |
| | DisplayTemplate | Gets or sets a template specifying a custom layout to display data values within column cells. This is a bindable property. |
| | EditTemplate | Gets or sets a template applied for the column values editor. This is a bindable property. |
| | FieldName | Gets or sets the column's unique field name. For bound columns, this property refers to a field in the underlying data source. For unbound columns, this property serves as a column's identifier.<br>This is a bindable property.<br>(Inherited from **GridColumn**) |
| | FixedStyle | Gets or sets whether a column is fixed. This is a bindable property. (Inherited from **GridColumn**) |
| | GroupInterval | Gets or sets how data rows are grouped when grouping by the current column is applied. This is a bindable property. (Inherited from **GridColumn**) |
| | HeaderTemplate | Gets or sets a template used to define the column headers visual representation. (Inherited from **GridColumn**) |
| | ImmediateUpdateAutoFilter | Gets or sets whether the column's filter condition is updated as soon as an end-user modifies the text within the auto filter panel's cell. This is a bindable property. (Inherited from **GridColumn**) |
| | IsGrouped | Gets or sets whether the grid's data is grouped by the current column. This is a bindable property. (Inherited from **GridColumn**) |
| | IsReadOnly | Gets or sets whether end-users are allowed to change cell values in the column. This is a bindable property. (Inherited from **GridColumn**) |
| | IsUnbound | Gets whether the column is unbound. (Inherited from **GridColumn**) |
| | IsVisible | Gets or sets whether the column is visible in the grid. This is a bindable property. (Inherited from **GridColumn**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | MinWidth | Gets or sets the minimum column width allowed. This is a bindable property. (Inherited from **GridColumn**) |
| | SerializationTypeName | Gets the string containing the column type name. (Inherited from **GridColumn**) |
| | SortIndex | Gets or sets the column's position among sorted columns. This is a bindable property. (Inherited from **GridColumn**) |
| | SortMode | Specifies how column values are sorted and grouped. This is a bindable property. (Inherited from **GridColumn**) |
| | SortOrder | Gets or sets the column's sort order. This is a bindable property. (Inherited from **GridColumn**) |
| | UnboundExpression | Gets or sets an expression used to evaluate values for the current Unbound Columns. This is a bindable property. (Inherited from **GridColumn**) |
| | UnboundType | Gets or sets the data type and binding mode of the column. This is a bindable property. (Inherited from **GridColumn**) |
| | Width | Gets or sets the column width. This is a bindable property. (Inherited from **GridColumn**) |

**Public Methods**

| | Name | Description |
|---|---|---|
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Equals | (Inherited from **System.Object**) |
| S | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | ToString | (Inherited from **System.Object**) |

**Public Fields**

| | Name | Description |
|---|---|---|
| | ActualWidthProperty | Identifies the GridColumn.ActualWidth bindable property. (Inherited from **GridColumn**) |
| | AllowAutoFilterProperty | Identifies the GridColumn.AllowAutoFilter bindable property. (Inherited from **GridColumn**) |
| | AllowGroupProperty | Identifies the GridColumn.AllowGroup bindable property. (Inherited from **GridColumn**) |
| | AllowSortProperty | Identifies the GridColumn.AllowSort bindable property. (Inherited from **GridColumn**) |
| | AutoFilterConditionProperty | Identifies the GridColumn.AutoFilterCondition bindable property. (Inherited from **GridColumn**) |
| | AutoFilterValueProperty | Identifies the GridColumn.AutoFilterValue bindable property. (Inherited from **GridColumn**) |
| | BindingContextProperty | (Inherited from **Xamarin.Forms.BindableObject**) |
| | CaptionProperty | Identifies the GridColumn.Caption bindable property. (Inherited from **GridColumn**) |
| | ColumnFilterModeProperty | Identifies the GridColumn.ColumnFilterMode bindable property. (Inherited from **GridColumn**) |
| | ContentAlignmentProperty | Identifies the GridColumn.ContentAlignment bindable property. (Inherited from **GridColumn**) |
| | DisplayFormatProperty | Identifies the GridColumn.DisplayFormat bindable property. (Inherited from **GridColumn**) |
| | DisplayTemplateProperty | Identifies the DisplayTemplate bindable property. |
| | EditTemplateProperty | Identifies the EditTemplate bindable property. |
| | FieldNameProperty | Identifies the GridColumn.FieldName bindable property. (Inherited from **GridColumn**) |
| | FixedStyleProperty | Identifies the GridColumn.FixedStyle bindable property. (Inherited from |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | | **GridColumn**) |
| S | GroupIntervalProperty | Identifies the GridColumn.GroupInterval bindable property. (Inherited from **GridColumn**) |
| S | HeaderTemplateProperty | Identifies the GridColumn.HeaderTemplate bindable property. (Inherited from **GridColumn**) |
| S | ImmediateUpdateAutoFilterProperty | Identifies the GridColumn.ImmediateUpdateAutoFilter bindable property. (Inherited from **GridColumn**) |
| S | IsGroupedProperty | Identifies the GridColumn.IsGrouped bindable property. (Inherited from **GridColumn**) |
| S | IsReadOnlyProperty | Identifies the GridColumn.IsReadOnly bindable property. (Inherited from **GridColumn**) |
| S | IsVisibleProperty | Identifies the GridColumn.IsVisible bindable property. (Inherited from **GridColumn**) |
| S | MinWidthProperty | Identifies the GridColumn.MinWidth bindable property. (Inherited from **GridColumn**) |
| S | SortIndexProperty | Identifies the GridColumn.SortIndex bindable property. (Inherited from **GridColumn**) |
| S | SortModeProperty | Identifies the GridColumn.SortMode bindable property. (Inherited from **GridColumn**) |
| S | SortOrderProperty | Identifies the GridColumn.SortOrder bindable property. (Inherited from **GridColumn**) |
| S | UnboundExpressionProperty | Identifies the GridColumn.UnboundExpression bindable property. (Inherited from **GridColumn**) |
| S | UnboundTypeProperty | Identifies the GridColumn.UnboundType bindable property. (Inherited from **GridColumn**) |
| S | WidthProperty | Identifies the GridColumn.Width bindable property. (Inherited from **GridColumn**) |

**Public Events**

| | Name | Description |
|---|---|---|
| | BindingContextChanged | (Inherited from **Xamarin.Forms.BindableObject**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | PropertyChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanging | (Inherited from **Xamarin.Forms.BindableObject**) |

**See Also**
TemplateColumn Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# TemplateColumn Constructor

Initializes a new instance of the TemplateColumn class with default settings.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**
**C#**

```
public TemplateColumn()
```

**See Also**
TemplateColumn Class
TemplateColumn Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# TemplateColumn Properties

A column type that allows you to define a custom template for column cells.

The following tables list the members exposed by the TemplateColumn type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | TemplateColumn | Initializes a new instance of the TemplateColumn class with default settings. |

### Public Properties

| | Name | Description |
|---|---|---|
| | ActualCaption | Gets the column caption as it is displayed in a column header. (Inherited from **GridColumn**) |
| | ActualWidth | Gets the column width. This is a bindable property. (Inherited from **GridColumn**) |
| | AllowAutoFilter | Gets or sets whether an end-user can filter data by the column's values using the auto-filter panel. This is a bindable property. (Inherited from **GridColumn**) |
| | AllowGroup | Gets or sets whether end-users are allowed to group data by the current column. This is a bindable property. (Inherited from **GridColumn**) |
| | AllowSort | Gets or sets whether end-users are allowed to sort data by the column's values. This is a bindable property. (Inherited from **GridColumn**) |
| | AutoFilterCondition | Gets or sets the comparison operator used to create filter conditions for the current column via the auto-filter panel. This is a bindable property. (Inherited from **GridColumn**) |
| | AutoFilterValue | Gets or sets the value in the auto-filter panel for the column. This is a bindable property. (Inherited from **GridColumn**) |
| | BindingContext | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Caption | Gets or sets the column's display caption. This is a bindable property. (Inherited from **GridColumn**) |
| | ColumnFilterMode | Gets or sets how column values are filtered via the auto-filter panel. This is a bindable property. (Inherited from **GridColumn**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | ContentAlignment | Gets or sets the content alignment within column cells. This is a bindable property. (Inherited from **GridColumn**) |
| | DisplayFormat | Gets or sets formatting for values in the column. This is a bindable property. (Inherited from **GridColumn**) |
| | DisplayTemplate | Gets or sets a template specifying a custom layout to display data values within column cells. This is a bindable property. |
| | EditTemplate | Gets or sets a template applied for the column values editor. This is a bindable property. |
| | FieldName | Gets or sets the column's unique field name. For bound columns, this property refers to a field in the underlying data source. For unbound columns, this property serves as a column's identifier.<br>This is a bindable property.<br>(Inherited from **GridColumn**) |
| | FixedStyle | Gets or sets whether a column is fixed. This is a bindable property. (Inherited from **GridColumn**) |
| | GroupInterval | Gets or sets how data rows are grouped when grouping by the current column is applied. This is a bindable property. (Inherited from **GridColumn**) |
| | HeaderTemplate | Gets or sets a template used to define the column headers visual representation. (Inherited from **GridColumn**) |
| | ImmediateUpdateAutoFilter | Gets or sets whether the column's filter condition is updated as soon as an end-user modifies the text within the auto filter panel's cell. This is a bindable property. (Inherited from **GridColumn**) |
| | IsGrouped | Gets or sets whether the grid's data is grouped by the current column. This is a bindable property. (Inherited from **GridColumn**) |
| | IsReadOnly | Gets or sets whether end-users are allowed to change cell values in the column. This is a bindable property. (Inherited from **GridColumn**) |
| | IsUnbound | Gets whether the column is unbound. (Inherited from **GridColumn**) |
| | IsVisible | Gets or sets whether the column is visible in the grid. This is a bindable property. (Inherited from **GridColumn**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | MinWidth | Gets or sets the minimum column width allowed. This is a bindable property. (Inherited from **GridColumn**) |
| | SerializationTypeName | Gets the string containing the column type name. (Inherited from **GridColumn**) |
| | SortIndex | Gets or sets the column's position among sorted columns. This is a bindable property. (Inherited from **GridColumn**) |
| | SortMode | Specifies how column values are sorted and grouped. This is a bindable property. (Inherited from **GridColumn**) |
| | SortOrder | Gets or sets the column's sort order. This is a bindable property. (Inherited from **GridColumn**) |
| | UnboundExpression | Gets or sets an expression used to evaluate values for the current Unbound Columns. This is a bindable property. (Inherited from **GridColumn**) |
| | UnboundType | Gets or sets the data type and binding mode of the column. This is a bindable property. (Inherited from **GridColumn**) |
| | Width | Gets or sets the column width. This is a bindable property. (Inherited from **GridColumn**) |

**Public Methods**

| | Name | Description |
|---|---|---|
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Equals | (Inherited from **System.Object**) |
| S | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | ToString | (Inherited from **System.Object**) |

**Public Fields**

| | Name | Description |
|---|---|---|
| | ActualWidthProperty | Identifies the GridColumn.ActualWidth bindable property. (Inherited from **GridColumn**) |
| | AllowAutoFilterProperty | Identifies the GridColumn.AllowAutoFilter bindable property. (Inherited from **GridColumn**) |
| | AllowGroupProperty | Identifies the GridColumn.AllowGroup bindable property. (Inherited from **GridColumn**) |
| | AllowSortProperty | Identifies the GridColumn.AllowSort bindable property. (Inherited from **GridColumn**) |
| | AutoFilterConditionProperty | Identifies the GridColumn.AutoFilterCondition bindable property. (Inherited from **GridColumn**) |
| | AutoFilterValueProperty | Identifies the GridColumn.AutoFilterValue bindable property. (Inherited from **GridColumn**) |
| | BindingContextProperty | (Inherited from **Xamarin.Forms.BindableObject**) |
| | CaptionProperty | Identifies the GridColumn.Caption bindable property. (Inherited from **GridColumn**) |
| | ColumnFilterModeProperty | Identifies the GridColumn.ColumnFilterMode bindable property. (Inherited from **GridColumn**) |
| | ContentAlignmentProperty | Identifies the GridColumn.ContentAlignment bindable property. (Inherited from **GridColumn**) |
| | DisplayFormatProperty | Identifies the GridColumn.DisplayFormat bindable property. (Inherited from **GridColumn**) |
| | DisplayTemplateProperty | Identifies the DisplayTemplate bindable property. |
| | EditTemplateProperty | Identifies the EditTemplate bindable property. |
| | FieldNameProperty | Identifies the GridColumn.FieldName bindable property. (Inherited from **GridColumn**) |
| | FixedStyleProperty | Identifies the GridColumn.FixedStyle bindable property. (Inherited from |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | | **GridColumn**) |
| S | GroupIntervalProperty | Identifies the GridColumn.GroupInterval bindable property. (Inherited from **GridColumn**) |
| S | HeaderTemplateProperty | Identifies the GridColumn.HeaderTemplate bindable property. (Inherited from **GridColumn**) |
| S | ImmediateUpdateAutoFilterProperty | Identifies the GridColumn.ImmediateUpdateAutoFilter bindable property. (Inherited from **GridColumn**) |
| S | IsGroupedProperty | Identifies the GridColumn.IsGrouped bindable property. (Inherited from **GridColumn**) |
| S | IsReadOnlyProperty | Identifies the GridColumn.IsReadOnly bindable property. (Inherited from **GridColumn**) |
| S | IsVisibleProperty | Identifies the GridColumn.IsVisible bindable property. (Inherited from **GridColumn**) |
| S | MinWidthProperty | Identifies the GridColumn.MinWidth bindable property. (Inherited from **GridColumn**) |
| S | SortIndexProperty | Identifies the GridColumn.SortIndex bindable property. (Inherited from **GridColumn**) |
| S | SortModeProperty | Identifies the GridColumn.SortMode bindable property. (Inherited from **GridColumn**) |
| S | SortOrderProperty | Identifies the GridColumn.SortOrder bindable property. (Inherited from **GridColumn**) |
| S | UnboundExpressionProperty | Identifies the GridColumn.UnboundExpression bindable property. (Inherited from **GridColumn**) |
| S | UnboundTypeProperty | Identifies the GridColumn.UnboundType bindable property. (Inherited from **GridColumn**) |
| S | WidthProperty | Identifies the GridColumn.Width bindable property. (Inherited from **GridColumn**) |

**Public Events**

| | Name | Description |
|---|---|---|
| | BindingContextChanged | (Inherited from **Xamarin.Forms.BindableObject**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | PropertyChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanging | (Inherited from **Xamarin.Forms.BindableObject**) |

**See Also**
TemplateColumn Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# DisplayTemplate Property

Gets or sets a template specifying a custom layout to display data values within column cells. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public DataTemplate DisplayTemplate { get; set; }
```

### Property Value

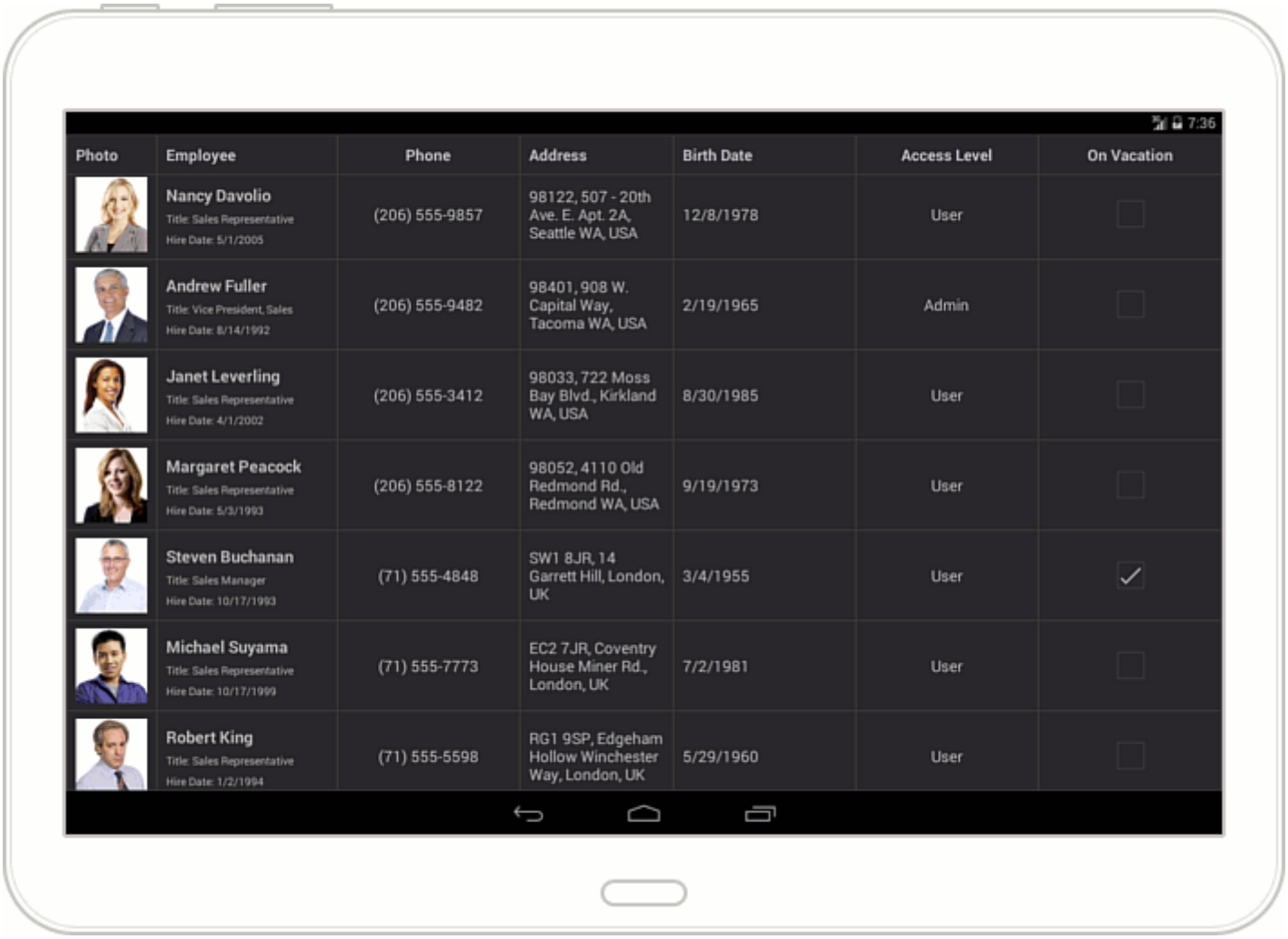
A Xamarin.Forms.DataTemplate object.

### Example

| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T208848. | |

This example shows how to add columns to GridControl to display and edit data of different types. The grid is bound to a collection of **Employee** objects. Each **Employee** object contains an employee's photo (image), name, position, phone, address (strings), hire and birth dates (DateTime values), a value that specifies an employee's access level, and a Boolean value indicating whether an employee is on vacation.

© 2018 Developer Express Inc.




Grid columns are stored in the GridControl.Columns collection. An individual column is specified by a GridColumn descendant, which is appropriate to the type of data contained in this column. In this example, to display information on employees and allow end-users to edit it, the following columns are created in the grid.

| Grid Column | Description |
|---|---|
| Photo<br>ImageColumn | This column displays employee photos which are images added to a project as embedded resources. |
| Employee<br>TemplateColumn | A cell in this column displays three employee properties: Name, Position and HireDate. A cell appearance is defined via a template. Each cell contains three Xamarin.Forms.Label elements within a Xamarin.Forms.Grid. Each Label element is bound to a property of the **Employee** class.<br><br>A cell template data context is specified by the CellData object. Its CellData.Value property provides access to a value of a data field assigned to the column's GridColumn.FieldName property. In this example, a column cell displays not only this field value but also values of two more fields. So, it is required to specify the whole data row as a binding source for the template. To do this, set the BindingContext property of the Grid to CellData.Source. Note that BindingContext specified for the Grid is inherited by all its children. This means that all Label elements have the same BindingContext as the Grid, and you can specify |

© 2018 Developer Express Inc.

| | |
|---|---|
| | their simple bindings to properties of that object (Employee).<br><br>Use the GridColumn.AllowSort, GridColumn.AllowGroup and GridColumn.IsReadOnly properties to prevent end-users from sorting and grouping data in the grid by this column and disable data editing. |
| Phone<br>NumberColumn | The employee's **Phone** property is of the string type. The keyboard allowing text input is automatically displayed when an end-user activates this column cell to edit an employee's phone. |
| Address<br>TextColumn | This column is bound to the employee's **Address** string property. The keyboard allowing text input is automatically displayed when an end-user activates this column cell to edit an employee's address. |
| Birth Date<br>DateColumn | This column displays and allows editing employee birth days. The GridColumn.DisplayFormat property specifies the format of displaying dates. |
| Access Level<br>PickerColumn | The employee's **Access** property is of the **AccessLevel** type which is an enumeration with values **Admin** and **User**. To allow editing an employee's access level by selecting one of two enumeration values, this property is bound to the grid's picker column. |
| On Vacation<br>SwitchColumn | The employee's **OnVacation** property is of the Boolean type, so it is bound to the grid's switch column which allows editing a cell value by selecting between two states. |

**C#**

```
 (Employee.cs)
public enum AccessLevel {
    Admin,
    User
}
public class Employee : ModelObject {
    string name;
    public string Name {
        get { return name; }
        set {
            name = value;
            if (Photo == null)
                Photo = ImageSource.FromResource (value.Replace (" ", String.Empty));
        }
    }
    public Employee(string name) {
        this.Name = name;
    }
    public ImageSource Photo { get; set; }
    public DateTime BirthDate { get; set; }
    public DateTime HireDate { get; set; }
    public string Position { get; set; }
    public string Address { get; set; }
    public string Phone { get; set; }
    public AccessLevel Access { get; set; }
    public bool OnVacation { get; set; }
}
```

**C#**

```
 (EmployeesRepository.cs)
public abstract class EmployeesRepository {
    readonly ObservableCollection<Employee> employees;
    public EmployeesRepository() {
        this.employees = new ObservableCollection<Employee>();
    }
    public ObservableCollection<Employee> Employees {
        get { return employees; }
    }
}
public class TestEmployeesRepository : EmployeesRepository {
    public TestEmployeesRepository() : base() {
        GenerateCustomers();
    }
    void GenerateCustomers() {
        Employees.Add(
            new Employee("Nancy Davolio") {
                BirthDate = new DateTime(1978, 12, 8),
                HireDate = new DateTime(2005, 5, 1),
                Position = "Sales Representative",
                Address = "98122, 507 - 20th Ave. E. Apt. 2A, Seattle WA, USA",
                Phone = "(206) 555-9857",
                Access = AccessLevel.User,
                OnVacation = false
            }
        );
        Employees.Add(
            new Employee("Andrew Fuller") {
                BirthDate = new DateTime(1965, 2, 19),
                HireDate = new DateTime(1992, 8, 14),
                Position = "Vice President, Sales",
                Address = "98401, 908 W. Capital Way, Tacoma WA, USA",
                Phone = "(206) 555-9482",
                Access = AccessLevel.Admin,
                OnVacation = false
            }
        );
        Employees.Add(
            new Employee("Janet Leverling") {
                BirthDate = new DateTime(1985, 8, 30),
                HireDate = new DateTime(2002, 4, 1),
                Position = "Sales Representative",
                Address = "98033, 722 Moss Bay Blvd., Kirkland WA, USA",
                Phone = "(206) 555-3412",
                Access = AccessLevel.User,
                OnVacation = false
            }
        );
        Employees.Add(
            new Employee("Margaret Peacock") {
                BirthDate = new DateTime(1973, 9, 19),
                HireDate = new DateTime(1993, 5, 3),
                Position = "Sales Representative",
                Address = "98052, 4110 Old Redmond Rd., Redmond WA, USA",
                Phone = "(206) 555-8122",
                Access = AccessLevel.User,
                OnVacation = false
            }
        );
        Employees.Add(
            new Employee("Steven Buchanan") {
                BirthDate = new DateTime(1955, 3, 4),
                HireDate = new DateTime(1993, 10, 17),
                Position = "Sales Manager",
                Address = "SW1 8JR, 14 Garrett Hill, London, UK",
```

© 2018 Developer Express Inc.

```
                Phone = "(71) 555-4848",
                Access = AccessLevel.User,
                OnVacation = true
            }
        );
        Employees.Add(
            new Employee("Michael Suyama") {
                BirthDate = new DateTime(1981, 7, 2),
                HireDate = new DateTime(1999, 10, 17),
                Position = "Sales Representative",
                Address = "EC2 7JR, Coventry House Miner Rd., London, UK",
                Phone = "(71) 555-7773",
                Access = AccessLevel.User,
                OnVacation = false
            }
        );
        Employees.Add(
            new Employee("Robert King") {
                BirthDate = new DateTime(1960, 5, 29),
                HireDate = new DateTime(1994, 1, 2),
                Position = "Sales Representative",
                Address = "RG1 9SP, Edgeham Hollow Winchester Way, London, UK",
                Phone = "(71) 555-5598",
                Access = AccessLevel.User,
                OnVacation = false
            }
        );
        Employees.Add(
            new Employee("Laura Callahan") {
                BirthDate = new DateTime(1985, 1, 9),
                HireDate = new DateTime(2004, 3, 5),
                Position = "Inside Sales Coordinator",
                Address = "98105, 4726 - 11th Ave. N.E., Seattle WA, USA",
                Phone = "(206) 555-1189",
                Access = AccessLevel.User,
                OnVacation = true
            }
        );
        Employees.Add(
            new Employee("Anne Dodsworth") {
                BirthDate = new DateTime(1980, 1, 27),
                HireDate = new DateTime(2004, 11, 15),
                Position = "Sales Representative",
                Address = "WG2 7LT, 7 Houndstooth Rd., London, UK",
                Phone = "(71) 555-4444",
                Access = AccessLevel.User,
                OnVacation = false
            }
        );
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl ItemsSource="{Binding Employees}" RowHeight="100">
    <dxGrid:GridControl.Columns>
        <dxGrid:ImageColumn FieldName="Photo" Width="100" />
        <dxGrid:TemplateColumn FieldName="Name" Caption="Employee" Width="
                               IsReadOnly="true" AllowGroup="False" AllowS
          <dxGrid:TemplateColumn.DisplayTemplate>
            <DataTemplate>
              <Grid BindingContext="{Binding Source}" Padding="0,15,0,15">
                <Grid.RowDefinitions>
                  <RowDefinition Height="Auto" />
                  <RowDefinition Height="Auto" />
                  <RowDefinition Height="Auto" />
                </Grid.RowDefinitions>
                <Label Text="{Binding Name}" Font="Bold, 18" Grid.Row="0"
                <Label Text="{Binding Position, StringFormat = 'Title: {0}
                <Label Text="{Binding HireDate, StringFormat = 'Hire Date:
              </Grid>
            </DataTemplate>
          </dxGrid:TemplateColumn.DisplayTemplate>
        </dxGrid:TemplateColumn>
        <dxGrid:NumberColumn FieldName="Phone" ContentAlignment="Center"/>
        <dxGrid:TextColumn FieldName="Address" Width="170"/>
        <dxGrid:DateColumn FieldName="BirthDate" DisplayFormat="d"/>
        <dxGrid:PickerColumn FieldName="Access" Caption="Access Level" Con
        <dxGrid:SwitchColumn FieldName="OnVacation"/>
    </dxGrid:GridControl.Columns>
</dxGrid:GridControl>
```

## See Also

TemplateColumn Class
TemplateColumn Members
DevExpress.Mobile.DataGrid Namespace
EditTemplate

© 2018 Developer Express Inc.

# EditTemplate Property

Gets or sets a template applied for the column values editor. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public DataTemplate EditTemplate { get; set; }
```

### Property Value

A Xamarin.Forms.DataTemplate object.

### Remarks

GridControl provides various ways to customize the appearance of its elements. For simple customizations (such as modifying colors, fonts, etc.) use appropriate customizer objects. If you need to completely modify the appearance of a particular grid element, you can define a custom template for it.

The **EditTemplate** property allows you to define a template for the inplace edit form. For this purpose, create a column of the TemplateColumn type, assign the required template to the column's **EditTemplate** property and add the templated column to the GridControl.Columns collection.

To customize the popup edit form, follow the steps from the How to: Implement a Custom Row Edit Form document.

| Tip |
| --- |
| Also, you can define a data template for the column cells when they display column data values and for the column headers. For this purpose, use the DisplayTemplate or GridColumn.HeaderTemplate property respectively. |

### See Also

TemplateColumn Class
TemplateColumn Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# TextColumn Class

A grid column to display and edit text values.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

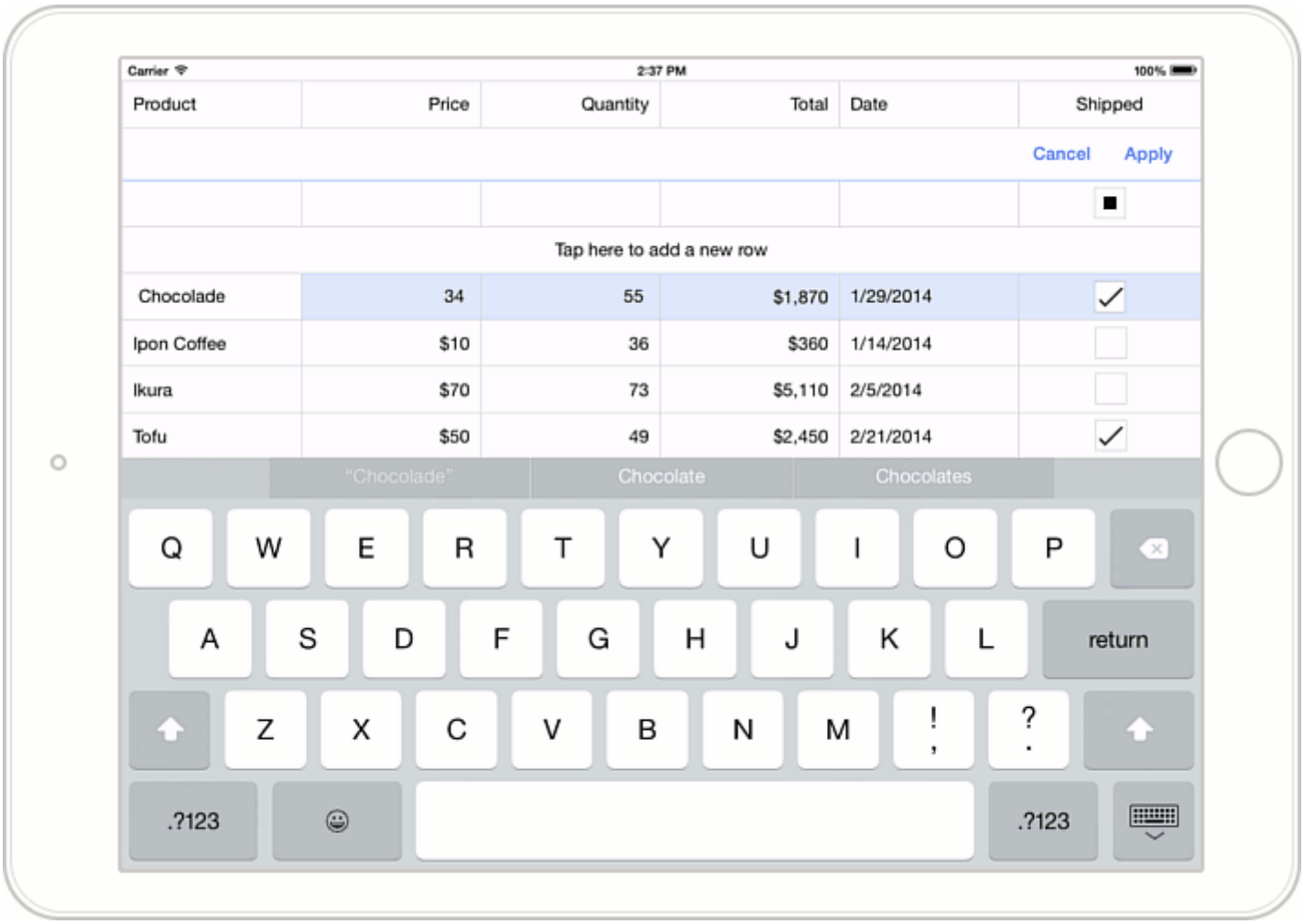
**C#**

```
public class TextColumn : GridColumn
```

### Remarks

In GridControl, data fields are represented as columns and records are displayed as rows. So, to display data, a grid should contain columns bound to fields of the underlying data source.

An individual column is specified by the **TextColumn**, NumberColumn, DateColumn or SwitchColumn object (a GridColumn descendant) depending on the type of data displayed in this column. Use columns of the **TextColumn** type to display text values and allow end-user to edit them. When an end-user activates a text column cell (double-taps a cell, or touches and holds a cell and then selects **Edit Cell** from the invoked menu), the keyboard is automatically displayed, allowing text input.




Column objects provide a set of properties to adjust column settings (for example, GridColumn.DisplayFormat, GridColumn.Width, GridColumn.Caption, GridColumn.IsVisible, etc.).

GridControl stores its columns in the GridControl.Columns collection, and displays them in the order in which they are added to this collection.

### Example

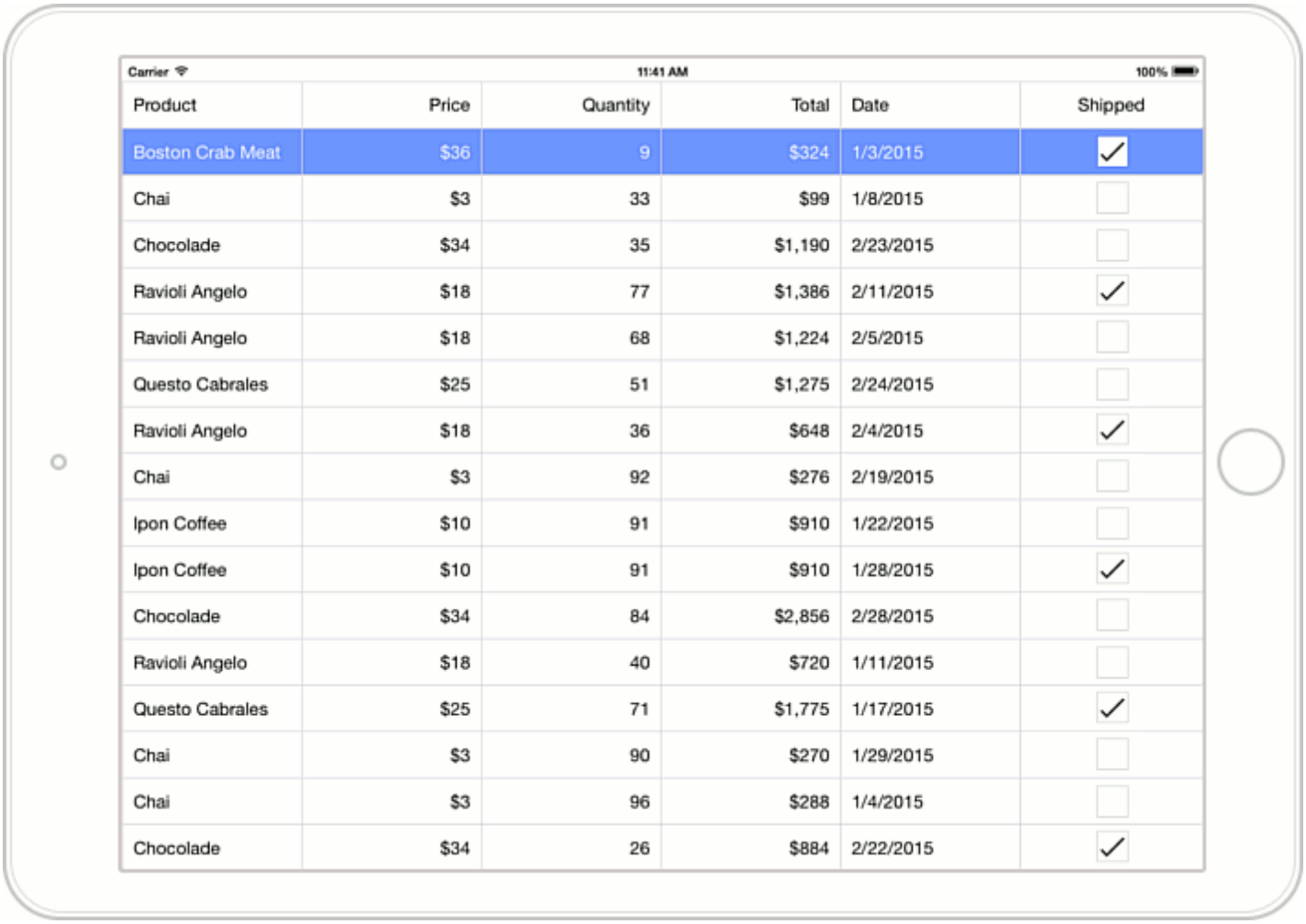
**Show Me**

© 2018 Developer Express Inc.

| | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T178764. | |

This example shows how to create and customize grid columns for displaying and editing data of different types (text, numbers, dates and Boolean values). The specified collection contains columns bound to the data source fields (*Product.Name*, *Product.UnitPrice*, *Quantity*, *Date* and *Shipped*) and one unbound column (*Total*) displaying data values calculated according to a formula based on the values of other columns.

The image below illustrates the result.




© 2018 Developer Express Inc.

**Xaml**

```
 (MainPage.xaml)
<dxGrid:GridControl.Columns>
        <dxGrid:TextColumn FieldName="Product.Name" Caption = "Product" Width = "170"
                           SortOrder = "Descending" SortIndex = "0"/>
        <dxGrid:NumberColumn FieldName="Product.UnitPrice" Caption = "Price" DisplayFormat="C0"/>
        <dxGrid:NumberColumn FieldName="Quantity"
                             SortOrder = "Ascending" SortIndex = "1"/>
        <dxGrid:NumberColumn FieldName="Total"
                             UnboundType="Integer" UnboundExpression="[Quantity] * [Product.UnitPrice]"
                             DisplayFormat="C0" IsReadOnly="True"/>
        <dxGrid:DateColumn FieldName="Date" DisplayFormat="d"
                           IsGrouped = "true" GroupInterval = "Date"/>
        <dxGrid:SwitchColumn FieldName="Shipped" AllowSort = "False" AllowAutoFilter="false"/>
    </dxGrid:GridControl.Columns>
```

### Inheritance Hierarchy

System.Object
  Xamarin.Forms.BindableObject
    GridColumn
      **TextColumn**
        Derived classes

### See Also

TextColumn Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# TextColumn Members

A grid column to display and edit text values.

The following tables list the members exposed by the TextColumn type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | TextColumn | Initializes a new instance of the TextColumn class with default settings. |

### Public Properties

| | Name | Description |
|---|---|---|
| | ActualCaption | Gets the column caption as it is displayed in a column header. (Inherited from **GridColumn**) |
| | ActualWidth | Gets the column width. This is a bindable property. (Inherited from **GridColumn**) |
| | AllowAutoFilter | Gets or sets whether an end-user can filter data by the column's values using the auto-filter panel. This is a bindable property. (Inherited from **GridColumn**) |
| | AllowGroup | Gets or sets whether end-users are allowed to group data by the current column. This is a bindable property. (Inherited from **GridColumn**) |
| | AllowSort | Gets or sets whether end-users are allowed to sort data by the column's values. This is a bindable property. (Inherited from **GridColumn**) |
| | AutoFilterCondition | Gets or sets the comparison operator used to create filter conditions for the current column via the auto-filter panel. This is a bindable property. (Inherited from **GridColumn**) |
| | AutoFilterValue | Gets or sets the value in the auto-filter panel for the column. This is a bindable property. (Inherited from **GridColumn**) |
| | BindingContext | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Caption | Gets or sets the column's display caption. This is a bindable property. (Inherited from **GridColumn**) |
| | ColumnFilterMode | Gets or sets how column values are filtered via the auto-filter panel. This is a bindable property. (Inherited from **GridColumn**) |
| | ContentAlignment | Gets or sets the content alignment within column cells. This is a bindable |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | | property. (Inherited from **GridColumn**) |
| | DisplayFormat | Gets or sets formatting for values in the column. This is a bindable property. (Inherited from **GridColumn**) |
| | FieldName | Gets or sets the column's unique field name. For bound columns, this property refers to a field in the underlying data source. For unbound columns, this property serves as a column's identifier.<br>This is a bindable property.<br>(Inherited from **GridColumn**) |
| | FixedStyle | Gets or sets whether a column is fixed. This is a bindable property. (Inherited from **GridColumn**) |
| | GroupInterval | Gets or sets how data rows are grouped when grouping by the current column is applied. This is a bindable property. (Inherited from **GridColumn**) |
| | HeaderTemplate | Gets or sets a template used to define the column headers visual representation. (Inherited from **GridColumn**) |
| | ImmediateUpdateAutoFilter | Gets or sets whether the column's filter condition is updated as soon as an end-user modifies the text within the auto filter panel's cell. This is a bindable property. (Inherited from **GridColumn**) |
| | IsGrouped | Gets or sets whether the grid's data is grouped by the current column. This is a bindable property. (Inherited from **GridColumn**) |
| | IsReadOnly | Gets or sets whether end-users are allowed to change cell values in the column. This is a bindable property. (Inherited from **GridColumn**) |
| | IsUnbound | Gets whether the column is unbound. (Inherited from **GridColumn**) |
| | IsVisible | Gets or sets whether the column is visible in the grid. This is a bindable property. (Inherited from **GridColumn**) |
| | MinWidth | Gets or sets the minimum column width allowed. This is a bindable property. (Inherited from **GridColumn**) |
| | SerializationTypeName | Gets the string containing the column type name. (Inherited from **GridColumn**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | SortIndex | Gets or sets the column's position among sorted columns. This is a bindable property. (Inherited from **GridColumn**) |
| | SortMode | Specifies how column values are sorted and grouped. This is a bindable property. (Inherited from **GridColumn**) |
| | SortOrder | Gets or sets the column's sort order. This is a bindable property. (Inherited from **GridColumn**) |
| | UnboundExpression | Gets or sets an expression used to evaluate values for the current Unbound Columns. This is a bindable property. (Inherited from **GridColumn**) |
| | UnboundType | Gets or sets the data type and binding mode of the column. This is a bindable property. (Inherited from **GridColumn**) |
| | Width | Gets or sets the column width. This is a bindable property. (Inherited from **GridColumn**) |

**Public Methods**

| | Name | Description |
|---|---|---|
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ToString | (Inherited from **System.Object**) |

**Public Fields**

| | Name | Description |
|---|---|---|

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| S | ActualWidthProperty | Identifies the GridColumn.ActualWidth bindable property. (Inherited from **GridColumn**) |
| S | AllowAutoFilterProperty | Identifies the GridColumn.AllowAutoFilter bindable property. (Inherited from **GridColumn**) |
| S | AllowGroupProperty | Identifies the GridColumn.AllowGroup bindable property. (Inherited from **GridColumn**) |
| S | AllowSortProperty | Identifies the GridColumn.AllowSort bindable property. (Inherited from **GridColumn**) |
| S | AutoFilterConditionProperty | Identifies the GridColumn.AutoFilterCondition bindable property. (Inherited from **GridColumn**) |
| S | AutoFilterValueProperty | Identifies the GridColumn.AutoFilterValue bindable property. (Inherited from **GridColumn**) |
| S | BindingContextProperty | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | CaptionProperty | Identifies the GridColumn.Caption bindable property. (Inherited from **GridColumn**) |
| S | ColumnFilterModeProperty | Identifies the GridColumn.ColumnFilterMode bindable property. (Inherited from **GridColumn**) |
| S | ContentAlignmentProperty | Identifies the GridColumn.ContentAlignment bindable property. (Inherited from **GridColumn**) |
| S | DisplayFormatProperty | Identifies the GridColumn.DisplayFormat bindable property. (Inherited from **GridColumn**) |
| S | FieldNameProperty | Identifies the GridColumn.FieldName bindable property. (Inherited from **GridColumn**) |
| S | FixedStyleProperty | Identifies the GridColumn.FixedStyle bindable property. (Inherited from **GridColumn**) |
| S | GroupIntervalProperty | Identifies the GridColumn.GroupInterval bindable property. (Inherited from **GridColumn**) |
| S | HeaderTemplateProperty | Identifies the GridColumn.HeaderTemplate bindable property. (Inherited from **GridColumn**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | ImmediateUpdateAutoFilterProperty | Identifies the GridColumn.ImmediateUpdateAutoFilter bindable property. (Inherited from **GridColumn**) |
| | IsGroupedProperty | Identifies the GridColumn.IsGrouped bindable property. (Inherited from **GridColumn**) |
| | IsReadOnlyProperty | Identifies the GridColumn.IsReadOnly bindable property. (Inherited from **GridColumn**) |
| | IsVisibleProperty | Identifies the GridColumn.IsVisible bindable property. (Inherited from **GridColumn**) |
| | MinWidthProperty | Identifies the GridColumn.MinWidth bindable property. (Inherited from **GridColumn**) |
| | SortIndexProperty | Identifies the GridColumn.SortIndex bindable property. (Inherited from **GridColumn**) |
| | SortModeProperty | Identifies the GridColumn.SortMode bindable property. (Inherited from **GridColumn**) |
| | SortOrderProperty | Identifies the GridColumn.SortOrder bindable property. (Inherited from **GridColumn**) |
| | UnboundExpressionProperty | Identifies the GridColumn.UnboundExpression bindable property. (Inherited from **GridColumn**) |
| | UnboundTypeProperty | Identifies the GridColumn.UnboundType bindable property. (Inherited from **GridColumn**) |
| | WidthProperty | Identifies the GridColumn.Width bindable property. (Inherited from **GridColumn**) |

**Public Events**

| | Name | Description |
|---|---|---|
| | BindingContextChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanging | (Inherited from **Xamarin.Forms.BindableObject**) |

**See Also**
TextColumn Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# TextColumn Constructor

Initializes a new instance of the TextColumn class with default settings.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**
**C#**

```
public TextColumn()
```

**See Also**
TextColumn Class
TextColumn Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# ThemeFontAttributes Class

Allows you to specify required font attributes for the grid's themes.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public class ThemeFontAttributes : object
```

### Example

| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T297553. | |

This example demonstrates how to customize a theme used to paint a GridControl.

To apply a theme to a **GridControl**, assign its name to the ThemeManager.ThemeName property. The Themes class stores available theme names.

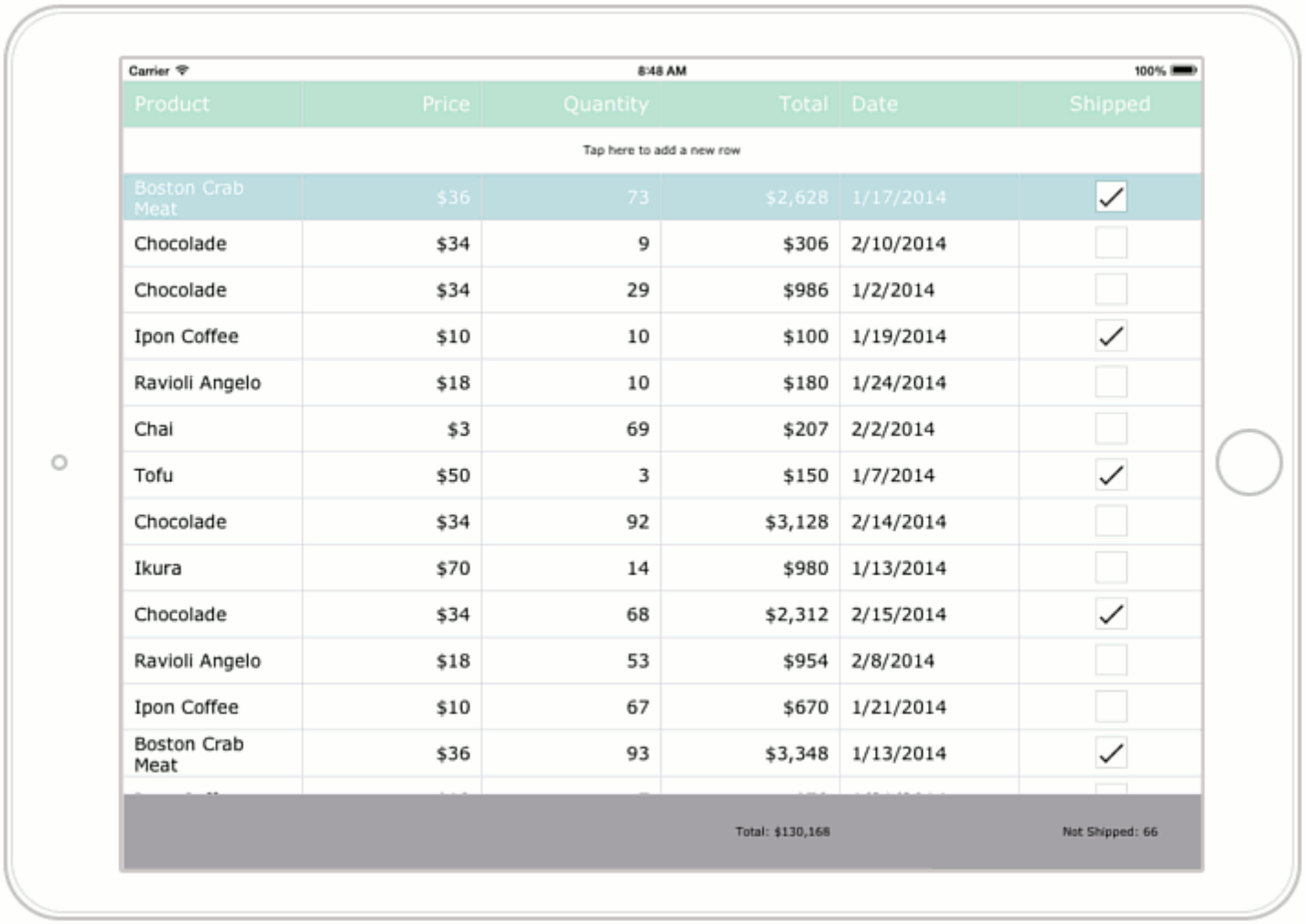
The ThemeBase class's properties provide access to different customizers, which you can adjust to change appearance settings (for example, font attributes, border and background colors, etc.) of the grid's corresponding visual elements (such as data cells, group rows, filter panel, etc.). In this example, the following customizers are used.

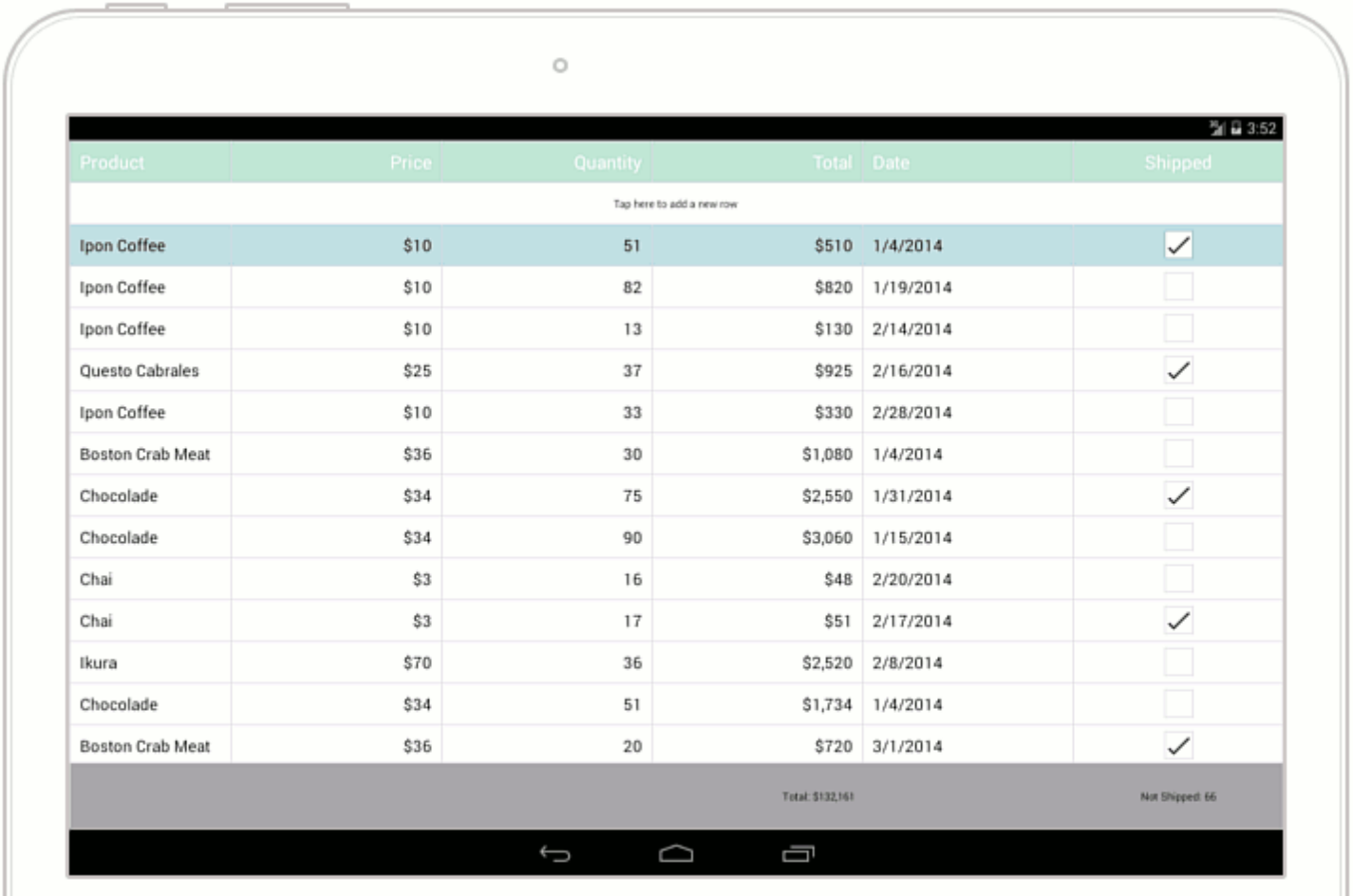
- ThemeBase.HeaderCustomizer
- ThemeBase.CellCustomizer
- ThemeBase.TotalSummaryCustomizer
- ThemeBase.NewItemRowCustomizer

| Important | |
|---|---|
| To apply the customization, call the ThemeManager.RefreshTheme method. | |

This example customizes the Light theme as it is shown in the image below.

© 2018 Developer Express Inc.


Carrier
8:48 AM
100%
Product
Price
Quantity
Total
Date
Shipped
Tap here to add a new row
Boston Crab Meat $36 73 $2,628 1/17/2014
Chocolade $34 9 $306 2/10/2014
Chocolade $34 29 $986 1/2/2014
Ipon Coffee $10 10 $100 1/19/2014
Ravioli Angelo $18 10 $180 1/24/2014
Chai $3 69 $207 2/2/2014
Tofu $50 3 $150 1/7/2014
Chocolade $34 92 $3,128 2/14/2014
Ikura $70 14 $980 1/13/2014
Chocolade $34 68 $2,312 2/15/2014
Ravioli Angelo $18 53 $954 2/8/2014
Ipon Coffee $10 67 $670 1/21/2014
Boston Crab Meat $36 93 $3,348 1/13/2014
Total: $130,168
Not Shipped: 66


3:52
Product
Price
Quantity
Total
Date
Shipped
Tap here to add a new row
Ipon Coffee $10 51 $510 1/4/2014
Ipon Coffee $10 82 $820 1/19/2014
Ipon Coffee $10 13 $130 2/14/2014
Questo Cabrales $25 37 $925 2/16/2014
Ipon Coffee $10 33 $330 2/28/2014
Boston Crab Meat $36 30 $1,080 1/4/2014
Chocolade $34 75 $2,550 1/31/2014
Chocolade $34 90 $3,060 1/15/2014
Chai $3 16 $48 2/20/2014
Chai $3 17 $51 2/17/2014
Ikura $70 36 $2,520 2/8/2014
Chocolade $34 51 $1,734 1/4/2014
Boston Crab Meat $36 20 $720 3/1/2014
Total: $132,161
Not Shipped: 66

C#

```
 (MainPage.xaml.cs)
using System;
using System.Collections.Generic;
using Xamarin.Forms;
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.DataGrid.Theme;
namespace ThemeCustomizing {
    public partial class MainPage : ContentPage {
        int count = 0;
        public MainPage() {
            InitializeComponent();
            TestOrdersRepository model = new TestOrdersRepository();
            BindingContext = model;
            ThemeManager.ThemeName = Themes.Light;
            // Header customization.
            ThemeManager.Theme.HeaderCustomizer.BackgroundColor = Color.FromRgb(187, 228, 208);
            ThemeFontAttributes myFont = new ThemeFontAttributes("Verdana",
                                         ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Large),
                                         FontAttributes.None, Color.White);
            ThemeManager.Theme.HeaderCustomizer.Font = myFont;
            // Cell customization.
            ThemeManager.Theme.CellCustomizer.SelectionColor = Color.FromRgb(186, 220, 225);
            ThemeFontAttributes myFont1 = new ThemeFontAttributes("Verdana",
                                         ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Medium),
                                         FontAttributes.None, Color.Black);
            ThemeManager.Theme.CellCustomizer.Font = myFont1;
            // Various customization.
            ThemeManager.Theme.TotalSumaryCustomizer.BackgroundColor = Color.FromRgb(163, 162, 168);
            ThemeFontAttributes myFont2 = new ThemeFontAttributes("Verdana",
                                 ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Default),
                                 FontAttributes.None, Color.Black);
            ThemeManager.Theme.TotalSummaryCustomizer.Font = myFont2;
            ThemeManager.Theme.NewItemRowCustomizer.Font = myFont2;
            ThemeManager.RefreshTheme();
        }
        void OnCalculateCustomSummary(object sender, CustomSummaryEventArgs e) {
            if (e.FieldName.ToString() == "Shipped")
            if (e.IsTotalSummary) {
                if (e.SummaryProcess == CustomSummaryProcess.Calculate) {
                    if (!(bool)e.FieldValue)
                        count++;
                    e.TotalValue = count;
                }
            }
        }
    }
}
```

**Inheritance Hierarchy**

System.Object
  **ThemeFontAttributes**

**See Also**
ThemeFontAttributes Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# ThemeFontAttributes Members

Allows you to specify required font attributes for the grid's themes.

The following tables list the members exposed by the ThemeFontAttributes type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | ThemeFontAttributes | Initializes a new instance of the ThemeFontAttributes class. |

### Public Properties

| | Name | Description |
|---|---|---|
| | Attributes | Gets the style (bold, italic, etc.) of the corresponding font. |
| | Color | Gets the font color. |
| | Family | Gets the name of the font family to which the corresponding font belongs. |
| | FontObject | Gets the object corresponding to the font. |
| | Size | Gets the size of the corresponding font in density-independent pixels. |

### Public Methods

| | Name | Description |
|---|---|---|
| | ApplyToLabel | Applies the corresponding font attributes to the specified Xamarin.Forms.Label object. |
| | Equals | Overloaded. Determines whether the specified object is equal to the current font. |
| S | Equals | (Inherited from **System.Object**) |
| S | FontSizeFromNamedSize | Converts the Xamarin.Forms.NamedSize enumeration's member to a double value. |
| | GetHashCode | Gets the hash code (a number) that corresponds to the value of the current font object. |
| | GetType | (Inherited from **System.Object**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

### See Also

ThemeFontAttributes Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# ThemeFontAttributes Constructor

Initializes a new instance of the ThemeFontAttributes class.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public ThemeFontAttributes(
   string family,
   Double size,
   FontAttributes attributes,
   Color color
)
```

### Parameters

*family*
A string value that specifies the name of the required font family. This value is assigned to the Family property.
*size*
A double value that specifies the size of the required font. This value is assigned to the Size property.
*attributes*
A Xamarin.Forms.FontAttributes enumeration member that specifies the font style (bold, italic, etc.). This value is assigned to the Attributes property.
*color*
A Xamarin.Forms.Color object that specifies the font color. This value is assigned to the Color property.

### Remarks

Instances of the ThemeFontAttributes class are automatically created, initialized and passed to the corresponding event handlers.

### Example

| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T297553. | |

This example demonstrates how to customize a theme used to paint a GridControl.

To apply a theme to a **GridControl**, assign its name to the ThemeManager.ThemeName property. The Themes class stores available theme names.

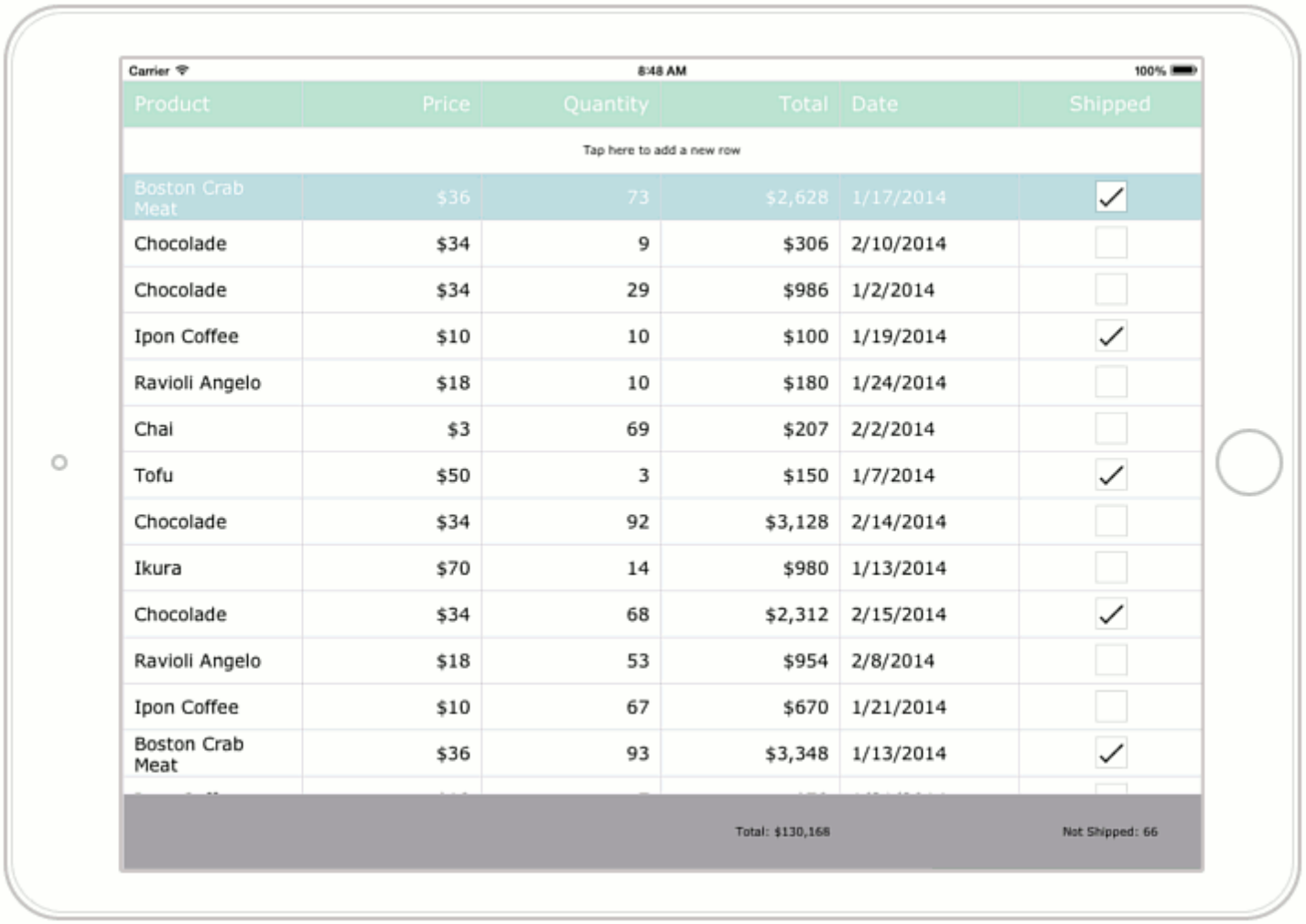
The ThemeBase class's properties provide access to different customizers, which you can adjust to change appearance settings (for example, font attributes, border and background colors, etc.) of the grid's corresponding visual elements (such as data cells, group rows, filter panel, etc.). In this example, the following customizers are used.

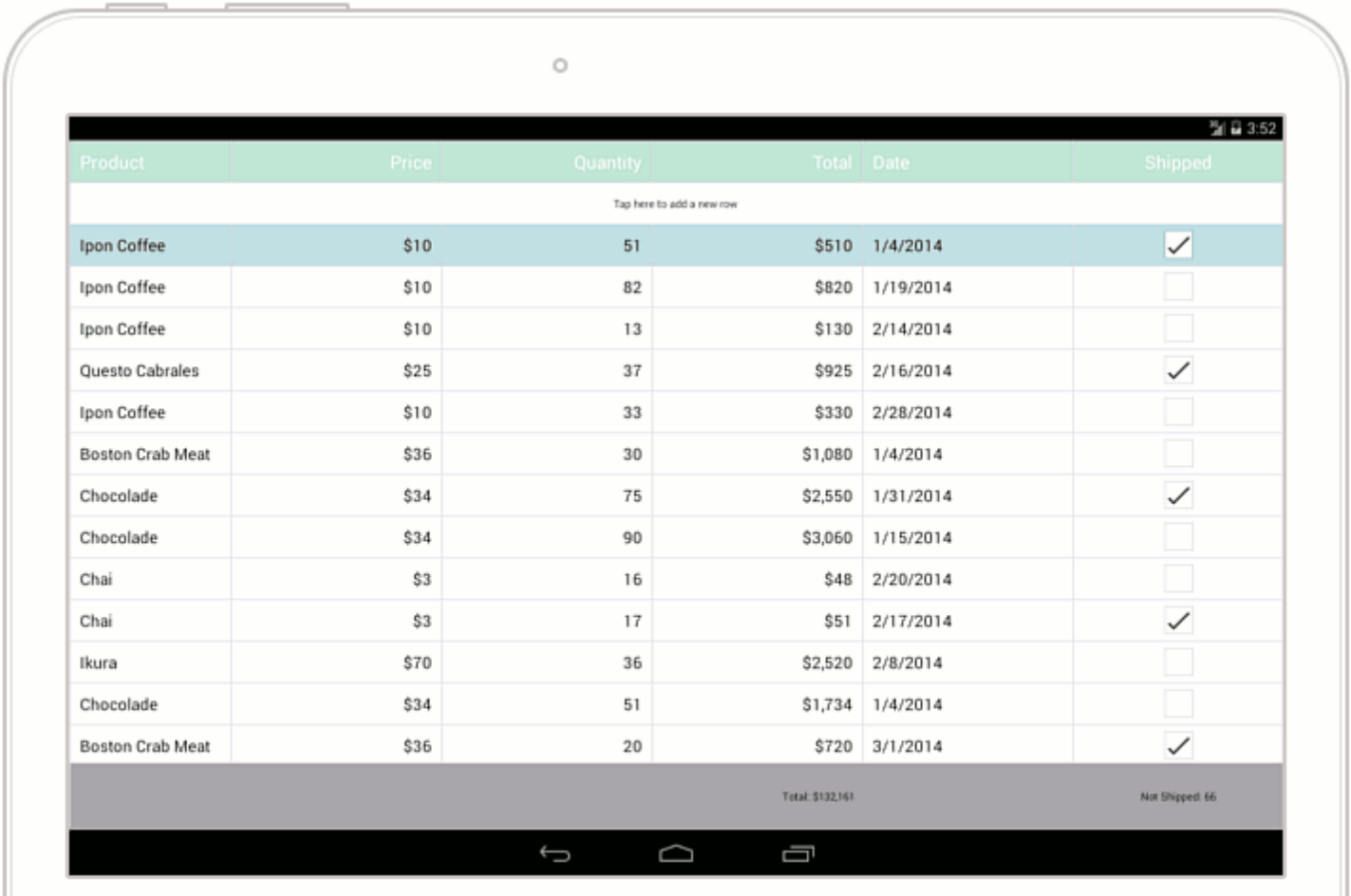
- ThemeBase.HeaderCustomizer
- ThemeBase.CellCustomizer
- ThemeBase.TotalSummaryCustomizer
- ThemeBase.NewItemRowCustomizer

| Important | |
|---|---|
| To apply the customization, call the ThemeManager.RefreshTheme method. | |

This example customizes the Light theme as it is shown in the image below.

© 2018 Developer Express Inc.


Carrier
8:48 AM
100%
Product
Price
Quantity
Total
Date
Shipped
Tap here to add a new row
Boston Crab Meat $36 73 $2,628 1/17/2014
Chocolade $34 9 $306 2/10/2014
Chocolade $34 29 $986 1/2/2014
Ipon Coffee $10 10 $100 1/19/2014
Ravioli Angelo $18 10 $180 1/24/2014
Chai $3 69 $207 2/2/2014
Tofu $50 3 $150 1/7/2014
Chocolade $34 92 $3,128 2/14/2014
Ikura $70 14 $980 1/13/2014
Chocolade $34 68 $2,312 2/15/2014
Ravioli Angelo $18 53 $954 2/8/2014
Ipon Coffee $10 67 $670 1/21/2014
Boston Crab Meat $36 93 $3,348 1/13/2014
Total: $130,168
Not Shipped: 66


3:52
Product
Price
Quantity
Total
Date
Shipped
Tap here to add a new row
Ipon Coffee $10 51 $510 1/4/2014
Ipon Coffee $10 82 $820 1/19/2014
Ipon Coffee $10 13 $130 2/14/2014
Questo Cabrales $25 37 $925 2/16/2014
Ipon Coffee $10 33 $330 2/28/2014
Boston Crab Meat $36 30 $1,080 1/4/2014
Chocolade $34 75 $2,550 1/31/2014
Chocolade $34 90 $3,060 1/15/2014
Chai $3 16 $48 2/20/2014
Chai $3 17 $51 2/17/2014
Ikura $70 36 $2,520 2/8/2014
Chocolade $34 51 $1,734 1/4/2014
Boston Crab Meat $36 20 $720 3/1/2014
Total: $132,161
Not Shipped: 66

C#

```
 (MainPage.xaml.cs)
using System;
using System.Collections.Generic;
using Xamarin.Forms;
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.DataGrid.Theme;
namespace ThemeCustomizing {
    public partial class MainPage : ContentPage {
        int count = 0;
        public MainPage() {
            InitializeComponent();
            TestOrdersRepository model = new TestOrdersRepository();
            BindingContext = model;
            ThemeManager.ThemeName = Themes.Light;
            // Header customization.
            ThemeManager.Theme.HeaderCustomizer.BackgroundColor = Color.FromRgb(187, 228, 208);
            ThemeFontAttributes myFont = new ThemeFontAttributes("Verdana",
                                         ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Large),
                                         FontAttributes.None, Color.White);
            ThemeManager.Theme.HeaderCustomizer.Font = myFont;
            // Cell customization.
            ThemeManager.Theme.CellCustomizer.SelectionColor = Color.FromRgb(186, 220, 225);
            ThemeFontAttributes myFont1 = new ThemeFontAttributes("Verdana",
                                         ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Medium),
                                         FontAttributes.None, Color.Black);
            ThemeManager.Theme.CellCustomizer.Font = myFont1;
            // Various customization.
            ThemeManager.Theme.TotalSumaryCustomizer.BackgroundColor = Color.FromRgb(163, 162, 168);
            ThemeFontAttributes myFont2 = new ThemeFontAttributes("Verdana",
                                 ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Default),
                                 FontAttributes.None, Color.Black);
            ThemeManager.Theme.TotalSummaryCustomizer.Font = myFont2;
            ThemeManager.Theme.NewItemRowCustomizer.Font = myFont2;
            ThemeManager.RefreshTheme();
        }
        void OnCalculateCustomSummary(object sender, CustomSummaryEventArgs e) {
            if (e.FieldName.ToString() == "Shipped")
            if (e.IsTotalSummary) {
                if (e.SummaryProcess == CustomSummaryProcess.Calculate) {
                    if (!(bool)e.FieldValue)
                        count++;
                    e.TotalValue = count;
                }
            }
        }
    }
}
```

**See Also**

ThemeFontAttributes Class
ThemeFontAttributes Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# ThemeFontAttributes Properties

Allows you to specify required font attributes for the grid's themes.

The following tables list the members exposed by the ThemeFontAttributes type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | ThemeFontAttributes | Initializes a new instance of the ThemeFontAttributes class. |

### Public Properties

| | Name | Description |
|---|---|---|
| | Attributes | Gets the style (bold, italic, etc.) of the corresponding font. |
| | Color | Gets the font color. |
| | Family | Gets the name of the font family to which the corresponding font belongs. |
| | FontObject | Gets the object corresponding to the font. |
| | Size | Gets the size of the corresponding font in density-independent pixels. |

### Public Methods

| | Name | Description |
|---|---|---|
| | ApplyToLabel | Applies the corresponding font attributes to the specified Xamarin.Forms.Label object. |
| | Equals | Overloaded. Determines whether the specified object is equal to the current font. |
| S | Equals | (Inherited from **System.Object**) |
| S | FontSizeFromNamedSize | Converts the Xamarin.Forms.NamedSize enumeration's member to a double value. |
| | GetHashCode | Gets the hash code (a number) that corresponds to the value of the current font object. |
| | GetType | (Inherited from **System.Object**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

**See Also**
ThemeFontAttributes Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# Attributes Property

Gets the style (bold, italic, etc.) of the corresponding font.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public FontAttributes Attributes { get; }
```

### Property Value

A Xamarin.Forms.FontAttributes enumeration member that specifies the font style (bold, italic, etc.).

### See Also

ThemeFontAttributes Class
ThemeFontAttributes Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# Color Property

Gets the font color.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public Color Color { get; }
```

### Property Value

A Xamarin.Forms.Color object specifying the color of the corresponding font.

### See Also

ThemeFontAttributes Class
ThemeFontAttributes Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# Family Property

Gets the name of the font family to which the corresponding font belongs.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public string Family { get; }
```

### Property Value

A string value that specifies the name of the font family.

### See Also

ThemeFontAttributes Class
ThemeFontAttributes Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# FontObject Property

Gets the object corresponding to the font.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public Font FontObject { get; }
```

### Property Value

A Xamarin.Forms.Font object that corresponds to the font.

### See Also

ThemeFontAttributes Class
ThemeFontAttributes Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# Size Property

Gets the size of the corresponding font in density-independent pixels.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public Double Size { get; }
```

### Property Value

A double value that specifies the size of the corresponding font.

### See Also

ThemeFontAttributes Class
ThemeFontAttributes Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# ThemeFontAttributes Methods

Allows you to specify required font attributes for the grid's themes.

The following tables list the members exposed by the ThemeFontAttributes type.

**Public Methods**

| | Name | Description |
|---|---|---|
| | ApplyToLabel | Applies the corresponding font attributes to the specified Xamarin.Forms.Label object. |
| | Equals | Overloaded. Determines whether the specified object is equal to the current font. |
| S | Equals | (Inherited from **System.Object**) |
| S | FontSizeFromNamedSize | Converts the Xamarin.Forms.NamedSize enumeration's member to a double value. |
| | GetHashCode | Gets the hash code (a number) that corresponds to the value of the current font object. |
| | GetType | (Inherited from **System.Object**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

**See Also**
ThemeFontAttributes Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# ApplyToLabel Method

Applies the corresponding font attributes to the specified Xamarin.Forms.Label object.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public void ApplyToLabel(
   Label label
)
```

### Parameters

*label*
The Xamarin.Forms.Label object to apply the attributes to.

### See Also

ThemeFontAttributes Class
ThemeFontAttributes Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# Equals Method

Determines whether the specified object is equal to the current font.

**Overload List**

| Name | Description |
|---|---|
| bool Equals(object obj) | Determines whether the specified object is equal to the current font. |
| static bool Equals(object objA, object objB) | Inherited from **System.Object** |

**See Also**
ThemeFontAttributes Class
ThemeFontAttributes Members
DevExpress.Mobile.DataGrid Namespace
ThemeFontAttributes.Equals Overload List

Determines whether the specified object is equal to the current font.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**
**C#**
```
public override bool Equals(
   object obj
)
```

**Parameters**
*obj*
The object to compare with the current font.

**Return Value**

**true** if the specified object is equal to the current font; otherwise **false**.

**See Also**
ThemeFontAttributes Class
ThemeFontAttributes Members
DevExpress.Mobile.DataGrid Namespace
ThemeFontAttributes.Equals Overload List

© 2018 Developer Express Inc.

# FontSizeFromNamedSize Method

Converts the Xamarin.Forms.NamedSize enumeration's member to a double value.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public static Double FontSizeFromNamedSize(
   NamedSize size
)
```

### Parameters

*size*
The Xamarin.Forms.NamedSize enumeration's member to be converted.

### Return Value

A double value that corresponds to a specified named size (large, small, etc.).

### Remarks

You can use the value returned by the **FontSizeFromNamedSize** method to specify sizes for new ThemeFontAttributes instances (see the example below).

### Example

| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T297553. | |

This example demonstrates how to customize a theme used to paint a GridControl.

To apply a theme to a **GridControl**, assign its name to the ThemeManager.ThemeName property. The Themes class stores available theme names.

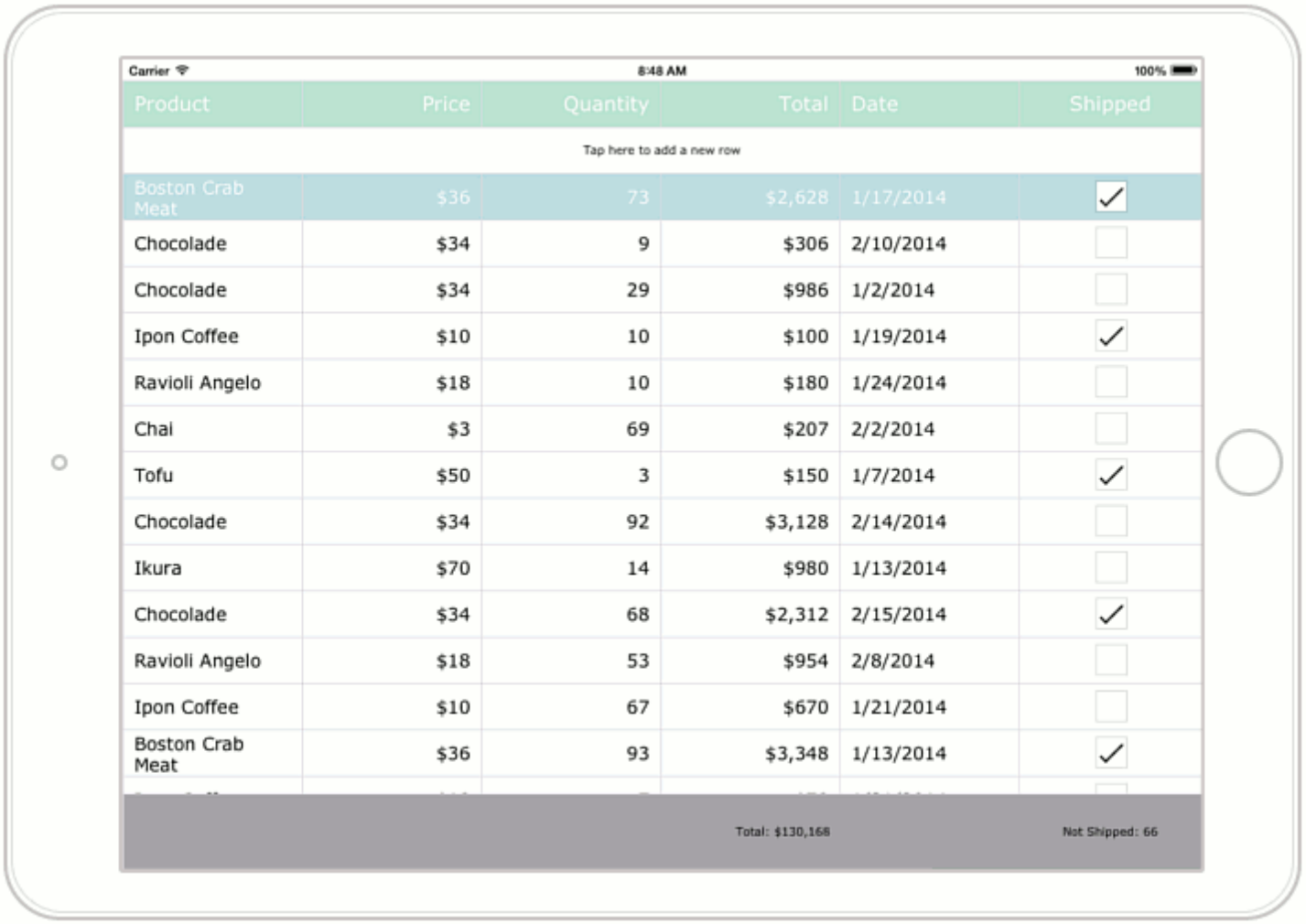
The ThemeBase class's properties provide access to different customizers, which you can adjust to change appearance settings (for example, font attributes, border and background colors, etc.) of the grid's corresponding visual elements (such as data cells, group rows, filter panel, etc.). In this example, the following customizers are used.

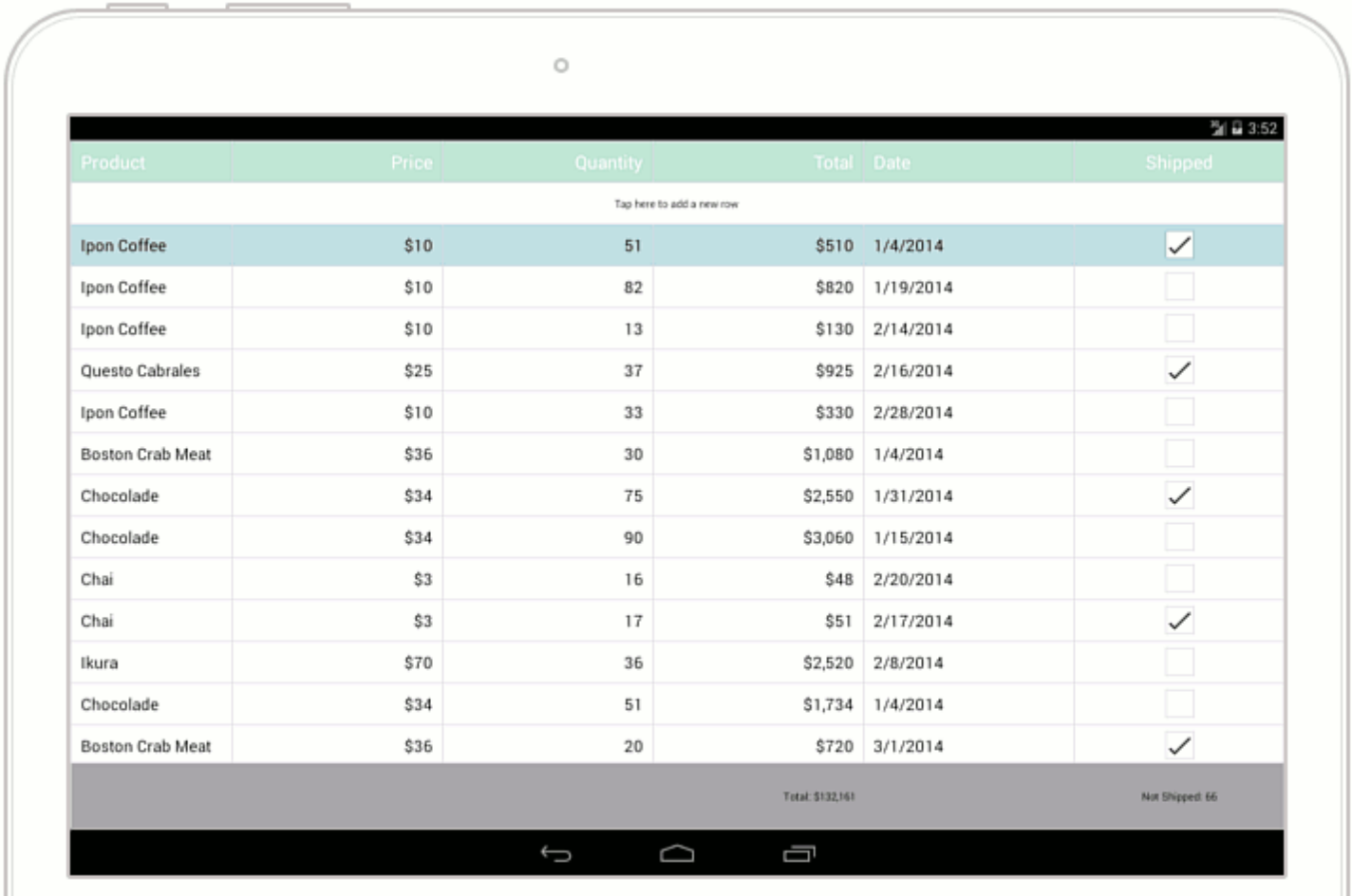
- ThemeBase.HeaderCustomizer
- ThemeBase.CellCustomizer
- ThemeBase.TotalSummaryCustomizer
- ThemeBase.NewItemRowCustomizer

| Important | |
|---|---|
| To apply the customization, call the ThemeManager.RefreshTheme method. | |

This example customizes the Light theme as it is shown in the image below.

© 2018 Developer Express Inc.


Carrier
8:48 AM
100%
Product
Price
Quantity
Total
Date
Shipped
Tap here to add a new row
Boston Crab Meat $36 73 $2,628 1/17/2014
Chocolade $34 9 $306 2/10/2014
Chocolade $34 29 $986 1/2/2014
Ipon Coffee $10 10 $100 1/19/2014
Ravioli Angelo $18 10 $180 1/24/2014
Chai $3 69 $207 2/2/2014
Tofu $50 3 $150 1/7/2014
Chocolade $34 92 $3,128 2/14/2014
Ikura $70 14 $980 1/13/2014
Chocolade $34 68 $2,312 2/15/2014
Ravioli Angelo $18 53 $954 2/8/2014
Ipon Coffee $10 67 $670 1/21/2014
Boston Crab Meat $36 93 $3,348 1/13/2014
Total: $130,168
Not Shipped: 66


3:52
Product
Price
Quantity
Total
Date
Shipped
Tap here to add a new row
Ipon Coffee $10 51 $510 1/4/2014
Ipon Coffee $10 82 $820 1/19/2014
Ipon Coffee $10 13 $130 2/14/2014
Questo Cabrales $25 37 $925 2/16/2014
Ipon Coffee $10 33 $330 2/28/2014
Boston Crab Meat $36 30 $1,080 1/4/2014
Chocolade $34 75 $2,550 1/31/2014
Chocolade $34 90 $3,060 1/15/2014
Chai $3 16 $48 2/20/2014
Chai $3 17 $51 2/17/2014
Ikura $70 36 $2,520 2/8/2014
Chocolade $34 51 $1,734 1/4/2014
Boston Crab Meat $36 20 $720 3/1/2014
Total: $132,161
Not Shipped: 66

**C#**

```
 (MainPage.xaml.cs)
using System;
using System.Collections.Generic;
using Xamarin.Forms;
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.DataGrid.Theme;
namespace ThemeCustomizing {
    public partial class MainPage : ContentPage {
        int count = 0;
        public MainPage() {
            InitializeComponent();
            TestOrdersRepository model = new TestOrdersRepository();
            BindingContext = model;
            ThemeManager.ThemeName = Themes.Light;
            // Header customization.
            ThemeManager.Theme.HeaderCustomizer.BackgroundColor = Color.FromRgb(187, 228, 208);
            ThemeFontAttributes myFont = new ThemeFontAttributes("Verdana",
                                         ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Large),
                                         FontAttributes.None, Color.White);
            ThemeManager.Theme.HeaderCustomizer.Font = myFont;
            // Cell customization.
            ThemeManager.Theme.CellCustomizer.SelectionColor = Color.FromRgb(186, 220, 225);
            ThemeFontAttributes myFont1 = new ThemeFontAttributes("Verdana",
                                         ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Medium),
                                         FontAttributes.None, Color.Black);
            ThemeManager.Theme.CellCustomizer.Font = myFont1;
            // Various customization.
            ThemeManager.Theme.TotalSumaryCustomizer.BackgroundColor = Color.FromRgb(163, 162, 168);
            ThemeFontAttributes myFont2 = new ThemeFontAttributes("Verdana",
                                 ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Default),
                                 FontAttributes.None, Color.Black);
            ThemeManager.Theme.TotalSummaryCustomizer.Font = myFont2;
            ThemeManager.Theme.NewItemRowCustomizer.Font = myFont2;
            ThemeManager.RefreshTheme();
        }
        void OnCalculateCustomSummary(object sender, CustomSummaryEventArgs e) {
            if (e.FieldName.ToString() == "Shipped")
            if (e.IsTotalSummary) {
                if (e.SummaryProcess == CustomSummaryProcess.Calculate) {
                    if (!(bool)e.FieldValue)
                        count++;
                    e.TotalValue = count;
                }
            }
        }
    }
}
```

**See Also**

ThemeFontAttributes Class
ThemeFontAttributes Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# GetHashCode Method

Gets the hash code (a number) that corresponds to the value of the current font object.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public override int GetHashCode()
```

### Return Value

An integer value representing the hash code for the font object.

### See Also

ThemeFontAttributes Class
ThemeFontAttributes Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# TopBottomRuleFormatCondition Class

Represents a top/bottom format condition.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public class TopBottomRuleFormatCondition : ExpressionConditionBase
```

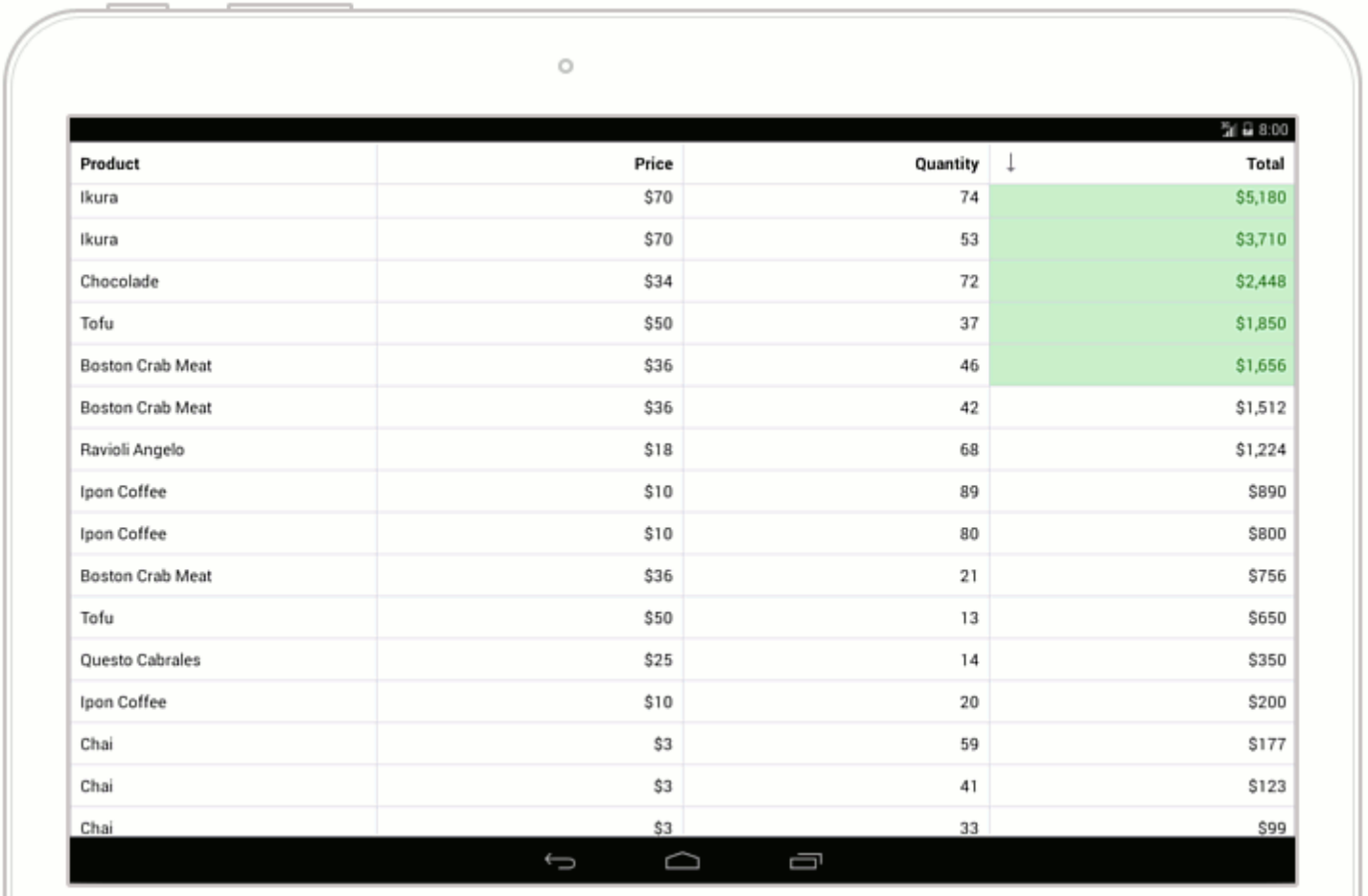
### Example

This example illustrates how to apply a top/bottom format to the **Total** column in a GridControl. The top/bottom format allows you to find the highest or lowest cell values. In this tutorial, the five top values in the **Total** column are highlighted.

© 2018 Developer Express Inc.

Carrier 2:56 PM 100%

| Product | Price | Quantity | Total |
|---|---|---|---|
| Ikura | $70 | 88 | $6,160 |
| Tofu | $50 | 96 | $4,800 |
| Tofu | $50 | 92 | $4,600 |
| Ravioli Angelo | $18 | 95 | $1,710 |
| Boston Crab Meat | $36 | 28 | $1,008 |
| Chocolade | $34 | 28 | $952 |
| Ravioli Angelo | $18 | 51 | $918 |
| Chocolade | $34 | 24 | $816 |
| Chocolade | $34 | 21 | $714 |
| Questo Cabrales | $25 | 28 | $700 |
| Boston Crab Meat | $36 | 18 | $648 |
| Ipon Coffee | $10 | 51 | $510 |
| Boston Crab Meat | $36 | 11 | $396 |
| Ravioli Angelo | $18 | 19 | $342 |
| Chocolade | $34 | 10 | $340 |
| Chocolade | $34 | 10 | $340 |



8:00

| Product | Price | Quantity | Total |
|---|---|---|---|
| Ikura | $70 | 74 | $5,180 |
| Ikura | $70 | 53 | $3,710 |
| Chocolade | $34 | 72 | $2,448 |
| Tofu | $50 | 37 | $1,850 |
| Boston Crab Meat | $36 | 46 | $1,656 |
| Boston Crab Meat | $36 | 42 | $1,512 |
| Ravioli Angelo | $18 | 68 | $1,224 |
| Ipon Coffee | $10 | 89 | $890 |
| Ipon Coffee | $10 | 80 | $800 |
| Boston Crab Meat | $36 | 21 | $756 |
| Tofu | $50 | 13 | $650 |
| Questo Cabrales | $25 | 14 | $350 |
| Ipon Coffee | $10 | 20 | $200 |
| Chai | $3 | 59 | $177 |
| Chai | $3 | 41 | $123 |
| Chai | $3 | 33 | $99 |

1. Create a new TopBottomRuleFormatCondition class instance.
2. Set this object's FormatConditionBase.FieldName property to **Total**. This column provides values to test against the formatting rule and the specified format will be applied to this column.
3. Set the Rule property to **TopBottomRule.TopItems** to highlight the highest column values.
4. Use the Threshold property to specify the count of column cells with highest values to be highlighted.
5. Assign one of the predefined formats to the FormatConditionBase.PredefinedFormatName property. All available format names are listed in the Predefined Format Names document.
6. To apply a conditional formatting rule represented by the created **TopBottomRuleFormatCondition** object, add this object to the GridControl.FormatConditions collection.

**C#**

```
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.Core.ConditionalFormatting;
// ...
TopBottomRuleFormatCondition condition = new TopBottomRuleFormatCondition();
condition.FieldName = "Total";
condition.Rule = TopBottomRule.TopItems;
condition.Threshold = 5;
condition.PredefinedFormatName = "GreenFillWithDarkGreenText";
grid.FormatConditions.Add (condition);
```

### Inheritance Hierarchy

System.Object
  Xamarin.Forms.BindableObject
    FormatConditionBase
      ExpressionConditionBase
        **TopBottomRuleFormatCondition**

### See Also

TopBottomRuleFormatCondition Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# TopBottomRuleFormatCondition Members

Represents a top/bottom format condition.

The following tables list the members exposed by the TopBottomRuleFormatCondition type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | TopBottomRuleFormatCondition | Initializes a new instance of the TopBottomRuleFormatCondition class with default settings. |

### Public Properties

| | Name | Description |
|---|---|---|
| | ApplyToRow | Specifies whether to apply formatting settings to the entire row. This is a bindable property. (Inherited from **ExpressionConditionBase**) |
| | BindingContext | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Expression | Gets or sets the expression specifying to which cells the format should be applied. This is a bindable property. (Inherited from **ExpressionConditionBase**) |
| | FieldName | Gets or sets the name of the column which provides values to test against the current format rule, and to which the format should be applied. This is a bindable property. (Inherited from **FormatConditionBase**) |
| | Format | Gets or sets the formatting settings applied to cells whose values take part in conditional formatting. This is a bindable property. (Inherited from **ExpressionConditionBase**) |
| | Owner | Gets the format condition's owner. This is a bindable property. (Inherited from **FormatConditionBase**) |
| | PredefinedFormatName | Gets or sets the name of the predefined format to be applied to target cells. This is a bindable property. (Inherited from **FormatConditionBase**) |
| | Rule | Specifies the rule for the top/bottom conditional formatting. This is a bindable property. |
| | Threshold | Specifies the number or percentage of cells (or corresponding rows) to format. This is a bindable property. |
| | TypeName | Gets or sets the data type name. This property supports the internal infrastructure and is not intended to be |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | | used directly from your code. (Inherited from **FormatConditionBase**) |

### Public Methods

| | Name | Description |
|---|---|---|
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Equals | (Inherited from **System.Object**) |
| S | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ToString | (Inherited from **System.Object**) |

### Public Fields

| | Name | Description |
|---|---|---|
| S | ApplyToRowProperty | Identifies the ExpressionConditionBase.ApplyToRow bindable property. (Inherited from **ExpressionConditionBase**) |
| S | BindingContextProperty | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ExpressionProperty | Identifies the ExpressionConditionBase.Expression bindable property. (Inherited from **ExpressionConditionBase**) |
| S | FieldNameProperty | Identifies the FormatConditionBase.FieldName bindable property. (Inherited from **FormatConditionBase**) |
| S | FormatProperty | Identifies the ExpressionConditionBase.Format bindable property. (Inherited from **ExpressionConditionBase**) |
| S | OwnerProperty | Identifies the FormatConditionBase.Owner bindable |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | | property. (Inherited from **FormatConditionBase**) |
| | PredefinedFormatNameProperty | Identifies the FormatConditionBase.PredefinedFormatName bindable property. (Inherited from **FormatConditionBase**) |
| | RuleProperty | Identifies the Rule bindable property. |
| | ThresholdProperty | Identifies the Threshold bindable property. |

**Public Events**

| | Name | Description |
|---|---|---|
| | BindingContextChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanging | (Inherited from **Xamarin.Forms.BindableObject**) |

**See Also**
TopBottomRuleFormatCondition Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# TopBottomRuleFormatCondition Constructor

Initializes a new instance of the TopBottomRuleFormatCondition class with default settings.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**
**C#**
```
public TopBottomRuleFormatCondition()
```

**See Also**
TopBottomRuleFormatCondition Class
TopBottomRuleFormatCondition Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# TopBottomRuleFormatCondition Properties

Represents a top/bottom format condition.

The following tables list the members exposed by the TopBottomRuleFormatCondition type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | TopBottomRuleFormatCondition | Initializes a new instance of the TopBottomRuleFormatCondition class with default settings. |

### Public Properties

| | Name | Description |
|---|---|---|
| | ApplyToRow | Specifies whether to apply formatting settings to the entire row. This is a bindable property. (Inherited from **ExpressionConditionBase**) |
| | BindingContext | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Expression | Gets or sets the expression specifying to which cells the format should be applied. This is a bindable property. (Inherited from **ExpressionConditionBase**) |
| | FieldName | Gets or sets the name of the column which provides values to test against the current format rule, and to which the format should be applied. This is a bindable property. (Inherited from **FormatConditionBase**) |
| | Format | Gets or sets the formatting settings applied to cells whose values take part in conditional formatting. This is a bindable property. (Inherited from **ExpressionConditionBase**) |
| | Owner | Gets the format condition's owner. This is a bindable property. (Inherited from **FormatConditionBase**) |
| | PredefinedFormatName | Gets or sets the name of the predefined format to be applied to target cells. This is a bindable property. (Inherited from **FormatConditionBase**) |
| | Rule | Specifies the rule for the top/bottom conditional formatting. This is a bindable property. |
| | Threshold | Specifies the number or percentage of cells (or corresponding rows) to format. This is a bindable property. |
| | TypeName | Gets or sets the data type name. This property supports the internal infrastructure and is not intended to be |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | | used directly from your code. (Inherited from **FormatConditionBase**) |

### Public Methods

| | Name | Description |
|---|---|---|
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Equals | (Inherited from **System.Object**) |
| S | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ToString | (Inherited from **System.Object**) |

### Public Fields

| | Name | Description |
|---|---|---|
| S | ApplyToRowProperty | Identifies the ExpressionConditionBase.ApplyToRow bindable property. (Inherited from **ExpressionConditionBase**) |
| S | BindingContextProperty | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ExpressionProperty | Identifies the ExpressionConditionBase.Expression bindable property. (Inherited from **ExpressionConditionBase**) |
| S | FieldNameProperty | Identifies the FormatConditionBase.FieldName bindable property. (Inherited from **FormatConditionBase**) |
| S | FormatProperty | Identifies the ExpressionConditionBase.Format bindable property. (Inherited from **ExpressionConditionBase**) |
| S | OwnerProperty | Identifies the FormatConditionBase.Owner bindable |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | | property. (Inherited from **FormatConditionBase**) |
| | PredefinedFormatNameProperty | Identifies the FormatConditionBase.PredefinedFormatName bindable property. (Inherited from **FormatConditionBase**) |
| | RuleProperty | Identifies the Rule bindable property. |
| | ThresholdProperty | Identifies the Threshold bindable property. |

**Public Events**

| | Name | Description |
|---|---|---|
| | BindingContextChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanging | (Inherited from **Xamarin.Forms.BindableObject**) |

**See Also**
TopBottomRuleFormatCondition Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# Rule Property

Specifies the rule for the top/bottom conditional formatting. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public TopBottomRule Rule { get; set; }
```

### Property Value

A TopBottomRule enumeration value.

### Remarks

The **Rule** property is used to set a top/bottom rule that specifies which cells (or corresponding rows) should be formatted.

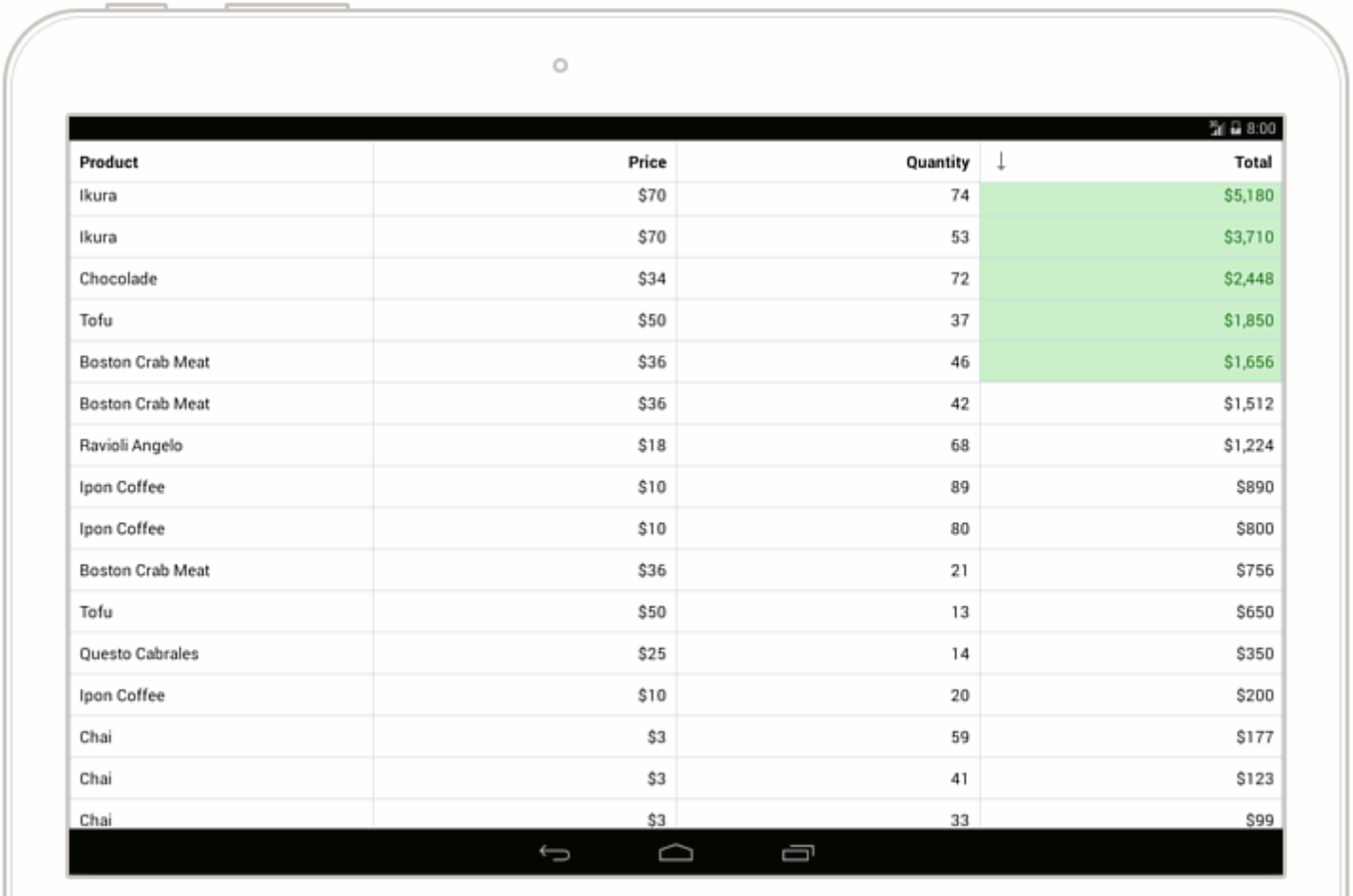
### Example

This example illustrates how to apply a top/bottom format to the **Total** column in a GridControl. The top/bottom format allows you to find the highest or lowest cell values. In this tutorial, the five top values in the **Total** column are highlighted.

© 2018 Developer Express Inc.

Carrier 2:56 PM 100%

| Product | Price | Quantity | ↓ Total |
|---|---|---|---|
| Ikura | $70 | 88 | $6,160 |
| Tofu | $50 | 96 | $4,800 |
| Tofu | $50 | 92 | $4,600 |
| Ravioli Angelo | $18 | 95 | $1,710 |
| Boston Crab Meat | $36 | 28 | $1,008 |
| Chocolade | $34 | 28 | $952 |
| Ravioli Angelo | $18 | 51 | $918 |
| Chocolade | $34 | 24 | $816 |
| Chocolade | $34 | 21 | $714 |
| Questo Cabrales | $25 | 28 | $700 |
| Boston Crab Meat | $36 | 18 | $648 |
| Ipon Coffee | $10 | 51 | $510 |
| Boston Crab Meat | $36 | 11 | $396 |
| Ravioli Angelo | $18 | 19 | $342 |
| Chocolade | $34 | 10 | $340 |
| Chocolade | $34 | 10 | $340 |



8:00

| Product | Price | Quantity | ↓ Total |
|---|---|---|---|
| Ikura | $70 | 74 | $5,180 |
| Ikura | $70 | 53 | $3,710 |
| Chocolade | $34 | 72 | $2,448 |
| Tofu | $50 | 37 | $1,850 |
| Boston Crab Meat | $36 | 46 | $1,656 |
| Boston Crab Meat | $36 | 42 | $1,512 |
| Ravioli Angelo | $18 | 68 | $1,224 |
| Ipon Coffee | $10 | 89 | $890 |
| Ipon Coffee | $10 | 80 | $800 |
| Boston Crab Meat | $36 | 21 | $756 |
| Tofu | $50 | 13 | $650 |
| Questo Cabrales | $25 | 14 | $350 |
| Ipon Coffee | $10 | 20 | $200 |
| Chai | $3 | 59 | $177 |
| Chai | $3 | 41 | $123 |
| Chai | $3 | 33 | $99 |

1. Create a new TopBottomRuleFormatCondition class instance.
2. Set this object's FormatConditionBase.FieldName property to **Total**. This column provides values to test against the formatting rule and the specified format will be applied to this column.
3. Set the Rule property to **TopBottomRule.TopItems** to highlight the highest column values.
4. Use the Threshold property to specify the count of column cells with highest values to be highlighted.
5. Assign one of the predefined formats to the FormatConditionBase.PredefinedFormatName property. All available format names are listed in the Predefined Format Names document.
6. To apply a conditional formatting rule represented by the created **TopBottomRuleFormatCondition** object, add this object to the GridControl.FormatConditions collection.

**C#**

```
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.Core.ConditionalFormatting;
// ...
TopBottomRuleFormatCondition condition = new TopBottomRuleFormatCondition();
condition.FieldName = "Total";
condition.Rule = TopBottomRule.TopItems;
condition.Threshold = 5;
condition.PredefinedFormatName = "GreenFillWithDarkGreenText";
grid.FormatConditions.Add (condition);
```

**See Also**
TopBottomRuleFormatCondition Class
TopBottomRuleFormatCondition Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# Threshold Property

Specifies the number or percentage of cells (or corresponding rows) to format. This is a bindable property.

**Namespace:** DevExpress.Mobile.DataGrid
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public Double Threshold { get; set; }
```

### Property Value

The number of cells to format.

### Remarks

The **Threshold** property is in effect for the following rules:

- TopBottomRule.TopItems
- TopBottomRule.TopPercent
- TopBottomRule.BottomItems
- TopBottomRule.BottomPercent

Depending on the rule (Rule), the **Threshold** property is used to specify either the number or percentage of cells (or corresponding rows) to format.

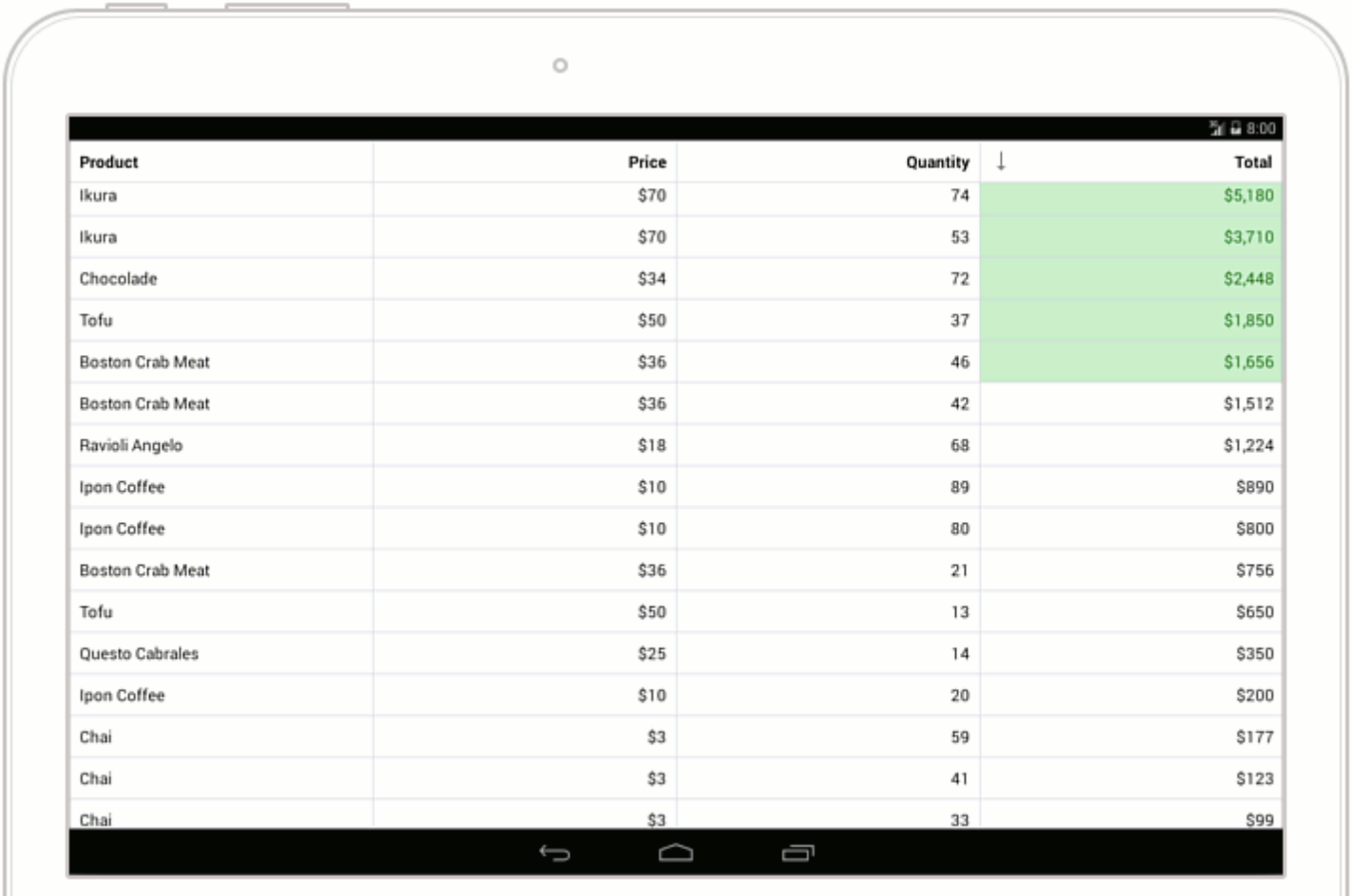
### Example

This example illustrates how to apply a top/bottom format to the **Total** column in a GridControl. The top/bottom format allows you to find the highest or lowest cell values. In this tutorial, the five top values in the **Total** column are highlighted.

© 2018 Developer Express Inc.

Carrier 2:56 PM 100%

| Product | Price | Quantity | ↓ Total |
|---|---|---|---|
| Ikura | $70 | 88 | $6,160 |
| Tofu | $50 | 96 | $4,800 |
| Tofu | $50 | 92 | $4,600 |
| Ravioli Angelo | $18 | 95 | $1,710 |
| Boston Crab Meat | $36 | 28 | $1,008 |
| Chocolade | $34 | 28 | $952 |
| Ravioli Angelo | $18 | 51 | $918 |
| Chocolade | $34 | 24 | $816 |
| Chocolade | $34 | 21 | $714 |
| Questo Cabrales | $25 | 28 | $700 |
| Boston Crab Meat | $36 | 18 | $648 |
| Ipon Coffee | $10 | 51 | $510 |
| Boston Crab Meat | $36 | 11 | $396 |
| Ravioli Angelo | $18 | 19 | $342 |
| Chocolade | $34 | 10 | $340 |
| Chocolade | $34 | 10 | $340 |



8:00

| Product | Price | Quantity | ↓ Total |
|---|---|---|---|
| Ikura | $70 | 74 | $5,180 |
| Ikura | $70 | 53 | $3,710 |
| Chocolade | $34 | 72 | $2,448 |
| Tofu | $50 | 37 | $1,850 |
| Boston Crab Meat | $36 | 46 | $1,656 |
| Boston Crab Meat | $36 | 42 | $1,512 |
| Ravioli Angelo | $18 | 68 | $1,224 |
| Ipon Coffee | $10 | 89 | $890 |
| Ipon Coffee | $10 | 80 | $800 |
| Boston Crab Meat | $36 | 21 | $756 |
| Tofu | $50 | 13 | $650 |
| Questo Cabrales | $25 | 14 | $350 |
| Ipon Coffee | $10 | 20 | $200 |
| Chai | $3 | 59 | $177 |
| Chai | $3 | 41 | $123 |
| Chai | $3 | 33 | $99 |

1. Create a new TopBottomRuleFormatCondition class instance.
2. Set this object's FormatConditionBase.FieldName property to **Total**. This column provides values to test against the formatting rule and the specified format will be applied to this column.
3. Set the Rule property to **TopBottomRule.TopItems** to highlight the highest column values.
4. Use the Threshold property to specify the count of column cells with highest values to be highlighted.
5. Assign one of the predefined formats to the FormatConditionBase.PredefinedFormatName property. All available format names are listed in the Predefined Format Names document.
6. To apply a conditional formatting rule represented by the created **TopBottomRuleFormatCondition** object, add this object to the GridControl.FormatConditions collection.

**C#**

```
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.Core.ConditionalFormatting;
// ...
TopBottomRuleFormatCondition condition = new TopBottomRuleFormatCondition();
condition.FieldName = "Total";
condition.Rule = TopBottomRule.TopItems;
condition.Threshold = 5;
condition.PredefinedFormatName = "GreenFillWithDarkGreenText";
grid.FormatConditions.Add (condition);
```

**See Also**
TopBottomRuleFormatCondition Class
TopBottomRuleFormatCondition Members
DevExpress.Mobile.DataGrid Namespace

© 2018 Developer Express Inc.

# DevExpress.Mobile.DataGrid.Theme

Contains classes that manipulate GridControl themes.

### Classes

| | Class | Description |
|---|---|---|
| | DarkTheme | Provides access to the **Dark** grid theme. |
| | LightTheme | Provides access to the **Light** grid theme. |
| | ThemeBase | Represents the base class for the GridControl themes. |
| | ThemeManager | Allows manipulating a GridControl's themes - apply them to a grid and customize the applied theme's appearance. |
| | Themes | Contains the available GridControl theme names. |

### Interfaces

| | Interface | Description |
|---|---|---|
| | IAndroidRefreshCustomizer | Defines classes that customize an Android refreshing bar. |
| | IAutoFilterPanelCustomizer | Defines classes that customize the Auto Filter panel. |
| | ICellCustomizer | Defines classes that customize grid cells. |
| | IColumnChooserCustomizer | Defines classes that customize a visual presentation of the Column Chooser dialog. |
| | IDialogFormCustomizer | Defines classes that customize both the Column Chooser dialog and the Edit Form. |
| | IEditorCustomizer | Defines classes that customize editors located in the Auto Filter panel, and the New Item Row and in-place edit forms. |
| | IFilterPanelCustomizer | Defines classes that customize the filter panel. |
| | IGridControlCustomizer | Defines classes that customize the DevExpress.Mobile.DataGrid.GridControl. |
| | IGroupCustomizer | Defines classes that customize group rows. |
| | IHeaderCustomizer | Defines classes that customize the grid header - the row that contains headers of all grid columns. |
| | IIOSRefreshCustomizer | Defines classes that customize an iOS refreshing icon. |
| | ILoadMoreViewCustomizer | Defines classes that customize the Load More panel. |
| | INewItemRowCustomizer | Defines classes that customize the New Item Row. |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | ISaveCancelButtonsPanelCustomizer | Defines classes that customize the **Apply/Cancel** panel, which appears when you either add a new row to a grid or edit grid data in the **Inplace** row edit mode. |
| | IThemeChangingHandler | Defines classes that react on a GridControl theme change. |
| | ITotalSummaryCustomizer | Defines classes that customize a grid's Total Summary. |
| | IWinPhoneRefreshCustomizer | Defines classes that customize the WinPhone refreshing bar. |

© 2018 Developer Express Inc.

# DarkTheme Class

Provides access to the **Dark** grid theme.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public class DarkTheme : ThemeBase
```

### Remarks

**Dark** is one of the predefined GridControl themes. The Themes class stores available theme names. To apply the **Dark** theme to a grid, assign the **Themes.Dark** field to the ThemeManager.ThemeName property.

### Inheritance Hierarchy

System.Object
  ThemeBase
    **DarkTheme**

### See Also

DarkTheme Members
DevExpress.Mobile.DataGrid.Theme Namespace
LightTheme
ThemeManager

© 2018 Developer Express Inc.

# DarkTheme Members

Provides access to the **Dark** grid theme.

The following tables list the members exposed by the DarkTheme type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | DarkTheme | Initializes a new instance of the DarkTheme class. |

### Public Properties

| | Name | Description |
|---|---|---|
| | AndroidRefreshCustomizer | Customizes the color of the Android refreshing bar. (Inherited from **ThemeBase**) |
| | AutoFilterPanelCustomizer | Customizes the Auto-Filter panel. (Inherited from **ThemeBase**) |
| | CellCustomizer | Customizes grid cell's visual attributes such as font, background color and so on. (Inherited from **ThemeBase**) |
| | ColumnChooserCustomizer | Customizes the Column Chooser. (Inherited from **ThemeBase**) |
| | DialogFormCustomizer | Customizes both the Column Chooser and the Popup Edit Form. (Inherited from **ThemeBase**) |
| | EditorCustomizer | Customizes editors located within the Auto-Filter panel. (Inherited from **ThemeBase**) |
| | FilterPanelCustomizer | Customizes the filter panel. (Inherited from **ThemeBase**) |
| | GridControlCustomizer | Customizes GridControl's visual attributes (such as background and border colors, border thickness and so on). (Inherited from **ThemeBase**) |
| | GroupCustomizer | Customizes group rows. (Inherited from **ThemeBase**) |
| | HeaderCustomizer | Customizes GridControl header's visual attributes (such as font, background and border color), the arrows shown while sorting grid values and so on. (Inherited from **ThemeBase**) |
| | IOSRefreshCustomizer | Customizes the color of the iOS refreshing icon. (Inherited from **ThemeBase**) |
| | LoadMoreViewCustomizer | Customizes the Load More panel shown when an end-user scrolls to the bottom of the grid. (Inherited from **ThemeBase**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | Name | Gets the name of the applied grid theme. (Inherited from **ThemeBase**) |
| | NewItemRowCustomizer | Customizes the New Item Row. (Inherited from **ThemeBase**) |
| | SaveCancelButtonsPanelCustomizer | Customizes the Apply/Cancel panel that appears when you either add a new row to a grid or edit grid data with the **Inplace** row edit mode. (Inherited from **ThemeBase**) |
| | TotalSummaryCustomizer | Customizes the total summaries appearance. (Inherited from **ThemeBase**) |
| | WinPhoneRefreshCustomizer | Customizes the color of the Windows Phone refreshing bar. (Inherited from **ThemeBase**) |

**Public Methods**

| | Name | Description |
|---|---|---|
| S | CreateDefaultFont | Overloaded. Creates and returns a **ThemeFontAttributes** object with the specified font attributes and color. (Inherited from **ThemeBase**) |
| S | CreateDefaultFontForFormEditors | Creates and returns a **ThemeFontAttributes** object with the default font attributes and the specified color. (Inherited from **ThemeBase**) |
| S | CreateDefaultPadding | Creates and returns the **Thickness** object that corresponds to the default paddings. (Inherited from **ThemeBase**) |
| S | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

**See Also**
DarkTheme Members
DevExpress.Mobile.DataGrid.Theme Namespace
LightTheme
ThemeManager

© 2018 Developer Express Inc.

# DarkTheme Constructor

Initializes a new instance of the DarkTheme class.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**
**C#**
```
public DarkTheme()
```

**See Also**
DarkTheme Class
DarkTheme Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# IAndroidRefreshCustomizer Interface

Defines classes that customize an Android refreshing bar.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public interface IAndroidRefreshCustomizer
```

### Remarks

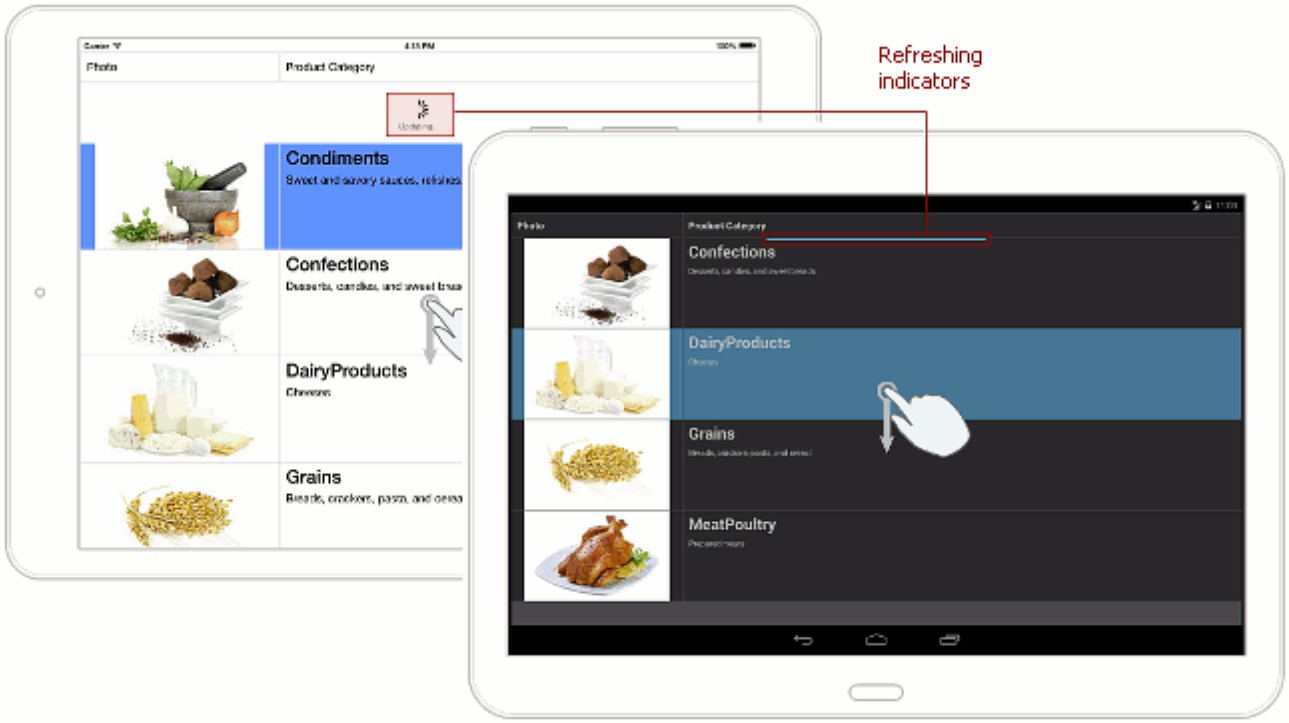
To customize the color of the Android refreshing bar, use the BarColor property of the object that implements the **IAndroidRefreshCustomizer** interface.

### Example

**Show Me**

A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T326911.

This example demonstrates how to customize a color of an indicator (icon or bar) shown when GridControl refreshes cell data.




To change the color of a refresh indicator, perform the steps below.

1. Activate a grid's **pull-to-refresh** functionality. For this purpose, assign **true** to the GridControl.IsPullToRefreshEnabled property.
2. Use the ThemeBase class's properties to access customizers used to change an indicator's color on different mobile platforms. In this example, the following customizers are used.
   - ThemeBase.AndroidRefreshCustomizer
   - ThemeBase.IOSRefreshCustomizer
3. Apply the customization by calling the ThemeManager.RefreshTheme method.

**C#**

© 2018 Developer Express Inc.

```
 (MainPage.xaml.cs)
using System;
using System.Collections.Generic;
using Xamarin.Forms;
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.DataGrid.Theme;
namespace RefreshCustomizers {
    public partial class MainPage : ContentPage {
        public MainPage() {
            InitializeComponent();
            TestOrdersRepository model = new TestOrdersRepository();
            BindingContext = model;
            ThemeManager.Theme.AndroidRefreshCustomizer.BarColor = Color.Blue;
            ThemeManager.Theme.IOSRefreshCustomizer.RefreshColor = Color.Blue;
            ThemeManager.RefreshTheme();
        }
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<?xml version="1.0" encoding="UTF-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="RefreshCustomizers.MainPage"
             xmlns:dxGrid="clr-namespace:DevExpress.Mobile.DataGrid;assembly=DevExpress.Mobile.Grid.v15.2"
      <ContentPage.Padding>
        <OnPlatform x:TypeArguments="Thickness" iOS="0, 20, 0, 0" />
    </ContentPage.Padding>
    <dxGrid:GridControl x:Name="grid" ItemsSource="{Binding Orders}"
                         NewItemRowVisibility = "false"
                        SortMode = "Multiple" AutoFilterPanelVisibility="true"
                        IsPullToRefreshEnabled = "true" >
         <dxGrid:GridControl.Columns>
             <dxGrid:TextColumn FieldName="Product.Name" Caption = "Product" Width = "170"
                                SortOrder = "Descending" SortIndex = "0"/>
             <dxGrid:NumberColumn FieldName="Product.UnitPrice" Caption = "Price" DisplayFormat="C0"/>
             <dxGrid:NumberColumn FieldName="Quantity"
                                  SortOrder = "Ascending" SortIndex = "1"/>
             <dxGrid:NumberColumn FieldName="Total"
                                  UnboundType="Integer" UnboundExpression="[Quantity] * [Product.UnitPrice
                                  DisplayFormat="C0" IsReadOnly="True"/>
             <dxGrid:DateColumn FieldName="Date" DisplayFormat="d" />
             <dxGrid:SwitchColumn FieldName="Shipped"/>
         </dxGrid:GridControl.Columns>
         <dxGrid:GridControl.GroupSummaries>
            <dxGrid:GridColumnSummary FieldName="Total" Type="Max"/>
        </dxGrid:GridControl.GroupSummaries>
        <dxGrid:GridControl.TotalSummaries>
            <dxGrid:GridColumnSummary FieldName="Total" Type="Sum"
                                      DisplayFormat= "Total: {0:C0}"/>
        </dxGrid:GridControl.TotalSummaries>
    </dxGrid:GridControl>
</ContentPage>
```

## Inheritance Hierarchy

## See Also

IAndroidRefreshCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
IIOSRefreshCustomizer

© 2018 Developer Express Inc.

# IAndroidRefreshCustomizer Members

Defines classes that customize an Android refreshing bar.

The following tables list the members exposed by the IAndroidRefreshCustomizer type.

**Public Properties**

| | Name | Description |
|---|---|---|
| | BarColor | Specifies a color of the Android refreshing bar. |

**See Also**
IAndroidRefreshCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
IIOSRefreshCustomizer

© 2018 Developer Express Inc.

# IAndroidRefreshCustomizer Properties

Defines classes that customize an Android refreshing bar.

The following tables list the members exposed by the IAndroidRefreshCustomizer type.

**Public Properties**

| | Name | Description |
|---|---|---|
| | BarColor | Specifies a color of the Android refreshing bar. |

**See Also**
IAndroidRefreshCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
IIOSRefreshCustomizer

© 2018 Developer Express Inc.

# BarColor Property

Specifies a color of the Android refreshing bar.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
Color BarColor { get; set; }
```

### Property Value

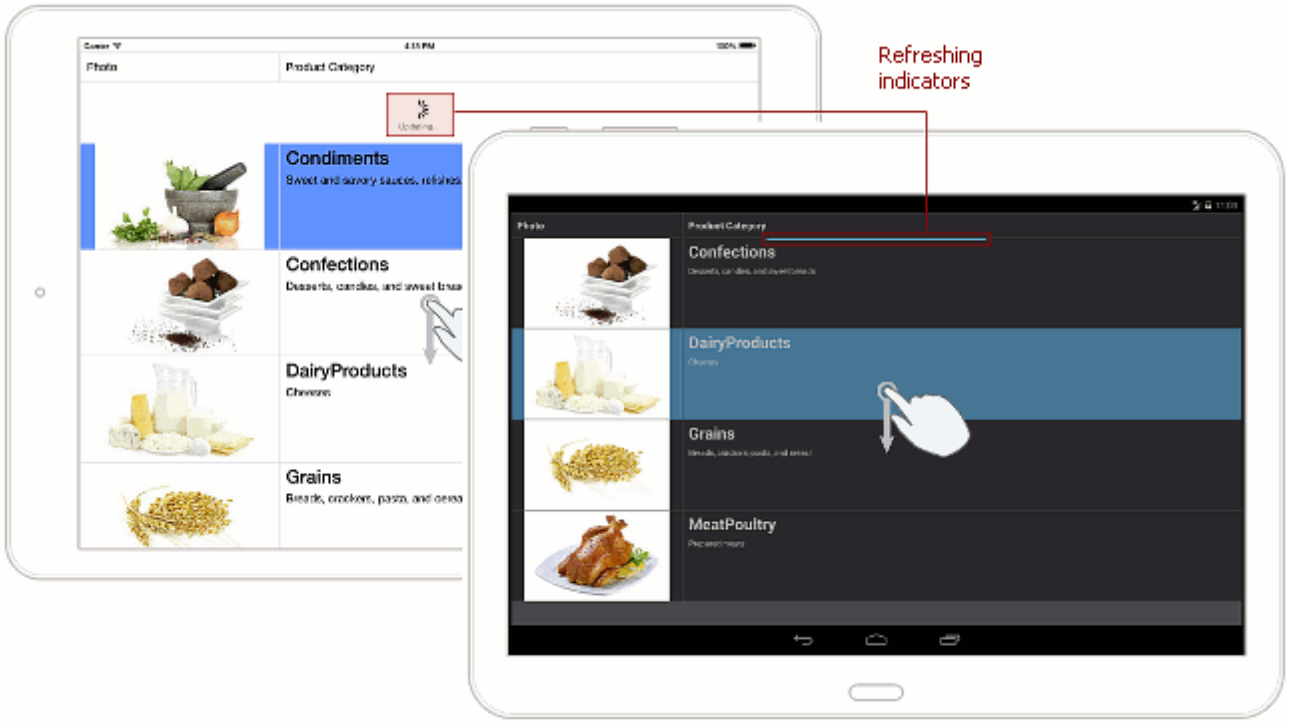
A Xamarin.Forms.Color object specifying the color of the Android refreshing bar.

### Example

**Show Me**

A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T326911.

This example demonstrates how to customize a color of an indicator (icon or bar) shown when GridControl refreshes cell data.




To change the color of a refresh indicator, perform the steps below.

1. Activate a grid's **pull-to-refresh** functionality. For this purpose, assign **true** to the GridControl.IsPullToRefreshEnabled property.
2. Use the ThemeBase class's properties to access customizers used to change an indicator's color on different mobile platforms. In this example, the following customizers are used.
   - ThemeBase.AndroidRefreshCustomizer
   - ThemeBase.IOSRefreshCustomizer
3. Apply the customization by calling the ThemeManager.RefreshTheme method.

**C#**

© 2018 Developer Express Inc.

(MainPage.xaml.cs)

```
using System;
using System.Collections.Generic;
using Xamarin.Forms;
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.DataGrid.Theme;
namespace RefreshCustomizers {
    public partial class MainPage : ContentPage {
        public MainPage() {
            InitializeComponent();
            TestOrdersRepository model = new TestOrdersRepository();
            BindingContext = model;
            ThemeManager.Theme.AndroidRefreshCustomizer.BarColor = Color.Blue;
            ThemeManager.Theme.IOSRefreshCustomizer.RefreshColor = Color.Blue;
            ThemeManager.RefreshTheme();
        }
    }
}
```

**Xaml**

(MainPage.xaml)

```
<?xml version="1.0" encoding="UTF-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="RefreshCustomizers.MainPage"
             xmlns:dxGrid="clr-namespace:DevExpress.Mobile.DataGrid;assembly=DevExpress.Mobile.Grid.v15.2"
      <ContentPage.Padding>
        <OnPlatform x:TypeArguments="Thickness" iOS="0, 20, 0, 0" />
    </ContentPage.Padding>
    <dxGrid:GridControl x:Name="grid" ItemsSource="{Binding Orders}"
                        NewItemRowVisibility = "false"
                        SortMode = "Multiple" AutoFilterPanelVisibility="true"
                        IsPullToRefreshEnabled = "true" >
         <dxGrid:GridControl.Columns>
            <dxGrid:TextColumn FieldName="Product.Name" Caption = "Product" Width = "170"
                               SortOrder = "Descending" SortIndex = "0"/>
            <dxGrid:NumberColumn FieldName="Product.UnitPrice" Caption = "Price" DisplayFormat="C0"/>
            <dxGrid:NumberColumn FieldName="Quantity"
                                 SortOrder = "Ascending" SortIndex = "1"/>
            <dxGrid:NumberColumn FieldName="Total"
                                 UnboundType="Integer" UnboundExpression="[Quantity] * [Product.UnitPrice
                                 DisplayFormat="C0" IsReadOnly="True"/>
            <dxGrid:DateColumn FieldName="Date" DisplayFormat="d" />
            <dxGrid:SwitchColumn FieldName="Shipped"/>
         </dxGrid:GridControl.Columns>
         <dxGrid:GridControl.GroupSummaries>
            <dxGrid:GridColumnSummary FieldName="Total" Type="Max"/>
        </dxGrid:GridControl.GroupSummaries>
        <dxGrid:GridControl.TotalSummaries>
            <dxGrid:GridColumnSummary FieldName="Total" Type="Sum"
                                      DisplayFormat= "Total: {0:C0}"/>
        </dxGrid:GridControl.TotalSummaries>
    </dxGrid:GridControl>
</ContentPage>
```

## See Also

IAndroidRefreshCustomizer Interface
IAndroidRefreshCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# IAutoFilterPanelCustomizer Interface

Defines classes that customize the Auto Filter panel.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public interface IAutoFilterPanelCustomizer
```

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

### Inheritance Hierarchy

### See Also

IAutoFilterPanelCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# IAutoFilterPanelCustomizer Members

Defines classes that customize the Auto Filter panel.

The following tables list the members exposed by the IAutoFilterPanelCustomizer type.

**Public Properties**

| | Name | Description |
|---|---|---|
| | BackgroundColor | Specifying the Auto Filter panel's background color. |
| | BorderColor | Specifies the Auto Filter panel's border color. |

**Public Methods**

| | Name | Description |
|---|---|---|
| | LocateContentInPanel | Locates the specified view within the Auto Filter panel's bounds. |

**See Also**
IAutoFilterPanelCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# IAutoFilterPanelCustomizer Properties

Defines classes that customize the Auto Filter panel.

The following tables list the members exposed by the IAutoFilterPanelCustomizer type.

**Public Properties**

| | Name | Description |
|---|---|---|
| | BackgroundColor | Specifying the Auto Filter panel's background color. |
| | BorderColor | Specifies the Auto Filter panel's border color. |

**Public Methods**

| | Name | Description |
|---|---|---|
| | LocateContentInPanel | Locates the specified view within the Auto Filter panel's bounds. |

**See Also**
IAutoFilterPanelCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# BackgroundColor Property

Specifying the Auto Filter panel's background color.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
Color BackgroundColor { get; set; }
```

### Property Value

A Xamarin.Forms.Color object specifying the border color.

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

### See Also

IAutoFilterPanelCustomizer Interface
IAutoFilterPanelCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# BorderColor Property

Specifies the Auto Filter panel's border color.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
Color BorderColor { get; set; }
```

### Property Value

A Xamarin.Forms.Color object specifying the border color.

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

### See Also

IAutoFilterPanelCustomizer Interface
IAutoFilterPanelCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# IAutoFilterPanelCustomizer Methods

Defines classes that customize the Auto Filter panel.

The following tables list the members exposed by the IAutoFilterPanelCustomizer type.

**Public Methods**

| | Name | Description |
|---|---|---|
| | LocateContentInPanel | Locates the specified view within the Auto Filter panel's bounds. |

**See Also**
IAutoFilterPanelCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# LocateContentInPanel Method

Locates the specified view within the Auto Filter panel's bounds.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
void LocateContentInPanel(
   View content,
   Rectangle contentBounds,
   bool isFirst,
   bool isLast,
   bool isLeftFixedColumnBorder,
   bool isRightFixedColumnBorder
)
```

### Parameters

*content*
A Xamarin.Forms.View object, specifying the view to be locatedin the **Auto Filter** panel.
*contentBounds*
A Xamarin.Forms.Rectangle object, specifying the **Auto Filter** panel's bounds.
*isFirst*
**true**, if the specified view is the first view in the **Auto Filter** panel; otherwise, **false**.
*isLast*
**true**, if the specified view is the last view in the **Auto Filter** panel; otherwise, **false**.
*isLeftFixedColumnBorder*
**true**, if the specified view belongs to the last column fixed to the left; otherwise, **false**.
*isRightFixedColumnBorder*
**true**, if the specified view belongs to the first column fixed to the right; otherwise, **false**.

### See Also

IAutoFilterPanelCustomizer Interface
IAutoFilterPanelCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# ICellCustomizer Interface

Defines classes that customize grid cells.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public interface ICellCustomizer
```

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

### Inheritance Hierarchy

### See Also

ICellCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
How to: Customize Themes
Themes Overview

© 2018 Developer Express Inc.

# ICellCustomizer Members

Defines classes that customize grid cells.

The following tables list the members exposed by the ICellCustomizer type.

**Public Properties**

| | Name | Description |
|---|---|---|
| | BackgroundColor | Specifies the cell background color. |
| | BorderColor | Specifies the color of the cell borders. |
| | Font | Specifies the font used to display data in grid cells. |
| | HighlightColor | Specifies the color used to highlight the cell content in custom user scenarios. |
| | Padding | Specifies the internal spaces between the cell content and borders. |
| | SelectionColor | Specifies the background color on selecting a cell. |
| | SelectionFontColor | Specifies the font color on selecting a cell. |

**Public Methods**

| | Name | Description |
|---|---|---|
| | CreateCheckedImageSource | Sets up an image source for checked switches shown in switch columns. |
| | CreateUnassignedImageSource | Sets up an image source for unassigned switches shown in switch columns. |
| | CreateUncheckedImageSource | Sets up an image source for unchecked switches shown in switch columns. |
| | LocateCellViewInRow | Locates the specified view within cell bounds. |

**See Also**
ICellCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
How to: Customize Themes
Themes Overview

© 2018 Developer Express Inc.

# ICellCustomizer Properties

Defines classes that customize grid cells.

The following tables list the members exposed by the ICellCustomizer type.

**Public Properties**

| | Name | Description |
|---|---|---|
| | BackgroundColor | Specifies the cell background color. |
| | BorderColor | Specifies the color of the cell borders. |
| | Font | Specifies the font used to display data in grid cells. |
| | HighlightColor | Specifies the color used to highlight the cell content in custom user scenarios. |
| | Padding | Specifies the internal spaces between the cell content and borders. |
| | SelectionColor | Specifies the background color on selecting a cell. |
| | SelectionFontColor | Specifies the font color on selecting a cell. |

**Public Methods**

| | Name | Description |
|---|---|---|
| | CreateCheckedImageSource | Sets up an image source for checked switches shown in switch columns. |
| | CreateUnassignedImageSource | Sets up an image source for unassigned switches shown in switch columns. |
| | CreateUncheckedImageSource | Sets up an image source for unchecked switches shown in switch columns. |
| | LocateCellViewInRow | Locates the specified view within cell bounds. |

**See Also**
ICellCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
How to: Customize Themes
Themes Overview

© 2018 Developer Express Inc.

# BackgroundColor Property

Specifies the cell background color.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
Color BackgroundColor { get; set; }
```

### Property Value

A Xamarin.Forms.Color object, that specifies the cell background color.

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

### See Also

ICellCustomizer Interface
ICellCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
BorderColor

© 2018 Developer Express Inc.

# BorderColor Property

Specifies the color of the cell borders.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
Color BorderColor { get; set; }
```

### Property Value

A Xamarin.Forms.Color object specifying the border color.

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

### See Also

ICellCustomizer Interface
ICellCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# Font Property

Specifies the font used to display data in grid cells.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
ThemeFontAttributes Font { get; set; }
```

### Property Value

The **ThemeFontAttributes** object, specifying font attributes.

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** | |
|---|---|
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. | |

### See Also

ICellCustomizer Interface
ICellCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
How to: Customize Themes
Themes Overview

© 2018 Developer Express Inc.

# HighlightColor Property

Specifies the color used to highlight the cell content in custom user scenarios.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
Color HighlightColor { get; set; }
```

### Property Value

A Xamarin.Forms.Color object specifying the color.

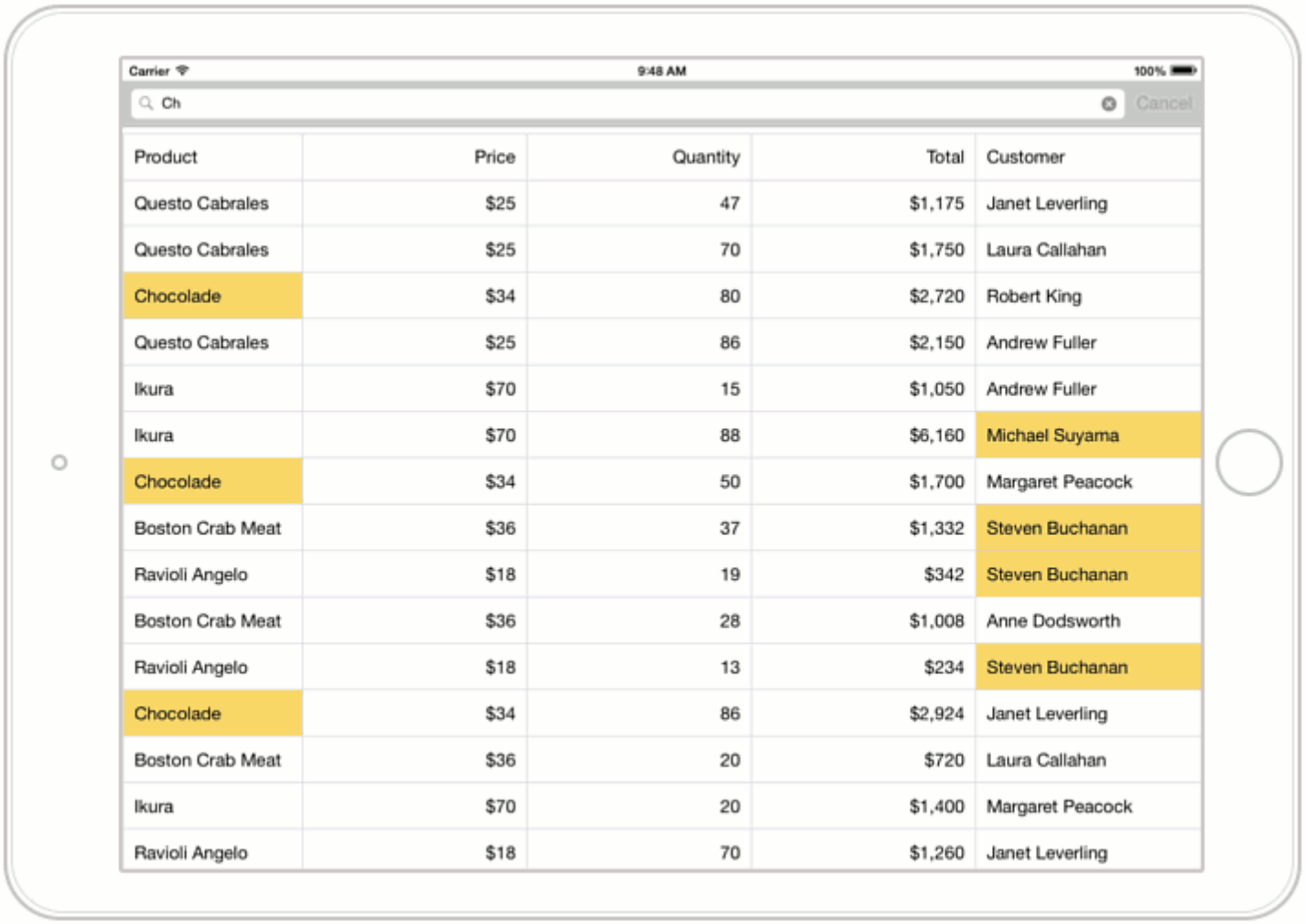
### Example

**Show Me**

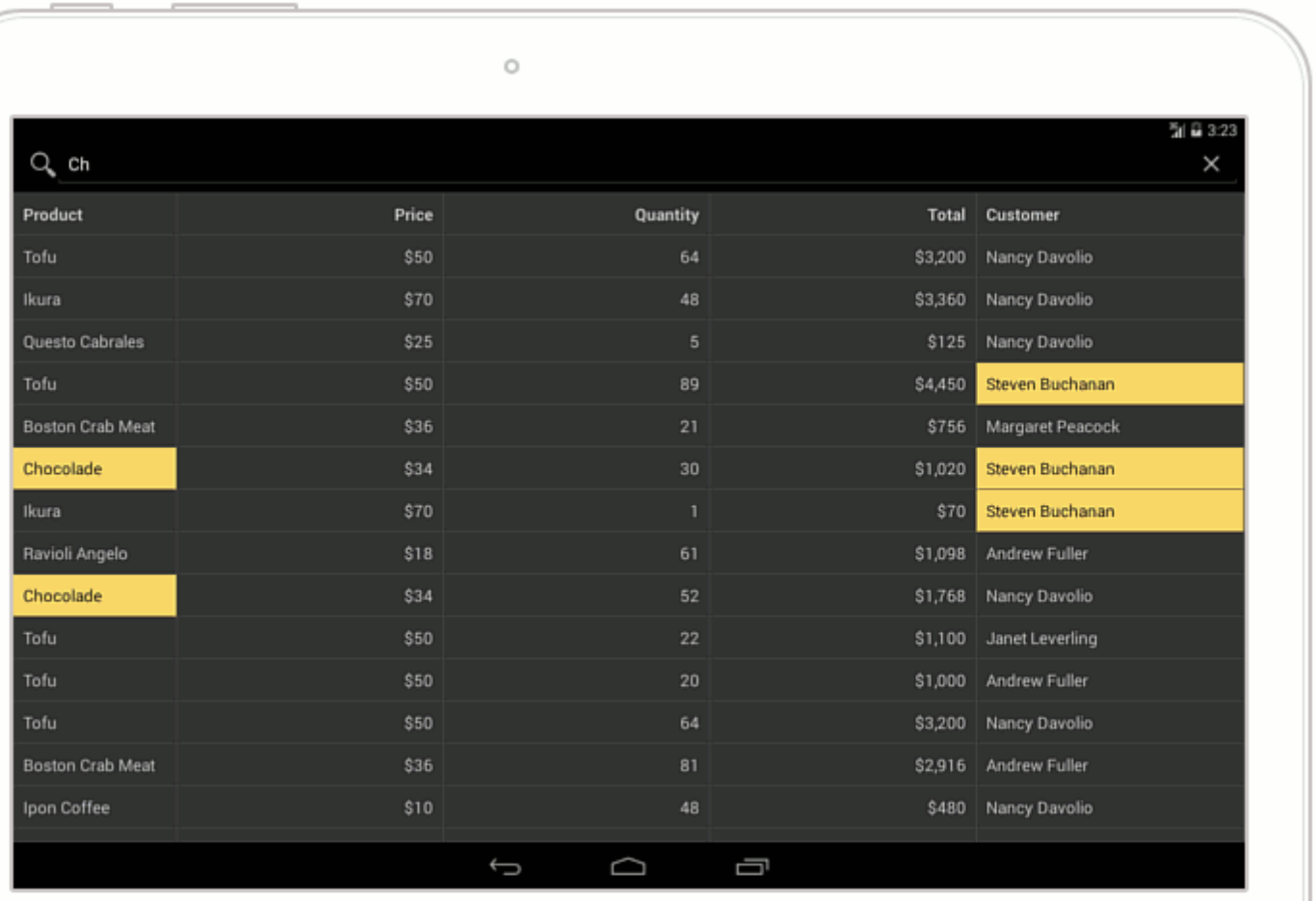
A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T294259.

This example demonstrates how to implement searching against grid columns - an end-user types in text in a search bar and all cells containing the entered string are highlighted on the fly.

© 2018 Developer Express Inc.




| Product | Price | Quantity | Total | Customer |
|---|---|---|---|---|
| Questo Cabrales | $25 | 47 | $1,175 | Janet Leverling |
| Questo Cabrales | $25 | 70 | $1,750 | Laura Callahan |
| Chocolade | $34 | 80 | $2,720 | Robert King |
| Questo Cabrales | $25 | 86 | $2,150 | Andrew Fuller |
| Ikura | $70 | 15 | $1,050 | Andrew Fuller |
| Ikura | $70 | 88 | $6,160 | Michael Suyama |
| Chocolade | $34 | 50 | $1,700 | Margaret Peacock |
| Boston Crab Meat | $36 | 37 | $1,332 | Steven Buchanan |
| Ravioli Angelo | $18 | 19 | $342 | Steven Buchanan |
| Boston Crab Meat | $36 | 28 | $1,008 | Anne Dodsworth |
| Ravioli Angelo | $18 | 13 | $234 | Steven Buchanan |
| Chocolade | $34 | 86 | $2,924 | Janet Leverling |
| Boston Crab Meat | $36 | 20 | $720 | Laura Callahan |
| Ikura | $70 | 20 | $1,400 | Margaret Peacock |
| Ravioli Angelo | $18 | 70 | $1,260 | Janet Leverling |




| Product | Price | Quantity | Total | Customer |
|---|---|---|---|---|
| Tofu | $50 | 64 | $3,200 | Nancy Davolio |
| Ikura | $70 | 48 | $3,360 | Nancy Davolio |
| Questo Cabrales | $25 | 5 | $125 | Nancy Davolio |
| Tofu | $50 | 89 | $4,450 | Steven Buchanan |
| Boston Crab Meat | $36 | 21 | $756 | Margaret Peacock |
| Chocolade | $34 | 30 | $1,020 | Steven Buchanan |
| Ikura | $70 | 1 | $70 | Steven Buchanan |
| Ravioli Angelo | $18 | 61 | $1,098 | Andrew Fuller |
| Chocolade | $34 | 52 | $1,768 | Nancy Davolio |
| Tofu | $50 | 22 | $1,100 | Janet Leverling |
| Tofu | $50 | 20 | $1,000 | Andrew Fuller |
| Tofu | $50 | 64 | $3,200 | Nancy Davolio |
| Boston Crab Meat | $36 | 81 | $2,916 | Andrew Fuller |
| Ipon Coffee | $10 | 48 | $480 | Nancy Davolio |

To implement this functionality, follow the steps below.

1. Place the **SearchBar** Xamarin.Forms standard view above the GridControl on the page.
2. Subscribe to the **SearchBar.TextChanged** event that is raised each time the text in the search bar is changed. In this event handler, get the currently entered search string and call the GridControl.Redraw method to recalculate the graphical information of the grid.
3. Before grid cells are repainted, the GridControl.CustomizeCell event is raised. Handle this event to apply the HighlightColor color to cells containing the string currently entered in the search bar.

**C#**

```
 (MainPage.xaml.cs)
using System;
using System.Collections.Generic;
using Xamarin.Forms;
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.DataGrid.Theme;
namespace GridSearchTextSample {
    public partial class MainPage : ContentPage {
        string searchText;
        public MainPage () {
            InitializeComponent ();
            TestOrdersRepository model = new TestOrdersRepository ();
            BindingContext = model;
            searchText = string.Empty;
        }
        void OnCustomizeCell(CustomizeCellEventArgs e) {
            if (string.IsNullOrEmpty(searchText))
                return;
            string cellText = e.DisplayText.ToUpper();
            if (cellText.Contains(searchText.ToUpper()) && !e.IsSelected) {
                e.BackgroundColor = ThemeManager.Theme.CellCustomizer.HighlightColor;
                e.ForeColor = Color.Black;
                e.Handled = true;
            }
        }
        void OnSearchTextChanged(object sender, EventArgs args) {
            SearchBar searchBar = (SearchBar)sender;
            searchText = searchBar.Text;
            grid.Redraw(false);
        }
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<?xml version="1.0" encoding="UTF-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="GridSearchTextSample.MainPage"
             xmlns:dxg="clr-namespace:DevExpress.Mobile.DataGrid;assembly=DevExpress.Mobile.Grid.v15.1">
    <ContentPage.Padding>
        <OnPlatform x:TypeArguments="Thickness" iOS="0, 20, 0, 0" />
    </ContentPage.Padding>
    <ContentPage.Content>
        <Grid>
            <Grid.RowDefinitions>
                <RowDefinition Height="auto"/>
                <RowDefinition Height="*"/>
            </Grid.RowDefinitions>
            <Grid.ColumnDefinitions>
                <ColumnDefinition Width="*"/>
            </Grid.ColumnDefinitions>
            <SearchBar Grid.Row="0" Grid.Column="0" x:Name="searchBar"
                       Placeholder="Enter a value for search"
                       TextChanged="OnSearchTextChanged"/>
            <dxg:GridControl Grid.Row="1" Grid.Column="0" x:Name="grid"
                             ItemsSource="{Binding Orders}"
                             CustomizeCell="OnCustomizeCell">
                <dxg:GridControl.Columns>
                    <dxg:TextColumn FieldName="Product.Name" Caption="Product" Width="170"/>
                    <dxg:NumberColumn FieldName="Product.UnitPrice" Caption="Price" DisplayFormat="C0"/>
                    <dxg:NumberColumn FieldName="Quantity"/>
                    <dxg:NumberColumn FieldName="Total"
                                      UnboundType="Integer" UnboundExpression="[Quantity] * [Product.Unit
                                      DisplayFormat="C0" IsReadOnly="True"/>
                    <dxg:TextColumn FieldName="Customer.Name" Caption="Customer"/>
                </dxg:GridControl.Columns>
            </dxg:GridControl>
        </Grid>
    </ContentPage.Content>
</ContentPage>
```

## See Also

ICellCustomizer Interface
ICellCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# Padding Property

Specifies the internal spaces between the cell content and borders.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
Thickness Padding { get; set; }
```

### Property Value

A Xamarin.Forms.Thickness object, representing the cell inner indents, measured in the density-independent pixels.

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

### See Also

ICellCustomizer Interface
ICellCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
How to: Customize Themes
Themes Overview

© 2018 Developer Express Inc.

# SelectionColor Property

Specifies the background color on selecting a cell.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
Color SelectionColor { get; set; }
```

### Property Value

A Xamarin.Forms.Color object specifying the background color on selecting the cell.

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

### See Also

ICellCustomizer Interface
ICellCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
How to: Customize Themes
Themes Overview

© 2018 Developer Express Inc.

# SelectionFontColor Property

Specifies the font color on selecting a cell.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
Color SelectionFontColor { get; set; }
```

### Property Value

A Xamarin.Forms.Color object specifying the font color on selecting the cell.

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

### See Also

ICellCustomizer Interface
ICellCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
How to: Customize Themes
Themes Overview

© 2018 Developer Express Inc.

# ICellCustomizer Methods

Defines classes that customize grid cells.

The following tables list the members exposed by the ICellCustomizer type.

**Public Methods**

| | Name | Description |
|---|---|---|
| | CreateCheckedImageSource | Sets up an image source for checked switches shown in switch columns. |
| | CreateUnassignedImageSource | Sets up an image source for unassigned switches shown in switch columns. |
| | CreateUncheckedImageSource | Sets up an image source for unchecked switches shown in switch columns. |
| | LocateCellViewInRow | Locates the specified view within cell bounds. |

**See Also**
ICellCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
How to: Customize Themes
Themes Overview

© 2018 Developer Express Inc.

# CreateCheckedImageSource Method

Sets up an image source for checked switches shown in switch columns.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
ImageSource CreateCheckedImageSource()
```

### Return Value

The Xamarin.Forms.ImageSource object, that specifies the image source.

### Remarks

In the GridControl, data fields are represented as columns and records are displayed as rows. So, to display data, a grid should contain columns bound to fields of the underlying data source.

An individual column is specified by the SwitchColumn, NumberColumn, TextColumn or DateColumn object (a GridColumn descendant) depending on the type of data displayed in this column. Use columns of the **SwitchColumn** type that contain switches within cells to allow end-users to change Boolean values.

The **CreateCheckedImageSource** method allows you to set up the image that will be shown in columns of the **SwitchColumn** type once the switch is set to True (is checked).

The **CreateCheckedImageSource** method is abstract. So, to change the image source shown in your grid, override the method in the ICellCustomizer descendants.

### See Also

ICellCustomizer Interface
ICellCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
CreateUnassignedImageSource
CreateUncheckedImageSource

© 2018 Developer Express Inc.

# CreateUnassignedImageSource Method

Sets up an image source for unassigned switches shown in switch columns.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
ImageSource CreateUnassignedImageSource()
```

### Return Value

The Xamarin.Forms.ImageSource object, that specifies the image source.

### Remarks

In the GridControl, data fields are represented as columns and records are displayed as rows. So, to display data, a grid should contain columns bound to fields of the underlying data source.

An individual column is specified by the SwitchColumn, NumberColumn, TextColumn or DateColumn object (a GridColumn descendant) depending on the type of data displayed in this column. Use columns of the **SwitchColumn** type that contain switches within cells to allow end-users to change Boolean values.

The **CreateUnassignedImageSource** method allows you to set up the image that will be shown in columns of the **SwitchColumn** type once the switch value is not specified.

The **CreateUnassignedImageSource** method is abstract. So, to change the image source shown in your grid, override the method in the ICellCustomizer descendants.

### See Also

ICellCustomizer Interface
ICellCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
CreateUncheckedImageSource
CreateCheckedImageSource

© 2018 Developer Express Inc.

# CreateUncheckedImageSource Method

Sets up an image source for unchecked switches shown in switch columns.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
ImageSource CreateUncheckedImageSource()
```

### Return Value

The Xamarin.Forms.ImageSource object, that specifies the image source.

### Remarks

In the GridControl, data fields are represented as columns and records are displayed as rows. So, to display data, a grid should contain columns bound to fields of the underlying data source.

An individual column is specified by the SwitchColumn, NumberColumn, TextColumn or DateColumn object (a GridColumn descendant) depending on the type of data displayed in this column. Use columns of the **SwitchColumn** type that contain switches within cells to allow end-users to change Boolean values.

The **CreateUncheckedImageSource** method allows you to set up the image that will be shown in columns of the **SwitchColumn** type once the switch is set to False (is not checked).

The **CreateUncheckedImageSource** method is abstract. So, to change the image source shown in your grid, override the method in the ICellCustomizer descendants.

### See Also

ICellCustomizer Interface
ICellCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
How to: Customize Themes
Themes Overview
CreateCheckedImageSource
CreateUnassignedImageSource

© 2018 Developer Express Inc.

# LocateCellViewInRow Method

Locates the specified view within cell bounds.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
void LocateCellViewInRow(
   View cell,
   Rectangle cellBounds,
   bool isFirst,
   bool isLast,
   bool isLeftFixedColumnBorder,
   bool isRightFixedColumnBorder
)
```

### Parameters

*cell*
A Xamarin.Forms.View object, specifying the view to be located within a cell.
*cellBounds*
A Xamarin.Forms.Rectangle object, specifying the cell bounds.
*isFirst*
**true**, if the specified view is the first view in a cell; otherwise, **false**.
*isLast*
**true**, if the specified view is the last view in a cell; otherwise, **false**.
*isLeftFixedColumnBorder*
**true**, if the specified view belongs to the last column fixed to the left; otherwise, **false**.
*isRightFixedColumnBorder*
**true**, if the specified view belongs to the first column fixed to the right; otherwise, **false**.

### See Also

ICellCustomizer Interface
ICellCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
How to: Customize Themes

© 2018 Developer Express Inc.

# IColumnChooserCustomizer Interface

Defines classes that customize a visual presentation of the Column Chooser dialog.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public interface IColumnChooserCustomizer
```

### Remarks

To change the font used to display column names in the **Column Chooser** dialog, use the LabelFont property.

| Note |
| --- |
| To customize the dialog and its appearance, use objects that implement the IDialogFormCustomizer interface. |

### Inheritance Hierarchy

### See Also

IColumnChooserCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
How to: Customize Themes

© 2018 Developer Express Inc.

# IColumnChooserCustomizer Members

Defines classes that customize a visual presentation of the Column Chooser dialog.

The following tables list the members exposed by the IColumnChooserCustomizer type.

**Public Properties**

| | Name | Description |
|---|---|---|
| | LabelFont | Specifies a font used to display column names in the Column Chooser dialog. |

**See Also**
IColumnChooserCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
How to: Customize Themes

© 2018 Developer Express Inc.

# IColumnChooserCustomizer Properties

Defines classes that customize a visual presentation of the Column Chooser dialog.

The following tables list the members exposed by the IColumnChooserCustomizer type.

**Public Properties**

| | Name | Description |
|---|---|---|
| | LabelFont | Specifies a font used to display column names in the Column Chooser dialog. |

**See Also**
IColumnChooserCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
How to: Customize Themes

© 2018 Developer Express Inc.

# LabelFont Property

Specifies a font used to display column names in the Column Chooser dialog.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
ThemeFontAttributes LabelFont { get; set; }
```

### Property Value

A **ThemeFontAttributes** object specifying the labels font.

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

### See Also

IColumnChooserCustomizer Interface
IColumnChooserCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# IDialogFormCustomizer Interface

Defines classes that customize both the Column Chooser dialog and the Edit Form.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public interface IDialogFormCustomizer
```

### Remarks

The GridControl allows editing its data either with an in-place row editor or an edit form, depending on the row edit mode set. To customize the in-place row editors, use the ThemeBase.EditorCustomizer property.

To customize the **Column Chooser** dialog, popup and area-based edit forms, use the objects that implement the **IDialogFormCustomizer** interface. For example, you can use objects returned by the ThemeBase.DialogFormCustomizer property.

| Note |
| --- |
| To customize the font used to display text in the **Column Chooser** dialog, use the IColumnChooserCustomizer.LabelFont property. |

### Inheritance Hierarchy

### See Also

IDialogFormCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
How to: Customize Themes
IColumnChooserCustomizer

© 2018 Developer Express Inc.

# IDialogFormCustomizer Members

Defines classes that customize both the Column Chooser dialog and the Edit Form.

The following tables list the members exposed by the IDialogFormCustomizer type.

**Public Properties**

| | Name | Description |
|---|---|---|
| | BackgroundColor | Specifies the background color of the Column Chooser dialog and the popup edit form. |
| | CaptionBackgroundColor | Specifies the caption's background color of the Column Chooser dialog, popup and full screen edit forms. |
| | CaptionFont | Specifies the font used to display captions of the Column Chooser dialog and the popup edit forms. |
| | FullScreenBackgroundColor | Specifies the background color for the **Full Screen Area** and **Grid Area** edit modes. |
| | FullScreenBorderColor | Specifies the color of the line dividing the **Full Screen Area** edit form from the caption and the content. |
| | FullScreenCaptionFont | Specifies the font used to display captions of the **Full Screen Area** and the **Grid Area** edit forms. |
| | PanelPadding | Specifies the internal space between the Column Chooser dialog's and the edit form's content and their edges. |

**See Also**
IDialogFormCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
How to: Customize Themes
IColumnChooserCustomizer

© 2018 Developer Express Inc.

# IDialogFormCustomizer Properties

Defines classes that customize both the Column Chooser dialog and the Edit Form.

The following tables list the members exposed by the IDialogFormCustomizer type.

**Public Properties**

| | Name | Description |
|---|---|---|
| | BackgroundColor | Specifies the background color of the Column Chooser dialog and the popup edit form. |
| | CaptionBackgroundColor | Specifies the caption's background color of the Column Chooser dialog, popup and full screen edit forms. |
| | CaptionFont | Specifies the font used to display captions of the Column Chooser dialog and the popup edit forms. |
| | FullScreenBackgroundColor | Specifies the background color for the **Full Screen Area** and **Grid Area** edit modes. |
| | FullScreenBorderColor | Specifies the color of the line dividing the **Full Screen Area** edit form from the caption and the content. |
| | FullScreenCaptionFont | Specifies the font used to display captions of the **Full Screen Area** and the **Grid Area** edit forms. |
| | PanelPadding | Specifies the internal space between the Column Chooser dialog's and the edit form's content and their edges. |

**See Also**
IDialogFormCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
How to: Customize Themes
IColumnChooserCustomizer

© 2018 Developer Express Inc.

# BackgroundColor Property

Specifies the background color of the Column Chooser dialog and the popup edit form.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
Color BackgroundColor { get; set; }
```

### Property Value

A Xamarin.Forms.Color object specifying the background color.

### Remarks

To customize the background color of the full screen edit forms, use the FullScreenBackgroundColor property.

To customize the caption's background color of the **Column Chooser** dialog and popup edit forms, use the CaptionBackgroundColor property.

### See Also

IDialogFormCustomizer Interface
IDialogFormCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
How to: Customize Themes
How to: Customize Popup Menu

© 2018 Developer Express Inc.

# CaptionBackgroundColor Property

Specifies the caption's background color of the Column Chooser dialog, popup and full screen edit forms.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
Color CaptionBackgroundColor { get; set; }
```

### Property Value

A Xamarin.Forms.Color object specifying the background color.

### Remarks

To customize the font used to display captions of the **Column Chooser** dialog and popup edit forms, use the CaptionFont property.

To customize the font used to display captions of full screen edit forms, use the FullScreenCaptionFont property.

To customize the background color for the entire **Column Chooser** dialog and popup edit forms, use the BackgroundColor property.

To customize the background color for the full screen edit forms, use the FullScreenBackgroundColor property.

### See Also

IDialogFormCustomizer Interface
IDialogFormCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
How to: Customize Themes
How to: Customize Popup Menu

© 2018 Developer Express Inc.

# CaptionFont Property

Specifies the font used to display captions of the Column Chooser dialog and the popup edit forms.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
ThemeFontAttributes CaptionFont { get; set; }
```

### Property Value

A **ThemeFontAttributes** object specifying the captions font.

### Remarks

To customize the captions background font, use the CaptionBackgroundColor property.

### See Also

IDialogFormCustomizer Interface
IDialogFormCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
How to: Customize Themes
How to: Customize Popup Menu

© 2018 Developer Express Inc.

# FullScreenBackgroundColor Property

Specifies the background color for the **Full Screen Area** and **Grid Area** edit modes.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
Color FullScreenBackgroundColor { get; set; }
```

### Property Value

A Xamarin.Forms.Color object specifying the background color.

### Remarks

To customize the background color for the **popup edit forms**, use the BackgroundColor property.

### See Also

IDialogFormCustomizer Interface
IDialogFormCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# FullScreenBorderColor Property

Specifies the color of the line dividing the **Full Screen Area** edit form from the caption and the content.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
Color FullScreenBorderColor { get; set; }
```

### Property Value

A Xamarin.Forms.Color object specifying the line color.

### Remarks

To customize the full screen edit form, also use the FullScreenBackgroundColor and FullScreenCaptionFont properties.

### See Also

IDialogFormCustomizer Interface
IDialogFormCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# FullScreenCaptionFont Property

Specifies the font used to display captions of the **Full Screen Area** and the **Grid Area** edit forms.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
ThemeFontAttributes FullScreenCaptionFont { get; set; }
```

### Property Value

A **ThemeFontAttributes** object specifying the font.

### Remarks

To customize the font used for displaying captions of the **popup edit forms**, use the CaptionFont property.

### See Also

IDialogFormCustomizer Interface
IDialogFormCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# PanelPadding Property

Specifies the internal space between the Column Chooser dialog's and the edit form's content and their edges.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
Thickness PanelPadding { get; set; }
```

### Property Value

A Xamarin.Forms.Thickness object, representing the **Column Chooser** dialog and the **edit form** inner indents, measured in the density-independent pixels.

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

### See Also

IDialogFormCustomizer Interface
IDialogFormCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
How to: Customize Themes
How to: Customize Popup Menu

© 2018 Developer Express Inc.

# IEditorCustomizer Interface

Defines classes that customize editors located in the Auto Filter panel, and the New Item Row and in-place edit forms.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public interface IEditorCustomizer
```

### Remarks

To customize an editor's background color, use the EditorBackgroundColor property.

To customize an editor's font, use the Font property (for in-place edit forms) and the FormEditorsFont property (for popup and area-based edit forms).

### Inheritance Hierarchy

### See Also

IEditorCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
How to: Customize Themes

© 2018 Developer Express Inc.

# IEditorCustomizer Members

Defines classes that customize editors located in the Auto Filter panel, and the New Item Row and in-place edit forms.

The following tables list the members exposed by the IEditorCustomizer type.

**Public Properties**

| | Name | Description |
|---|---|---|
| | EditorBackgroundColor | Specifies the background color of the focused editors located within the Auto Filter panel, the New Item Row and the in-palce data editors. |
| | Font | Specifies the font used to display data in the editors located within the Auto Filter panel, the New Item Row and in-palce edit forms. |
| | FormEditorsFont | Specifies the font used to display the text in the editors located within the popup and area-based edit forms. |
| | UnfocusedCellHighlighting | Specifies the color used to highlight unfocused editors located within the New Item Row and the in-place row editors. |

**Public Methods**

| | Name | Description |
|---|---|---|
| | LocateCellViewInRow | Locates the specified view within an editor's bounds. |

**See Also**
IEditorCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
How to: Customize Themes

© 2018 Developer Express Inc.

# IEditorCustomizer Properties

Defines classes that customize editors located in the Auto Filter panel, and the New Item Row and in-place edit forms.

The following tables list the members exposed by the IEditorCustomizer type.

**Public Properties**

| | Name | Description |
|---|---|---|
| | EditorBackgroundColor | Specifies the background color of the focused editors located within the Auto Filter panel, the New Item Row and the in-palce data editors. |
| | Font | Specifies the font used to display data in the editors located within the Auto Filter panel, the New Item Row and in-palce edit forms. |
| | FormEditorsFont | Specifies the font used to display the text in the editors located within the popup and area-based edit forms. |
| | UnfocusedCellHighlighting | Specifies the color used to highlight unfocused editors located within the New Item Row and the in-place row editors. |

**Public Methods**

| | Name | Description |
|---|---|---|
| | LocateCellViewInRow | Locates the specified view within an editor's bounds. |

**See Also**
IEditorCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
How to: Customize Themes

© 2018 Developer Express Inc.

# EditorBackgroundColor Property

Specifies the background color of the focused editors located within the Auto Filter panel, the New Item Row and the in-palce data editors.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
Color EditorBackgroundColor { get; set; }
```

### Property Value

A Xamarin.Forms.Color object, specifying the background color.

### Remarks

To customize the background color of unfocused editors, use the UnfocusedCellHighlighting property.

### See Also

IEditorCustomizer Interface
IEditorCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
How to: Customize Themes

© 2018 Developer Express Inc.

# Font Property

Specifies the font used to display data in the editors located within the Auto Filter panel, the New Item Row and in-palce edit forms.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
ThemeFontAttributes Font { get; set; }
```

### Property Value

The **ThemeFontAttributes** object, specifying font attributes.

### Remarks

To customize the font of the popup and area-based edit forms, use the FormEditorsFont property.

### See Also

IEditorCustomizer Interface
IEditorCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
How to: Customize Themes

© 2018 Developer Express Inc.

# FormEditorsFont Property

Specifies the font used to display the text in the editors located within the popup and area-based edit forms.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
ThemeFontAttributes FormEditorsFont { get; set; }
```

### Property Value

The **ThemeFontAttributes** object, specifying the font attributes.

### Remarks

To customize the font of the in-place edit forms, use the Font property.

### See Also

IEditorCustomizer Interface
IEditorCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
How to: Customize Themes

© 2018 Developer Express Inc.

# UnfocusedCellHighlighting Property

Specifies the color used to highlight unfocused editors located within the New Item Row and the in-place row editors.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
Color UnfocusedCellHighlighting { get; set; }
```

### Property Value

A Xamarin.Forms.Color object specifying the required color.

### Remarks

To specify the background color for focused editors, use the EditorBackgroundColor property.

### See Also

IEditorCustomizer Interface
IEditorCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
How to: Customize Themes

© 2018 Developer Express Inc.

# IEditorCustomizer Methods

Defines classes that customize editors located in the Auto Filter panel, and the New Item Row and in-place edit forms.

The following tables list the members exposed by the IEditorCustomizer type.

**Public Methods**

| | Name | Description |
|---|---|---|
| | LocateCellViewInRow | Locates the specified view within an editor's bounds. |

**See Also**
IEditorCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
How to: Customize Themes

© 2018 Developer Express Inc.

# LocateCellViewInRow Method

Locates the specified view within an editor's bounds.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
void LocateCellViewInRow(
   View cell,
   Rectangle cellBounds,
   bool isFirst,
   bool isLast,
   bool isLeftFixedColumnBorder,
   bool isRightFixedColumnBorder
)
```

### Parameters

*cell*
A Xamarin.Forms.View object, specifying the view to be located within an editor.
*cellBounds*
A Xamarin.Forms.Rectangle object, specifying an editor's bounds.
*isFirst*
**true**, if the specified view is the first view in an editor; otherwise, **false**.
*isLast*
**true**, if the specified view is the last view in an editor; otherwise, **false**.
*isLeftFixedColumnBorder*
**true**, if the specified view belongs to the last column fixed to the left; otherwise, **false**.
*isRightFixedColumnBorder*
**true**, if the specified view belongs to the first column fixed to the right; otherwise, **false**.

### See Also

IEditorCustomizer Interface
IEditorCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
How to: Customize Themes

© 2018 Developer Express Inc.

# IFilterPanelCustomizer Interface

Defines classes that customize the filter panel.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public interface IFilterPanelCustomizer
```

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

### Inheritance Hierarchy

### See Also

IFilterPanelCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# IFilterPanelCustomizer Members

Defines classes that customize the filter panel.

The following tables list the members exposed by the IFilterPanelCustomizer type.

**Public Properties**

| | Name | Description |
|---|---|---|
| | BackgroundColor | Specifies the filter panel's background color. |
| | BorderColor | Specifies the filter panel's border color. |
| | Font | Specifies the font used to display data in the filter panel. |
| | PanelPadding | Specifies the internal space between the filter panel's content and its edges. |

**Public Methods**

| | Name | Description |
|---|---|---|
| | CreateClearFilterImageSource | Sets up an image source for the "Clear Filter" icon shown in the filter panel. |
| | LocateContentInPanel | Locates the specified view within the filter panel bounds. |

**See Also**
IFilterPanelCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# IFilterPanelCustomizer Properties

Defines classes that customize the filter panel.

The following tables list the members exposed by the IFilterPanelCustomizer type.

**Public Properties**

| | Name | Description |
|---|---|---|
| | BackgroundColor | Specifies the filter panel's background color. |
| | BorderColor | Specifies the filter panel's border color. |
| | Font | Specifies the font used to display data in the filter panel. |
| | PanelPadding | Specifies the internal space between the filter panel's content and its edges. |

**Public Methods**

| | Name | Description |
|---|---|---|
| | CreateClearFilterImageSource | Sets up an image source for the "Clear Filter" icon shown in the filter panel. |
| | LocateContentInPanel | Locates the specified view within the filter panel bounds. |

**See Also**
IFilterPanelCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# BackgroundColor Property

Specifies the filter panel's background color.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
Color BackgroundColor { get; set; }
```

### Property Value

A Xamarin.Forms.Color object specifying the background color.

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

### See Also

IFilterPanelCustomizer Interface
IFilterPanelCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
How to: Customize Themes
Themes Overview

© 2018 Developer Express Inc.

# BorderColor Property

Specifies the filter panel's border color.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
Color BorderColor { get; set; }
```

### Property Value

A Xamrin.Forms.Color object specifying the border color.

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

### See Also

IFilterPanelCustomizer Interface
IFilterPanelCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# Font Property

Specifies the font used to display data in the filter panel.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
ThemeFontAttributes Font { get; set; }
```

### Property Value

The **ThemeFontAttributes** object, specifying font attributes.

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

### See Also

IFilterPanelCustomizer Interface
IFilterPanelCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
How to: Customize Themes
Themes Overview

© 2018 Developer Express Inc.

# PanelPadding Property

Specifies the internal space between the filter panel's content and its edges.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
Thickness PanelPadding { get; set; }
```

### Property Value

A Xamarin.Forms.Thickness object, representing the **filter panel** inner indents, measured in the density-independent pixels.

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

### See Also

IFilterPanelCustomizer Interface
IFilterPanelCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# IFilterPanelCustomizer Methods

Defines classes that customize the filter panel.

The following tables list the members exposed by the IFilterPanelCustomizer type.

**Public Methods**

| | Name | Description |
|---|---|---|
| | CreateClearFilterImageSource | Sets up an image source for the "Clear Filter" icon shown in the filter panel. |
| | LocateContentInPanel | Locates the specified view within the filter panel bounds. |

**See Also**
IFilterPanelCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# CreateClearFilterImageSource Method

Sets up an image source for the "Clear Filter" icon shown in the filter panel.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
ImageSource CreateClearFilterImageSource()
```

### Return Value

The Xamarin.Forms.ImageSource object, that specifies the image source.

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

The **CreateClearFilterImageSource** method is abstract. So, to change the image source shown in your grid, override the method in the IFilterPanelCustomizer descendants.

### See Also

IFilterPanelCustomizer Interface
IFilterPanelCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# LocateContentInPanel Method

Locates the specified view within the filter panel bounds.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
void LocateContentInPanel(
   View content,
   Rectangle contentBounds
)
```

### Parameters

*content*
A Xamarin.Forms.View object, specifying the view to be located within the **filter panel**.
*contentBounds*
A Xamarin.Forms.Rectangle object, specifying the **filter panel** bounds.

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

### See Also

IFilterPanelCustomizer Interface
IFilterPanelCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# IGridControlCustomizer Interface

Defines classes that customize the GridControl.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public interface IGridControlCustomizer
```

### Remarks

To customize the grid's background color, use the BackgroundColor property.

To customize the grid's borders, use the BorderColor and BorderThickness properties.

### Inheritance Hierarchy

### See Also

IGridControlCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
How to: Customize Themes

© 2018 Developer Express Inc.

# IGridControlCustomizer Members

Defines classes that customize the GridControl.

The following tables list the members exposed by the IGridControlCustomizer type.

**Public Properties**

| | Name | Description |
|---|---|---|
| | BackgroundColor | Specifies the GridControl's background color. |
| | BorderColor | Specifies the GridControl's border color. |
| | BorderThickness | Specifies the GridControl's border thickness, measured in density-independent pixels. |
| | PopupMenuHighlightColor | Specifies the highlight color of the GridControl's popup menu. |

**See Also**
IGridControlCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
How to: Customize Themes

© 2018 Developer Express Inc.

# IGridControlCustomizer Properties

Defines classes that customize the GridControl.

The following tables list the members exposed by the IGridControlCustomizer type.

**Public Properties**

| | Name | Description |
|---|---|---|
| | BackgroundColor | Specifies the GridControl's background color. |
| | BorderColor | Specifies the GridControl's border color. |
| | BorderThickness | Specifies the GridControl's border thickness, measured in density-independent pixels. |
| | PopupMenuHighlightColor | Specifies the highlight color of the GridControl's popup menu. |

**See Also**
IGridControlCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
How to: Customize Themes

© 2018 Developer Express Inc.

# BackgroundColor Property

Specifies the GridControl's background color.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
Color BackgroundColor { get; set; }
```

### Property Value

A Xamarin.Forms.Color object specifying the background color.

### Remarks

The **BackgroundColor** property's value is used to fill the empty area (if any) that is not occupied by grid rows.

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

### See Also

IGridControlCustomizer Interface
IGridControlCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# BorderColor Property

Specifies the GridControl's border color.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
Color BorderColor { get; set; }
```

### Property Value

A Xamrin.Forms.Color object specifying the border color.

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

### See Also

IGridControlCustomizer Interface
IGridControlCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# BorderThickness Property

Specifies the GridControl's border thickness, measured in density-independent pixels.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
Thickness BorderThickness { get; set; }
```

### Property Value

A Xamarin.Forms.Thickness object, representing the border thickness.

### See Also

IGridControlCustomizer Interface
IGridControlCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
How to: Customize Themes

© 2018 Developer Express Inc.

# PopupMenuHighlightColor Property

Specifies the highlight color of the GridControl's popup menu.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
Color PopupMenuHighlightColor { get; set; }
```

### Property Value

A Xamarin.Forms.Color object, specifying the highlight color.

### Remarks

To customize the **GridControl**'s highlight color, use the BackgroundColor property.

### See Also

IGridControlCustomizer Interface
IGridControlCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
How to: Customize Themes

© 2018 Developer Express Inc.

# IGroupCustomizer Interface

Defines classes that customize group rows.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public interface IGroupCustomizer
```

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

### Inheritance Hierarchy

### See Also

IGroupCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# IGroupCustomizer Members

Defines classes that customize group rows.

The following tables list the members exposed by the IGroupCustomizer type.

**Public Properties**

| | Name | Description |
|---|---|---|
| | BackgroundColor | Specifies the group rows' background color. |
| | Font | Specifies the font used to display data in group rows. |
| | Padding | Specifies the internal space between the group row's content and its edges. |

**Public Methods**

| | Name | Description |
|---|---|---|
| | CreateArrowRightImageSource | Sets up a source for the icon shown on the left of the collapsed group rows. |
| | LocateViewInRow | Locates the specified view within the group row's bounds. |

**See Also**
IGroupCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# IGroupCustomizer Properties

Defines classes that customize group rows.

The following tables list the members exposed by the IGroupCustomizer type.

**Public Properties**

| | Name | Description |
|---|---|---|
| | BackgroundColor | Specifies the group rows' background color. |
| | Font | Specifies the font used to display data in group rows. |
| | Padding | Specifies the internal space between the group row's content and its edges. |

**Public Methods**

| | Name | Description |
|---|---|---|
| | CreateArrowRightImageSource | Sets up a source for the icon shown on the left of the collapsed group rows. |
| | LocateViewInRow | Locates the specified view within the group row's bounds. |

**See Also**
IGroupCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# BackgroundColor Property

Specifies the group rows' background color.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
Color BackgroundColor { get; set; }
```

### Property Value

A Xamarin.Forms.Color object specifying the background color.

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

### See Also

IGroupCustomizer Interface
IGroupCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# Font Property

Specifies the font used to display data in group rows.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
ThemeFontAttributes Font { get; set; }
```

### Property Value

The **ThemeFontAttributes** object, specifying font attributes.

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

### See Also

IGroupCustomizer Interface
IGroupCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
How to: Customize Themes
Themes Overview

© 2018 Developer Express Inc.

# Padding Property

Specifies the internal space between the group row's content and its edges.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
Thickness Padding { get; set; }
```

### Property Value

A Xamarin.Forms.Thickness object, representing the **group row** inner indents, measured in the density-independent pixels.

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

### See Also

IGroupCustomizer Interface
IGroupCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# IGroupCustomizer Methods

Defines classes that customize group rows.

The following tables list the members exposed by the IGroupCustomizer type.

**Public Methods**

| | Name | Description |
|---|---|---|
| | CreateArrowRightImageSource | Sets up a source for the icon shown on the left of the collapsed group rows. |
| | LocateViewInRow | Locates the specified view within the group row's bounds. |

**See Also**
IGroupCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# CreateArrowRightImageSource Method

Sets up a source for the icon shown on the left of the collapsed group rows.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
ImageSource CreateArrowRightImageSource()
```

### Return Value

The Xamarin.Forms.ImageSource object, that specifies the image source.

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

The **CreateArrowRightImageSource** method is abstract. So, to change the image source shown in your grid, override the method in the IGroupCustomizer descendants.

### See Also

IGroupCustomizer Interface
IGroupCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# LocateViewInRow Method

Locates the specified view within the group row's bounds.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
void LocateViewInRow(
   View cell,
   Rectangle cellBounds
)
```

### Parameters

*cell*
A Xamarin.Forms.View object, specifying the view to be located within the group row.
*cellBounds*
A Xamarin.Forms.Rectangle object, specifying the group row's bounds.

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

### See Also

IGroupCustomizer Interface
IGroupCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# IHeaderCustomizer Interface

Defines classes that customize the grid header - the row that contains headers of all grid columns.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public interface IHeaderCustomizer
```

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

### Inheritance Hierarchy

### See Also

IHeaderCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# IHeaderCustomizer Members

Defines classes that customize the grid header - the row that contains headers of all grid columns.

The following tables list the members exposed by the IHeaderCustomizer type.

### Public Properties

| | Name | Description |
|---|---|---|
| | BackgroundColor | Specifies the grid header's background color. |
| | BorderColor | Specifies the grid header's border color. |
| | Font | Specifies the font used to display data in the grid header. |
| | Padding | Specifies the internal space between the grid header's content and its edges. |

### Public Methods

| | Name | Description |
|---|---|---|
|  | CreateArrowDownImageSource | Sets up a source for the icon that is shown in the header of the column sorted in descending order. |
|  | CreateArrowNoneImageSource | Sets up a source for the icon that is shown in the header of the unsorted column. |
|  | CreateArrowUpImageSource | Sets up a source for the icon that is shown in the header of the column sorted in ascending order. |
|  | LocateHeaderViewInPanel | Locates the specified view within the grid header's bounds. |

**See Also**
IHeaderCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# IHeaderCustomizer Properties

Defines classes that customize the grid header - the row that contains headers of all grid columns.

The following tables list the members exposed by the IHeaderCustomizer type.

**Public Properties**

| | Name | Description |
|---|---|---|
| | BackgroundColor | Specifies the grid header's background color. |
| | BorderColor | Specifies the grid header's border color. |
| | Font | Specifies the font used to display data in the grid header. |
| | Padding | Specifies the internal space between the grid header's content and its edges. |

**Public Methods**

| | Name | Description |
|---|---|---|
| | CreateArrowDownImageSource | Sets up a source for the icon that is shown in the header of the column sorted in descending order. |
| | CreateArrowNoneImageSource | Sets up a source for the icon that is shown in the header of the unsorted column. |
| | CreateArrowUpImageSource | Sets up a source for the icon that is shown in the header of the column sorted in ascending order. |
| | LocateHeaderViewInPanel | Locates the specified view within the grid header's bounds. |

**See Also**
IHeaderCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# BackgroundColor Property

Specifies the grid header's background color.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
Color BackgroundColor { get; set; }
```

### Property Value

A Xamarin.Forms.Color object specifying the background color.

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

### See Also

IHeaderCustomizer Interface
IHeaderCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
How to: Customize Themes
Themes Overview

© 2018 Developer Express Inc.

# BorderColor Property

Specifies the grid header's border color.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
Color BorderColor { get; set; }
```

### Property Value

A Xamrin.Forms.Color object specifying the border color.

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

### See Also

IHeaderCustomizer Interface
IHeaderCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# Font Property

Specifies the font used to display data in the grid header.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
ThemeFontAttributes Font { get; set; }
```

### Property Value

The **ThemeFontAttributes** object, specifying font attributes.

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

### See Also

IHeaderCustomizer Interface
IHeaderCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
How to: Customize Themes
Themes Overview

© 2018 Developer Express Inc.

# Padding Property

Specifies the internal space between the grid header's content and its edges.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
Thickness Padding { get; set; }
```

### Property Value

A Xamarin.Forms.Thickness object, representing the grid header inner indents, measured in the density-independent pixels.

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

### See Also

IHeaderCustomizer Interface
IHeaderCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# IHeaderCustomizer Methods

Defines classes that customize the grid header - the row that contains headers of all grid columns.

The following tables list the members exposed by the IHeaderCustomizer type.

**Public Methods**

| | Name | Description |
|---|---|---|
| | CreateArrowDownImageSource | Sets up a source for the icon that is shown in the header of the column sorted in descending order. |
| | CreateArrowNoneImageSource | Sets up a source for the icon that is shown in the header of the unsorted column. |
| | CreateArrowUpImageSource | Sets up a source for the icon that is shown in the header of the column sorted in ascending order. |
| | LocateHeaderViewInPanel | Locates the specified view within the grid header's bounds. |

**See Also**
IHeaderCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# CreateArrowDownImageSource Method

Sets up a source for the icon that is shown in the header of the column sorted in descending order.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
ImageSource CreateArrowDownImageSource()
```

### Return Value

The Xamarin.Forms.ImageSource object, that specifies the image source.

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

The **CreateArrowDownImageSource** method is abstract. So, to change the image source shown in your grid, override the method in the IHeaderCustomizer descendants.

### See Also

IHeaderCustomizer Interface
IHeaderCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
CreateArrowNoneImageSource
CreateArrowUpImageSource

© 2018 Developer Express Inc.

# CreateArrowNoneImageSource Method

Sets up a source for the icon that is shown in the header of the unsorted column.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
ImageSource CreateArrowNoneImageSource()
```

### Return Value

The Xamarin.Forms.ImageSource object, that specifies the image source.

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

The **CreateArrowNoneImageSource** method is abstract. So, to change the image source shown in your grid, override the method in the IHeaderCustomizer descendants.

### See Also

IHeaderCustomizer Interface
IHeaderCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
CreateArrowDownImageSource
CreateArrowUpImageSource

© 2018 Developer Express Inc.

# CreateArrowUpImageSource Method

Sets up a source for the icon that is shown in the header of the column sorted in ascending order.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
ImageSource CreateArrowUpImageSource()
```

### Return Value

The Xamarin.Forms.ImageSource object, that specifies the image source.

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

The **CreateArrowUpImageSource** method is abstract. So, to change the image source shown in your grid, override the method in the IHeaderCustomizer descendants.

### See Also

IHeaderCustomizer Interface
IHeaderCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
CreateArrowDownImageSource
CreateArrowNoneImageSource

© 2018 Developer Express Inc.

# LocateHeaderViewInPanel Method

Locates the specified view within the grid header's bounds.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
void LocateHeaderViewInPanel(
   View header,
   Rectangle headerBounds,
   bool isFirst,
   bool isLast,
   bool isLeftFixedColumnBorder,
   bool isRightFixedColumnBorder
)
```

### Parameters

*header*
A Xamarin.Forms.View object, specifying the view to be located within the grid header.
*headerBounds*
A Xamarin.Forms.Rectangle object, specifying the grid header's bounds.
*isFirst*
**true**, if the specified view is the first view in the grid header; otherwise, **false**.
*isLast*
**true**, if the specified view is the last view in the grid header; otherwise, **false**.
*isLeftFixedColumnBorder*
**true**, if the specified view belongs to the last column fixed to the left; otherwise, **false**.
*isRightFixedColumnBorder*
**true**, if the specified view belongs to the first column fixed to the right; otherwise, **false**.

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** | |
|---|---|
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. | |

### See Also

IHeaderCustomizer Interface
IHeaderCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# IIOSRefreshCustomizer Interface

Defines classes that customize an iOS refreshing icon.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public interface IIOSRefreshCustomizer
```

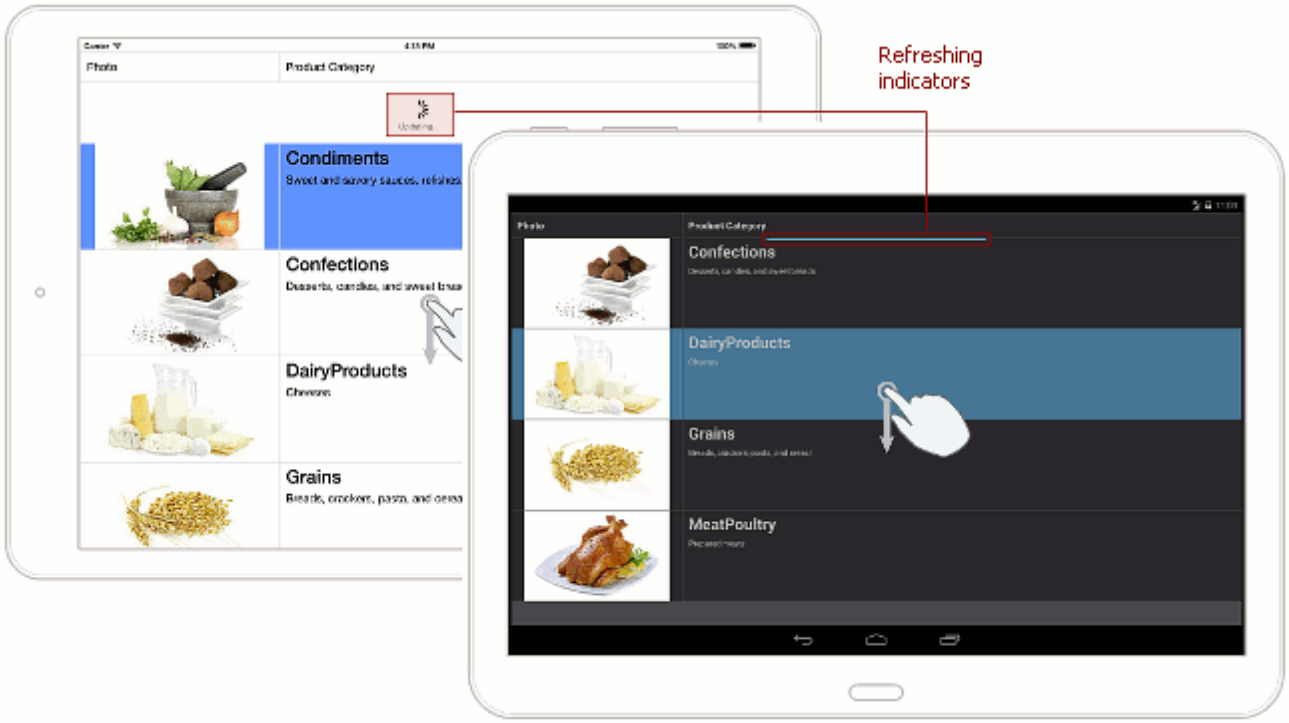
### Remarks

To customize the color of the iOS refreshing icon, use the RefreshColor property of an object that implements the **IIOSRefreshCustomizer** interface.

### Example

| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T326911. | |

This example demonstrates how to customize a color of an indicator (icon or bar) shown when GridControl refreshes cell data.




To change the color of a refresh indicator, perform the steps below.

1. Activate a grid's **pull-to-refresh** functionality. For this purpose, assign **true** to the GridControl.IsPullToRefreshEnabled property.
2. Use the ThemeBase class's properties to access customizers used to change an indicator's color on different mobile platforms. In this example, the following customizers are used.
   - ThemeBase.AndroidRefreshCustomizer
   - ThemeBase.IOSRefreshCustomizer
3. Apply the customization by calling the ThemeManager.RefreshTheme method.

| C# | |
|---|---|

© 2018 Developer Express Inc.

```
 (MainPage.xaml.cs)
using System;
using System.Collections.Generic;
using Xamarin.Forms;
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.DataGrid.Theme;
namespace RefreshCustomizers {
    public partial class MainPage : ContentPage {
        public MainPage() {
            InitializeComponent();
            TestOrdersRepository model = new TestOrdersRepository();
            BindingContext = model;
            ThemeManager.Theme.AndroidRefreshCustomizer.BarColor = Color.Blue;
            ThemeManager.Theme.IOSRefreshCustomizer.RefreshColor = Color.Blue;
            ThemeManager.RefreshTheme();
        }
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<?xml version="1.0" encoding="UTF-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="RefreshCustomizers.MainPage"
             xmlns:dxGrid="clr-namespace:DevExpress.Mobile.DataGrid;assembly=DevExpress.Mobile.Grid.v15.2"
      <ContentPage.Padding>
        <OnPlatform x:TypeArguments="Thickness" iOS="0, 20, 0, 0" />
    </ContentPage.Padding>
    <dxGrid:GridControl x:Name="grid" ItemsSource="{Binding Orders}"
                         NewItemRowVisibility = "false"
                        SortMode = "Multiple" AutoFilterPanelVisibility="true"
                        IsPullToRefreshEnabled = "true" >
         <dxGrid:GridControl.Columns>
            <dxGrid:TextColumn FieldName="Product.Name" Caption = "Product" Width = "170"
                               SortOrder = "Descending" SortIndex = "0"/>
            <dxGrid:NumberColumn FieldName="Product.UnitPrice" Caption = "Price" DisplayFormat="C0"/>
            <dxGrid:NumberColumn FieldName="Quantity"
                                 SortOrder = "Ascending" SortIndex = "1"/>
            <dxGrid:NumberColumn FieldName="Total"
                                 UnboundType="Integer" UnboundExpression="[Quantity] * [Product.UnitPrice
                                 DisplayFormat="C0" IsReadOnly="True"/>
            <dxGrid:DateColumn FieldName="Date" DisplayFormat="d" />
            <dxGrid:SwitchColumn FieldName="Shipped"/>
         </dxGrid:GridControl.Columns>
         <dxGrid:GridControl.GroupSummaries>
            <dxGrid:GridColumnSummary FieldName="Total" Type="Max"/>
        </dxGrid:GridControl.GroupSummaries>
        <dxGrid:GridControl.TotalSummaries>
            <dxGrid:GridColumnSummary FieldName="Total" Type="Sum"
                                      DisplayFormat= "Total: {0:C0}"/>
        </dxGrid:GridControl.TotalSummaries>
    </dxGrid:GridControl>
</ContentPage>
```

## Inheritance Hierarchy

## See Also

IIOSRefreshCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
IAndroidRefreshCustomizer

© 2018 Developer Express Inc.

# IIOSRefreshCustomizer Members

Defines classes that customize an iOS refreshing icon.

The following tables list the members exposed by the IIOSRefreshCustomizer type.

**Public Properties**

| | Name | Description |
|---|---|---|
| | RefreshColor | Specifies the color of the iOS refreshing icon. |

**See Also**
IIOSRefreshCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
IAndroidRefreshCustomizer

© 2018 Developer Express Inc.

# IIOSRefreshCustomizer Properties

Defines classes that customize an iOS refreshing icon.

The following tables list the members exposed by the IIOSRefreshCustomizer type.

**Public Properties**

| | Name | Description |
|---|---|---|
| | RefreshColor | Specifies the color of the iOS refreshing icon. |

**See Also**
IIOSRefreshCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
IAndroidRefreshCustomizer

© 2018 Developer Express Inc.

# RefreshColor Property

Specifies the color of the iOS refreshing icon.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
Color RefreshColor { get; set; }
```

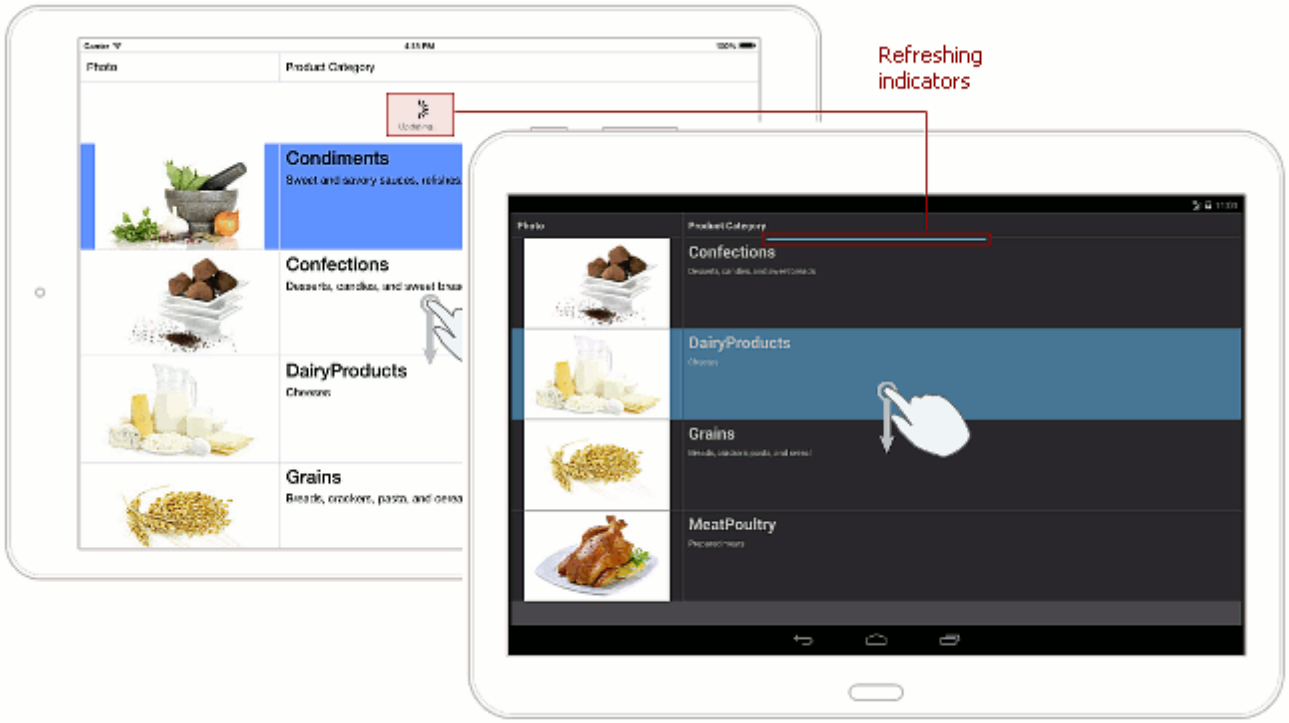
### Property Value

A Xamarin.Forms.Color object specifying the color of the iOS refreshing icon.

### Example

| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T326911. | |

This example demonstrates how to customize a color of an indicator (icon or bar) shown when GridControl refreshes cell data.




To change the color of a refresh indicator, perform the steps below.

1. Activate a grid's **pull-to-refresh** functionality. For this purpose, assign **true** to the GridControl.IsPullToRefreshEnabled property.
2. Use the ThemeBase class's properties to access customizers used to change an indicator's color on different mobile platforms. In this example, the following customizers are used.
   - ThemeBase.AndroidRefreshCustomizer
   - ThemeBase.IOSRefreshCustomizer
3. Apply the customization by calling the ThemeManager.RefreshTheme method.

| C# | |
|---|---|

© 2018 Developer Express Inc.

(MainPage.xaml.cs)

```
using System;
using System.Collections.Generic;
using Xamarin.Forms;
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.DataGrid.Theme;
namespace RefreshCustomizers {
    public partial class MainPage : ContentPage {
        public MainPage() {
            InitializeComponent();
            TestOrdersRepository model = new TestOrdersRepository();
            BindingContext = model;
            ThemeManager.Theme.AndroidRefreshCustomizer.BarColor = Color.Blue;
            ThemeManager.Theme.IOSRefreshCustomizer.RefreshColor = Color.Blue;
            ThemeManager.RefreshTheme();
        }
    }
}
```

**Xaml**

(MainPage.xaml)

```
<?xml version="1.0" encoding="UTF-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="RefreshCustomizers.MainPage"
             xmlns:dxGrid="clr-namespace:DevExpress.Mobile.DataGrid;assembly=DevExpress.Mobile.Grid.v15.2"
      <ContentPage.Padding>
        <OnPlatform x:TypeArguments="Thickness" iOS="0, 20, 0, 0" />
    </ContentPage.Padding>
    <dxGrid:GridControl x:Name="grid" ItemsSource="{Binding Orders}"
                        NewItemRowVisibility = "false"
                       SortMode = "Multiple" AutoFilterPanelVisibility="true"
                       IsPullToRefreshEnabled = "true" >
         <dxGrid:GridControl.Columns>
            <dxGrid:TextColumn FieldName="Product.Name" Caption = "Product" Width = "170"
                               SortOrder = "Descending" SortIndex = "0"/>
            <dxGrid:NumberColumn FieldName="Product.UnitPrice" Caption = "Price" DisplayFormat="C0"/>
            <dxGrid:NumberColumn FieldName="Quantity"
                                 SortOrder = "Ascending" SortIndex = "1"/>
            <dxGrid:NumberColumn FieldName="Total"
                                 UnboundType="Integer" UnboundExpression="[Quantity] * [Product.UnitPrice
                                 DisplayFormat="C0" IsReadOnly="True"/>
            <dxGrid:DateColumn FieldName="Date" DisplayFormat="d" />
            <dxGrid:SwitchColumn FieldName="Shipped"/>
         </dxGrid:GridControl.Columns>
         <dxGrid:GridControl.GroupSummaries>
            <dxGrid:GridColumnSummary FieldName="Total" Type="Max"/>
        </dxGrid:GridControl.GroupSummaries>
        <dxGrid:GridControl.TotalSummaries>
            <dxGrid:GridColumnSummary FieldName="Total" Type="Sum"
                                      DisplayFormat= "Total: {0:C0}"/>
        </dxGrid:GridControl.TotalSummaries>
    </dxGrid:GridControl>
</ContentPage>
```

## See Also

IIOSRefreshCustomizer Interface
IIOSRefreshCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# ILoadMoreViewCustomizer Interface

Defines classes that customize the Load More panel.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public interface ILoadMoreViewCustomizer
```

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

### Inheritance Hierarchy

### See Also

ILoadMoreViewCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# ILoadMoreViewCustomizer Members

Defines classes that customize the Load More panel.

The following tables list the members exposed by the ILoadMoreViewCustomizer type.

**Public Properties**

| | Name | Description |
|---|---|---|
| | ActivityViewColor | Specifies the color of an activity indicator shown in the Load More panel. |
| | BackgroundColor | Specifies the Load More panel's background color. |
| | BorderColor | Specifies the Load More panel's border color. |
| | Font | Specifies the font used to display text in the Load More panel. |

**Public Methods**

| | Name | Description |
|---|---|---|
|  | LocateViewInRow | Locates the specified view within the Load More panel's bounds. |

**See Also**
ILoadMoreViewCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# ILoadMoreViewCustomizer Properties

Defines classes that customize the Load More panel.

The following tables list the members exposed by the ILoadMoreViewCustomizer type.

**Public Properties**

| | Name | Description |
|---|---|---|
| | ActivityViewColor | Specifies the color of an activity indicator shown in the Load More panel. |
| | BackgroundColor | Specifies the Load More panel's background color. |
| | BorderColor | Specifies the Load More panel's border color. |
| | Font | Specifies the font used to display text in the Load More panel. |

**Public Methods**

| | Name | Description |
|---|---|---|
|  | LocateViewInRow | Locates the specified view within the Load More panel's bounds. |

**See Also**
ILoadMoreViewCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# ActivityViewColor Property

Specifies the color of an activity indicator shown in the Load More panel.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
Color ActivityViewColor { get; set; }
```

### Property Value

A Xamarin.Forms.Color object specifying the activity indicator's color.

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

### See Also

ILoadMoreViewCustomizer Interface
ILoadMoreViewCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# BackgroundColor Property

Specifies the Load More panel's background color.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
Color BackgroundColor { get; set; }
```

### Property Value

A Xamarin.Forms.Color object specifying the background color.

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

### See Also

ILoadMoreViewCustomizer Interface
ILoadMoreViewCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# BorderColor Property

Specifies the Load More panel's border color.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
Color BorderColor { get; set; }
```

### Property Value

A Xamarin.Forms.Color object specifying the border color.

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

### See Also

ILoadMoreViewCustomizer Interface
ILoadMoreViewCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# Font Property

Specifies the font used to display text in the Load More panel.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
ThemeFontAttributes Font { get; set; }
```

### Property Value

A **ThemeFontAttributes** object specifying the font.

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

### See Also

ILoadMoreViewCustomizer Interface
ILoadMoreViewCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# ILoadMoreViewCustomizer Methods

Defines classes that customize the Load More panel.

The following tables list the members exposed by the ILoadMoreViewCustomizer type.

**Public Methods**

| | Name | Description |
|---|---|---|
| | LocateViewInRow | Locates the specified view within the Load More panel's bounds. |

**See Also**
ILoadMoreViewCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# LocateViewInRow Method

Locates the specified view within the Load More panel's bounds.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
void LocateViewInRow(
   View view,
   Rectangle cellBounds
)
```

### Parameters

*view*
A Xamarin.Forms.View object, specifying the view to be located within the **Load More** panel.
*cellBounds*
A Xamarin.Forms.Rectangle object, specifying the **Load More** panel's bounds.

### See Also

ILoadMoreViewCustomizer Interface
ILoadMoreViewCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# INewItemRowCustomizer Interface

Defines classes that customize the New Item Row.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public interface INewItemRowCustomizer
```

### Example

| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T297553. | |

This example demonstrates how to customize a theme used to paint a GridControl.

To apply a theme to a **GridControl**, assign its name to the ThemeManager.ThemeName property. The Themes class stores available theme names.

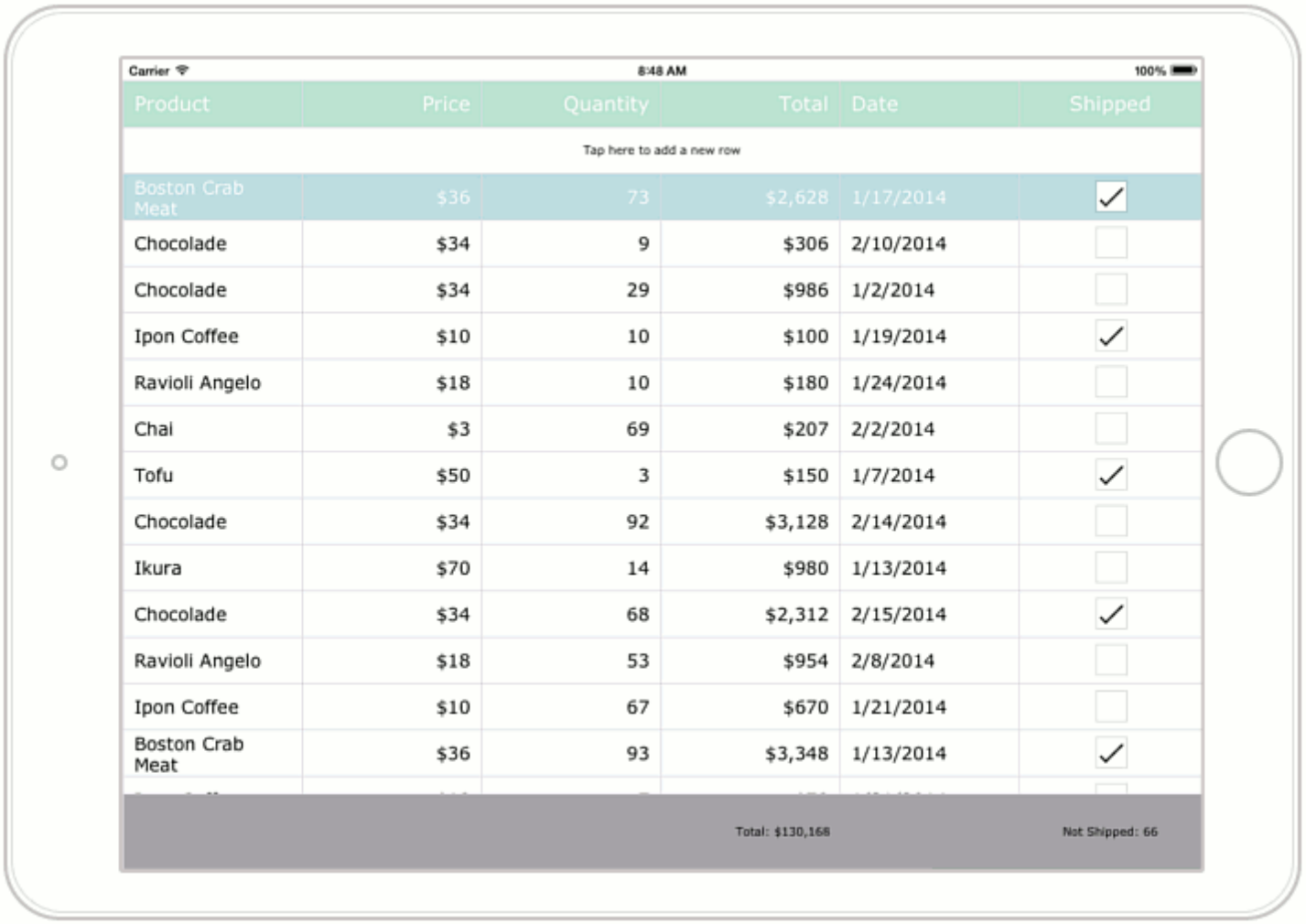
The ThemeBase class's properties provide access to different customizers, which you can adjust to change appearance settings (for example, font attributes, border and background colors, etc.) of the grid's corresponding visual elements (such as data cells, group rows, filter panel, etc.). In this example, the following customizers are used.

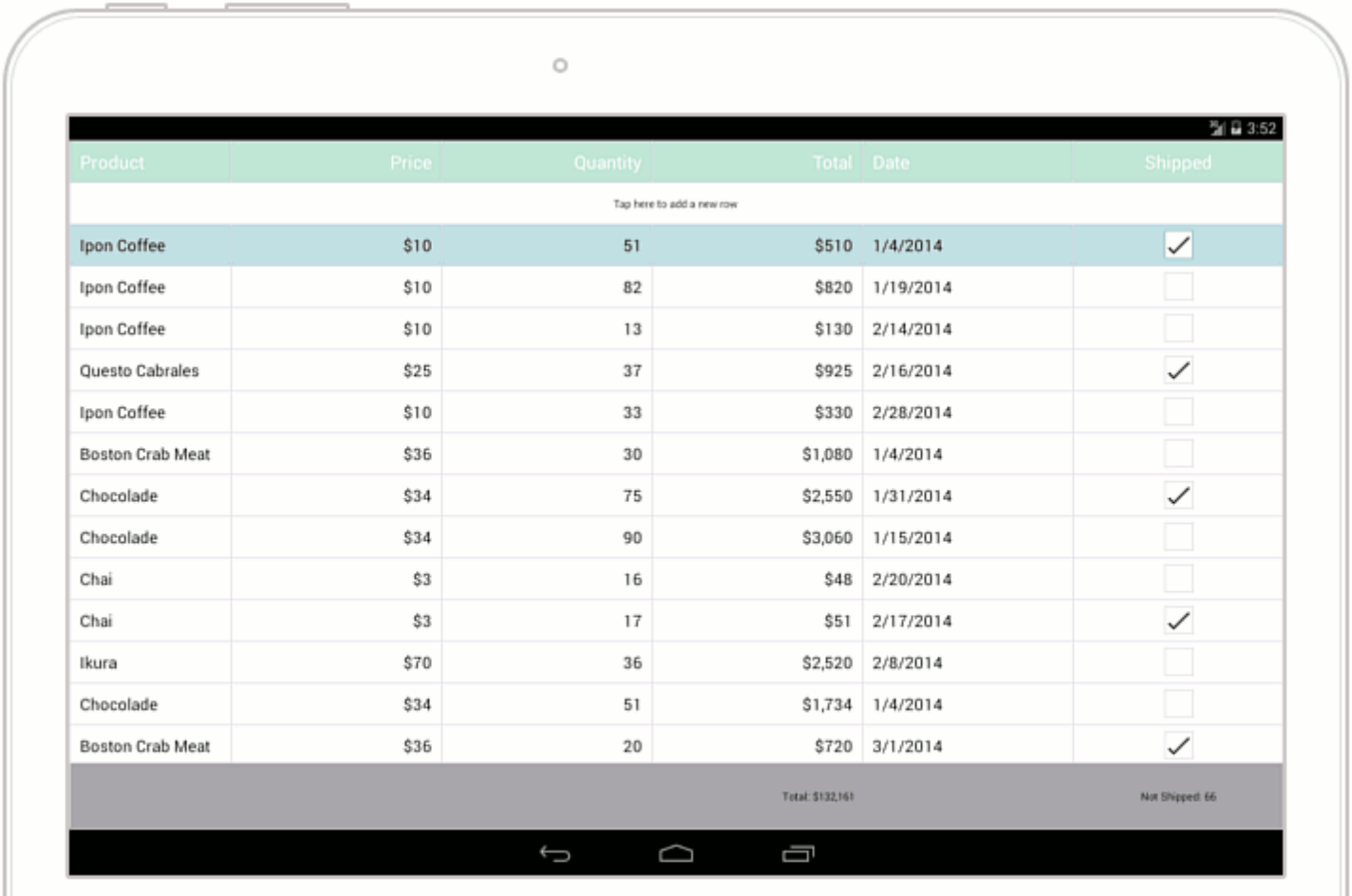
- ThemeBase.HeaderCustomizer
- ThemeBase.CellCustomizer
- ThemeBase.TotalSummaryCustomizer
- ThemeBase.NewItemRowCustomizer

| Important | |
|---|---|
| To apply the customization, call the ThemeManager.RefreshTheme method. | |

This example customizes the Light theme as it is shown in the image below.

© 2018 Developer Express Inc.


Carrier
8:48 AM
100%
Product
Price
Quantity
Total
Date
Shipped
Tap here to add a new row
Boston Crab Meat $36 73 $2,628 1/17/2014
Chocolade $34 9 $306 2/10/2014
Chocolade $34 29 $986 1/2/2014
Ipon Coffee $10 10 $100 1/19/2014
Ravioli Angelo $18 10 $180 1/24/2014
Chai $3 69 $207 2/2/2014
Tofu $50 3 $150 1/7/2014
Chocolade $34 92 $3,128 2/14/2014
Ikura $70 14 $980 1/13/2014
Chocolade $34 68 $2,312 2/15/2014
Ravioli Angelo $18 53 $954 2/8/2014
Ipon Coffee $10 67 $670 1/21/2014
Boston Crab Meat $36 93 $3,348 1/13/2014
Total: $130,168
Not Shipped: 66


3:52
Product
Price
Quantity
Total
Date
Shipped
Tap here to add a new row
Ipon Coffee $10 51 $510 1/4/2014
Ipon Coffee $10 82 $820 1/19/2014
Ipon Coffee $10 13 $130 2/14/2014
Questo Cabrales $25 37 $925 2/16/2014
Ipon Coffee $10 33 $330 2/28/2014
Boston Crab Meat $36 30 $1,080 1/4/2014
Chocolade $34 75 $2,550 1/31/2014
Chocolade $34 90 $3,060 1/15/2014
Chai $3 16 $48 2/20/2014
Chai $3 17 $51 2/17/2014
Ikura $70 36 $2,520 2/8/2014
Chocolade $34 51 $1,734 1/4/2014
Boston Crab Meat $36 20 $720 3/1/2014
Total: $132,161
Not Shipped: 66

**C#**

```
 (MainPage.xaml.cs)
using System;
using System.Collections.Generic;
using Xamarin.Forms;
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.DataGrid.Theme;
namespace ThemeCustomizing {
    public partial class MainPage : ContentPage {
        int count = 0;
        public MainPage() {
            InitializeComponent();
            TestOrdersRepository model = new TestOrdersRepository();
            BindingContext = model;
            ThemeManager.ThemeName = Themes.Light;
            // Header customization.
            ThemeManager.Theme.HeaderCustomizer.BackgroundColor = Color.FromRgb(187, 228, 208);
            ThemeFontAttributes myFont = new ThemeFontAttributes("Verdana",
                                         ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Large),
                                         FontAttributes.None, Color.White);
            ThemeManager.Theme.HeaderCustomizer.Font = myFont;
            // Cell customization.
            ThemeManager.Theme.CellCustomizer.SelectionColor = Color.FromRgb(186, 220, 225);
            ThemeFontAttributes myFont1 = new ThemeFontAttributes("Verdana",
                                         ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Medium),
                                         FontAttributes.None, Color.Black);
            ThemeManager.Theme.CellCustomizer.Font = myFont1;
            // Various customization.
            ThemeManager.Theme.TotalSumaryCustomizer.BackgroundColor = Color.FromRgb(163, 162, 168);
            ThemeFontAttributes myFont2 = new ThemeFontAttributes("Verdana",
                                 ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Default),
                                 FontAttributes.None, Color.Black);
            ThemeManager.Theme.TotalSummaryCustomizer.Font = myFont2;
            ThemeManager.Theme.NewItemRowCustomizer.Font = myFont2;
            ThemeManager.RefreshTheme();
        }
        void OnCalculateCustomSummary(object sender, CustomSummaryEventArgs e) {
            if (e.FieldName.ToString() == "Shipped")
            if (e.IsTotalSummary) {
                if (e.SummaryProcess == CustomSummaryProcess.Calculate) {
                    if (!(bool)e.FieldValue)
                        count++;
                    e.TotalValue = count;
                }
            }
        }
    }
}
```

**Inheritance Hierarchy**

**See Also**
INewItemRowCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# INewItemRowCustomizer Members

Defines classes that customize the New Item Row.

The following tables list the members exposed by the INewItemRowCustomizer type.

**Public Properties**

| | Name | Description |
|---|---|---|
| | BackgroundColor | Specifies the New Item Row's background color. |
| | BorderColor | Specifies the New Item Row's border color. |
| | Font | Specifies the font used to display text in the New Item Row. |

**Public Methods**

| | Name | Description |
|---|---|---|
| | LocateCellViewInRow | Locates the specified view within the New Item Row's bounds. |

**See Also**
INewItemRowCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# INewItemRowCustomizer Properties

Defines classes that customize the New Item Row.

The following tables list the members exposed by the INewItemRowCustomizer type.

**Public Properties**

| | Name | Description |
|---|---|---|
| | BackgroundColor | Specifies the New Item Row's background color. |
| | BorderColor | Specifies the New Item Row's border color. |
| | Font | Specifies the font used to display text in the New Item Row. |

**Public Methods**

| | Name | Description |
|---|---|---|
| | LocateCellViewInRow | Locates the specified view within the New Item Row's bounds. |

**See Also**
INewItemRowCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# BackgroundColor Property

Specifies the New Item Row's background color.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
Color BackgroundColor { get; set; }
```

### Property Value

A Xamarin.Forms.Color object specifying the background color.

### Example

| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T297553. | |

This example demonstrates how to customize a theme used to paint a GridControl.

To apply a theme to a **GridControl**, assign its name to the ThemeManager.ThemeName property. The Themes class stores available theme names.

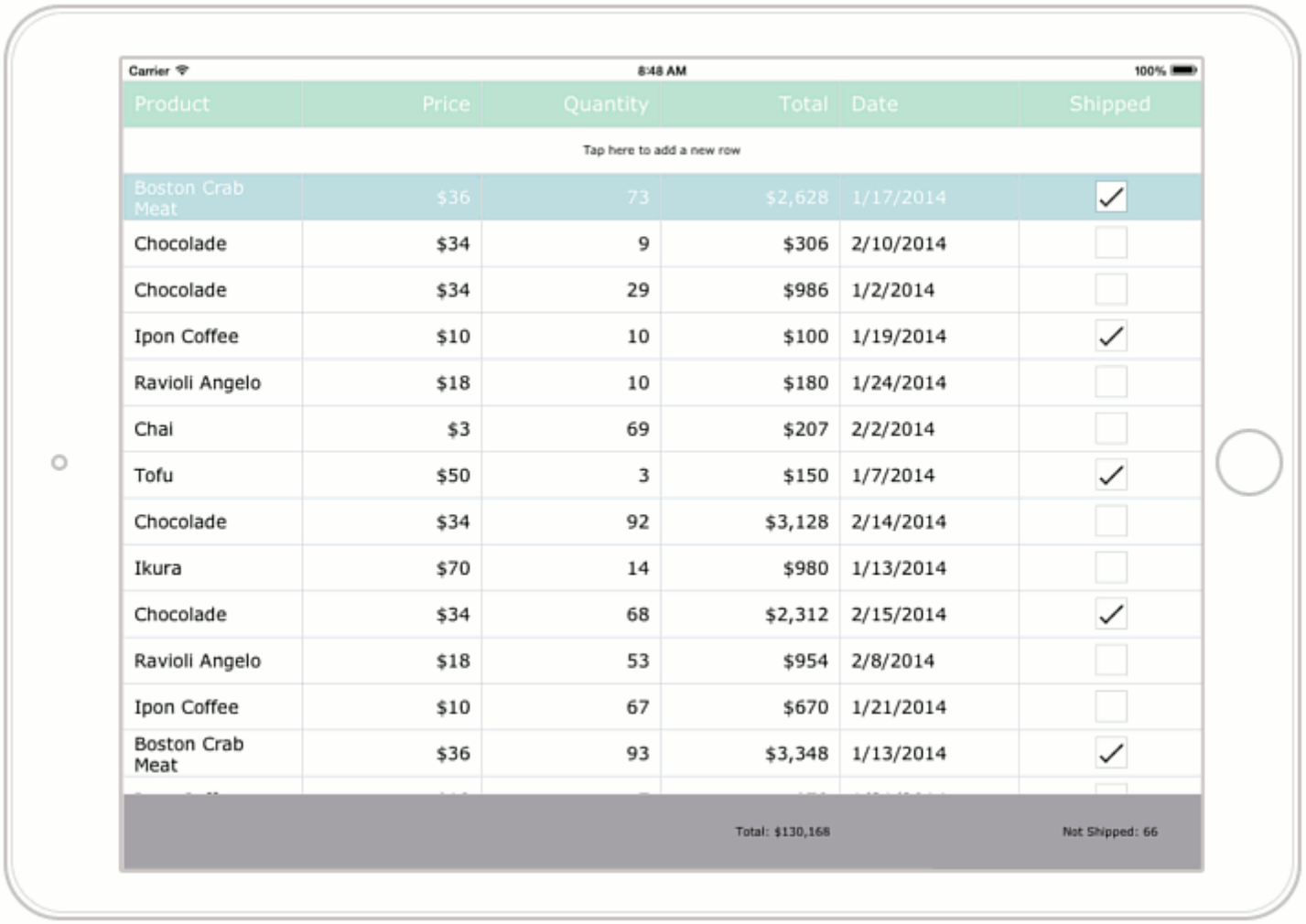
The ThemeBase class's properties provide access to different customizers, which you can adjust to change appearance settings (for example, font attributes, border and background colors, etc.) of the grid's corresponding visual elements (such as data cells, group rows, filter panel, etc.). In this example, the following customizers are used.

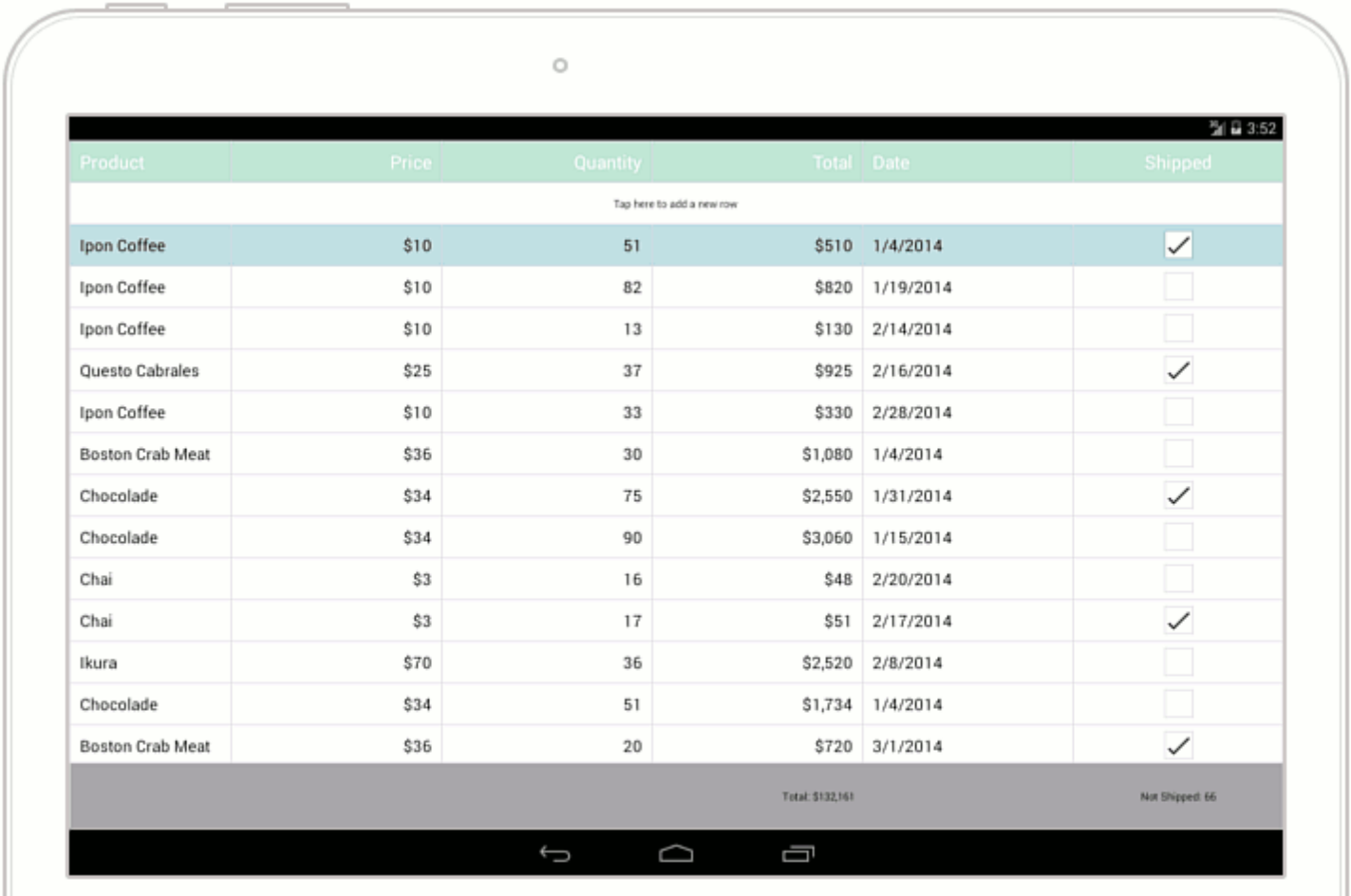
- ThemeBase.HeaderCustomizer
- ThemeBase.CellCustomizer
- ThemeBase.TotalSummaryCustomizer
- ThemeBase.NewItemRowCustomizer

| Important | |
|---|---|
| To apply the customization, call the ThemeManager.RefreshTheme method. | |

This example customizes the Light theme as it is shown in the image below.

© 2018 Developer Express Inc.


Carrier
8:48 AM
100%
Product
Price
Quantity
Total
Date
Shipped
Tap here to add a new row
Boston Crab Meat $36 73 $2,628 1/17/2014
Chocolade $34 9 $306 2/10/2014
Chocolade $34 29 $986 1/2/2014
Ipon Coffee $10 10 $100 1/19/2014
Ravioli Angelo $18 10 $180 1/24/2014
Chai $3 69 $207 2/2/2014
Tofu $50 3 $150 1/7/2014
Chocolade $34 92 $3,128 2/14/2014
Ikura $70 14 $980 1/13/2014
Chocolade $34 68 $2,312 2/15/2014
Ravioli Angelo $18 53 $954 2/8/2014
Ipon Coffee $10 67 $670 1/21/2014
Boston Crab Meat $36 93 $3,348 1/13/2014
Total: $130,168
Not Shipped: 66


3:52
Product
Price
Quantity
Total
Date
Shipped
Tap here to add a new row
Ipon Coffee $10 51 $510 1/4/2014
Ipon Coffee $10 82 $820 1/19/2014
Ipon Coffee $10 13 $130 2/14/2014
Questo Cabrales $25 37 $925 2/16/2014
Ipon Coffee $10 33 $330 2/28/2014
Boston Crab Meat $36 30 $1,080 1/4/2014
Chocolade $34 75 $2,550 1/31/2014
Chocolade $34 90 $3,060 1/15/2014
Chai $3 16 $48 2/20/2014
Chai $3 17 $51 2/17/2014
Ikura $70 36 $2,520 2/8/2014
Chocolade $34 51 $1,734 1/4/2014
Boston Crab Meat $36 20 $720 3/1/2014
Total: $132,161
Not Shipped: 66

C#

```
 (MainPage.xaml.cs)
using System;
using System.Collections.Generic;
using Xamarin.Forms;
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.DataGrid.Theme;
namespace ThemeCustomizing {
    public partial class MainPage : ContentPage {
        int count = 0;
        public MainPage() {
            InitializeComponent();
            TestOrdersRepository model = new TestOrdersRepository();
            BindingContext = model;
            ThemeManager.ThemeName = Themes.Light;
            // Header customization.
            ThemeManager.Theme.HeaderCustomizer.BackgroundColor = Color.FromRgb(187, 228, 208);
            ThemeFontAttributes myFont = new ThemeFontAttributes("Verdana",
                                         ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Large),
                                         FontAttributes.None, Color.White);
            ThemeManager.Theme.HeaderCustomizer.Font = myFont;
            // Cell customization.
            ThemeManager.Theme.CellCustomizer.SelectionColor = Color.FromRgb(186, 220, 225);
            ThemeFontAttributes myFont1 = new ThemeFontAttributes("Verdana",
                                         ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Medium),
                                         FontAttributes.None, Color.Black);
            ThemeManager.Theme.CellCustomizer.Font = myFont1;
            // Various customization.
            ThemeManager.Theme.TotalSumaryCustomizer.BackgroundColor = Color.FromRgb(163, 162, 168);
            ThemeFontAttributes myFont2 = new ThemeFontAttributes("Verdana",
                                 ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Default),
                                 FontAttributes.None, Color.Black);
            ThemeManager.Theme.TotalSummaryCustomizer.Font = myFont2;
            ThemeManager.Theme.NewItemRowCustomizer.Font = myFont2;
            ThemeManager.RefreshTheme();
        }
        void OnCalculateCustomSummary(object sender, CustomSummaryEventArgs e) {
            if (e.FieldName.ToString() == "Shipped")
            if (e.IsTotalSummary) {
                if (e.SummaryProcess == CustomSummaryProcess.Calculate) {
                    if (!(bool)e.FieldValue)
                        count++;
                    e.TotalValue = count;
                }
            }
        }
    }
}
```

**See Also**

INewItemRowCustomizer Interface
INewItemRowCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# BorderColor Property

Specifies the New Item Row's border color.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
Color BorderColor { get; set; }
```

### Property Value

A Xamarin.Forms.Color object specifying the border color.

### Example

| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T297553. | |

This example demonstrates how to customize a theme used to paint a GridControl.

To apply a theme to a **GridControl**, assign its name to the ThemeManager.ThemeName property. The Themes class stores available theme names.

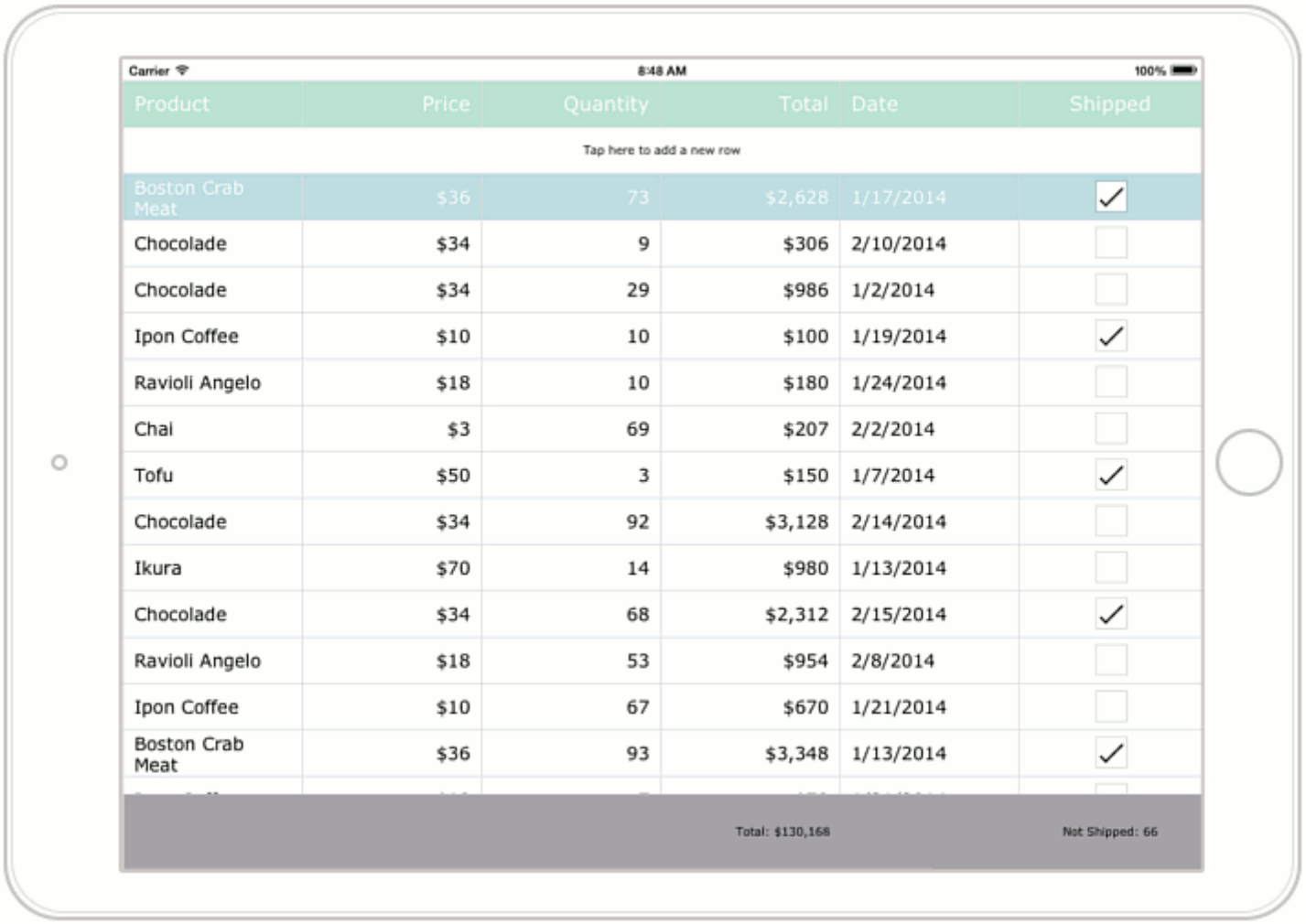
The ThemeBase class's properties provide access to different customizers, which you can adjust to change appearance settings (for example, font attributes, border and background colors, etc.) of the grid's corresponding visual elements (such as data cells, group rows, filter panel, etc.). In this example, the following customizers are used.

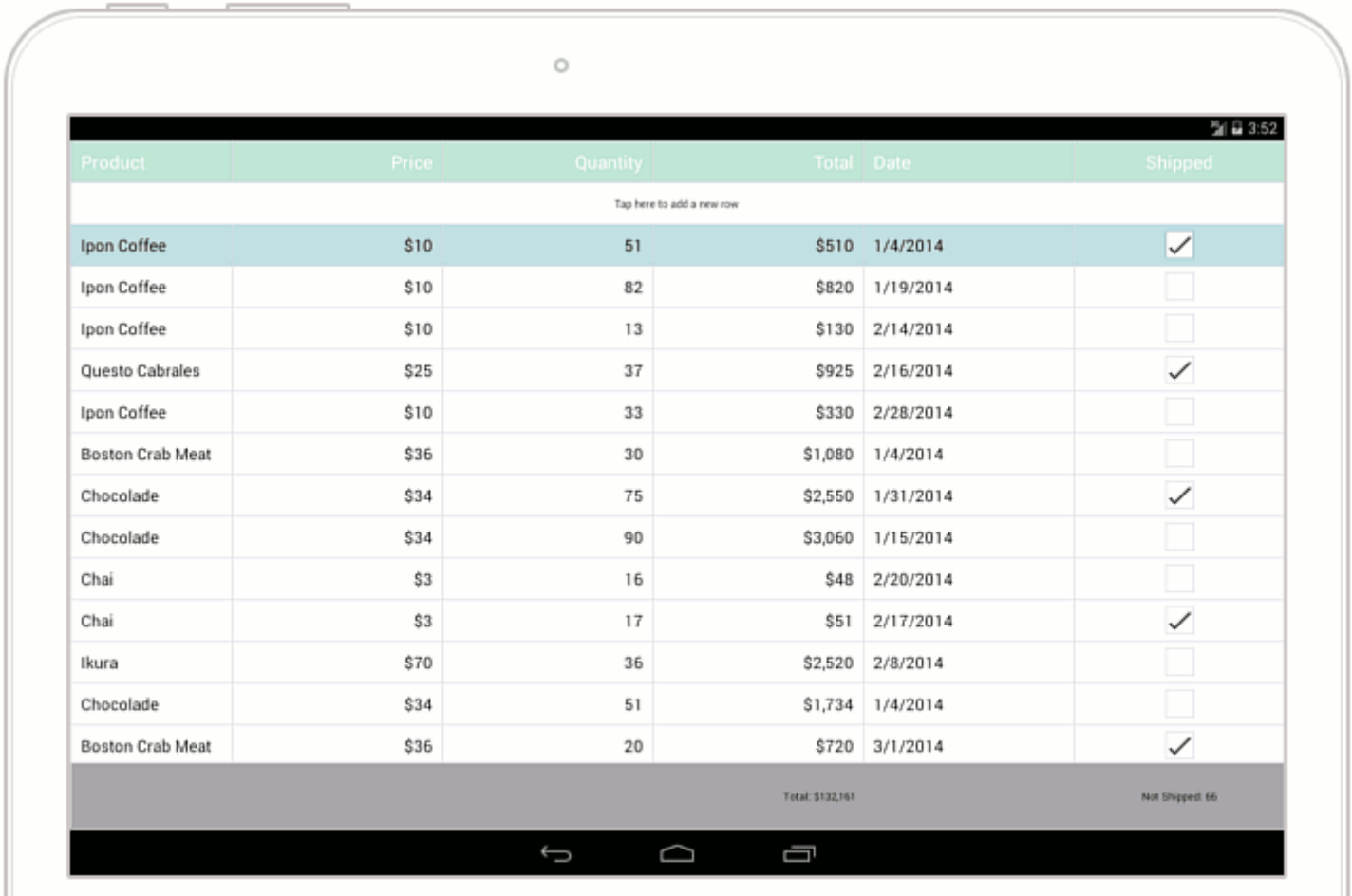
- ThemeBase.HeaderCustomizer
- ThemeBase.CellCustomizer
- ThemeBase.TotalSummaryCustomizer
- ThemeBase.NewItemRowCustomizer

| Important | |
|---|---|
| To apply the customization, call the ThemeManager.RefreshTheme method. | |

This example customizes the Light theme as it is shown in the image below.

© 2018 Developer Express Inc.


Carrier
8:48 AM
100%
Product
Price
Quantity
Total
Date
Shipped
Tap here to add a new row
Boston Crab Meat $36 73 $2,628 1/17/2014
Chocolade $34 9 $306 2/10/2014
Chocolade $34 29 $986 1/2/2014
Ipon Coffee $10 10 $100 1/19/2014
Ravioli Angelo $18 10 $180 1/24/2014
Chai $3 69 $207 2/2/2014
Tofu $50 3 $150 1/7/2014
Chocolade $34 92 $3,128 2/14/2014
Ikura $70 14 $980 1/13/2014
Chocolade $34 68 $2,312 2/15/2014
Ravioli Angelo $18 53 $954 2/8/2014
Ipon Coffee $10 67 $670 1/21/2014
Boston Crab Meat $36 93 $3,348 1/13/2014
Total: $130,168
Not Shipped: 66


3:52
Product
Price
Quantity
Total
Date
Shipped
Tap here to add a new row
Ipon Coffee $10 51 $510 1/4/2014
Ipon Coffee $10 82 $820 1/19/2014
Ipon Coffee $10 13 $130 2/14/2014
Questo Cabrales $25 37 $925 2/16/2014
Ipon Coffee $10 33 $330 2/28/2014
Boston Crab Meat $36 30 $1,080 1/4/2014
Chocolade $34 75 $2,550 1/31/2014
Chocolade $34 90 $3,060 1/15/2014
Chai $3 16 $48 2/20/2014
Chai $3 17 $51 2/17/2014
Ikura $70 36 $2,520 2/8/2014
Chocolade $34 51 $1,734 1/4/2014
Boston Crab Meat $36 20 $720 3/1/2014
Total: $132,161
Not Shipped: 66

C#

```
 (MainPage.xaml.cs)
using System;
using System.Collections.Generic;
using Xamarin.Forms;
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.DataGrid.Theme;
namespace ThemeCustomizing {
    public partial class MainPage : ContentPage {
        int count = 0;
        public MainPage() {
            InitializeComponent();
            TestOrdersRepository model = new TestOrdersRepository();
            BindingContext = model;
            ThemeManager.ThemeName = Themes.Light;
            // Header customization.
            ThemeManager.Theme.HeaderCustomizer.BackgroundColor = Color.FromRgb(187, 228, 208);
            ThemeFontAttributes myFont = new ThemeFontAttributes("Verdana",
                                         ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Large),
                                         FontAttributes.None, Color.White);
            ThemeManager.Theme.HeaderCustomizer.Font = myFont;
            // Cell customization.
            ThemeManager.Theme.CellCustomizer.SelectionColor = Color.FromRgb(186, 220, 225);
            ThemeFontAttributes myFont1 = new ThemeFontAttributes("Verdana",
                                         ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Medium),
                                         FontAttributes.None, Color.Black);
            ThemeManager.Theme.CellCustomizer.Font = myFont1;
            // Various customization.
            ThemeManager.Theme.TotalSumaryCustomizer.BackgroundColor = Color.FromRgb(163, 162, 168);
            ThemeFontAttributes myFont2 = new ThemeFontAttributes("Verdana",
                                 ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Default),
                                 FontAttributes.None, Color.Black);
            ThemeManager.Theme.TotalSummaryCustomizer.Font = myFont2;
            ThemeManager.Theme.NewItemRowCustomizer.Font = myFont2;
            ThemeManager.RefreshTheme();
        }
        void OnCalculateCustomSummary(object sender, CustomSummaryEventArgs e) {
            if (e.FieldName.ToString() == "Shipped")
            if (e.IsTotalSummary) {
                if (e.SummaryProcess == CustomSummaryProcess.Calculate) {
                    if (!(bool)e.FieldValue)
                        count++;
                    e.TotalValue = count;
                }
            }
        }
    }
}
```

**See Also**

INewItemRowCustomizer Interface
INewItemRowCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# Font Property

Specifies the font used to display text in the New Item Row.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
ThemeFontAttributes Font { get; set; }
```

### Property Value

A **ThemeFontAttributes** object that specifies the font.

### Example

| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T297553. | |

This example demonstrates how to customize a theme used to paint a GridControl.

To apply a theme to a **GridControl**, assign its name to the ThemeManager.ThemeName property. The Themes class stores available theme names.

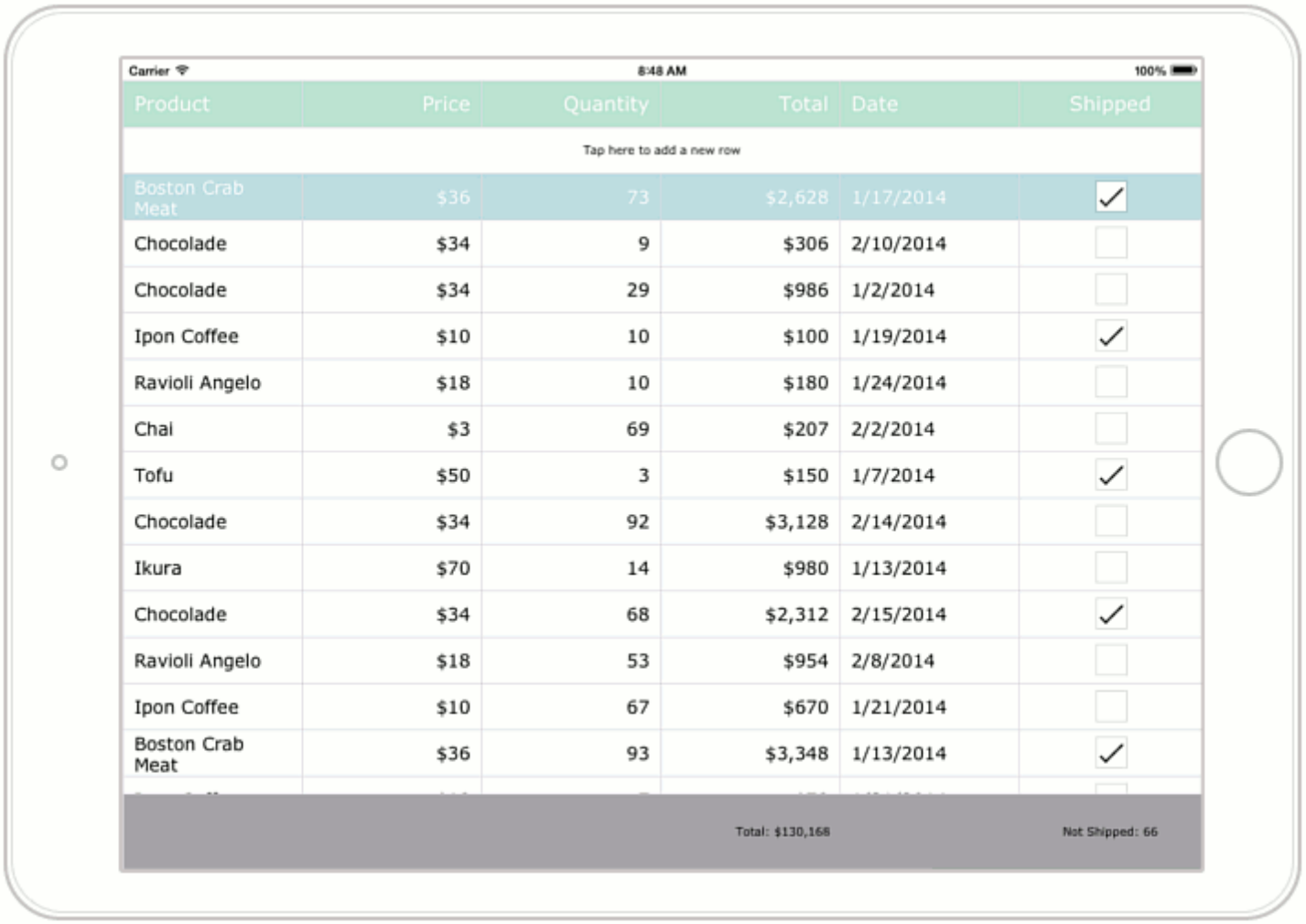
The ThemeBase class's properties provide access to different customizers, which you can adjust to change appearance settings (for example, font attributes, border and background colors, etc.) of the grid's corresponding visual elements (such as data cells, group rows, filter panel, etc.). In this example, the following customizers are used.

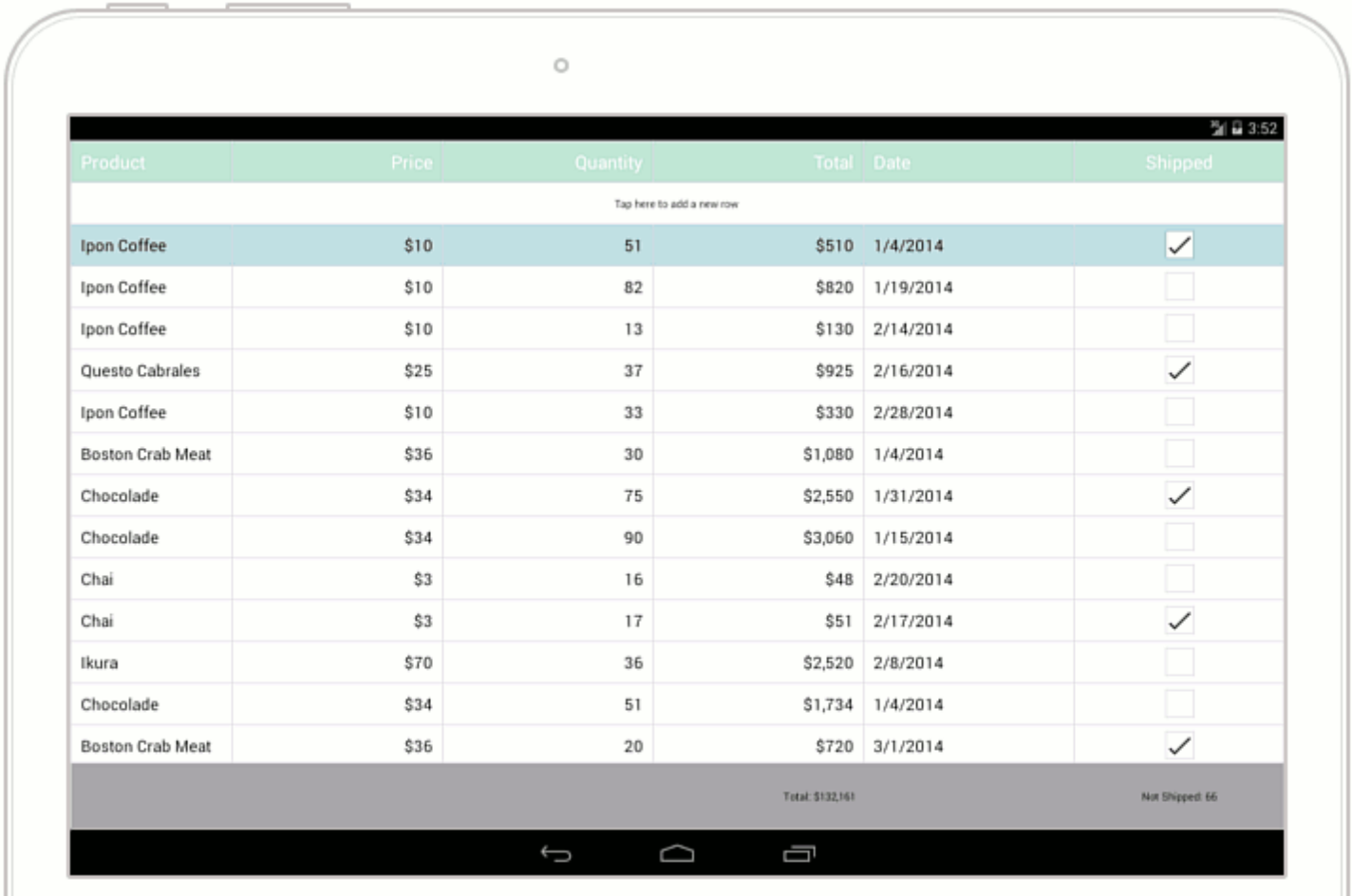
- ThemeBase.HeaderCustomizer
- ThemeBase.CellCustomizer
- ThemeBase.TotalSummaryCustomizer
- ThemeBase.NewItemRowCustomizer

| Important | |
|---|---|
| To apply the customization, call the ThemeManager.RefreshTheme method. | |

This example customizes the Light theme as it is shown in the image below.

© 2018 Developer Express Inc.


Carrier
8:48 AM
100%
Product
Price
Quantity
Total
Date
Shipped
Tap here to add a new row
Boston Crab Meat $36 73 $2,628 1/17/2014
Chocolade $34 9 $306 2/10/2014
Chocolade $34 29 $986 1/2/2014
Ipon Coffee $10 10 $100 1/19/2014
Ravioli Angelo $18 10 $180 1/24/2014
Chai $3 69 $207 2/2/2014
Tofu $50 3 $150 1/7/2014
Chocolade $34 92 $3,128 2/14/2014
Ikura $70 14 $980 1/13/2014
Chocolade $34 68 $2,312 2/15/2014
Ravioli Angelo $18 53 $954 2/8/2014
Ipon Coffee $10 67 $670 1/21/2014
Boston Crab Meat $36 93 $3,348 1/13/2014
Total: $130,168
Not Shipped: 66


3:52
Product
Price
Quantity
Total
Date
Shipped
Tap here to add a new row
Ipon Coffee $10 51 $510 1/4/2014
Ipon Coffee $10 82 $820 1/19/2014
Ipon Coffee $10 13 $130 2/14/2014
Questo Cabrales $25 37 $925 2/16/2014
Ipon Coffee $10 33 $330 2/28/2014
Boston Crab Meat $36 30 $1,080 1/4/2014
Chocolade $34 75 $2,550 1/31/2014
Chocolade $34 90 $3,060 1/15/2014
Chai $3 16 $48 2/20/2014
Chai $3 17 $51 2/17/2014
Ikura $70 36 $2,520 2/8/2014
Chocolade $34 51 $1,734 1/4/2014
Boston Crab Meat $36 20 $720 3/1/2014
Total: $132,161
Not Shipped: 66

C#

```
 (MainPage.xaml.cs)
using System;
using System.Collections.Generic;
using Xamarin.Forms;
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.DataGrid.Theme;
namespace ThemeCustomizing {
    public partial class MainPage : ContentPage {
        int count = 0;
        public MainPage() {
            InitializeComponent();
            TestOrdersRepository model = new TestOrdersRepository();
            BindingContext = model;
            ThemeManager.ThemeName = Themes.Light;
            // Header customization.
            ThemeManager.Theme.HeaderCustomizer.BackgroundColor = Color.FromRgb(187, 228, 208);
            ThemeFontAttributes myFont = new ThemeFontAttributes("Verdana",
                                         ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Large),
                                         FontAttributes.None, Color.White);
            ThemeManager.Theme.HeaderCustomizer.Font = myFont;
            // Cell customization.
            ThemeManager.Theme.CellCustomizer.SelectionColor = Color.FromRgb(186, 220, 225);
            ThemeFontAttributes myFont1 = new ThemeFontAttributes("Verdana",
                                         ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Medium),
                                         FontAttributes.None, Color.Black);
            ThemeManager.Theme.CellCustomizer.Font = myFont1;
            // Various customization.
            ThemeManager.Theme.TotalSumaryCustomizer.BackgroundColor = Color.FromRgb(163, 162, 168);
            ThemeFontAttributes myFont2 = new ThemeFontAttributes("Verdana",
                                 ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Default),
                                 FontAttributes.None, Color.Black);
            ThemeManager.Theme.TotalSummaryCustomizer.Font = myFont2;
            ThemeManager.Theme.NewItemRowCustomizer.Font = myFont2;
            ThemeManager.RefreshTheme();
        }
        void OnCalculateCustomSummary(object sender, CustomSummaryEventArgs e) {
            if (e.FieldName.ToString() == "Shipped")
            if (e.IsTotalSummary) {
                if (e.SummaryProcess == CustomSummaryProcess.Calculate) {
                    if (!(bool)e.FieldValue)
                        count++;
                    e.TotalValue = count;
                }
            }
        }
    }
}
```

**See Also**

INewItemRowCustomizer Interface
INewItemRowCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# INewItemRowCustomizer Methods

Defines classes that customize the New Item Row.

The following tables list the members exposed by the INewItemRowCustomizer type.

**Public Methods**

| | Name | Description |
|---|---|---|
| | LocateCellViewInRow | Locates the specified view within the New Item Row's bounds. |

**See Also**
INewItemRowCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# LocateCellViewInRow Method

Locates the specified view within the New Item Row's bounds.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
void LocateCellViewInRow(
   View cell,
   Rectangle cellBounds,
   bool isFirst,
   bool isLast,
   bool isLeftFixedColumnBorder,
   bool isRightFixedColumnBorder
)
```

### Parameters

*cell*
A Xamarin.Forms.View object, specifying the view to be located within the **Add New Item** row.
*cellBounds*
A Xamarin.Forms.Rectangle object, specifying the the **Add New Item** row's bounds.
*isFirst*
**true**, if the specified view is first view in the **Add New Item** row; otherwise, **false**.
*isLast*
**true**, if the specified view is last view in the **Add New Item** row; otherwise, **false**.
*isLeftFixedColumnBorder*
**true**, if the specified view belongs to the last column fixed to the left; otherwise, **false**.
*isRightFixedColumnBorder*
**true**, if the specified view belongs to the first column fixed to the right; otherwise, **false**.

### See Also

INewItemRowCustomizer Interface
INewItemRowCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# ISaveCancelButtonsPanelCustomizer Interface

Defines classes that customize the **Apply/Cancel** panel, which appears when you either add a new row to a grid or edit grid data in the **Inplace** row edit mode.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public interface ISaveCancelButtonsPanelCustomizer
```

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

### Inheritance Hierarchy

### See Also

ISaveCancelButtonsPanelCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# ISaveCancelButtonsPanelCustomizer Members

Defines classes that customize the **Apply/Cancel** panel, which appears when you either add a new row to a grid or edit grid data in the **Inplace** row edit mode.

The following tables list the members exposed by the ISaveCancelButtonsPanelCustomizer type.

**Public Properties**

| | Name | Description |
|---|---|---|
| | BackgroundColor | Specifies the background color of the **Apply/Cancel** panel, which appears when you either add a new row to a grid or edit grid data with the **Inplace** row edit mode. |
| | BorderColor | Specifies the border color of the **Apply/Cancel** panel, which appears when you either add a new row to a grid or edit grid data with the **Inplace** row edit mode. |
| | Font | Specifies the font used to display text in the **Apply/Cancel** panel, which appears when you either add a new row to a grid or edit grid data with the **Inplace** row edit mode. |

**See Also**
ISaveCancelButtonsPanelCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# ISaveCancelButtonsPanelCustomizer Properties

Defines classes that customize the **Apply/Cancel** panel, which appears when you either add a new row to a grid or edit grid data in the **Inplace** row edit mode.

The following tables list the members exposed by the ISaveCancelButtonsPanelCustomizer type.

**Public Properties**

| | Name | Description |
|---|---|---|
| | BackgroundColor | Specifies the background color of the **Apply/Cancel** panel, which appears when you either add a new row to a grid or edit grid data with the **Inplace** row edit mode. |
| | BorderColor | Specifies the border color of the **Apply/Cancel** panel, which appears when you either add a new row to a grid or edit grid data with the **Inplace** row edit mode. |
| | Font | Specifies the font used to display text in the **Apply/Cancel** panel, which appears when you either add a new row to a grid or edit grid data with the **Inplace** row edit mode. |

**See Also**
ISaveCancelButtonsPanelCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# BackgroundColor Property

Specifies the background color of the **Apply/Cancel** panel, which appears when you either add a new row to a grid or edit grid data with the **Inplace** row edit mode.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
Color BackgroundColor { get; set; }
```

### Property Value

A Xamarin.Forms.Color object specifying the background color.

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

### See Also

ISaveCancelButtonsPanelCustomizer Interface
ISaveCancelButtonsPanelCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# BorderColor Property

Specifies the border color of the **Apply/Cancel** panel, which appears when you either add a new row to a grid or edit grid data with the **Inplace** row edit mode.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
Color BorderColor { get; set; }
```

### Property Value

A Xamarin.Forms.Color object specifying the border color.

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

### See Also

ISaveCancelButtonsPanelCustomizer Interface
ISaveCancelButtonsPanelCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# Font Property

Specifies the font used to display text in the **Apply/Cancel** panel, which appears when you either add a new row to a grid or edit grid data with the **Inplace** row edit mode.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
ThemeFontAttributes Font { get; set; }
```

### Property Value

A **ThemeFontAttributes** object specifying the desired font.

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

### See Also

ISaveCancelButtonsPanelCustomizer Interface
ISaveCancelButtonsPanelCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# IThemeChangingHandler Interface

Defines classes that react on a GridControl theme change.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public interface IThemeChangingHandler
```

### Remarks

Use this interface's OnThemeChanged method to track the moment the theme changes.

### Inheritance Hierarchy

### See Also

IThemeChangingHandler Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# IThemeChangingHandler Members

Defines classes that react on a GridControl theme change.

The following tables list the members exposed by the IThemeChangingHandler type.

**Public Methods**

| | Name | Description |
|---|---|---|
| | OnThemeChanged | This method is called when a grid theme is changed. |

**See Also**
IThemeChangingHandler Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# IThemeChangingHandler Methods

Defines classes that react on a GridControl theme change.

The following tables list the members exposed by the IThemeChangingHandler type.

**Public Methods**

| | Name | Description |
|---|---|---|
| | OnThemeChanged | This method is called when a grid theme is changed. |

**See Also**
IThemeChangingHandler Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# OnThemeChanged Method

This method is called when a grid theme is changed.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
void OnThemeChanged()
```

### Remarks

Use this method to force your custom control to react on a theme change. To do this, follow the steps below:

1. Implement the IThemeChangingHandler interface for your control.

**C#**

```
public class CustomButton : IThemeChangingHandler {
     public CustomButton() {
     }
}
```

2. Call the ThemeManager.AddThemeChangedHandler method from your control's constructor to add the control to the list of delegates handling the ThemeManager.ThemeChanged event.

**C#**

```
public class CustomButton : IThemeChangingHandler {
     public CustomButton() {
          ThemeManger.AddThemeChagedHandler(this);
     }
}
```

3. Implement the **OnThemeChanged** method for your control. For example, it may call your control's method used to repaint it.

**C#**

```
public class CustomButton : IThemeChangingHandler {
     public CustomButton() {
          ThemeManger.AddThemeChagedHandler(this);
     }
     void OnThemeChanged(object Sender, EventArgs e) {
          RepaintButton();
     }
}
```

### See Also

IThemeChangingHandler Interface
IThemeChangingHandler Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# ITotalSummaryCustomizer Interface

Defines classes that customize a grid's Total Summary.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public interface ITotalSummaryCustomizer
```

### Example

| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T297553. | |

This example demonstrates how to customize a theme used to paint a GridControl.

To apply a theme to a **GridControl**, assign its name to the ThemeManager.ThemeName property. The Themes class stores available theme names.

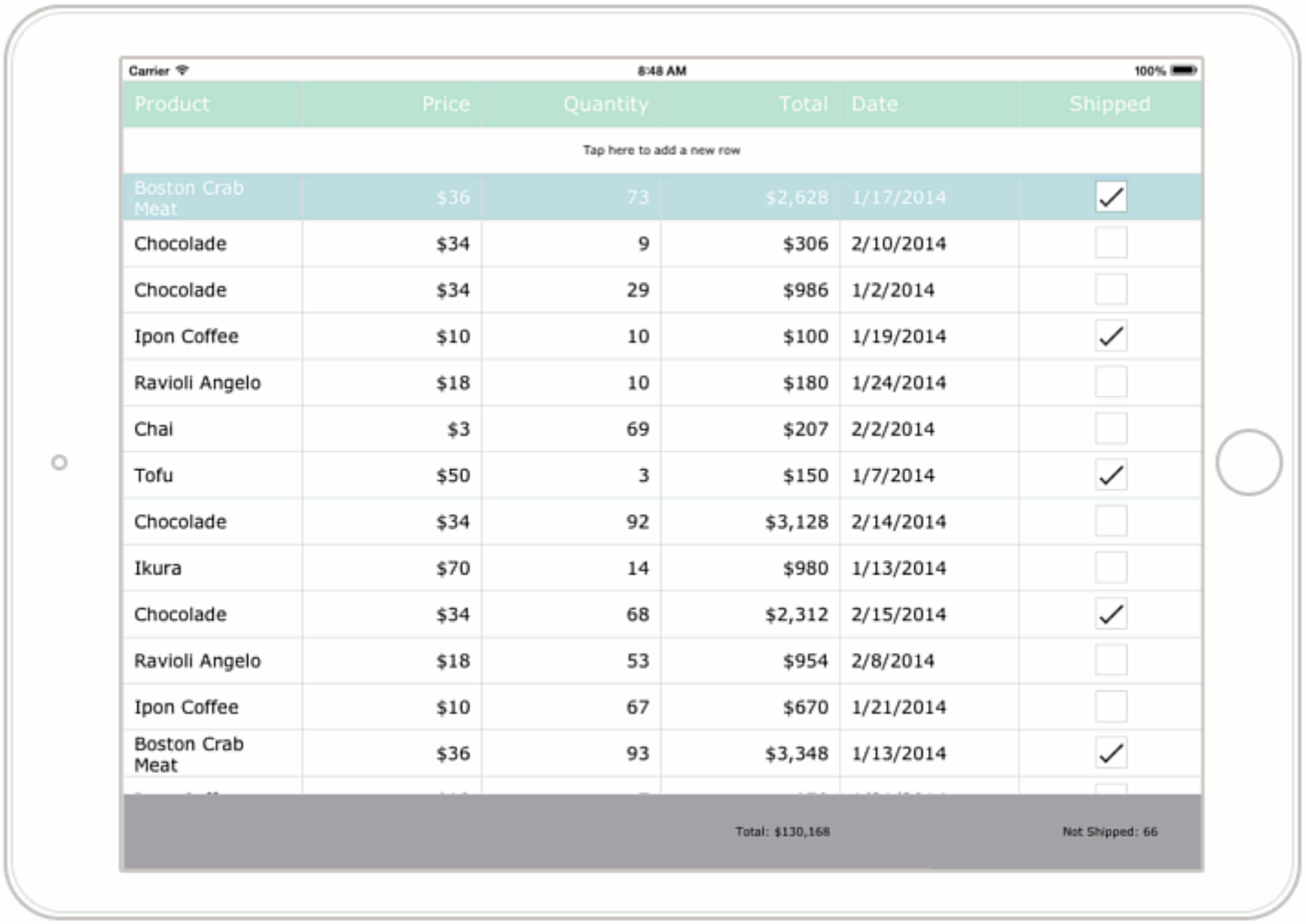
The ThemeBase class's properties provide access to different customizers, which you can adjust to change appearance settings (for example, font attributes, border and background colors, etc.) of the grid's corresponding visual elements (such as data cells, group rows, filter panel, etc.). In this example, the following customizers are used.

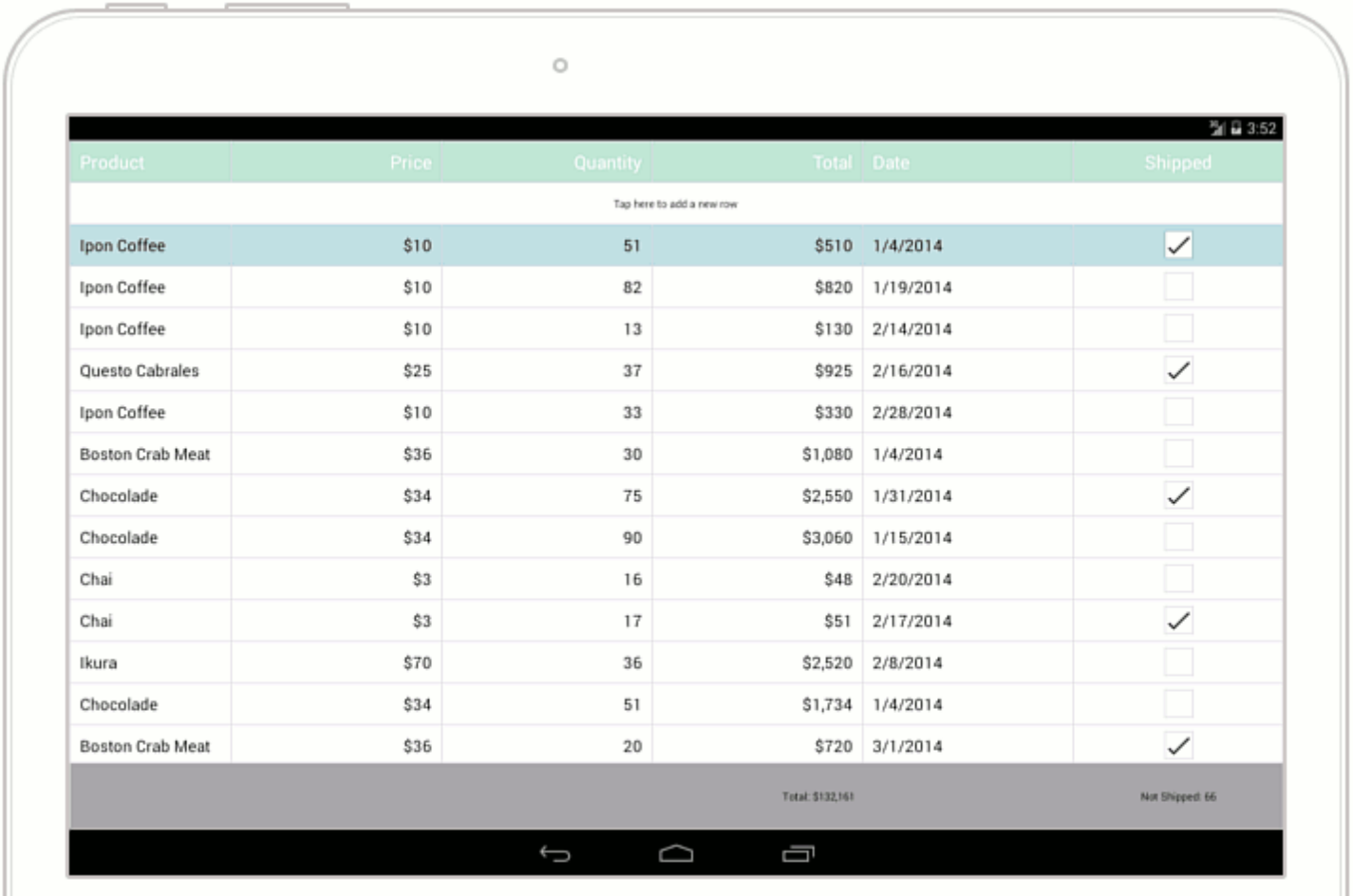
- ThemeBase.HeaderCustomizer
- ThemeBase.CellCustomizer
- ThemeBase.TotalSummaryCustomizer
- ThemeBase.NewItemRowCustomizer

| Important | |
|---|---|
| To apply the customization, call the ThemeManager.RefreshTheme method. | |

This example customizes the Light theme as it is shown in the image below.

© 2018 Developer Express Inc.


Carrier
8:48 AM
100%
Product
Price
Quantity
Total
Date
Shipped
Tap here to add a new row
Boston Crab Meat $36 73 $2,628 1/17/2014
Chocolade $34 9 $306 2/10/2014
Chocolade $34 29 $986 1/2/2014
Ipon Coffee $10 10 $100 1/19/2014
Ravioli Angelo $18 10 $180 1/24/2014
Chai $3 69 $207 2/2/2014
Tofu $50 3 $150 1/7/2014
Chocolade $34 92 $3,128 2/14/2014
Ikura $70 14 $980 1/13/2014
Chocolade $34 68 $2,312 2/15/2014
Ravioli Angelo $18 53 $954 2/8/2014
Ipon Coffee $10 67 $670 1/21/2014
Boston Crab Meat $36 93 $3,348 1/13/2014
Total: $130,168
Not Shipped: 66


3:52
Product
Price
Quantity
Total
Date
Shipped
Tap here to add a new row
Ipon Coffee $10 51 $510 1/4/2014
Ipon Coffee $10 82 $820 1/19/2014
Ipon Coffee $10 13 $130 2/14/2014
Questo Cabrales $25 37 $925 2/16/2014
Ipon Coffee $10 33 $330 2/28/2014
Boston Crab Meat $36 30 $1,080 1/4/2014
Chocolade $34 75 $2,550 1/31/2014
Chocolade $34 90 $3,060 1/15/2014
Chai $3 16 $48 2/20/2014
Chai $3 17 $51 2/17/2014
Ikura $70 36 $2,520 2/8/2014
Chocolade $34 51 $1,734 1/4/2014
Boston Crab Meat $36 20 $720 3/1/2014
Total: $132,161
Not Shipped: 66

C#

```
 (MainPage.xaml.cs)
using System;
using System.Collections.Generic;
using Xamarin.Forms;
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.DataGrid.Theme;
namespace ThemeCustomizing {
    public partial class MainPage : ContentPage {
        int count = 0;
        public MainPage() {
            InitializeComponent();
            TestOrdersRepository model = new TestOrdersRepository();
            BindingContext = model;
            ThemeManager.ThemeName = Themes.Light;
            // Header customization.
            ThemeManager.Theme.HeaderCustomizer.BackgroundColor = Color.FromRgb(187, 228, 208);
            ThemeFontAttributes myFont = new ThemeFontAttributes("Verdana",
                                         ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Large),
                                         FontAttributes.None, Color.White);
            ThemeManager.Theme.HeaderCustomizer.Font = myFont;
            // Cell customization.
            ThemeManager.Theme.CellCustomizer.SelectionColor = Color.FromRgb(186, 220, 225);
            ThemeFontAttributes myFont1 = new ThemeFontAttributes("Verdana",
                                         ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Medium),
                                         FontAttributes.None, Color.Black);
            ThemeManager.Theme.CellCustomizer.Font = myFont1;
            // Various customization.
            ThemeManager.Theme.TotalSumaryCustomizer.BackgroundColor = Color.FromRgb(163, 162, 168);
            ThemeFontAttributes myFont2 = new ThemeFontAttributes("Verdana",
                                 ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Default),
                                 FontAttributes.None, Color.Black);
            ThemeManager.Theme.TotalSummaryCustomizer.Font = myFont2;
            ThemeManager.Theme.NewItemRowCustomizer.Font = myFont2;
            ThemeManager.RefreshTheme();
        }
        void OnCalculateCustomSummary(object sender, CustomSummaryEventArgs e) {
            if (e.FieldName.ToString() == "Shipped")
            if (e.IsTotalSummary) {
                if (e.SummaryProcess == CustomSummaryProcess.Calculate) {
                    if (!(bool)e.FieldValue)
                        count++;
                    e.TotalValue = count;
                }
            }
        }
    }
}
```

**Inheritance Hierarchy**

**See Also**
ITotalSummaryCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# ITotalSummaryCustomizer Members

Defines classes that customize a grid's Total Summary.

The following tables list the members exposed by the ITotalSummaryCustomizer type.

**Public Properties**

| | Name | Description |
|---|---|---|
| | BackgroundColor | Specifies the Total Summary's background color. |
| | BorderColor | Specifies the Total Summary's border color. |
| | Font | Specifies the font used to display text in the Total Summary. |
| | Padding | Specifies the internal space between the Total Summary's content and its edge, in density-independent pixels. |

**Public Methods**

| | Name | Description |
|---|---|---|
| | LocateSummaryViewInPanel | Locates the specified view within the Total Summary's bounds. |

**See Also**
ITotalSummaryCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# ITotalSummaryCustomizer Properties

Defines classes that customize a grid's Total Summary.

The following tables list the members exposed by the ITotalSummaryCustomizer type.

**Public Properties**

| | Name | Description |
|---|---|---|
| | BackgroundColor | Specifies the Total Summary's background color. |
| | BorderColor | Specifies the Total Summary's border color. |
| | Font | Specifies the font used to display text in the Total Summary. |
| | Padding | Specifies the internal space between the Total Summary's content and its edge, in density-independent pixels. |

**Public Methods**

| | Name | Description |
|---|---|---|
| | LocateSummaryViewInPanel | Locates the specified view within the Total Summary's bounds. |

**See Also**
ITotalSummaryCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# BackgroundColor Property

Specifies the Total Summary's background color.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
Color BackgroundColor { get; set; }
```

### Property Value

A Xamarin.Forms.Color object specifying the background color.

### Example

| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T297553. | |

This example demonstrates how to customize a theme used to paint a GridControl.

To apply a theme to a **GridControl**, assign its name to the ThemeManager.ThemeName property. The Themes class stores available theme names.

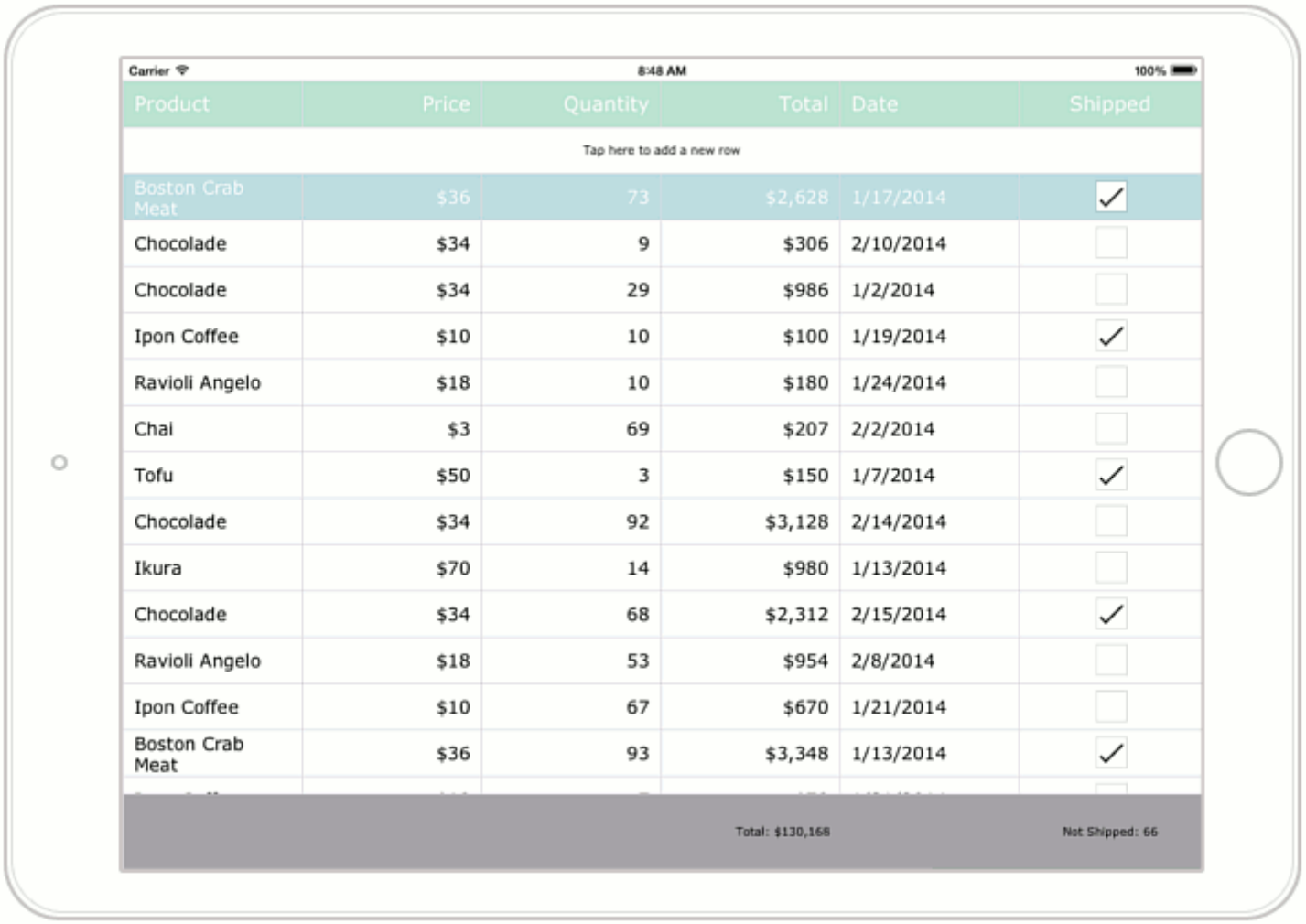
The ThemeBase class's properties provide access to different customizers, which you can adjust to change appearance settings (for example, font attributes, border and background colors, etc.) of the grid's corresponding visual elements (such as data cells, group rows, filter panel, etc.). In this example, the following customizers are used.

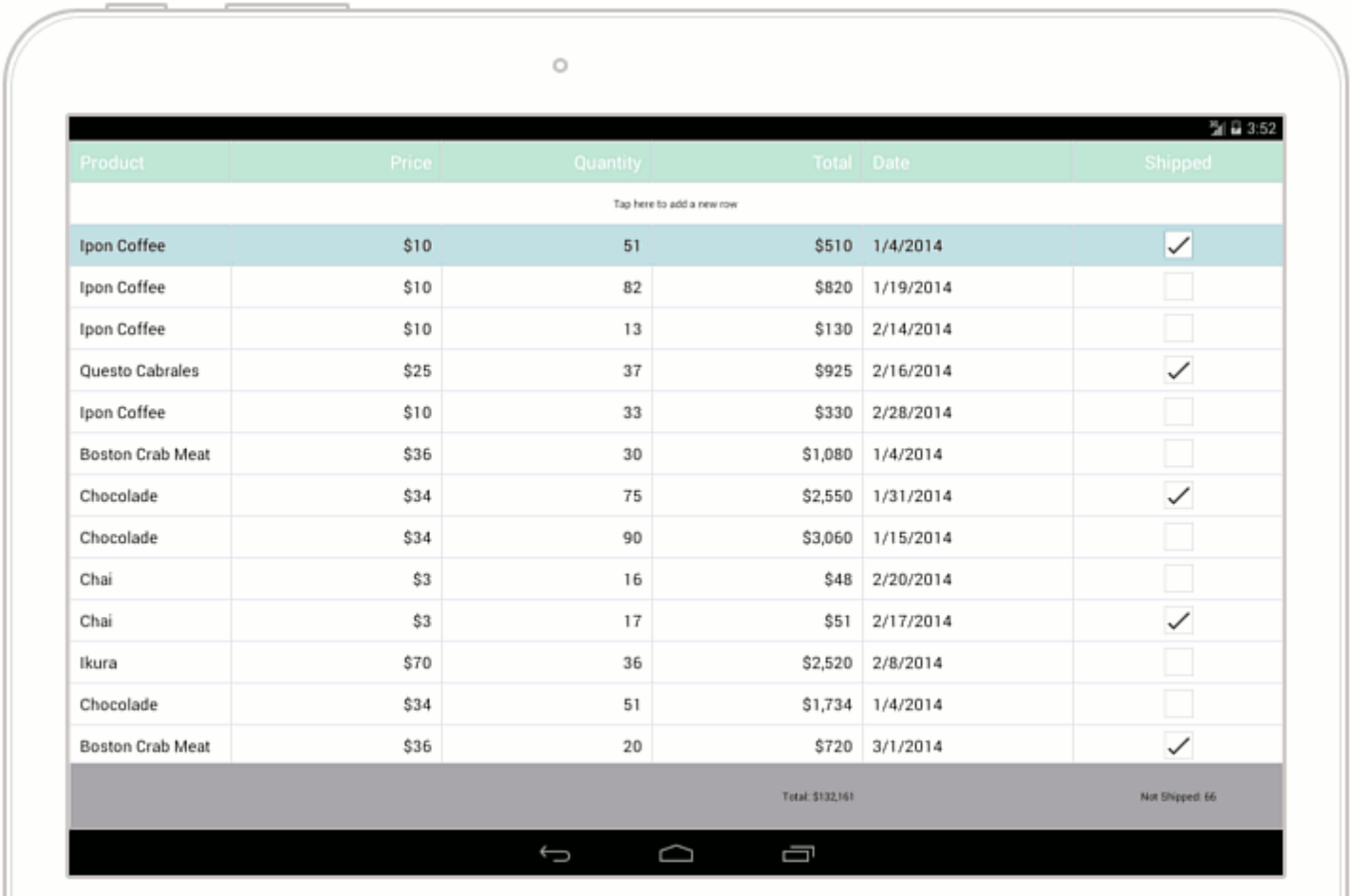
- ThemeBase.HeaderCustomizer
- ThemeBase.CellCustomizer
- ThemeBase.TotalSummaryCustomizer
- ThemeBase.NewItemRowCustomizer

| Important | |
|---|---|
| To apply the customization, call the ThemeManager.RefreshTheme method. | |

This example customizes the Light theme as it is shown in the image below.

© 2018 Developer Express Inc.


Carrier
8:48 AM
100%
Product
Price
Quantity
Total
Date
Shipped
Tap here to add a new row
Boston Crab Meat $36 73 $2,628 1/17/2014
Chocolade $34 9 $306 2/10/2014
Chocolade $34 29 $986 1/2/2014
Ipon Coffee $10 10 $100 1/19/2014
Ravioli Angelo $18 10 $180 1/24/2014
Chai $3 69 $207 2/2/2014
Tofu $50 3 $150 1/7/2014
Chocolade $34 92 $3,128 2/14/2014
Ikura $70 14 $980 1/13/2014
Chocolade $34 68 $2,312 2/15/2014
Ravioli Angelo $18 53 $954 2/8/2014
Ipon Coffee $10 67 $670 1/21/2014
Boston Crab Meat $36 93 $3,348 1/13/2014
Total: $130,168
Not Shipped: 66


3:52
Product
Price
Quantity
Total
Date
Shipped
Tap here to add a new row
Ipon Coffee $10 51 $510 1/4/2014
Ipon Coffee $10 82 $820 1/19/2014
Ipon Coffee $10 13 $130 2/14/2014
Questo Cabrales $25 37 $925 2/16/2014
Ipon Coffee $10 33 $330 2/28/2014
Boston Crab Meat $36 30 $1,080 1/4/2014
Chocolade $34 75 $2,550 1/31/2014
Chocolade $34 90 $3,060 1/15/2014
Chai $3 16 $48 2/20/2014
Chai $3 17 $51 2/17/2014
Ikura $70 36 $2,520 2/8/2014
Chocolade $34 51 $1,734 1/4/2014
Boston Crab Meat $36 20 $720 3/1/2014
Total: $132,161
Not Shipped: 66

C#

```
 (MainPage.xaml.cs)
using System;
using System.Collections.Generic;
using Xamarin.Forms;
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.DataGrid.Theme;
namespace ThemeCustomizing {
    public partial class MainPage : ContentPage {
        int count = 0;
        public MainPage() {
            InitializeComponent();
            TestOrdersRepository model = new TestOrdersRepository();
            BindingContext = model;
            ThemeManager.ThemeName = Themes.Light;
            // Header customization.
            ThemeManager.Theme.HeaderCustomizer.BackgroundColor = Color.FromRgb(187, 228, 208);
            ThemeFontAttributes myFont = new ThemeFontAttributes("Verdana",
                                         ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Large),
                                         FontAttributes.None, Color.White);
            ThemeManager.Theme.HeaderCustomizer.Font = myFont;
            // Cell customization.
            ThemeManager.Theme.CellCustomizer.SelectionColor = Color.FromRgb(186, 220, 225);
            ThemeFontAttributes myFont1 = new ThemeFontAttributes("Verdana",
                                         ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Medium),
                                         FontAttributes.None, Color.Black);
            ThemeManager.Theme.CellCustomizer.Font = myFont1;
            // Various customization.
            ThemeManager.Theme.TotalSumaryCustomizer.BackgroundColor = Color.FromRgb(163, 162, 168);
            ThemeFontAttributes myFont2 = new ThemeFontAttributes("Verdana",
                                 ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Default),
                                 FontAttributes.None, Color.Black);
            ThemeManager.Theme.TotalSummaryCustomizer.Font = myFont2;
            ThemeManager.Theme.NewItemRowCustomizer.Font = myFont2;
            ThemeManager.RefreshTheme();
        }
        void OnCalculateCustomSummary(object sender, CustomSummaryEventArgs e) {
            if (e.FieldName.ToString() == "Shipped")
            if (e.IsTotalSummary) {
                if (e.SummaryProcess == CustomSummaryProcess.Calculate) {
                    if (!(bool)e.FieldValue)
                        count++;
                    e.TotalValue = count;
                }
            }
        }
    }
}
```

**See Also**

ITotalSummaryCustomizer Interface
ITotalSummaryCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# BorderColor Property

Specifies the Total Summary's border color.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
Color BorderColor { get; set; }
```

### Property Value

A Xamarin.Forms.Color object that specifies the border color.

### Example

| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T297553. | |

This example demonstrates how to customize a theme used to paint a GridControl.

To apply a theme to a **GridControl**, assign its name to the ThemeManager.ThemeName property. The Themes class stores available theme names.

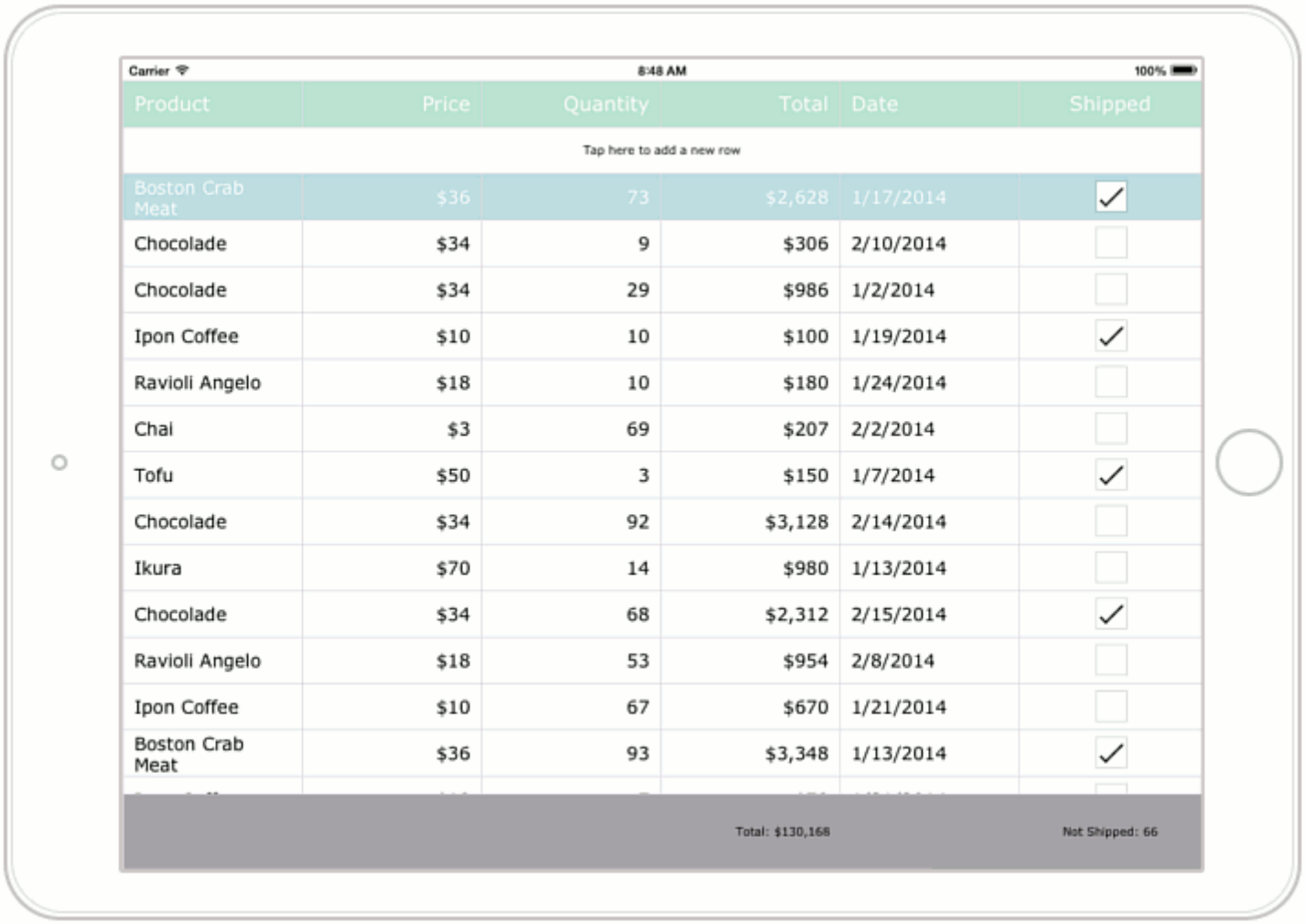
The ThemeBase class's properties provide access to different customizers, which you can adjust to change appearance settings (for example, font attributes, border and background colors, etc.) of the grid's corresponding visual elements (such as data cells, group rows, filter panel, etc.). In this example, the following customizers are used.

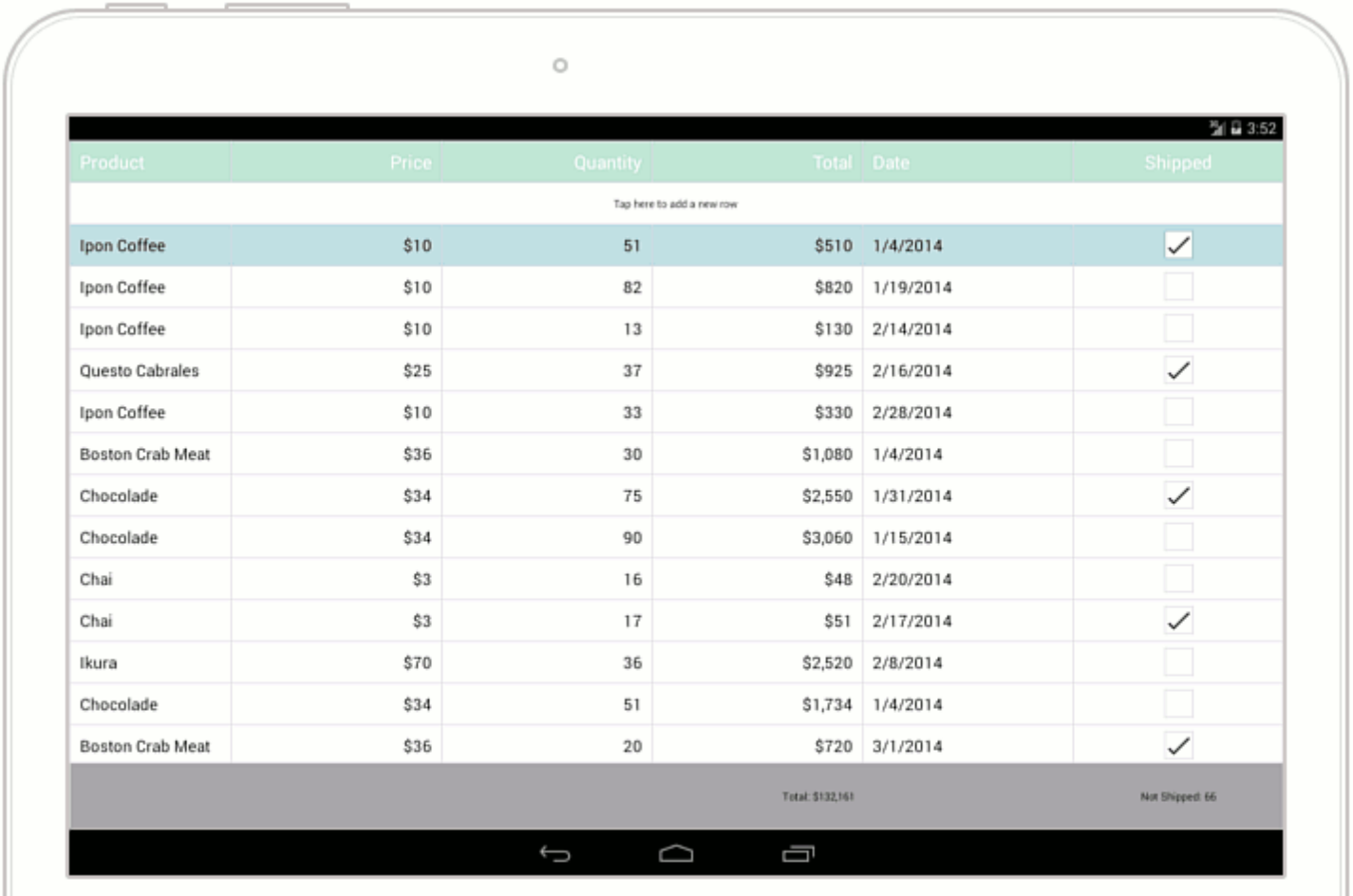
- ThemeBase.HeaderCustomizer
- ThemeBase.CellCustomizer
- ThemeBase.TotalSummaryCustomizer
- ThemeBase.NewItemRowCustomizer

| Important | |
|---|---|
| To apply the customization, call the ThemeManager.RefreshTheme method. | |

This example customizes the Light theme as it is shown in the image below.

© 2018 Developer Express Inc.


Carrier
8:48 AM
100%
Product
Price
Quantity
Total
Date
Shipped
Tap here to add a new row
Boston Crab Meat $36 73 $2,628 1/17/2014
Chocolade $34 9 $306 2/10/2014
Chocolade $34 29 $986 1/2/2014
Ipon Coffee $10 10 $100 1/19/2014
Ravioli Angelo $18 10 $180 1/24/2014
Chai $3 69 $207 2/2/2014
Tofu $50 3 $150 1/7/2014
Chocolade $34 92 $3,128 2/14/2014
Ikura $70 14 $980 1/13/2014
Chocolade $34 68 $2,312 2/15/2014
Ravioli Angelo $18 53 $954 2/8/2014
Ipon Coffee $10 67 $670 1/21/2014
Boston Crab Meat $36 93 $3,348 1/13/2014
Total: $130,168
Not Shipped: 66


3:52
Product
Price
Quantity
Total
Date
Shipped
Tap here to add a new row
Ipon Coffee $10 51 $510 1/4/2014
Ipon Coffee $10 82 $820 1/19/2014
Ipon Coffee $10 13 $130 2/14/2014
Questo Cabrales $25 37 $925 2/16/2014
Ipon Coffee $10 33 $330 2/28/2014
Boston Crab Meat $36 30 $1,080 1/4/2014
Chocolade $34 75 $2,550 1/31/2014
Chocolade $34 90 $3,060 1/15/2014
Chai $3 16 $48 2/20/2014
Chai $3 17 $51 2/17/2014
Ikura $70 36 $2,520 2/8/2014
Chocolade $34 51 $1,734 1/4/2014
Boston Crab Meat $36 20 $720 3/1/2014
Total: $132,161
Not Shipped: 66

**C#**

```
 (MainPage.xaml.cs)
using System;
using System.Collections.Generic;
using Xamarin.Forms;
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.DataGrid.Theme;
namespace ThemeCustomizing {
    public partial class MainPage : ContentPage {
        int count = 0;
        public MainPage() {
            InitializeComponent();
            TestOrdersRepository model = new TestOrdersRepository();
            BindingContext = model;
            ThemeManager.ThemeName = Themes.Light;
            // Header customization.
            ThemeManager.Theme.HeaderCustomizer.BackgroundColor = Color.FromRgb(187, 228, 208);
            ThemeFontAttributes myFont = new ThemeFontAttributes("Verdana",
                                         ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Large),
                                         FontAttributes.None, Color.White);
            ThemeManager.Theme.HeaderCustomizer.Font = myFont;
            // Cell customization.
            ThemeManager.Theme.CellCustomizer.SelectionColor = Color.FromRgb(186, 220, 225);
            ThemeFontAttributes myFont1 = new ThemeFontAttributes("Verdana",
                                         ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Medium),
                                         FontAttributes.None, Color.Black);
            ThemeManager.Theme.CellCustomizer.Font = myFont1;
            // Various customization.
            ThemeManager.Theme.TotalSumaryCustomizer.BackgroundColor = Color.FromRgb(163, 162, 168);
            ThemeFontAttributes myFont2 = new ThemeFontAttributes("Verdana",
                                 ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Default),
                                 FontAttributes.None, Color.Black);
            ThemeManager.Theme.TotalSummaryCustomizer.Font = myFont2;
            ThemeManager.Theme.NewItemRowCustomizer.Font = myFont2;
            ThemeManager.RefreshTheme();
        }
        void OnCalculateCustomSummary(object sender, CustomSummaryEventArgs e) {
            if (e.FieldName.ToString() == "Shipped")
            if (e.IsTotalSummary) {
                if (e.SummaryProcess == CustomSummaryProcess.Calculate) {
                    if (!(bool)e.FieldValue)
                        count++;
                    e.TotalValue = count;
                }
            }
        }
    }
}
```

**See Also**

ITotalSummaryCustomizer Interface
ITotalSummaryCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# Font Property

Specifies the font used to display text in the Total Summary.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
ThemeFontAttributes Font { get; set; }
```

### Property Value

A **ThemeFontAttributes** object specifying the desired font.

### Example

| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T297553. | |

This example demonstrates how to customize a theme used to paint a GridControl.

To apply a theme to a **GridControl**, assign its name to the ThemeManager.ThemeName property. The Themes class stores available theme names.

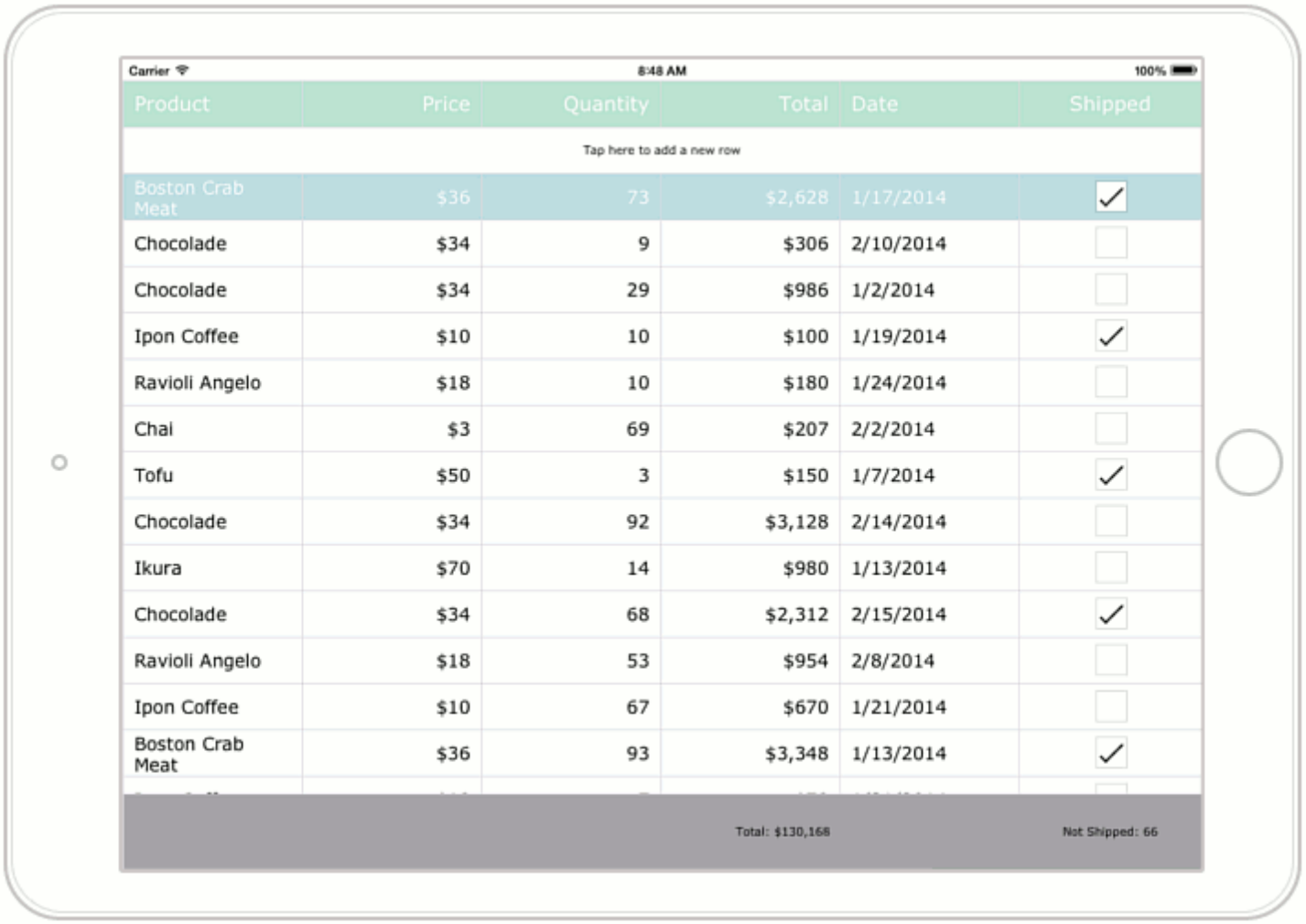
The ThemeBase class's properties provide access to different customizers, which you can adjust to change appearance settings (for example, font attributes, border and background colors, etc.) of the grid's corresponding visual elements (such as data cells, group rows, filter panel, etc.). In this example, the following customizers are used.

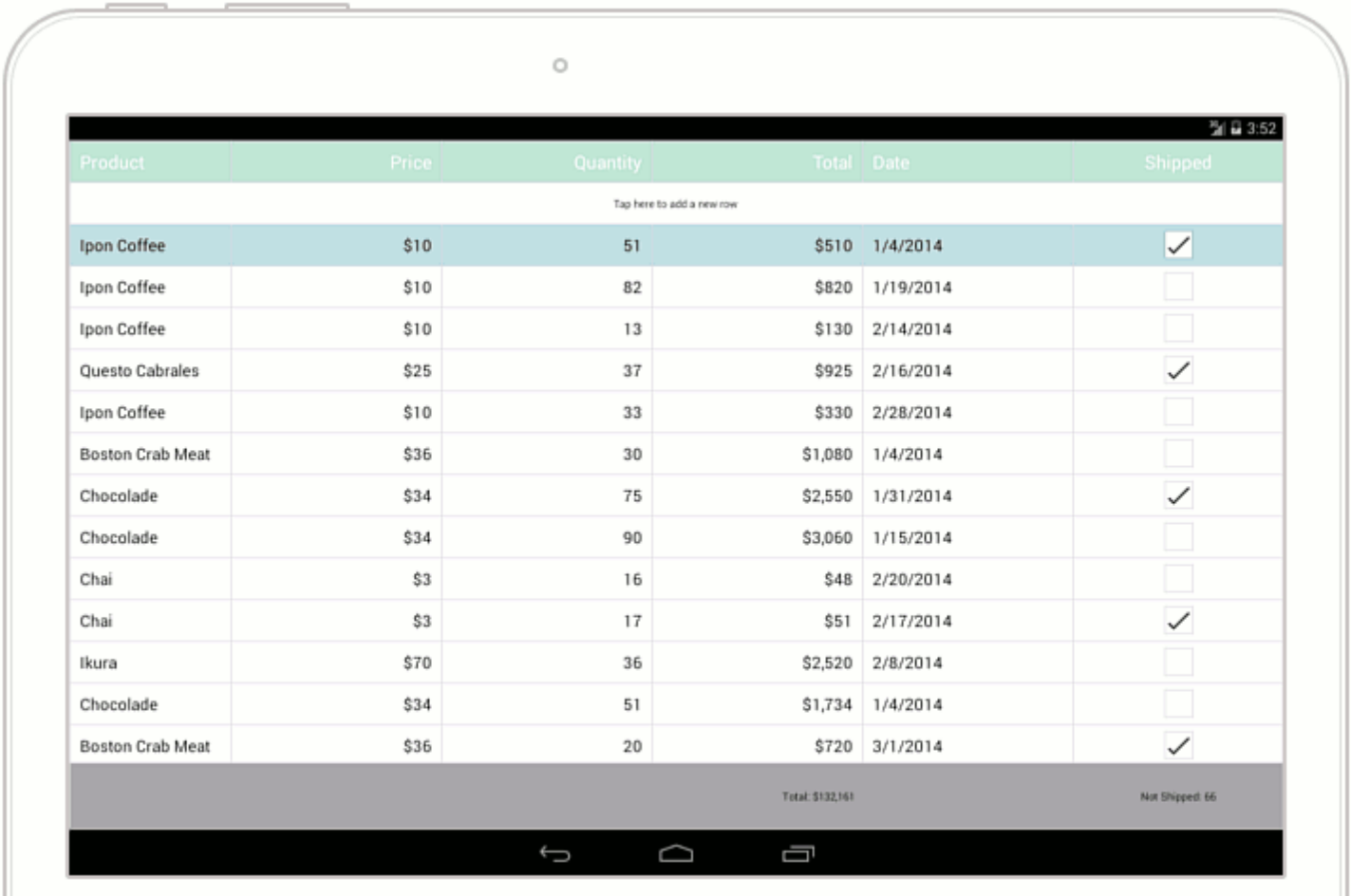
- ThemeBase.HeaderCustomizer
- ThemeBase.CellCustomizer
- ThemeBase.TotalSummaryCustomizer
- ThemeBase.NewItemRowCustomizer

| Important | |
|---|---|
| To apply the customization, call the ThemeManager.RefreshTheme method. | |

This example customizes the Light theme as it is shown in the image below.

© 2018 Developer Express Inc.


Carrier
8:48 AM
100%
Product
Price
Quantity
Total
Date
Shipped
Tap here to add a new row
Boston Crab Meat $36 73 $2,628 1/17/2014
Chocolade $34 9 $306 2/10/2014
Chocolade $34 29 $986 1/2/2014
Ipon Coffee $10 10 $100 1/19/2014
Ravioli Angelo $18 10 $180 1/24/2014
Chai $3 69 $207 2/2/2014
Tofu $50 3 $150 1/7/2014
Chocolade $34 92 $3,128 2/14/2014
Ikura $70 14 $980 1/13/2014
Chocolade $34 68 $2,312 2/15/2014
Ravioli Angelo $18 53 $954 2/8/2014
Ipon Coffee $10 67 $670 1/21/2014
Boston Crab Meat $36 93 $3,348 1/13/2014
Total: $130,168
Not Shipped: 66


3:52
Product
Price
Quantity
Total
Date
Shipped
Tap here to add a new row
Ipon Coffee $10 51 $510 1/4/2014
Ipon Coffee $10 82 $820 1/19/2014
Ipon Coffee $10 13 $130 2/14/2014
Questo Cabrales $25 37 $925 2/16/2014
Ipon Coffee $10 33 $330 2/28/2014
Boston Crab Meat $36 30 $1,080 1/4/2014
Chocolade $34 75 $2,550 1/31/2014
Chocolade $34 90 $3,060 1/15/2014
Chai $3 16 $48 2/20/2014
Chai $3 17 $51 2/17/2014
Ikura $70 36 $2,520 2/8/2014
Chocolade $34 51 $1,734 1/4/2014
Boston Crab Meat $36 20 $720 3/1/2014
Total: $132,161
Not Shipped: 66

C#

```
 (MainPage.xaml.cs)
using System;
using System.Collections.Generic;
using Xamarin.Forms;
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.DataGrid.Theme;
namespace ThemeCustomizing {
    public partial class MainPage : ContentPage {
        int count = 0;
        public MainPage() {
            InitializeComponent();
            TestOrdersRepository model = new TestOrdersRepository();
            BindingContext = model;
            ThemeManager.ThemeName = Themes.Light;
            // Header customization.
            ThemeManager.Theme.HeaderCustomizer.BackgroundColor = Color.FromRgb(187, 228, 208);
            ThemeFontAttributes myFont = new ThemeFontAttributes("Verdana",
                                         ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Large),
                                         FontAttributes.None, Color.White);
            ThemeManager.Theme.HeaderCustomizer.Font = myFont;
            // Cell customization.
            ThemeManager.Theme.CellCustomizer.SelectionColor = Color.FromRgb(186, 220, 225);
            ThemeFontAttributes myFont1 = new ThemeFontAttributes("Verdana",
                                         ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Medium),
                                         FontAttributes.None, Color.Black);
            ThemeManager.Theme.CellCustomizer.Font = myFont1;
            // Various customization.
            ThemeManager.Theme.TotalSumaryCustomizer.BackgroundColor = Color.FromRgb(163, 162, 168);
            ThemeFontAttributes myFont2 = new ThemeFontAttributes("Verdana",
                                 ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Default),
                                 FontAttributes.None, Color.Black);
            ThemeManager.Theme.TotalSummaryCustomizer.Font = myFont2;
            ThemeManager.Theme.NewItemRowCustomizer.Font = myFont2;
            ThemeManager.RefreshTheme();
        }
        void OnCalculateCustomSummary(object sender, CustomSummaryEventArgs e) {
            if (e.FieldName.ToString() == "Shipped")
            if (e.IsTotalSummary) {
                if (e.SummaryProcess == CustomSummaryProcess.Calculate) {
                    if (!(bool)e.FieldValue)
                        count++;
                    e.TotalValue = count;
                }
            }
        }
    }
}
```

**See Also**
ITotalSummaryCustomizer Interface
ITotalSummaryCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# Padding Property

Specifies the internal space between the Total Summary's content and its edge, in density-independent pixels.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
Thickness Padding { get; set; }
```

### Property Value

A Xamarin.Forms.Thickness object, representing the **Total Summary**'s inner indents, measured in density-independent pixels.

### Example

| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T297553. | |

This example demonstrates how to customize a theme used to paint a GridControl.

To apply a theme to a **GridControl**, assign its name to the ThemeManager.ThemeName property. The Themes class stores available theme names.

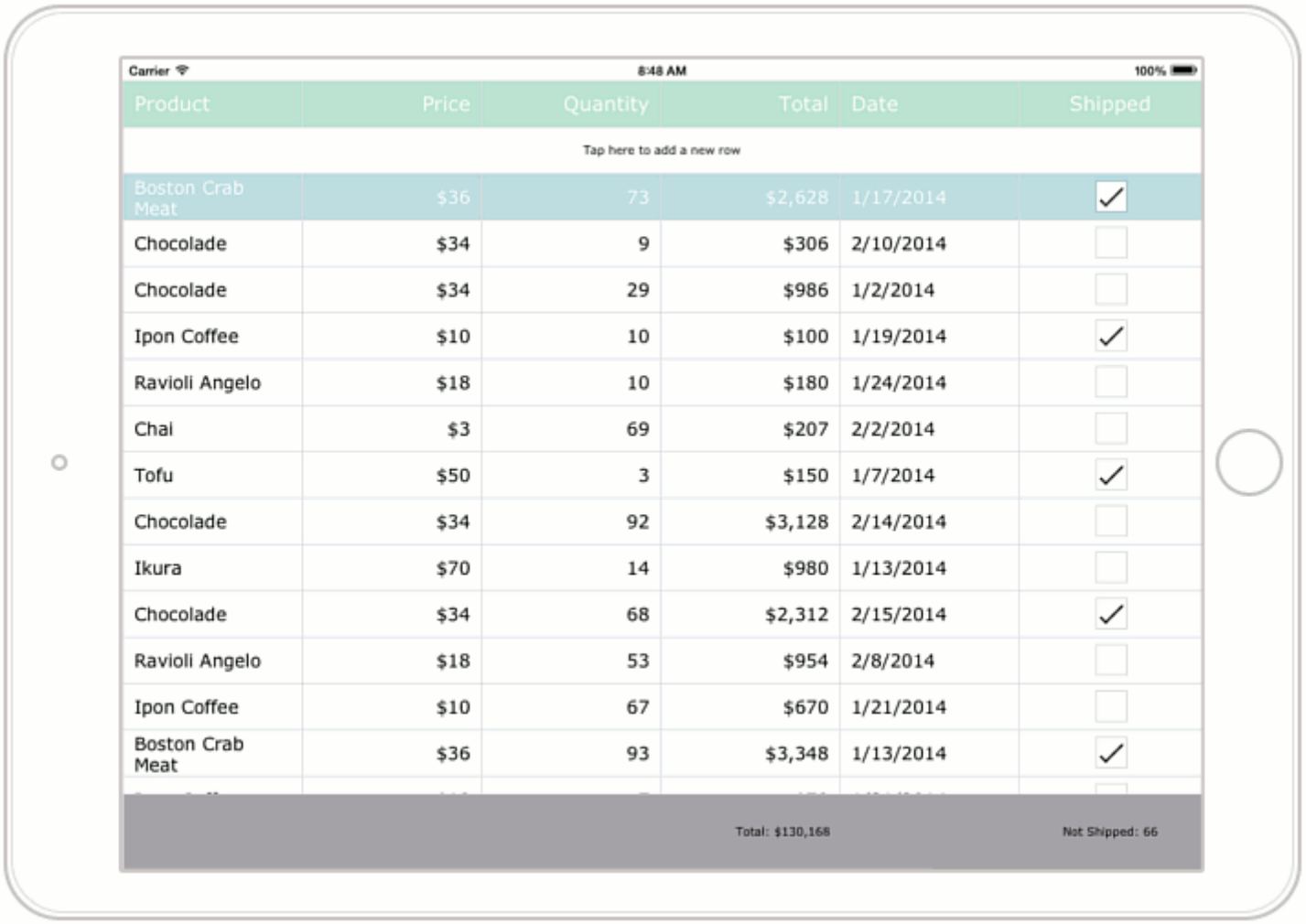
The ThemeBase class's properties provide access to different customizers, which you can adjust to change appearance settings (for example, font attributes, border and background colors, etc.) of the grid's corresponding visual elements (such as data cells, group rows, filter panel, etc.). In this example, the following customizers are used.

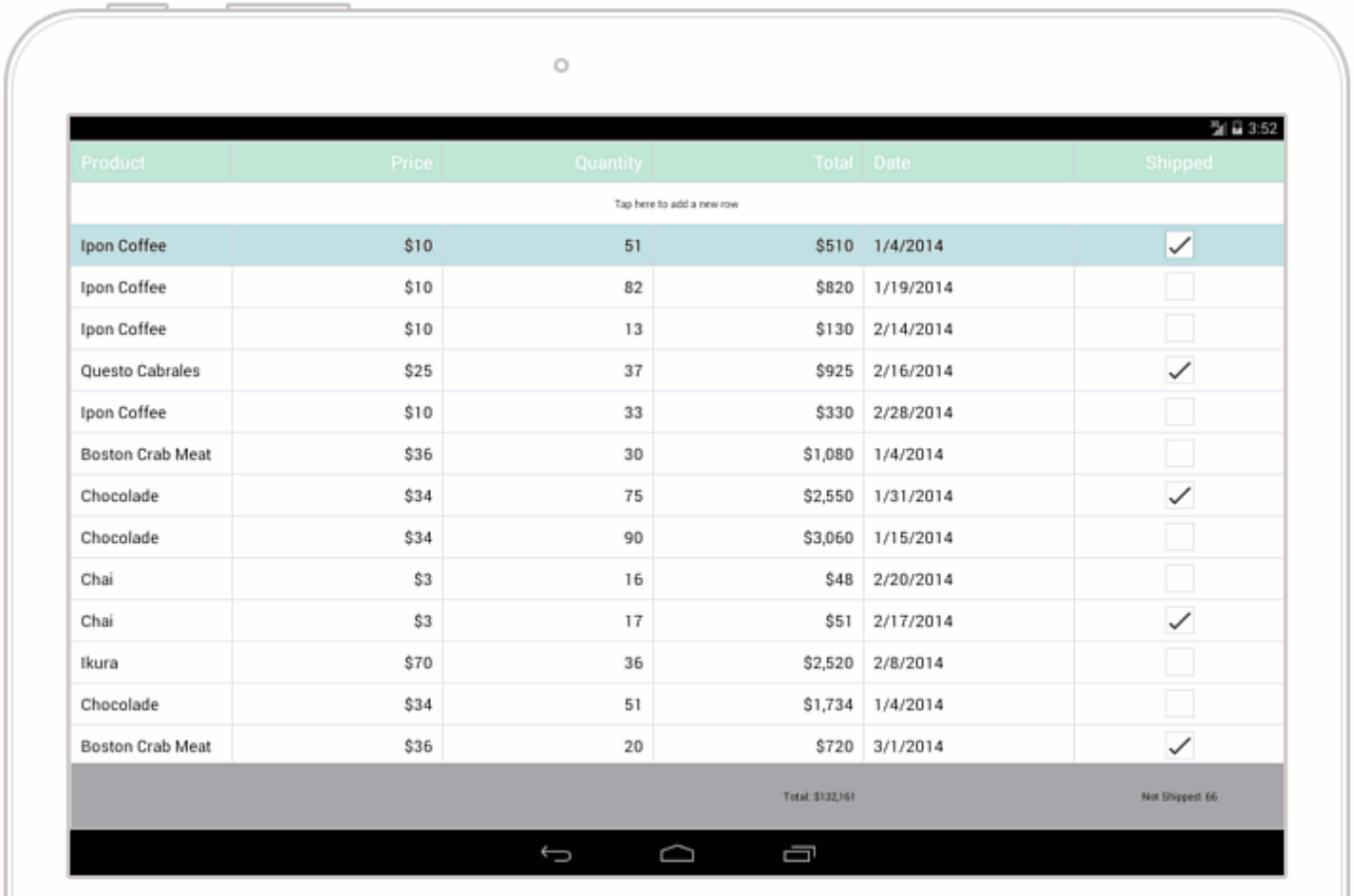
- ThemeBase.HeaderCustomizer
- ThemeBase.CellCustomizer
- ThemeBase.TotalSummaryCustomizer
- ThemeBase.NewItemRowCustomizer

| Important | |
|---|---|
| To apply the customization, call the ThemeManager.RefreshTheme method. | |

This example customizes the Light theme as it is shown in the image below.

© 2018 Developer Express Inc.


Carrier
8:48 AM
100%
Product
Price
Quantity
Total
Date
Shipped
Tap here to add a new row
Boston Crab Meat $36 73 $2,628 1/17/2014
Chocolade $34 9 $306 2/10/2014
Chocolade $34 29 $986 1/2/2014
Ipon Coffee $10 10 $100 1/19/2014
Ravioli Angelo $18 10 $180 1/24/2014
Chai $3 69 $207 2/2/2014
Tofu $50 3 $150 1/7/2014
Chocolade $34 92 $3,128 2/14/2014
Ikura $70 14 $980 1/13/2014
Chocolade $34 68 $2,312 2/15/2014
Ravioli Angelo $18 53 $954 2/8/2014
Ipon Coffee $10 67 $670 1/21/2014
Boston Crab Meat $36 93 $3,348 1/13/2014
Total: $130,168
Not Shipped: 66


3:52
Product
Price
Quantity
Total
Date
Shipped
Tap here to add a new row
Ipon Coffee $10 51 $510 1/4/2014
Ipon Coffee $10 82 $820 1/19/2014
Ipon Coffee $10 13 $130 2/14/2014
Questo Cabrales $25 37 $925 2/16/2014
Ipon Coffee $10 33 $330 2/28/2014
Boston Crab Meat $36 30 $1,080 1/4/2014
Chocolade $34 75 $2,550 1/31/2014
Chocolade $34 90 $3,060 1/15/2014
Chai $3 16 $48 2/20/2014
Chai $3 17 $51 2/17/2014
Ikura $70 36 $2,520 2/8/2014
Chocolade $34 51 $1,734 1/4/2014
Boston Crab Meat $36 20 $720 3/1/2014
Total: $132,161
Not Shipped: 66

**C#**

```
 (MainPage.xaml.cs)
using System;
using System.Collections.Generic;
using Xamarin.Forms;
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.DataGrid.Theme;
namespace ThemeCustomizing {
    public partial class MainPage : ContentPage {
        int count = 0;
        public MainPage() {
            InitializeComponent();
            TestOrdersRepository model = new TestOrdersRepository();
            BindingContext = model;
            ThemeManager.ThemeName = Themes.Light;
            // Header customization.
            ThemeManager.Theme.HeaderCustomizer.BackgroundColor = Color.FromRgb(187, 228, 208);
            ThemeFontAttributes myFont = new ThemeFontAttributes("Verdana",
                                         ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Large),
                                         FontAttributes.None, Color.White);
            ThemeManager.Theme.HeaderCustomizer.Font = myFont;
            // Cell customization.
            ThemeManager.Theme.CellCustomizer.SelectionColor = Color.FromRgb(186, 220, 225);
            ThemeFontAttributes myFont1 = new ThemeFontAttributes("Verdana",
                                         ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Medium),
                                         FontAttributes.None, Color.Black);
            ThemeManager.Theme.CellCustomizer.Font = myFont1;
            // Various customization.
            ThemeManager.Theme.TotalSumaryCustomizer.BackgroundColor = Color.FromRgb(163, 162, 168);
            ThemeFontAttributes myFont2 = new ThemeFontAttributes("Verdana",
                                 ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Default),
                                 FontAttributes.None, Color.Black);
            ThemeManager.Theme.TotalSummaryCustomizer.Font = myFont2;
            ThemeManager.Theme.NewItemRowCustomizer.Font = myFont2;
            ThemeManager.RefreshTheme();
        }
        void OnCalculateCustomSummary(object sender, CustomSummaryEventArgs e) {
            if (e.FieldName.ToString() == "Shipped")
            if (e.IsTotalSummary) {
                if (e.SummaryProcess == CustomSummaryProcess.Calculate) {
                    if (!(bool)e.FieldValue)
                        count++;
                    e.TotalValue = count;
                }
            }
        }
    }
}
```

**See Also**

ITotalSummaryCustomizer Interface
ITotalSummaryCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# ITotalSummaryCustomizer Methods

Defines classes that customize a grid's Total Summary.

The following tables list the members exposed by the ITotalSummaryCustomizer type.

**Public Methods**

| | Name | Description |
|---|---|---|
| | LocateSummaryViewInPanel | Locates the specified view within the Total Summary's bounds. |

**See Also**
ITotalSummaryCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# LocateSummaryViewInPanel Method

Locates the specified view within the Total Summary's bounds.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
void LocateSummaryViewInPanel(
   View summaryView,
   Rectangle summaryBounds,
   bool isFirst,
   bool isLast,
   bool isLeftFixedColumnBorder,
   bool isRightFixedColumnBorder
)
```

### Parameters

*summaryView*
A Xamarin.Forms.View object, specifying the view to be located within the **Total Summary**.
*summaryBounds*
A Xamarin.Forms.Rectangle object, specifying the **Total Summary**'s bounds.
*isFirst*
**true**, if the specified view is the first view in the **Total Summary**; otherwise, **false**.
*isLast*
**true**, if the specified view is the last view in the **Total Summary**; otherwise, **false**.
*isLeftFixedColumnBorder*
**true**, if the specified view belongs to the last column fixed to the left; otherwise, **false**.
*isRightFixedColumnBorder*
**true**, if the specified view belongs to the first column fixed to the right; otherwise, **false**.

### See Also

ITotalSummaryCustomizer Interface
ITotalSummaryCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# IWinPhoneRefreshCustomizer Interface

Defines classes that customize the WinPhone refreshing bar.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public interface IWinPhoneRefreshCustomizer
```

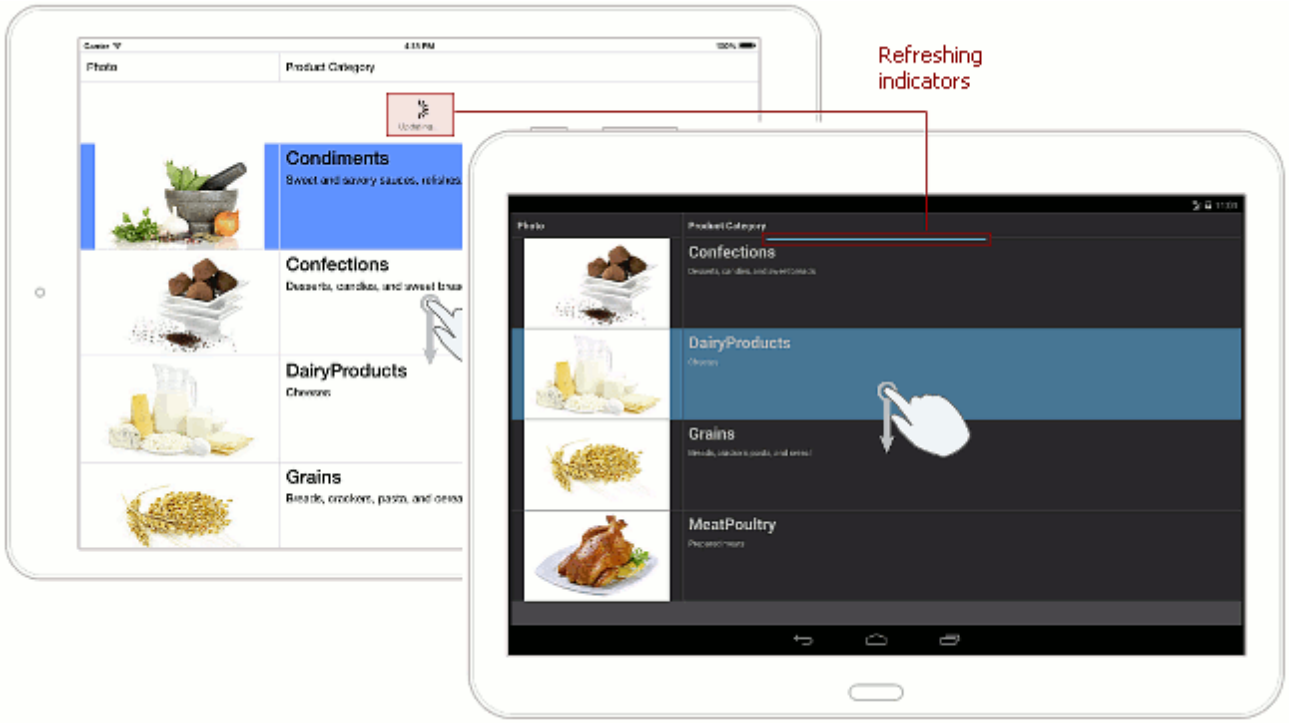
### Remarks

To customize the color of the WinPhone refreshing bar, use the BarColor property of an object that implements the **IWinPhoneRefreshCustomizer** interface.

### Example

| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T326911. | |

This example demonstrates how to customize a color of an indicator (icon or bar) shown when GridControl refreshes cell data.




To change the color of a refresh indicator, perform the steps below.

1. Activate a grid's **pull-to-refresh** functionality. For this purpose, assign **true** to the GridControl.IsPullToRefreshEnabled property.
2. Use the ThemeBase class's properties to access customizers used to change an indicator's color on different mobile platforms. In this example, the following customizers are used.
   - ThemeBase.AndroidRefreshCustomizer
   - ThemeBase.IOSRefreshCustomizer
3. Apply the customization by calling the ThemeManager.RefreshTheme method.

| C# | |
|---|---|

© 2018 Developer Express Inc.

```
 (MainPage.xaml.cs)
using System;
using System.Collections.Generic;
using Xamarin.Forms;
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.DataGrid.Theme;
namespace RefreshCustomizers {
    public partial class MainPage : ContentPage {
        public MainPage() {
            InitializeComponent();
            TestOrdersRepository model = new TestOrdersRepository();
            BindingContext = model;
            ThemeManager.Theme.AndroidRefreshCustomizer.BarColor = Color.Blue;
            ThemeManager.Theme.IOSRefreshCustomizer.RefreshColor = Color.Blue;
            ThemeManager.RefreshTheme();
        }
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<?xml version="1.0" encoding="UTF-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="RefreshCustomizers.MainPage"
             xmlns:dxGrid="clr-namespace:DevExpress.Mobile.DataGrid;assembly=DevExpress.Mobile.Grid.v15.2"
      <ContentPage.Padding>
        <OnPlatform x:TypeArguments="Thickness" iOS="0, 20, 0, 0" />
    </ContentPage.Padding>
    <dxGrid:GridControl x:Name="grid" ItemsSource="{Binding Orders}"
                        NewItemRowVisibility = "false"
                       SortMode = "Multiple" AutoFilterPanelVisibility="true"
                       IsPullToRefreshEnabled = "true" >
         <dxGrid:GridControl.Columns>
             <dxGrid:TextColumn FieldName="Product.Name" Caption = "Product" Width = "170"
                                SortOrder = "Descending" SortIndex = "0"/>
             <dxGrid:NumberColumn FieldName="Product.UnitPrice" Caption = "Price" DisplayFormat="C0"/>
             <dxGrid:NumberColumn FieldName="Quantity"
                                  SortOrder = "Ascending" SortIndex = "1"/>
             <dxGrid:NumberColumn FieldName="Total"
                                  UnboundType="Integer" UnboundExpression="[Quantity] * [Product.UnitPrice
                                  DisplayFormat="C0" IsReadOnly="True"/>
             <dxGrid:DateColumn FieldName="Date" DisplayFormat="d" />
             <dxGrid:SwitchColumn FieldName="Shipped"/>
         </dxGrid:GridControl.Columns>
         <dxGrid:GridControl.GroupSummaries>
            <dxGrid:GridColumnSummary FieldName="Total" Type="Max"/>
        </dxGrid:GridControl.GroupSummaries>
        <dxGrid:GridControl.TotalSummaries>
            <dxGrid:GridColumnSummary FieldName="Total" Type="Sum"
                                      DisplayFormat= "Total: {0:C0}"/>
        </dxGrid:GridControl.TotalSummaries>
    </dxGrid:GridControl>
</ContentPage>
```

## Inheritance Hierarchy

## See Also

IWinPhoneRefreshCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
IAndroidRefreshCustomizer
IIOSRefreshCustomizer

© 2018 Developer Express Inc.

# IWinPhoneRefreshCustomizer Members

Defines classes that customize the WinPhone refreshing bar.

The following tables list the members exposed by the IWinPhoneRefreshCustomizer type.

**Public Properties**

| | Name | Description |
|---|---|---|
| | BarColor | Specifies the color of the WinPhone refreshing bar. |

**See Also**
IWinPhoneRefreshCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
IAndroidRefreshCustomizer
IIOSRefreshCustomizer

© 2018 Developer Express Inc.

# IWinPhoneRefreshCustomizer Properties

Defines classes that customize the WinPhone refreshing bar.

The following tables list the members exposed by the IWinPhoneRefreshCustomizer type.

**Public Properties**

| | Name | Description |
|---|---|---|
| | BarColor | Specifies the color of the WinPhone refreshing bar. |

**See Also**
IWinPhoneRefreshCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace
IAndroidRefreshCustomizer
IIOSRefreshCustomizer

© 2018 Developer Express Inc.

# BarColor Property

Specifies the color of the WinPhone refreshing bar.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
Color BarColor { get; set; }
```

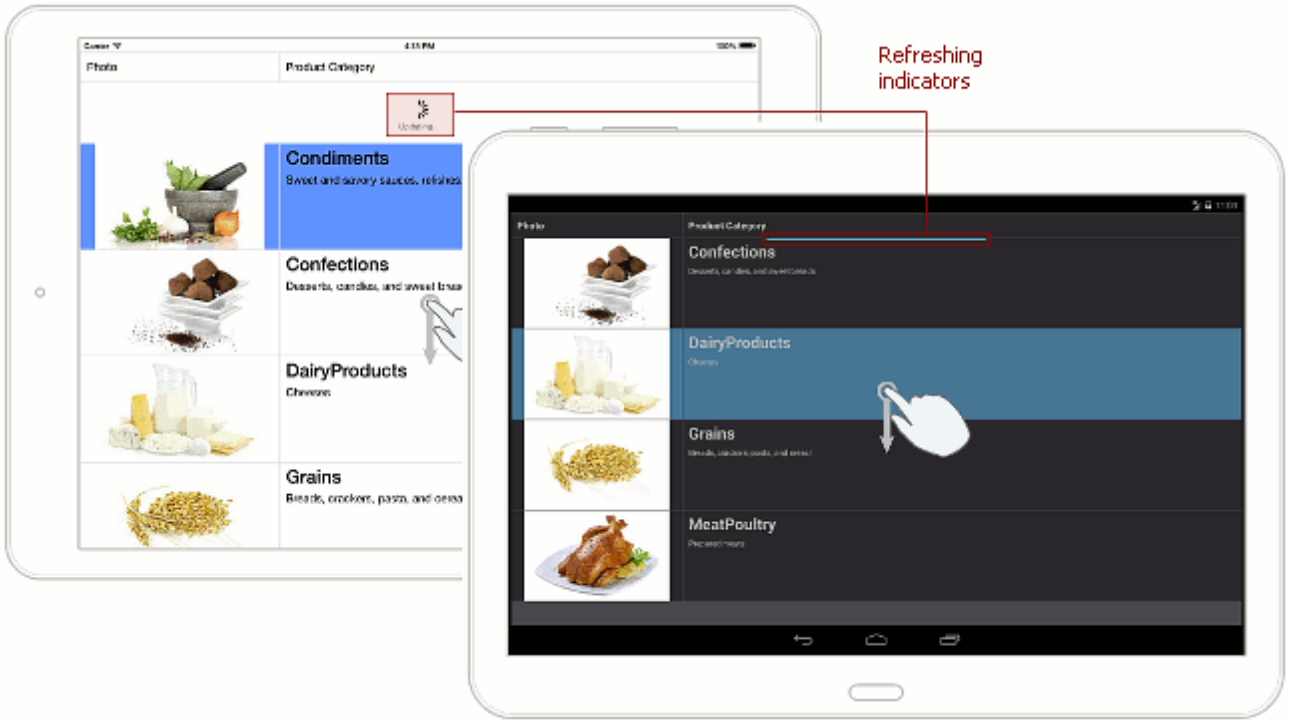
### Property Value

A Xamarin.Forms.Color object specifying the color of the WinPhone refreshing bar.

### Example

| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T326911. | |

This example demonstrates how to customize a color of an indicator (icon or bar) shown when GridControl refreshes cell data.




To change the color of a refresh indicator, perform the steps below.

1. Activate a grid's **pull-to-refresh** functionality. For this purpose, assign **true** to the GridControl.IsPullToRefreshEnabled property.
2. Use the ThemeBase class's properties to access customizers used to change an indicator's color on different mobile platforms. In this example, the following customizers are used.
   - ThemeBase.AndroidRefreshCustomizer
   - ThemeBase.IOSRefreshCustomizer
3. Apply the customization by calling the ThemeManager.RefreshTheme method.

| C# | |
|---|---|

© 2018 Developer Express Inc.

```
 (MainPage.xaml.cs)
using System;
using System.Collections.Generic;
using Xamarin.Forms;
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.DataGrid.Theme;
namespace RefreshCustomizers {
    public partial class MainPage : ContentPage {
        public MainPage() {
            InitializeComponent();
            TestOrdersRepository model = new TestOrdersRepository();
            BindingContext = model;
            ThemeManager.Theme.AndroidRefreshCustomizer.BarColor = Color.Blue;
            ThemeManager.Theme.IOSRefreshCustomizer.RefreshColor = Color.Blue;
            ThemeManager.RefreshTheme();
        }
    }
}
```

**Xaml**

```
 (MainPage.xaml)
<?xml version="1.0" encoding="UTF-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="RefreshCustomizers.MainPage"
             xmlns:dxGrid="clr-namespace:DevExpress.Mobile.DataGrid;assembly=DevExpress.Mobile.Grid.v15.2"
      <ContentPage.Padding>
        <OnPlatform x:TypeArguments="Thickness" iOS="0, 20, 0, 0" />
    </ContentPage.Padding>
    <dxGrid:GridControl x:Name="grid" ItemsSource="{Binding Orders}"
                         NewItemRowVisibility = "false"
                        SortMode = "Multiple" AutoFilterPanelVisibility="true"
                        IsPullToRefreshEnabled = "true" >
         <dxGrid:GridControl.Columns>
             <dxGrid:TextColumn FieldName="Product.Name" Caption = "Product" Width = "170"
                                SortOrder = "Descending" SortIndex = "0"/>
             <dxGrid:NumberColumn FieldName="Product.UnitPrice" Caption = "Price" DisplayFormat="C0"/>
             <dxGrid:NumberColumn FieldName="Quantity"
                                  SortOrder = "Ascending" SortIndex = "1"/>
             <dxGrid:NumberColumn FieldName="Total"
                                  UnboundType="Integer" UnboundExpression="[Quantity] * [Product.UnitPrice
                                  DisplayFormat="C0" IsReadOnly="True"/>
             <dxGrid:DateColumn FieldName="Date" DisplayFormat="d" />
             <dxGrid:SwitchColumn FieldName="Shipped"/>
         </dxGrid:GridControl.Columns>
         <dxGrid:GridControl.GroupSummaries>
            <dxGrid:GridColumnSummary FieldName="Total" Type="Max"/>
        </dxGrid:GridControl.GroupSummaries>
        <dxGrid:GridControl.TotalSummaries>
            <dxGrid:GridColumnSummary FieldName="Total" Type="Sum"
                                      DisplayFormat= "Total: {0:C0}"/>
        </dxGrid:GridControl.TotalSummaries>
    </dxGrid:GridControl>
</ContentPage>
```

## See Also

IWinPhoneRefreshCustomizer Interface
IWinPhoneRefreshCustomizer Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# LightTheme Class

Provides access to the **Light** grid theme.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public class LightTheme : ThemeBase
```

### Remarks

**Light** is one of the predefined GridControl themes. The Themes class stores available theme names. To apply the **Light** theme to a grid, assign the **Themes.Light** field to the ThemeManager.ThemeName property.

### Inheritance Hierarchy

System.Object
  ThemeBase
    **LightTheme**

### See Also

LightTheme Members
DevExpress.Mobile.DataGrid.Theme Namespace
DarkTheme
ThemeManager

© 2018 Developer Express Inc.

# LightTheme Members

Provides access to the **Light** grid theme.

The following tables list the members exposed by the LightTheme type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | LightTheme | Initializes a new instance of the LightTheme class. |

### Public Properties

| | Name | Description |
|---|---|---|
| | AndroidRefreshCustomizer | Customizes the color of the Android refreshing bar. (Inherited from **ThemeBase**) |
| | AutoFilterPanelCustomizer | Customizes the Auto-Filter panel. (Inherited from **ThemeBase**) |
| | CellCustomizer | Customizes grid cell's visual attributes such as font, background color and so on. (Inherited from **ThemeBase**) |
| | ColumnChooserCustomizer | Customizes the Column Chooser. (Inherited from **ThemeBase**) |
| | DialogFormCustomizer | Customizes both the Column Chooser and the Popup Edit Form. (Inherited from **ThemeBase**) |
| | EditorCustomizer | Customizes editors located within the Auto-Filter panel. (Inherited from **ThemeBase**) |
| | FilterPanelCustomizer | Customizes the filter panel. (Inherited from **ThemeBase**) |
| | GridControlCustomizer | Customizes GridControl's visual attributes (such as background and border colors, border thickness and so on). (Inherited from **ThemeBase**) |
| | GroupCustomizer | Customizes group rows. (Inherited from **ThemeBase**) |
| | HeaderCustomizer | Customizes GridControl header's visual attributes (such as font, background and border color), the arrows shown while sorting grid values and so on. (Inherited from **ThemeBase**) |
| | IOSRefreshCustomizer | Customizes the color of the iOS refreshing icon. (Inherited from **ThemeBase**) |
| | LoadMoreViewCustomizer | Customizes the Load More panel shown when an end-user scrolls to the bottom of the grid. (Inherited from **ThemeBase**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | Name | Gets the name of the applied grid theme. (Inherited from **ThemeBase**) |
| | NewItemRowCustomizer | Customizes the New Item Row. (Inherited from **ThemeBase**) |
| | SaveCancelButtonsPanelCustomizer | Customizes the Apply/Cancel panel that appears when you either add a new row to a grid or edit grid data with the **Inplace** row edit mode. (Inherited from **ThemeBase**) |
| | TotalSummaryCustomizer | Customizes the total summaries appearance. (Inherited from **ThemeBase**) |
| | WinPhoneRefreshCustomizer | Customizes the color of the Windows Phone refreshing bar. (Inherited from **ThemeBase**) |

**Public Methods**

| | Name | Description |
|---|---|---|
| S | CreateDefaultFont | Overloaded. Creates and returns a **ThemeFontAttributes** object with the specified font attributes and color. (Inherited from **ThemeBase**) |
| S | CreateDefaultFontForFormEditors | Creates and returns a **ThemeFontAttributes** object with the default font attributes and the specified color. (Inherited from **ThemeBase**) |
| S | CreateDefaultPadding | Creates and returns the **Thickness** object that corresponds to the default paddings. (Inherited from **ThemeBase**) |
| S | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

**See Also**
LightTheme Members
DevExpress.Mobile.DataGrid.Theme Namespace
DarkTheme
ThemeManager

© 2018 Developer Express Inc.

# LightTheme Constructor

Initializes a new instance of the LightTheme class.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**
**C#**
```
public LightTheme()
```

**See Also**
LightTheme Class
LightTheme Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# ThemeBase Class

Represents the base class for the GridControl themes.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public abstract class ThemeBase : object
```

### Remarks

The **ThemeBase** class serves as a base for the LightTheme and DarkTheme classes. Use these classes to access the predefined grid themes (**Light** or **Dark** respectively) and customize their appearance. For more information on customizing grid themes, see the example below.

### Example

| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T297553. | |

This example demonstrates how to customize a theme used to paint a GridControl.

To apply a theme to a **GridControl**, assign its name to the ThemeManager.ThemeName property. The Themes class stores available theme names.

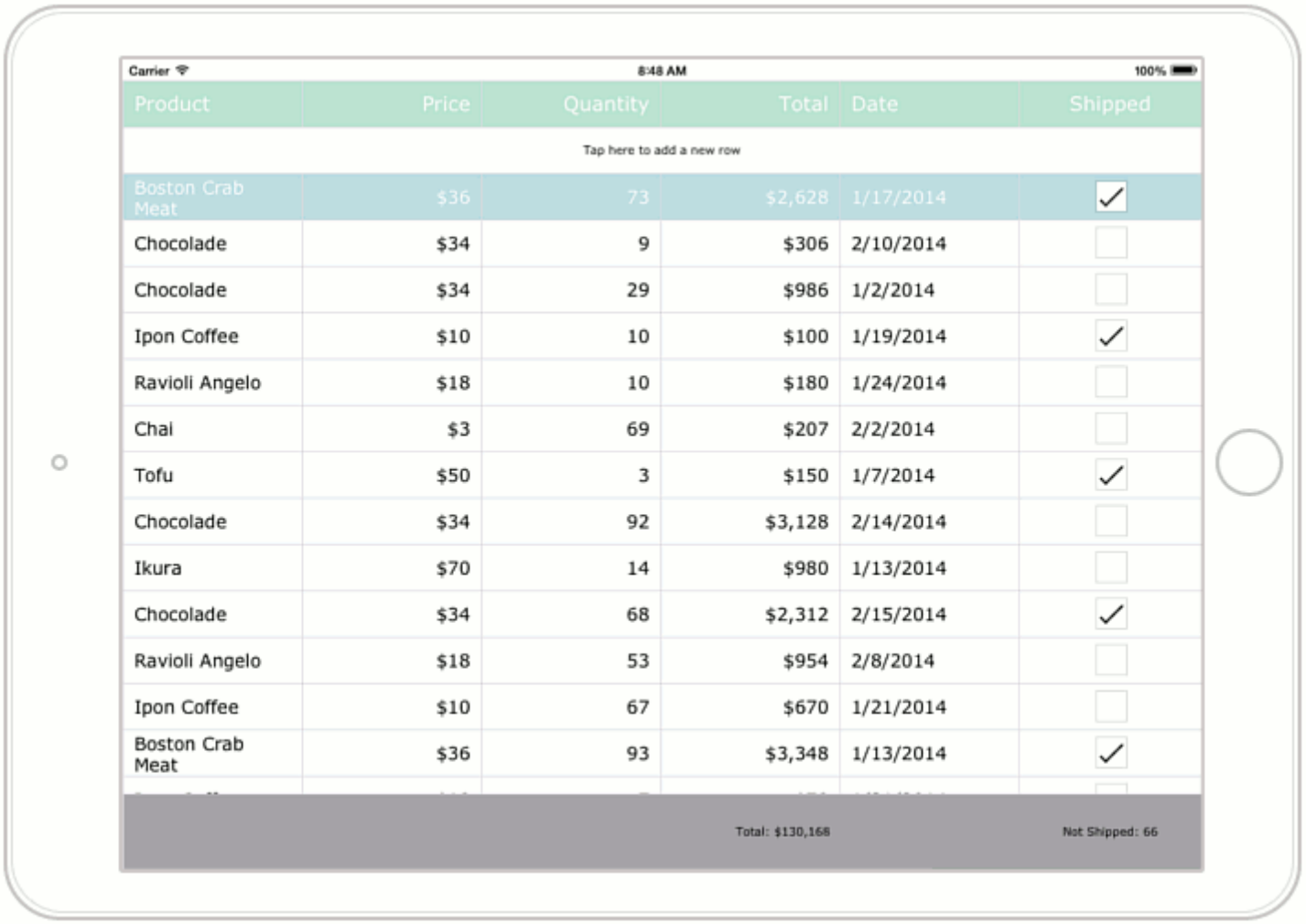
The ThemeBase class's properties provide access to different customizers, which you can adjust to change appearance settings (for example, font attributes, border and background colors, etc.) of the grid's corresponding visual elements (such as data cells, group rows, filter panel, etc.). In this example, the following customizers are used.

- HeaderCustomizer
- CellCustomizer
- TotalSummaryCustomizer
- NewItemRowCustomizer

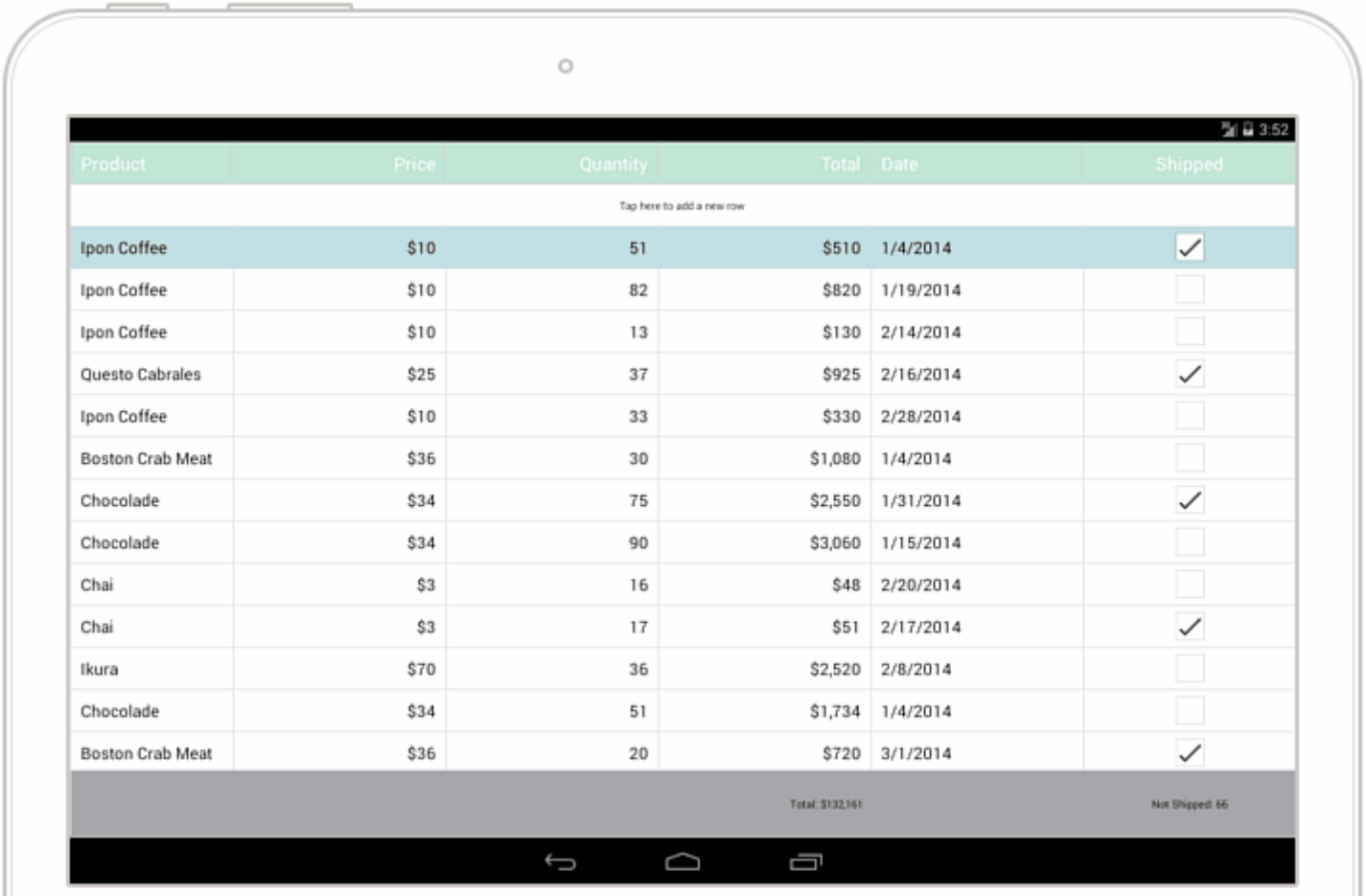
| Important | |
|---|---|
| To apply the customization, call the ThemeManager.RefreshTheme method. | |

This example customizes the Light theme as it is shown in the image below.

© 2018 Developer Express Inc.




| Product | Price | Quantity | Total | Date | Shipped |
|---|---|---|---|---|---|
| Tap here to add a new row | | | | | |
| Boston Crab Meat | $36 | 73 | $2,628 | 1/17/2014 | ✓ |
| Chocolade | $34 | 9 | $306 | 2/10/2014 | |
| Chocolade | $34 | 29 | $986 | 1/2/2014 | |
| Ipon Coffee | $10 | 10 | $100 | 1/19/2014 | ✓ |
| Ravioli Angelo | $18 | 10 | $180 | 1/24/2014 | |
| Chai | $3 | 69 | $207 | 2/2/2014 | |
| Tofu | $50 | 3 | $150 | 1/7/2014 | ✓ |
| Chocolade | $34 | 92 | $3,128 | 2/14/2014 | |
| Ikura | $70 | 14 | $980 | 1/13/2014 | |
| Chocolade | $34 | 68 | $2,312 | 2/15/2014 | ✓ |
| Ravioli Angelo | $18 | 53 | $954 | 2/8/2014 | |
| Ipon Coffee | $10 | 67 | $670 | 1/21/2014 | |
| Boston Crab Meat | $36 | 93 | $3,348 | 1/13/2014 | ✓ |
| | | | Total: $130,168 | | Not Shipped: 66 |




| Product | Price | Quantity | Total | Date | Shipped |
|---|---|---|---|---|---|
| Tap here to add a new row | | | | | |
| Ipon Coffee | $10 | 51 | $510 | 1/4/2014 | ✓ |
| Ipon Coffee | $10 | 82 | $820 | 1/19/2014 | |
| Ipon Coffee | $10 | 13 | $130 | 2/14/2014 | |
| Questo Cabrales | $25 | 37 | $925 | 2/16/2014 | ✓ |
| Ipon Coffee | $10 | 33 | $330 | 2/28/2014 | |
| Boston Crab Meat | $36 | 30 | $1,080 | 1/4/2014 | |
| Chocolade | $34 | 75 | $2,550 | 1/31/2014 | ✓ |
| Chocolade | $34 | 90 | $3,060 | 1/15/2014 | |
| Chai | $3 | 16 | $48 | 2/20/2014 | |
| Chai | $3 | 17 | $51 | 2/17/2014 | ✓ |
| Ikura | $70 | 36 | $2,520 | 2/8/2014 | |
| Chocolade | $34 | 51 | $1,734 | 1/4/2014 | |
| Boston Crab Meat | $36 | 20 | $720 | 3/1/2014 | ✓ |
| | | | Total: $132,161 | | Not Shipped: 66 |

C#

```
 (MainPage.xaml.cs)
using System;
using System.Collections.Generic;
using Xamarin.Forms;
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.DataGrid.Theme;
namespace ThemeCustomizing {
    public partial class MainPage : ContentPage {
        int count = 0;
        public MainPage() {
            InitializeComponent();
            TestOrdersRepository model = new TestOrdersRepository();
            BindingContext = model;
            ThemeManager.ThemeName = Themes.Light;
            // Header customization.
            ThemeManager.Theme.HeaderCustomizer.BackgroundColor = Color.FromRgb(187, 228, 208);
            ThemeFontAttributes myFont = new ThemeFontAttributes("Verdana",
                                         ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Large),
                                         FontAttributes.None, Color.White);
            ThemeManager.Theme.HeaderCustomizer.Font = myFont;
            // Cell customization.
            ThemeManager.Theme.CellCustomizer.SelectionColor = Color.FromRgb(186, 220, 225);
            ThemeFontAttributes myFont1 = new ThemeFontAttributes("Verdana",
                                         ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Medium),
                                         FontAttributes.None, Color.Black);
            ThemeManager.Theme.CellCustomizer.Font = myFont1;
            // Various customization.
            ThemeManager.Theme.TotalSumaryCustomizer.BackgroundColor = Color.FromRgb(163, 162, 168);
            ThemeFontAttributes myFont2 = new ThemeFontAttributes("Verdana",
                                 ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Default),
                                 FontAttributes.None, Color.Black);
            ThemeManager.Theme.TotalSummaryCustomizer.Font = myFont2;
            ThemeManager.Theme.NewItemRowCustomizer.Font = myFont2;
            ThemeManager.RefreshTheme();
        }
        void OnCalculateCustomSummary(object sender, CustomSummaryEventArgs e) {
            if (e.FieldName.ToString() == "Shipped")
            if (e.IsTotalSummary) {
                if (e.SummaryProcess == CustomSummaryProcess.Calculate) {
                    if (!(bool)e.FieldValue)
                        count++;
                    e.TotalValue = count;
                }
            }
        }
    }
}
```

**Inheritance Hierarchy**

System.Object
  **ThemeBase**
    Derived classes

**See Also**
ThemeBase Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# ThemeBase Members

Represents the base class for the GridControl themes.

The following tables list the members exposed by the ThemeBase type.

### Public Properties

| | Name | Description |
|---|---|---|
| | AndroidRefreshCustomizer | Customizes the color of the Android refreshing bar. |
| | AutoFilterPanelCustomizer | Customizes the Auto-Filter panel. |
| | CellCustomizer | Customizes grid cell's visual attributes such as font, background color and so on. |
| | ColumnChooserCustomizer | Customizes the Column Chooser. |
| | DialogFormCustomizer | Customizes both the Column Chooser and the Popup Edit Form. |
| | EditorCustomizer | Customizes editors located within the Auto-Filter panel. |
| | FilterPanelCustomizer | Customizes the filter panel. |
| | GridControlCustomizer | Customizes GridControl's visual attributes (such as background and border colors, border thickness and so on). |
| | GroupCustomizer | Customizes group rows. |
| | HeaderCustomizer | Customizes GridControl header's visual attributes (such as font, background and border color), the arrows shown while sorting grid values and so on. |
| | IOSRefreshCustomizer | Customizes the color of the iOS refreshing icon. |
| | LoadMoreViewCustomizer | Customizes the Load More panel shown when an end-user scrolls to the bottom of the grid. |
| | Name | Gets the name of the applied grid theme. |
| | NewItemRowCustomizer | Customizes the New Item Row. |
| | SaveCancelButtonsPanelCustomizer | Customizes the Apply/Cancel panel that appears when you either add a new row to a grid or edit grid data with the **Inplace** row edit mode. |
| | TotalSummaryCustomizer | Customizes the total summaries appearance. |
| | WinPhoneRefreshCustomizer | Customizes the color of the Windows Phone refreshing bar. |

### Public Methods

© 2018 Developer Express Inc.

| | Name | Description |
|---|---|---|
| | CreateDefaultFont | Overloaded. Creates and returns a **ThemeFontAttributes** object with the default font attributes and the specified color. |
| | CreateDefaultFontForFormEditors | Creates and returns a **ThemeFontAttributes** object with the default font attributes and the specified color. |
| | CreateDefaultPadding | Creates and returns the **Thickness** object that corresponds to the default paddings. |
| | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

**See Also**
ThemeBase Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# ThemeBase Properties

Represents the base class for the GridControl themes.

The following tables list the members exposed by the ThemeBase type.

### Public Properties

| | Name | Description |
|---|---|---|
| | AndroidRefreshCustomizer | Customizes the color of the Android refreshing bar. |
| | AutoFilterPanelCustomizer | Customizes the Auto-Filter panel. |
| | CellCustomizer | Customizes grid cell's visual attributes such as font, background color and so on. |
| | ColumnChooserCustomizer | Customizes the Column Chooser. |
| | DialogFormCustomizer | Customizes both the Column Chooser and the Popup Edit Form. |
| | EditorCustomizer | Customizes editors located within the Auto-Filter panel. |
| | FilterPanelCustomizer | Customizes the filter panel. |
| | GridControlCustomizer | Customizes GridControl's visual attributes (such as background and border colors, border thickness and so on). |
| | GroupCustomizer | Customizes group rows. |
| | HeaderCustomizer | Customizes GridControl header's visual attributes (such as font, background and border color), the arrows shown while sorting grid values and so on. |
| | IOSRefreshCustomizer | Customizes the color of the iOS refreshing icon. |
| | LoadMoreViewCustomizer | Customizes the Load More panel shown when an end-user scrolls to the bottom of the grid. |
| | Name | Gets the name of the applied grid theme. |
| | NewItemRowCustomizer | Customizes the New Item Row. |
| | SaveCancelButtonsPanelCustomizer | Customizes the Apply/Cancel panel that appears when you either add a new row to a grid or edit grid data with the **Inplace** row edit mode. |
| | TotalSummaryCustomizer | Customizes the total summaries appearance. |
| | WinPhoneRefreshCustomizer | Customizes the color of the Windows Phone refreshing bar. |

### Public Methods

© 2018 Developer Express Inc.

| | Name | Description |
|---|---|---|
| | CreateDefaultFont | Overloaded. Creates and returns a **ThemeFontAttributes** object with the default font attributes and the specified color. |
| | CreateDefaultFontForFormEditors | Creates and returns a **ThemeFontAttributes** object with the default font attributes and the specified color. |
| | CreateDefaultPadding | Creates and returns the **Thickness** object that corresponds to the default paddings. |
| | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

**See Also**
ThemeBase Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# AndroidRefreshCustomizer Property

Customizes the color of the Android refreshing bar.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public IAndroidRefreshCustomizer AndroidRefreshCustomizer { get; set; }
```

### Property Value

An object of the type which implements the **IAndroidRefreshCustomizer** interface.

### Remarks

To enable the pull-to-refresh functionality, follow the steps from the How to: Implement Pull-to-Refresh document. After the feature has been added to a grid, you can customize the Android refreshing bar via the **AndroidRefreshCustomizer** property.

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

### See Also

ThemeBase Class
ThemeBase Members
DevExpress.Mobile.DataGrid.Theme Namespace
IOSRefreshCustomizer

© 2018 Developer Express Inc.

# AutoFilterPanelCustomizer Property

Customizes the Auto-Filter panel.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public IAutoFilterPanelCustomizer AutoFilterPanelCustomizer { get; set; }
```

### Property Value

An object of the type which implements the **IAutoFilterPanelCustomizer** interface.

### Remarks

To enable filtering data by columns, follow the steps from the Lesson 7 - Filter Data document. After the Auto-Filter panel is added to a grid, you can customize it using the **AutoFilterPanelCustomizer** property.

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** | |
|---|---|
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. | |

### See Also

ThemeBase Class
ThemeBase Members
DevExpress.Mobile.DataGrid.Theme Namespace
EditorCustomizer
FilterPanelCustomizer

© 2018 Developer Express Inc.

# CellCustomizer Property

Customizes grid cell's visual attributes such as font, background color and so on.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public ICellCustomizer CellCustomizer { get; set; }
```

### Property Value

An object of the type which implements the **ICellCustomizer** interface.

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

### Example

| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T297553. | |

This example demonstrates how to customize a theme used to paint a GridControl.

To apply a theme to a **GridControl**, assign its name to the ThemeManager.ThemeName property. The Themes class stores available theme names.

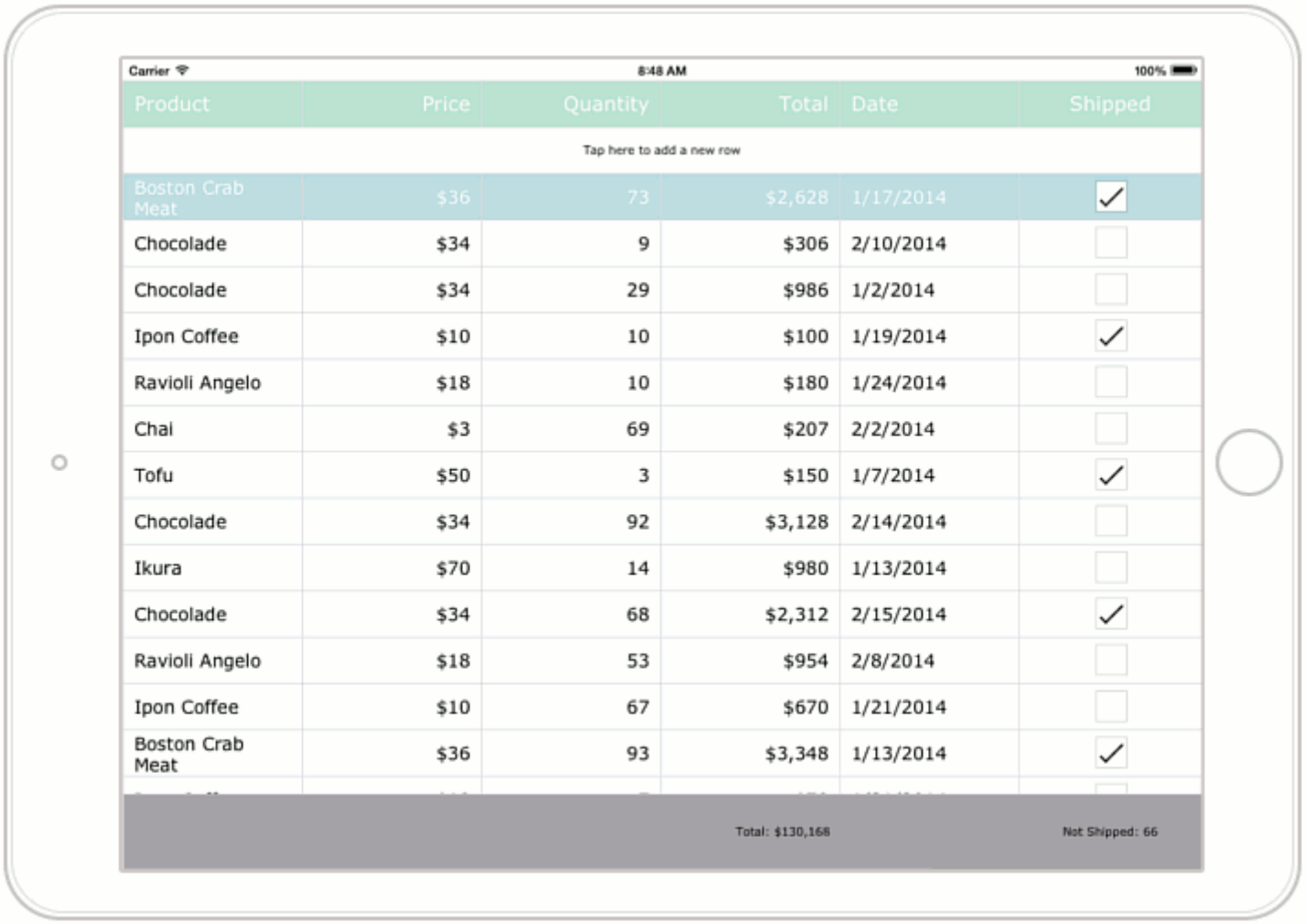
The ThemeBase class's properties provide access to different customizers, which you can adjust to change appearance settings (for example, font attributes, border and background colors, etc.) of the grid's corresponding visual elements (such as data cells, group rows, filter panel, etc.). In this example, the following customizers are used.

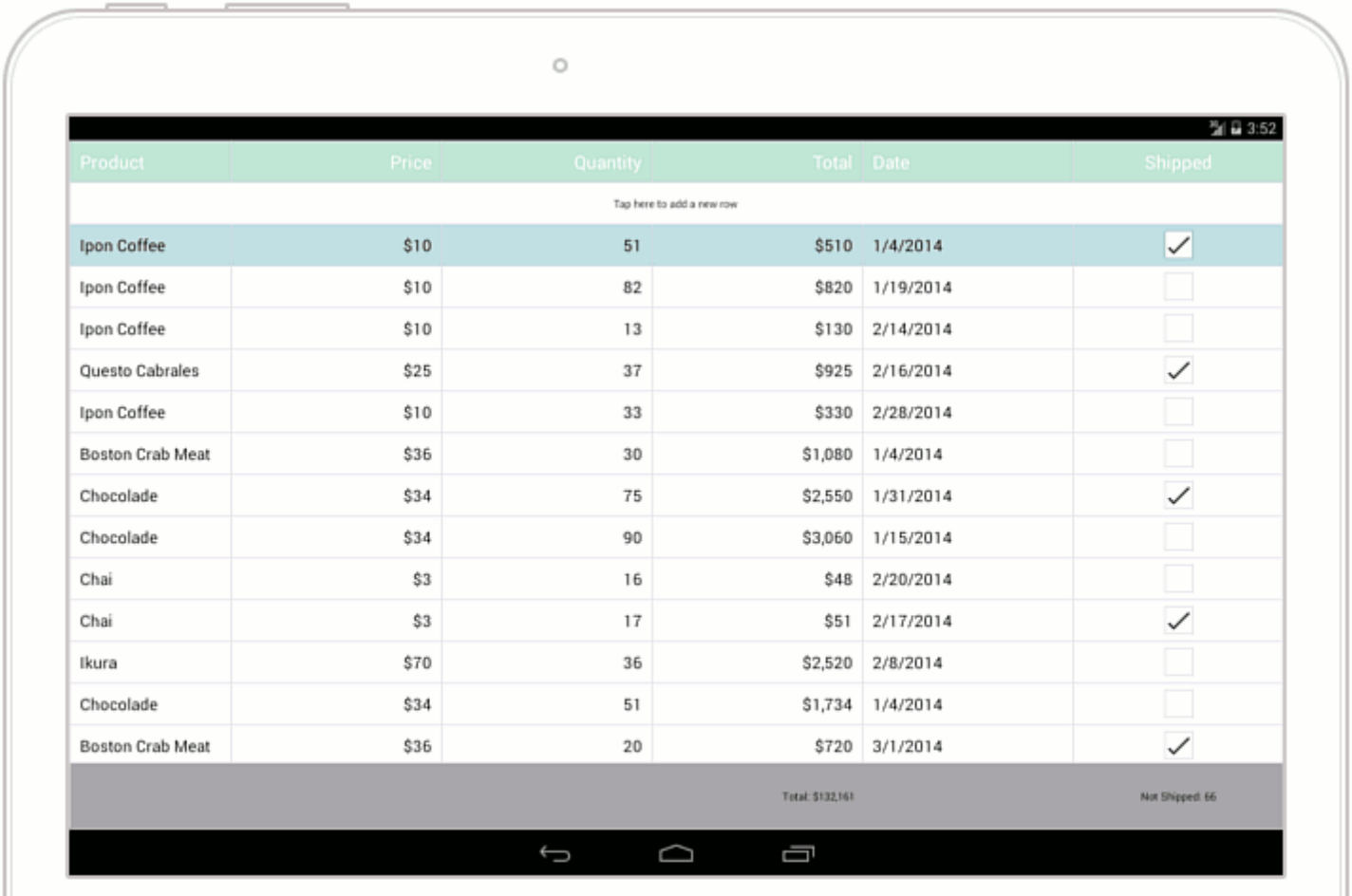
- HeaderCustomizer
- CellCustomizer
- TotalSummaryCustomizer
- NewItemRowCustomizer

| Important | |
|---|---|
| To apply the customization, call the ThemeManager.RefreshTheme method. | |

This example customizes the Light theme as it is shown in the image below.

© 2018 Developer Express Inc.


Carrier
8:48 AM
100%
Product
Price
Quantity
Total
Date
Shipped
Tap here to add a new row
Boston Crab Meat $36 73 $2,628 1/17/2014
Chocolade $34 9 $306 2/10/2014
Chocolade $34 29 $986 1/2/2014
Ipon Coffee $10 10 $100 1/19/2014
Ravioli Angelo $18 10 $180 1/24/2014
Chai $3 69 $207 2/2/2014
Tofu $50 3 $150 1/7/2014
Chocolade $34 92 $3,128 2/14/2014
Ikura $70 14 $980 1/13/2014
Chocolade $34 68 $2,312 2/15/2014
Ravioli Angelo $18 53 $954 2/8/2014
Ipon Coffee $10 67 $670 1/21/2014
Boston Crab Meat $36 93 $3,348 1/13/2014
Total: $130,168
Not Shipped: 66


3:52
Product
Price
Quantity
Total
Date
Shipped
Tap here to add a new row
Ipon Coffee $10 51 $510 1/4/2014
Ipon Coffee $10 82 $820 1/19/2014
Ipon Coffee $10 13 $130 2/14/2014
Questo Cabrales $25 37 $925 2/16/2014
Ipon Coffee $10 33 $330 2/28/2014
Boston Crab Meat $36 30 $1,080 1/4/2014
Chocolade $34 75 $2,550 1/31/2014
Chocolade $34 90 $3,060 1/15/2014
Chai $3 16 $48 2/20/2014
Chai $3 17 $51 2/17/2014
Ikura $70 36 $2,520 2/8/2014
Chocolade $34 51 $1,734 1/4/2014
Boston Crab Meat $36 20 $720 3/1/2014
Total: $132,161
Not Shipped: 66

**C#**

```
 (MainPage.xaml.cs)
using System;
using System.Collections.Generic;
using Xamarin.Forms;
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.DataGrid.Theme;
namespace ThemeCustomizing {
    public partial class MainPage : ContentPage {
        int count = 0;
        public MainPage() {
            InitializeComponent();
            TestOrdersRepository model = new TestOrdersRepository();
            BindingContext = model;
            ThemeManager.ThemeName = Themes.Light;
            // Header customization.
            ThemeManager.Theme.HeaderCustomizer.BackgroundColor = Color.FromRgb(187, 228, 208);
            ThemeFontAttributes myFont = new ThemeFontAttributes("Verdana",
                                         ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Large),
                                         FontAttributes.None, Color.White);
            ThemeManager.Theme.HeaderCustomizer.Font = myFont;
            // Cell customization.
            ThemeManager.Theme.CellCustomizer.SelectionColor = Color.FromRgb(186, 220, 225);
            ThemeFontAttributes myFont1 = new ThemeFontAttributes("Verdana",
                                         ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Medium),
                                         FontAttributes.None, Color.Black);
            ThemeManager.Theme.CellCustomizer.Font = myFont1;
            // Various customization.
            ThemeManager.Theme.TotalSumaryCustomizer.BackgroundColor = Color.FromRgb(163, 162, 168);
            ThemeFontAttributes myFont2 = new ThemeFontAttributes("Verdana",
                                 ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Default),
                                 FontAttributes.None, Color.Black);
            ThemeManager.Theme.TotalSummaryCustomizer.Font = myFont2;
            ThemeManager.Theme.NewItemRowCustomizer.Font = myFont2;
            ThemeManager.RefreshTheme();
        }
        void OnCalculateCustomSummary(object sender, CustomSummaryEventArgs e) {
            if (e.FieldName.ToString() == "Shipped")
            if (e.IsTotalSummary) {
                if (e.SummaryProcess == CustomSummaryProcess.Calculate) {
                    if (!(bool)e.FieldValue)
                        count++;
                    e.TotalValue = count;
                }
            }
        }
    }
}
```

**See Also**

ThemeBase Class
ThemeBase Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# ColumnChooserCustomizer Property

Customizes the Column Chooser.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public IColumnChooserCustomizer ColumnChooserCustomizer { get; set; }
```

### Property Value

An object of the type which implements the **IColumnChooserCustomizer** interface.

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

### See Also

ThemeBase Class
ThemeBase Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# DialogFormCustomizer Property

Customizes both the Column Chooser and the Popup Edit Form.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public IDialogFormCustomizer DialogFormCustomizer { get; set; }
```

### Property Value

An object of the type which implements the **IDialogFormCustomizer** interface.

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

### See Also

ThemeBase Class
ThemeBase Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# EditorCustomizer Property

Customizes editors located within the Auto-Filter panel.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public IEditorCustomizer EditorCustomizer { get; set; }
```

### Property Value

An object of the type which implements the **IEditorCustomizer** interface.

### Remarks

To enable filtering data by columns, follow the steps from the Lesson 7 - Filter Data document. After the Auto-Filter panel is added to a grid, you can customize it's editors using the **EditorCustomizer** property.

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** | |
|---|---|
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. | |

### See Also

ThemeBase Class
ThemeBase Members
DevExpress.Mobile.DataGrid.Theme Namespace
AutoFilterPanelCustomizer
FilterPanelCustomizer

© 2018 Developer Express Inc.

# FilterPanelCustomizer Property

Customizes the filter panel.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public IFilterPanelCustomizer FilterPanelCustomizer { get; set; }
```

### Property Value

An object of the type which implements the **IFilterPanelCustomizer** interface.

### Remarks

To enable filtering data by columns, follow the steps from the Lesson 7 - Filter Data document. After you apply a filter to a grid, the filter panel appears by default at the bottom of the grid. You can customize this panel using the **FilterPanelCustomizer** property.

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** | |
|---|---|
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. | |

### See Also

ThemeBase Class
ThemeBase Members
DevExpress.Mobile.DataGrid.Theme Namespace
AutoFilterPanelCustomizer
EditorCustomizer

© 2018 Developer Express Inc.

# GridControlCustomizer Property

Customizes GridControl's visual attributes (such as background and border colors, border thickness and so on).

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public IGridControlCustomizer GridControlCustomizer { get; set; }
```

### Property Value

An object of the type which implements the **IGridControlCustomizer** interface.

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

### See Also

ThemeBase Class
ThemeBase Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# GroupCustomizer Property

Customizes group rows.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public IGroupCustomizer GroupCustomizer { get; set; }
```

### Property Value

An object of the type which implements the **IGroupCustomizer** interface.

### Remarks

To enable grouping data by columns, follow the steps from the Lesson 4 - Group Data document. After the functionality is added to a grid, you can customize group rows using the **GroupCustomizer** property.

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** | |
|---|---|
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. | |

### See Also

ThemeBase Class
ThemeBase Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# HeaderCustomizer Property

Customizes GridControl header's visual attributes (such as font, background and border color), the arrows shown while sorting grid values and so on.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public IHeaderCustomizer HeaderCustomizer { get; set; }
```

### Property Value

An object of the type which implements the **IHeaderCustomizer** interface.

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

### Example

| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T297553. | |

This example demonstrates how to customize a theme used to paint a GridControl.

To apply a theme to a **GridControl**, assign its name to the ThemeManager.ThemeName property. The Themes class stores available theme names.

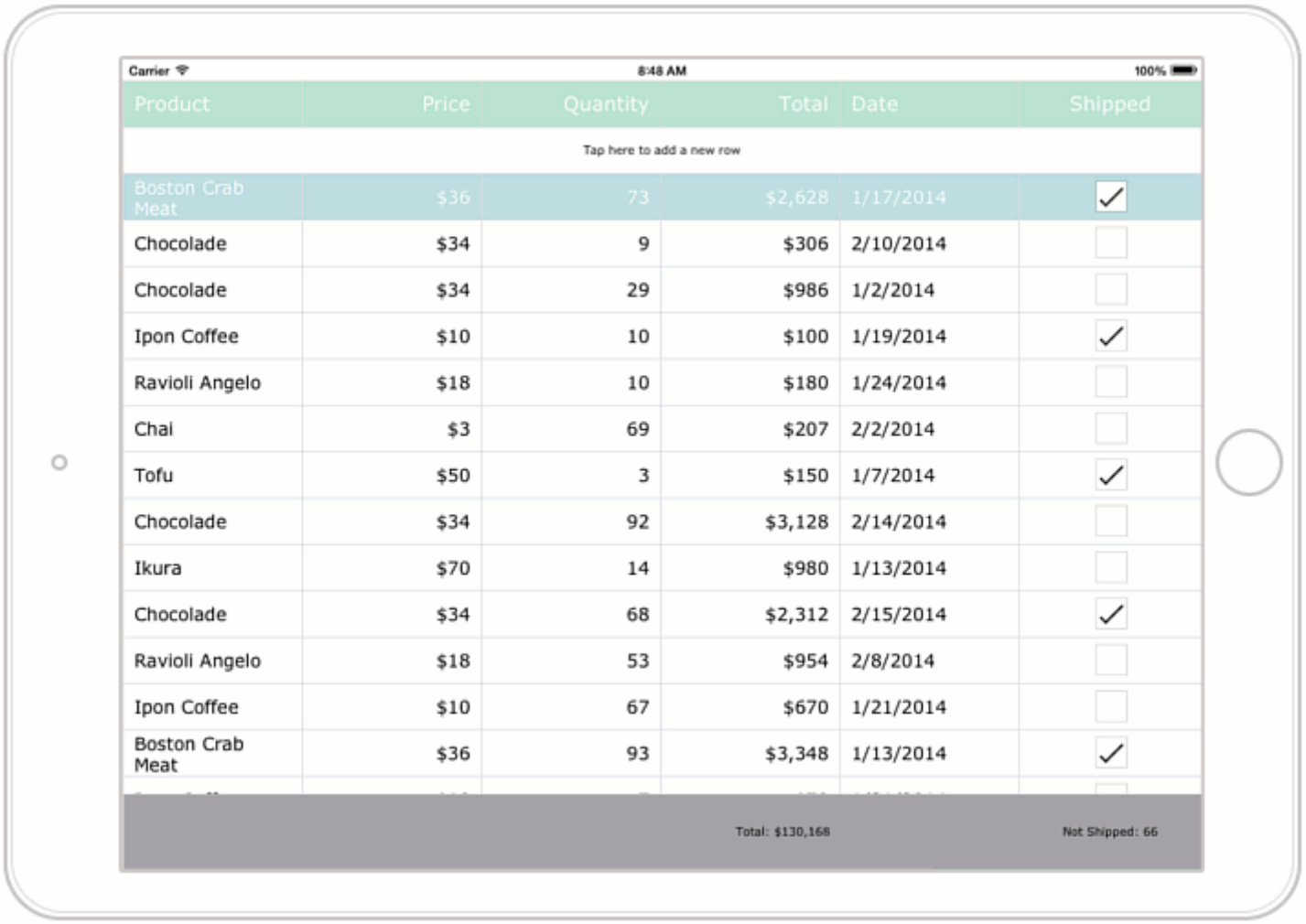
The ThemeBase class's properties provide access to different customizers, which you can adjust to change appearance settings (for example, font attributes, border and background colors, etc.) of the grid's corresponding visual elements (such as data cells, group rows, filter panel, etc.). In this example, the following customizers are used.

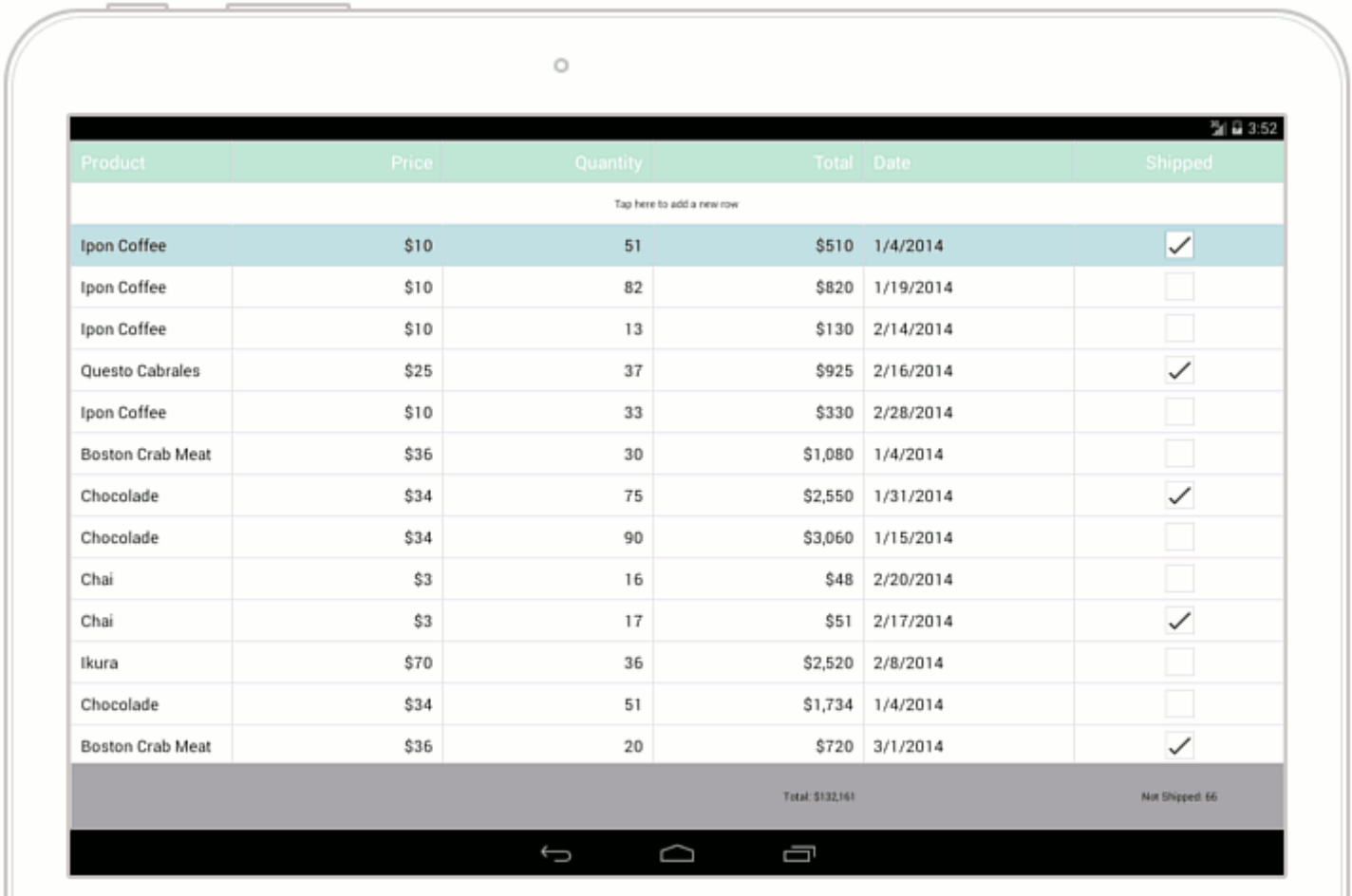
- HeaderCustomizer
- CellCustomizer
- TotalSummaryCustomizer
- NewItemRowCustomizer

| Important | |
|---|---|
| To apply the customization, call the ThemeManager.RefreshTheme method. | |

This example customizes the Light theme as it is shown in the image below.

© 2018 Developer Express Inc.


Carrier
8:48 AM
100%
Product
Price
Quantity
Total
Date
Shipped
Tap here to add a new row
Boston Crab Meat $36 73 $2,628 1/17/2014
Chocolade $34 9 $306 2/10/2014
Chocolade $34 29 $986 1/2/2014
Ipon Coffee $10 10 $100 1/19/2014
Ravioli Angelo $18 10 $180 1/24/2014
Chai $3 69 $207 2/2/2014
Tofu $50 3 $150 1/7/2014
Chocolade $34 92 $3,128 2/14/2014
Ikura $70 14 $980 1/13/2014
Chocolade $34 68 $2,312 2/15/2014
Ravioli Angelo $18 53 $954 2/8/2014
Ipon Coffee $10 67 $670 1/21/2014
Boston Crab Meat $36 93 $3,348 1/13/2014
Total: $130,168
Not Shipped: 66


3:52
Product
Price
Quantity
Total
Date
Shipped
Tap here to add a new row
Ipon Coffee $10 51 $510 1/4/2014
Ipon Coffee $10 82 $820 1/19/2014
Ipon Coffee $10 13 $130 2/14/2014
Questo Cabrales $25 37 $925 2/16/2014
Ipon Coffee $10 33 $330 2/28/2014
Boston Crab Meat $36 30 $1,080 1/4/2014
Chocolade $34 75 $2,550 1/31/2014
Chocolade $34 90 $3,060 1/15/2014
Chai $3 16 $48 2/20/2014
Chai $3 17 $51 2/17/2014
Ikura $70 36 $2,520 2/8/2014
Chocolade $34 51 $1,734 1/4/2014
Boston Crab Meat $36 20 $720 3/1/2014
Total: $132,161
Not Shipped: 66

C#

```
 (MainPage.xaml.cs)
using System;
using System.Collections.Generic;
using Xamarin.Forms;
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.DataGrid.Theme;
namespace ThemeCustomizing {
    public partial class MainPage : ContentPage {
        int count = 0;
        public MainPage() {
            InitializeComponent();
            TestOrdersRepository model = new TestOrdersRepository();
            BindingContext = model;
            ThemeManager.ThemeName = Themes.Light;
            // Header customization.
            ThemeManager.Theme.HeaderCustomizer.BackgroundColor = Color.FromRgb(187, 228, 208);
            ThemeFontAttributes myFont = new ThemeFontAttributes("Verdana",
                                         ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Large),
                                         FontAttributes.None, Color.White);
            ThemeManager.Theme.HeaderCustomizer.Font = myFont;
            // Cell customization.
            ThemeManager.Theme.CellCustomizer.SelectionColor = Color.FromRgb(186, 220, 225);
            ThemeFontAttributes myFont1 = new ThemeFontAttributes("Verdana",
                                         ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Medium),
                                         FontAttributes.None, Color.Black);
            ThemeManager.Theme.CellCustomizer.Font = myFont1;
            // Various customization.
            ThemeManager.Theme.TotalSumaryCustomizer.BackgroundColor = Color.FromRgb(163, 162, 168);
            ThemeFontAttributes myFont2 = new ThemeFontAttributes("Verdana",
                                 ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Default),
                                 FontAttributes.None, Color.Black);
            ThemeManager.Theme.TotalSummaryCustomizer.Font = myFont2;
            ThemeManager.Theme.NewItemRowCustomizer.Font = myFont2;
            ThemeManager.RefreshTheme();
        }
        void OnCalculateCustomSummary(object sender, CustomSummaryEventArgs e) {
            if (e.FieldName.ToString() == "Shipped")
            if (e.IsTotalSummary) {
                if (e.SummaryProcess == CustomSummaryProcess.Calculate) {
                    if (!(bool)e.FieldValue)
                        count++;
                    e.TotalValue = count;
                }
            }
        }
    }
}
```

**See Also**

ThemeBase Class
ThemeBase Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# IOSRefreshCustomizer Property

Customizes the color of the iOS refreshing icon.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public IIOSRefreshCustomizer IOSRefreshCustomizer { get; set; }
```

### Property Value

An object of the type which implements the **IIOSRefreshCustomizer** interface.

### Remarks

To enable the pull-to-refresh functionality, follow the steps from the How to: Implement Pull-to-Refresh document. After the feature has been added to a grid, you can customize the iOS refreshing icon via the **IOSRefreshCustomizer** property.

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

### See Also

ThemeBase Class
ThemeBase Members
DevExpress.Mobile.DataGrid.Theme Namespace
AndroidRefreshCustomizer

© 2018 Developer Express Inc.

# LoadMoreViewCustomizer Property

Customizes the Load More panel shown when an end-user scrolls to the bottom of the grid.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public ILoadMoreViewCustomizer LoadMoreViewCustomizer { get; set; }
```

### Property Value

An object of the type which implements the **ILoadMoreViewCustomizer** interface.

### Remarks

To enable the load-more feature, follow the steps from the How to: Implement Load-More document. After this functionality is added to a grid, you can customize the displayed panel using the **LoadMoreViewCustomizer** property.

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** | |
|---|---|
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. | |

### See Also

ThemeBase Class
ThemeBase Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# Name Property

Gets the name of the applied grid theme.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public string Name { get; }
```

### Property Value

A string that specifies the theme name.

### See Also

ThemeBase Class
ThemeBase Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# NewItemRowCustomizer Property

Customizes the New Item Row.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public INewItemRowCustomizer NewItemRowCustomizer { get; set; }
```

### Property Value

An object of the type which implements the **INewItemRowCustomizer** interface.

### Remarks

To enable the **New Item Row** feature, follow the steps from the Lesson 3 - Add New Item Row document. After this functionality is added to a grid, you can customize the displayed panel using the **NewItemRowCustomizer** property.

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

### Example

| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T297553. | |

This example demonstrates how to customize a theme used to paint a GridControl.

To apply a theme to a **GridControl**, assign its name to the ThemeManager.ThemeName property. The Themes class stores available theme names.

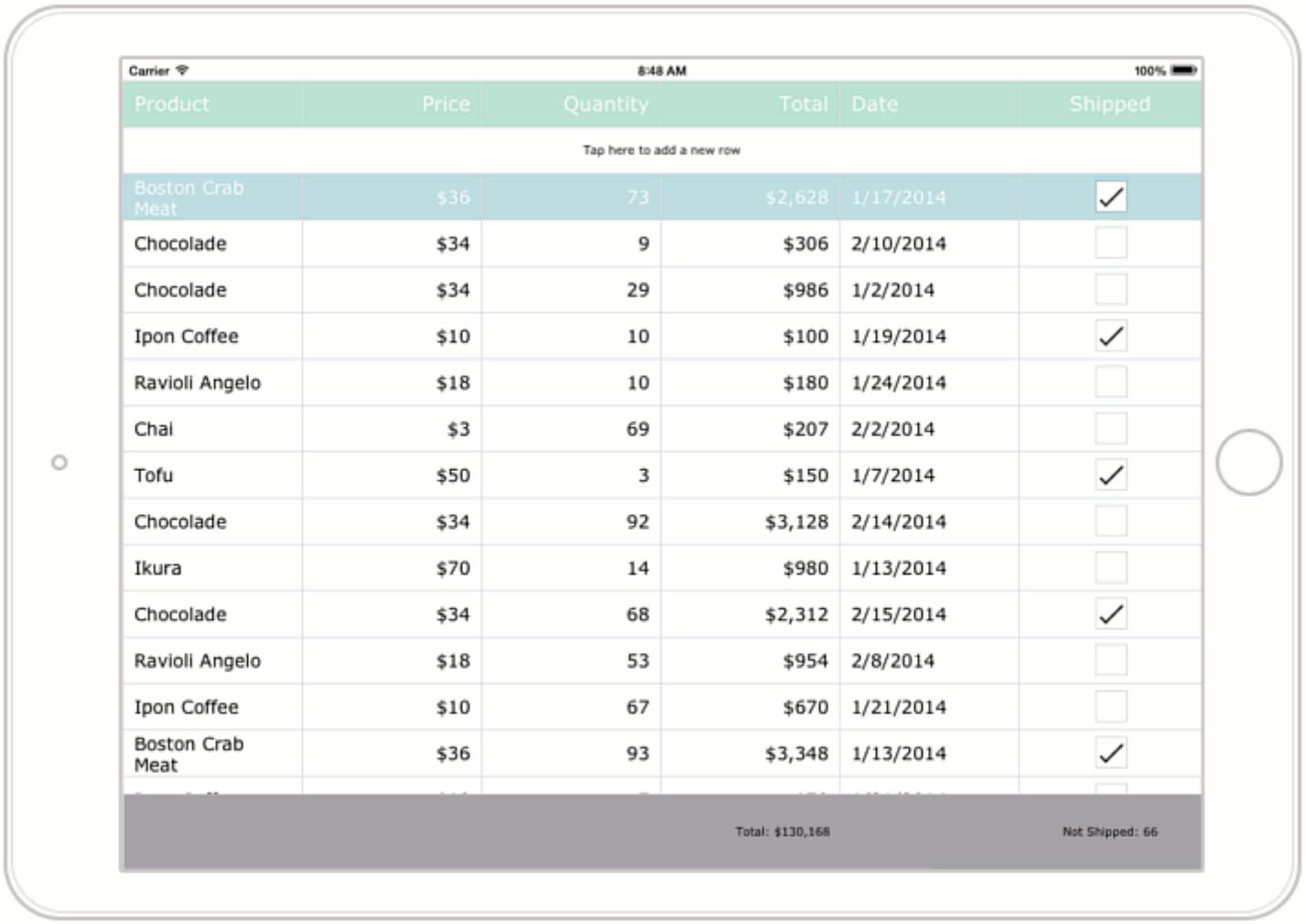
The ThemeBase class's properties provide access to different customizers, which you can adjust to change appearance settings (for example, font attributes, border and background colors, etc.) of the grid's corresponding visual elements (such as data cells, group rows, filter panel, etc.). In this example, the following customizers are used.

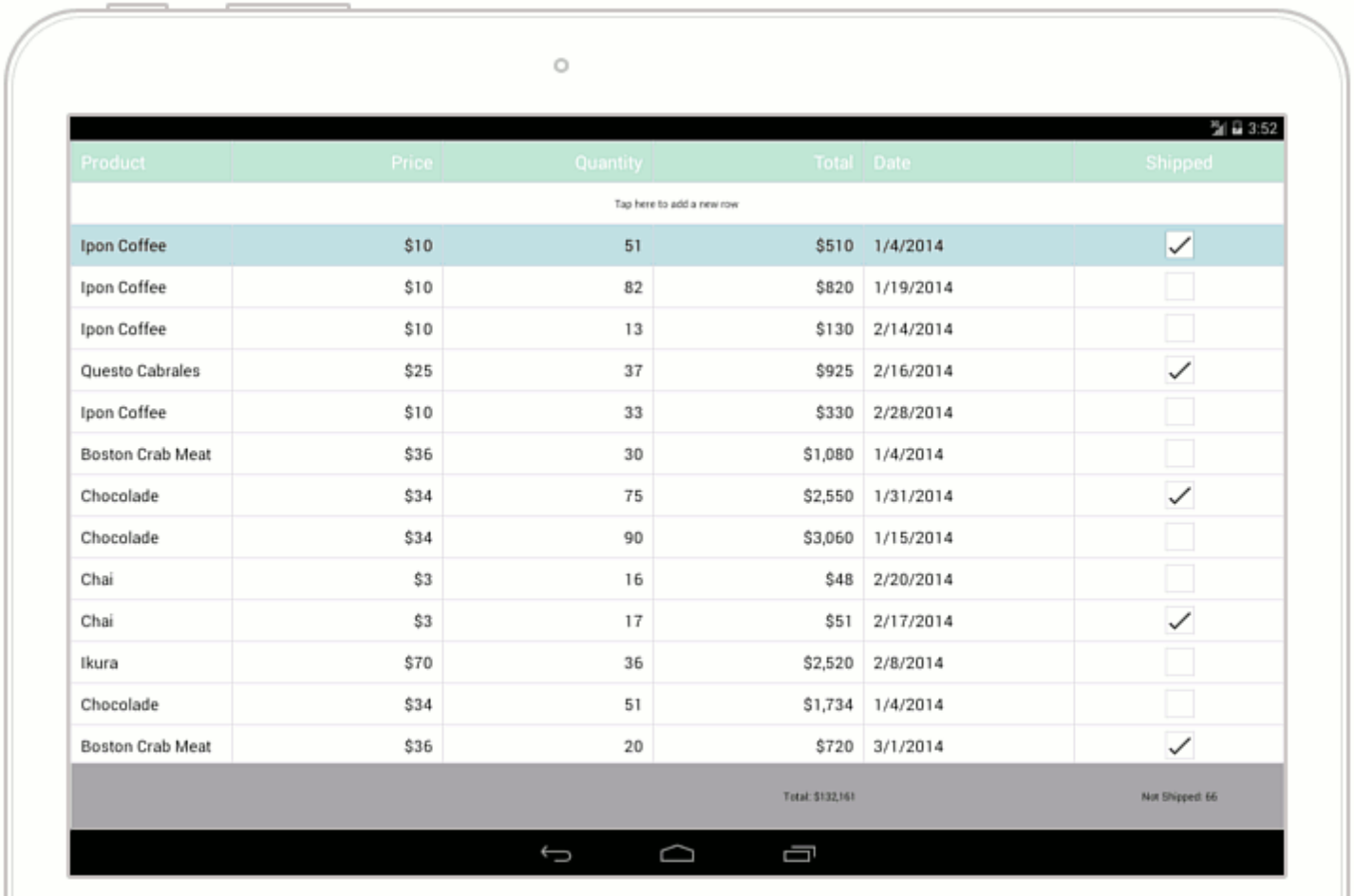
- HeaderCustomizer
- CellCustomizer
- TotalSummaryCustomizer
- NewItemRowCustomizer

| Important | |
|---|---|
| To apply the customization, call the ThemeManager.RefreshTheme method. | |

This example customizes the Light theme as it is shown in the image below.

© 2018 Developer Express Inc.


Carrier
8:48 AM
100%
Product
Price
Quantity
Total
Date
Shipped
Tap here to add a new row
Boston Crab Meat $36 73 $2,628 1/17/2014
Chocolade $34 9 $306 2/10/2014
Chocolade $34 29 $986 1/2/2014
Ipon Coffee $10 10 $100 1/19/2014
Ravioli Angelo $18 10 $180 1/24/2014
Chai $3 69 $207 2/2/2014
Tofu $50 3 $150 1/7/2014
Chocolade $34 92 $3,128 2/14/2014
Ikura $70 14 $980 1/13/2014
Chocolade $34 68 $2,312 2/15/2014
Ravioli Angelo $18 53 $954 2/8/2014
Ipon Coffee $10 67 $670 1/21/2014
Boston Crab Meat $36 93 $3,348 1/13/2014
Total: $130,168
Not Shipped: 66


3:52
Product
Price
Quantity
Total
Date
Shipped
Tap here to add a new row
Ipon Coffee $10 51 $510 1/4/2014
Ipon Coffee $10 82 $820 1/19/2014
Ipon Coffee $10 13 $130 2/14/2014
Questo Cabrales $25 37 $925 2/16/2014
Ipon Coffee $10 33 $330 2/28/2014
Boston Crab Meat $36 30 $1,080 1/4/2014
Chocolade $34 75 $2,550 1/31/2014
Chocolade $34 90 $3,060 1/15/2014
Chai $3 16 $48 2/20/2014
Chai $3 17 $51 2/17/2014
Ikura $70 36 $2,520 2/8/2014
Chocolade $34 51 $1,734 1/4/2014
Boston Crab Meat $36 20 $720 3/1/2014
Total: $132,161
Not Shipped: 66

C#

```
 (MainPage.xaml.cs)
using System;
using System.Collections.Generic;
using Xamarin.Forms;
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.DataGrid.Theme;
namespace ThemeCustomizing {
    public partial class MainPage : ContentPage {
        int count = 0;
        public MainPage() {
            InitializeComponent();
            TestOrdersRepository model = new TestOrdersRepository();
            BindingContext = model;
            ThemeManager.ThemeName = Themes.Light;
            // Header customization.
            ThemeManager.Theme.HeaderCustomizer.BackgroundColor = Color.FromRgb(187, 228, 208);
            ThemeFontAttributes myFont = new ThemeFontAttributes("Verdana",
                                         ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Large),
                                         FontAttributes.None, Color.White);
            ThemeManager.Theme.HeaderCustomizer.Font = myFont;
            // Cell customization.
            ThemeManager.Theme.CellCustomizer.SelectionColor = Color.FromRgb(186, 220, 225);
            ThemeFontAttributes myFont1 = new ThemeFontAttributes("Verdana",
                                         ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Medium),
                                         FontAttributes.None, Color.Black);
            ThemeManager.Theme.CellCustomizer.Font = myFont1;
            // Various customization.
            ThemeManager.Theme.TotalSumaryCustomizer.BackgroundColor = Color.FromRgb(163, 162, 168);
            ThemeFontAttributes myFont2 = new ThemeFontAttributes("Verdana",
                                 ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Default),
                                 FontAttributes.None, Color.Black);
            ThemeManager.Theme.TotalSummaryCustomizer.Font = myFont2;
            ThemeManager.Theme.NewItemRowCustomizer.Font = myFont2;
            ThemeManager.RefreshTheme();
        }
        void OnCalculateCustomSummary(object sender, CustomSummaryEventArgs e) {
            if (e.FieldName.ToString() == "Shipped")
            if (e.IsTotalSummary) {
                if (e.SummaryProcess == CustomSummaryProcess.Calculate) {
                    if (!(bool)e.FieldValue)
                        count++;
                    e.TotalValue = count;
                }
            }
        }
    }
}
```

**See Also**

ThemeBase Class
ThemeBase Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# SaveCancelButtonsPanelCustomizer Property

Customizes the Apply/Cancel panel that appears when you either add a new row to a grid or edit grid data with the **Inplace** row edit mode.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public ISaveCancelButtonsPanelCustomizer SaveCancelButtonsPanelCustomizer { get; set; }
```

### Property Value

An object of the type which implements the **ISaveCancelButtonsPanelCustomizer** interface.

### Remarks

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

### See Also

ThemeBase Class
ThemeBase Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# TotalSummaryCustomizer Property

Customizes the total summaries appearance.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public ITotalSummaryCustomizer TotalSummaryCustomizer { get; set; }
```

### Property Value

An object of the type which implements the **ITotalSummaryCustomizer** interface.

### Remarks

To enable the **total summaries** feature, follow the steps from the Lesson 5 - Calculate Summaries document. After this functionality is added to a grid, you can customize the displayed panel using the **TotalSummaryCustomizer** property.

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

### Example

| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T297553. | |

This example demonstrates how to customize a theme used to paint a GridControl.

To apply a theme to a **GridControl**, assign its name to the ThemeManager.ThemeName property. The Themes class stores available theme names.

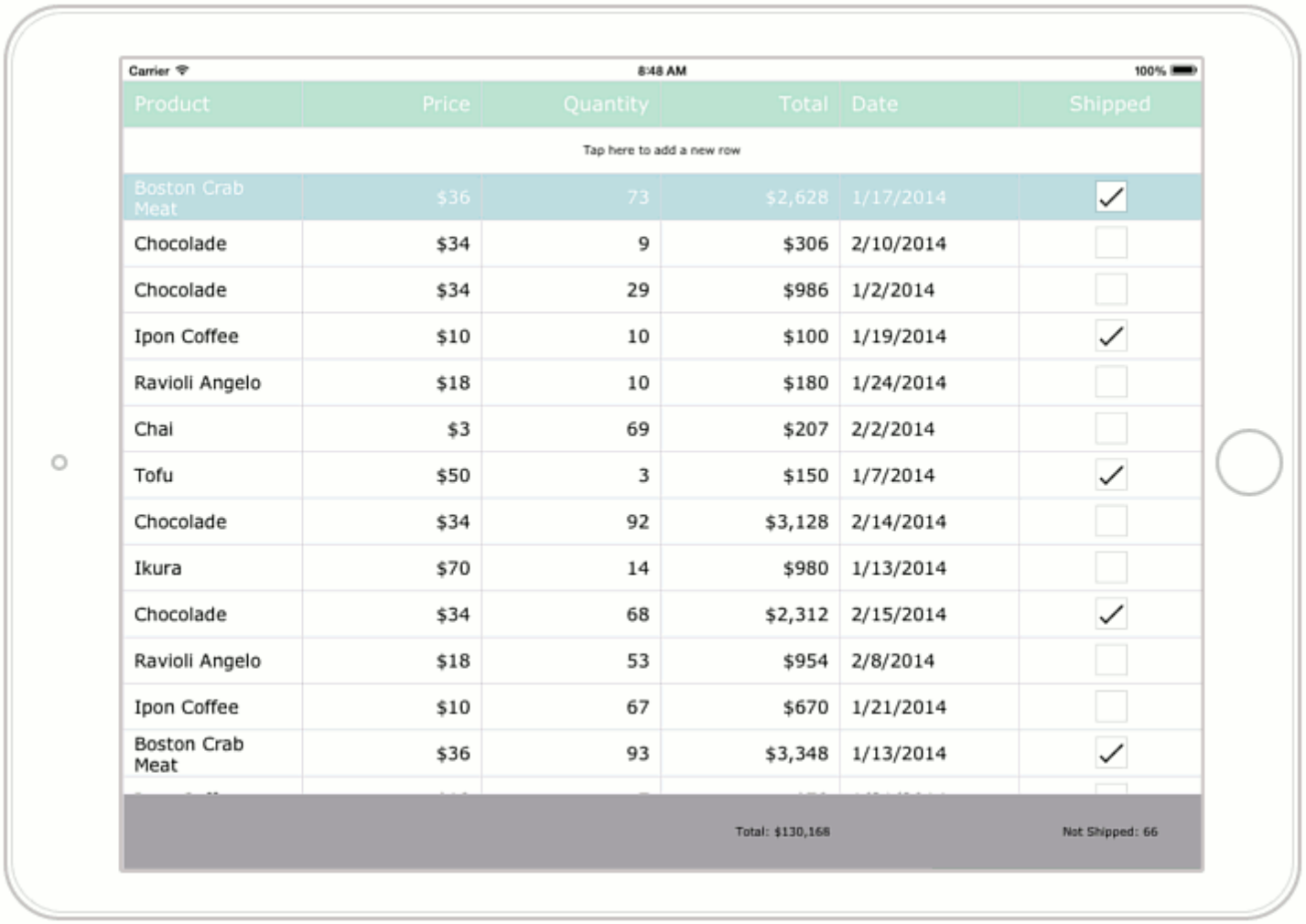
The ThemeBase class's properties provide access to different customizers, which you can adjust to change appearance settings (for example, font attributes, border and background colors, etc.) of the grid's corresponding visual elements (such as data cells, group rows, filter panel, etc.). In this example, the following customizers are used.

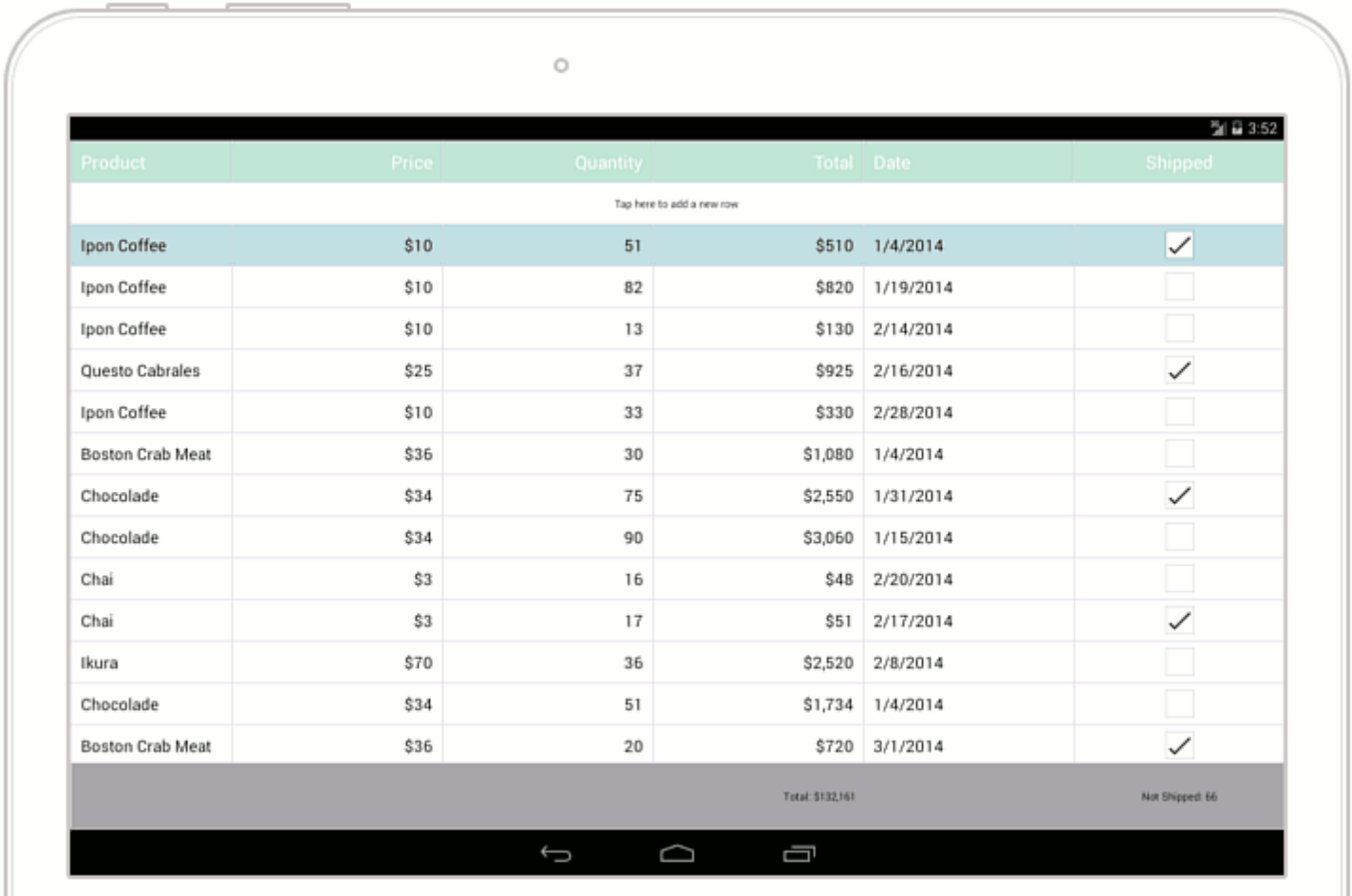
- HeaderCustomizer
- CellCustomizer
- TotalSummaryCustomizer
- NewItemRowCustomizer

| Important | |
|---|---|
| To apply the customization, call the ThemeManager.RefreshTheme method. | |

This example customizes the Light theme as it is shown in the image below.

© 2018 Developer Express Inc.


Carrier
8:48 AM
100%
Product
Price
Quantity
Total
Date
Shipped
Tap here to add a new row
Boston Crab Meat $36 73 $2,628 1/17/2014
Chocolade $34 9 $306 2/10/2014
Chocolade $34 29 $986 1/2/2014
Ipon Coffee $10 10 $100 1/19/2014
Ravioli Angelo $18 10 $180 1/24/2014
Chai $3 69 $207 2/2/2014
Tofu $50 3 $150 1/7/2014
Chocolade $34 92 $3,128 2/14/2014
Ikura $70 14 $980 1/13/2014
Chocolade $34 68 $2,312 2/15/2014
Ravioli Angelo $18 53 $954 2/8/2014
Ipon Coffee $10 67 $670 1/21/2014
Boston Crab Meat $36 93 $3,348 1/13/2014
Total: $130,168
Not Shipped: 66


3:52
Product
Price
Quantity
Total
Date
Shipped
Tap here to add a new row
Ipon Coffee $10 51 $510 1/4/2014
Ipon Coffee $10 82 $820 1/19/2014
Ipon Coffee $10 13 $130 2/14/2014
Questo Cabrales $25 37 $925 2/16/2014
Ipon Coffee $10 33 $330 2/28/2014
Boston Crab Meat $36 30 $1,080 1/4/2014
Chocolade $34 75 $2,550 1/31/2014
Chocolade $34 90 $3,060 1/15/2014
Chai $3 16 $48 2/20/2014
Chai $3 17 $51 2/17/2014
Ikura $70 36 $2,520 2/8/2014
Chocolade $34 51 $1,734 1/4/2014
Boston Crab Meat $36 20 $720 3/1/2014
Total: $132,161
Not Shipped: 66

C#

```
 (MainPage.xaml.cs)
using System;
using System.Collections.Generic;
using Xamarin.Forms;
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.DataGrid.Theme;
namespace ThemeCustomizing {
    public partial class MainPage : ContentPage {
        int count = 0;
        public MainPage() {
            InitializeComponent();
            TestOrdersRepository model = new TestOrdersRepository();
            BindingContext = model;
            ThemeManager.ThemeName = Themes.Light;
            // Header customization.
            ThemeManager.Theme.HeaderCustomizer.BackgroundColor = Color.FromRgb(187, 228, 208);
            ThemeFontAttributes myFont = new ThemeFontAttributes("Verdana",
                                         ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Large),
                                         FontAttributes.None, Color.White);
            ThemeManager.Theme.HeaderCustomizer.Font = myFont;
            // Cell customization.
            ThemeManager.Theme.CellCustomizer.SelectionColor = Color.FromRgb(186, 220, 225);
            ThemeFontAttributes myFont1 = new ThemeFontAttributes("Verdana",
                                         ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Medium),
                                         FontAttributes.None, Color.Black);
            ThemeManager.Theme.CellCustomizer.Font = myFont1;
            // Various customization.
            ThemeManager.Theme.TotalSumaryCustomizer.BackgroundColor = Color.FromRgb(163, 162, 168);
            ThemeFontAttributes myFont2 = new ThemeFontAttributes("Verdana",
                                 ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Default),
                                 FontAttributes.None, Color.Black);
            ThemeManager.Theme.TotalSummaryCustomizer.Font = myFont2;
            ThemeManager.Theme.NewItemRowCustomizer.Font = myFont2;
            ThemeManager.RefreshTheme();
        }
        void OnCalculateCustomSummary(object sender, CustomSummaryEventArgs e) {
            if (e.FieldName.ToString() == "Shipped")
            if (e.IsTotalSummary) {
                if (e.SummaryProcess == CustomSummaryProcess.Calculate) {
                    if (!(bool)e.FieldValue)
                        count++;
                    e.TotalValue = count;
                }
            }
        }
    }
}
```

**See Also**

ThemeBase Class
ThemeBase Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# WinPhoneRefreshCustomizer Property

Customizes the color of the Windows Phone refreshing bar.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public IWinPhoneRefreshCustomizer WinPhoneRefreshCustomizer { get; set; }
```

### Property Value

An object of the type which implements the **IWinPhoneRefreshCustomizer** interface.

### Remarks

To enable the pull-to-refresh functionality, follow the steps from the How to: Implement Pull-to-Refresh document. After the feature has been added to a grid, you can customize the Windows Phone refreshing bar via the **WinPhoneRefreshCustomizer** property.

GridControl provides you with two predefined themes - **Light** and **Dark**. The **Default** theme depends on the mobile platform. For iOS applications, the **Default** theme is **Light**, for Android - is **Dark**. To apply another theme to a grid, use the ThemeManager.ThemeName property. Then, you can customize the applied theme (if needed). For more information, see the How to: Customize Themes document.

| **Important** |
| --- |
| To apply customization to a grid, call the ThemeManager.RefreshTheme method. |

### See Also

ThemeBase Class
ThemeBase Members
DevExpress.Mobile.DataGrid.Theme Namespace
AndroidRefreshCustomizer
IOSRefreshCustomizer

© 2018 Developer Express Inc.

# ThemeBase Methods

Represents the base class for the GridControl themes.

The following tables list the members exposed by the ThemeBase type.

**Public Methods**

| | Name | Description |
|---|---|---|
| | CreateDefaultFont | Overloaded. Creates and returns a **ThemeFontAttributes** object with the default font attributes and the specified color. |
| | CreateDefaultFontForFormEditors | Creates and returns a **ThemeFontAttributes** object with the default font attributes and the specified color. |
| | CreateDefaultPadding | Creates and returns the **Thickness** object that corresponds to the default paddings. |
| | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

**See Also**
ThemeBase Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# CreateDefaultFont Method

Creates and returns a **ThemeFontAttributes** object with the default font attributes and the specified color.

**Overload List**

| Name | Description |
|---|---|
| static ThemeFontAttributes CreateDefaultFont(Color color) | Creates and returns a **ThemeFontAttributes** object with the default font attributes and the specified color. |
| static ThemeFontAttributes CreateDefaultFont(Color color, FontAttributes attributes) | Creates and returns a **ThemeFontAttributes** object with the specified font attributes and color. |

**See Also**
ThemeBase Class
ThemeBase Members
DevExpress.Mobile.DataGrid.Theme Namespace
ThemeBase.CreateDefaultFont Overload List

Creates and returns a **ThemeFontAttributes** object with the default font attributes and the specified color.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**
**C#**
```
public static ThemeFontAttributes CreateDefaultFont(
   Color color
)
```

**Parameters**
*color*
A color of the font to be returned.

**Return Value**

An object of the **ThemeFontAttributes** type.

**Remarks**

This method returns font attributes according to the device on which the application is currently running (either a phone or a tablet). To not take the device into account, call the CreateDefaultFontForFormEditors method.

**See Also**
ThemeBase Class
ThemeBase Members
DevExpress.Mobile.DataGrid.Theme Namespace
ThemeBase.CreateDefaultFont Overload List

Creates and returns a **ThemeFontAttributes** object with the specified font attributes and color.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**
**C#**
```
public static ThemeFontAttributes CreateDefaultFont(
   Color color,
   FontAttributes attributes
)
```

**Parameters**
*color*
The color of the font to be returned.

© 2018 Developer Express Inc.

*attributes*
A **FontAttributes** object that describes the font to be returned.

### Return Value

An object of the **ThemeFontAttributes** type.

### Remarks

This method returns font attributes according to the device on which the application is currently running (either a phone or a tablet). To not take the device into account, call the CreateDefaultFontForFormEditors method.

### See Also

ThemeBase Class
ThemeBase Members
DevExpress.Mobile.DataGrid.Theme Namespace
ThemeBase.CreateDefaultFont Overload List

© 2018 Developer Express Inc.

# CreateDefaultFontForFormEditors Method

Creates and returns a **ThemeFontAttributes** object with the default font attributes and the specified color.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public static ThemeFontAttributes CreateDefaultFontForFormEditors(
    Color color
)
```

### Parameters

*color*
A color of the font to be returned.

### Return Value

An object of the **ThemeFontAttributes** type.

### Remarks

This method does not track the device on which the application is currently running (either a phone or a tablet). To take the device into account, call the CreateDefaultFont method.

### See Also

ThemeBase Class
ThemeBase Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# CreateDefaultPadding Method

Creates and returns the **Thickness** object that corresponds to the default paddings.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public static Thickness CreateDefaultPadding()
```

### Return Value

An object of the Xamarin.Forms.Thickness type.

### See Also

ThemeBase Class
ThemeBase Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# ThemeManager Class

Allows manipulating a GridControl's themes - apply them to a grid and customize the applied theme's appearance.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public class ThemeManager : object
```

### Remarks

GridControl supports two built-in themes - **Light** and **Dark**. The Themes class stores the theme names. To apply the desired theme to a grid, assign an appropriate **Themes** field to the ThemeName property. For more information, refer to How to: Apply Themes.

You can use the applied theme as is, or customize its appearance (if required). To customize the current theme, use ThemeBase class's properties. To learn more, see the example below.

### Example

| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T297553. | |

This example demonstrates how to customize a theme used to paint a GridControl.

To apply a theme to a **GridControl**, assign its name to the ThemeName property. The Themes class stores available theme names.

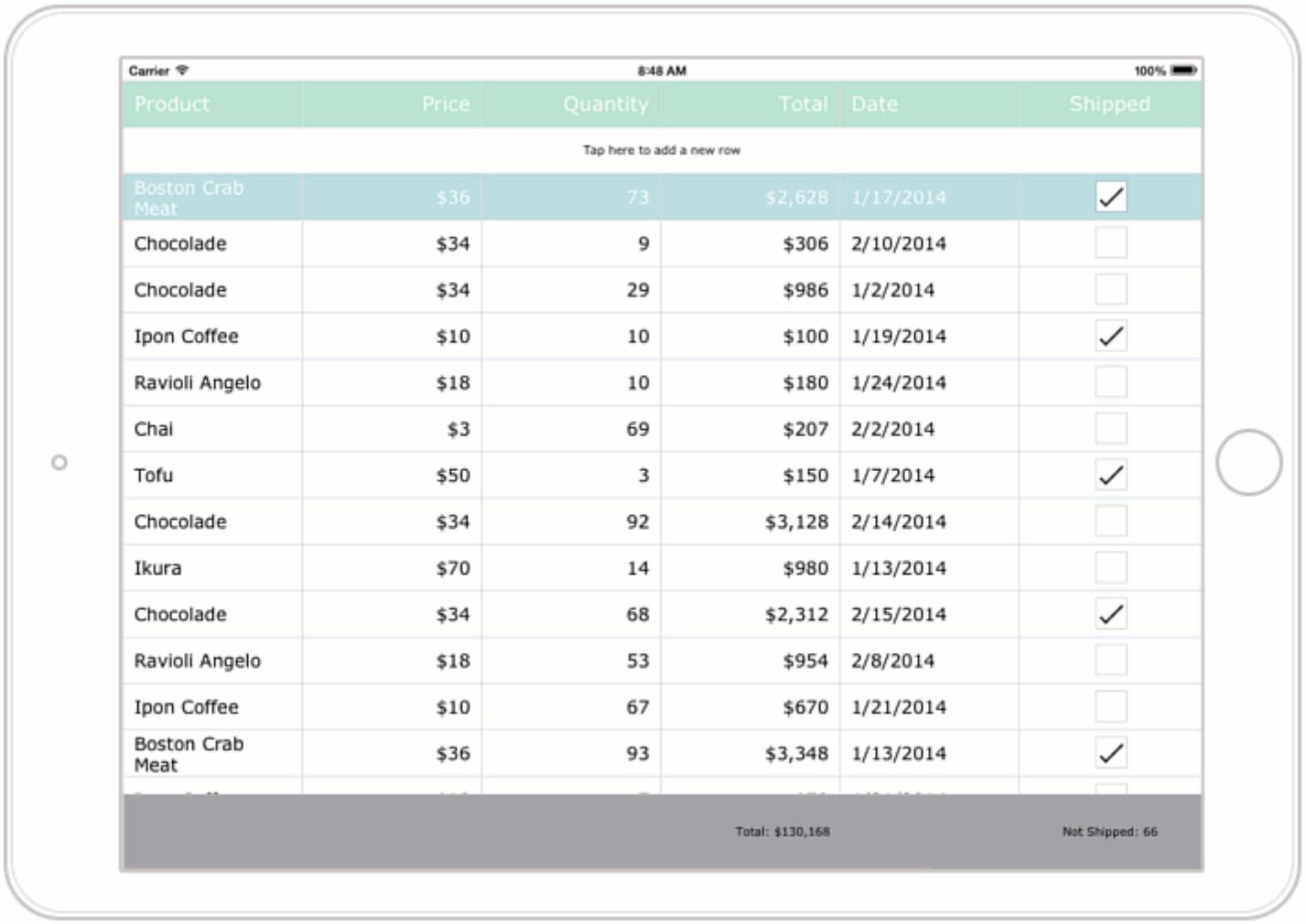
The ThemeBase class's properties provide access to different customizers, which you can adjust to change appearance settings (for example, font attributes, border and background colors, etc.) of the grid's corresponding visual elements (such as data cells, group rows, filter panel, etc.). In this example, the following customizers are used.

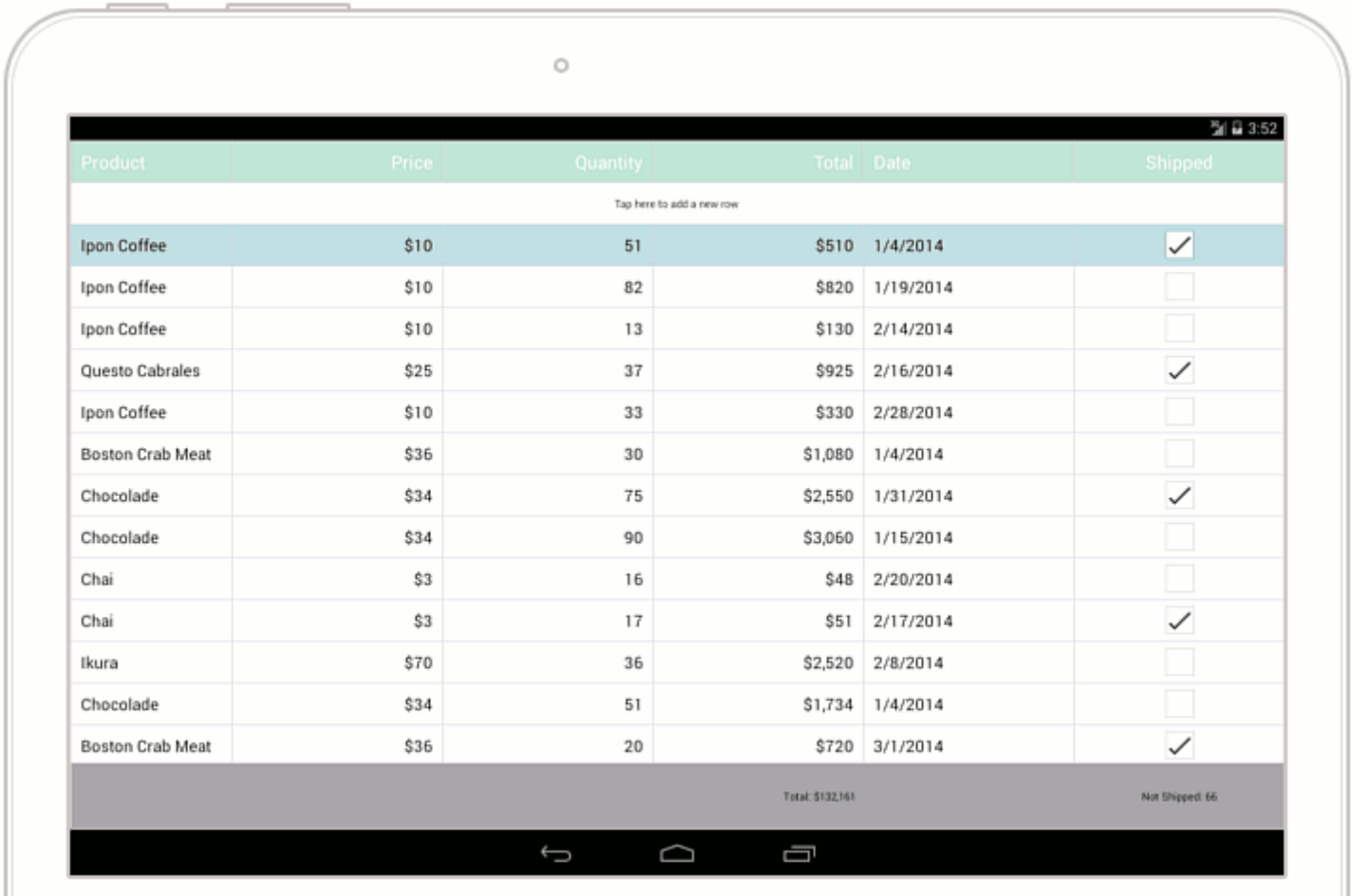
- ThemeBase.HeaderCustomizer
- ThemeBase.CellCustomizer
- ThemeBase.TotalSummaryCustomizer
- ThemeBase.NewItemRowCustomizer

| Important | |
|---|---|
| To apply the customization, call the RefreshTheme method. | |

This example customizes the Light theme as it is shown in the image below.

© 2018 Developer Express Inc.


Carrier
8:48 AM
100%
Product
Price
Quantity
Total
Date
Shipped
Tap here to add a new row
Boston Crab Meat $36 73 $2,628 1/17/2014
Chocolade $34 9 $306 2/10/2014
Chocolade $34 29 $986 1/2/2014
Ipon Coffee $10 10 $100 1/19/2014
Ravioli Angelo $18 10 $180 1/24/2014
Chai $3 69 $207 2/2/2014
Tofu $50 3 $150 1/7/2014
Chocolade $34 92 $3,128 2/14/2014
Ikura $70 14 $980 1/13/2014
Chocolade $34 68 $2,312 2/15/2014
Ravioli Angelo $18 53 $954 2/8/2014
Ipon Coffee $10 67 $670 1/21/2014
Boston Crab Meat $36 93 $3,348 1/13/2014
Total: $130,168
Not Shipped: 66


3:52
Product
Price
Quantity
Total
Date
Shipped
Tap here to add a new row
Ipon Coffee $10 51 $510 1/4/2014
Ipon Coffee $10 82 $820 1/19/2014
Ipon Coffee $10 13 $130 2/14/2014
Questo Cabrales $25 37 $925 2/16/2014
Ipon Coffee $10 33 $330 2/28/2014
Boston Crab Meat $36 30 $1,080 1/4/2014
Chocolade $34 75 $2,550 1/31/2014
Chocolade $34 90 $3,060 1/15/2014
Chai $3 16 $48 2/20/2014
Chai $3 17 $51 2/17/2014
Ikura $70 36 $2,520 2/8/2014
Chocolade $34 51 $1,734 1/4/2014
Boston Crab Meat $36 20 $720 3/1/2014
Total: $132,161
Not Shipped: 66

**C#**

```
 (MainPage.xaml.cs)
using System;
using System.Collections.Generic;
using Xamarin.Forms;
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.DataGrid.Theme;
namespace ThemeCustomizing {
    public partial class MainPage : ContentPage {
        int count = 0;
        public MainPage() {
            InitializeComponent();
            TestOrdersRepository model = new TestOrdersRepository();
            BindingContext = model;
            ThemeManager.ThemeName = Themes.Light;
            // Header customization.
            ThemeManager.Theme.HeaderCustomizer.BackgroundColor = Color.FromRgb(187, 228, 208);
            ThemeFontAttributes myFont = new ThemeFontAttributes("Verdana",
                                         ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Large),
                                         FontAttributes.None, Color.White);
            ThemeManager.Theme.HeaderCustomizer.Font = myFont;
            // Cell customization.
            ThemeManager.Theme.CellCustomizer.SelectionColor = Color.FromRgb(186, 220, 225);
            ThemeFontAttributes myFont1 = new ThemeFontAttributes("Verdana",
                                         ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Medium),
                                         FontAttributes.None, Color.Black);
            ThemeManager.Theme.CellCustomizer.Font = myFont1;
            // Various customization.
            ThemeManager.Theme.TotalSumaryCustomizer.BackgroundColor = Color.FromRgb(163, 162, 168);
            ThemeFontAttributes myFont2 = new ThemeFontAttributes("Verdana",
                                 ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Default),
                                 FontAttributes.None, Color.Black);
            ThemeManager.Theme.TotalSummaryCustomizer.Font = myFont2;
            ThemeManager.Theme.NewItemRowCustomizer.Font = myFont2;
            ThemeManager.RefreshTheme();
        }
        void OnCalculateCustomSummary(object sender, CustomSummaryEventArgs e) {
            if (e.FieldName.ToString() == "Shipped")
            if (e.IsTotalSummary) {
                if (e.SummaryProcess == CustomSummaryProcess.Calculate) {
                    if (!(bool)e.FieldValue)
                        count++;
                    e.TotalValue = count;
                }
            }
        }
    }
}
```

**Inheritance Hierarchy**

System.Object
  **ThemeManager**

**See Also**
ThemeManager Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# ThemeManager Members

Allows manipulating a GridControl's themes - apply them to a grid and customize the applied theme's appearance.

The following tables list the members exposed by the ThemeManager type.

### Public Properties

| | Name | Description |
|---|---|---|
| | Theme | Provides access to the current grid theme. |
| | ThemeName | Gets or sets the currently applied theme's name. |

### Public Methods

| | Name | Description |
|---|---|---|
| | AddThemeChangedHandler | Adds a weak reference to a delegate that will be used to handle the ThemeChanged event. |
| | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | ReferenceEquals | (Inherited from **System.Object**) |
| | RefreshTheme | Refreshes the theme that is currently applied to a GridControl. |
| | ToString | (Inherited from **System.Object**) |

### Public Events

| | Name | Description |
|---|---|---|
| | ThemeChanged | Occurs after you either apply a new theme to a grid or call the RefreshTheme method. |

**See Also**
ThemeManager Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# ThemeManager Properties

Allows manipulating a GridControl's themes - apply them to a grid and customize the applied theme's appearance.

The following tables list the members exposed by the ThemeManager type.

### Public Properties

| | Name | Description |
|---|---|---|
| | Theme | Provides access to the current grid theme. |
| | ThemeName | Gets or sets the currently applied theme's name. |

### Public Methods

| | Name | Description |
|---|---|---|
| | AddThemeChangedHandler | Adds a weak reference to a delegate that will be used to handle the ThemeChanged event. |
| | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | ReferenceEquals | (Inherited from **System.Object**) |
| | RefreshTheme | Refreshes the theme that is currently applied to a GridControl. |
| | ToString | (Inherited from **System.Object**) |

### Public Events

| | Name | Description |
|---|---|---|
| | ThemeChanged | Occurs after you either apply a new theme to a grid or call the RefreshTheme method. |

**See Also**
ThemeManager Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# Theme Property

Provides access to the current grid theme.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public static ThemeBase Theme { get; set; }
```

### Property Value

A ThemeBase class descendant.

### Remarks

The **Theme** property allows you to customize the theme that is applied to a grid. For more information, see the example below.

### Example

| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T297553. | |

This example demonstrates how to customize a theme used to paint a GridControl.

To apply a theme to a **GridControl**, assign its name to the ThemeName property. The Themes class stores available theme names.

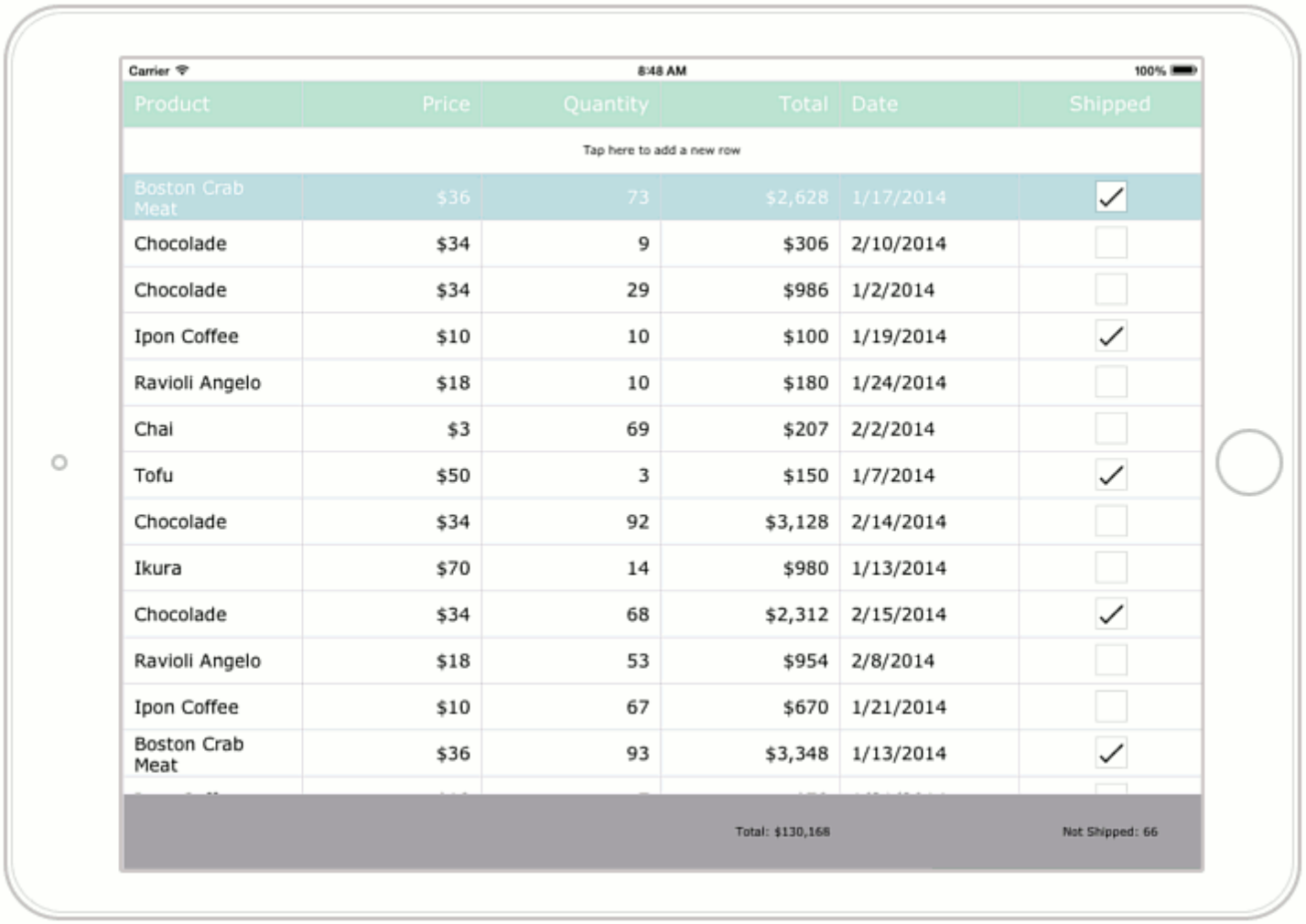
The ThemeBase class's properties provide access to different customizers, which you can adjust to change appearance settings (for example, font attributes, border and background colors, etc.) of the grid's corresponding visual elements (such as data cells, group rows, filter panel, etc.). In this example, the following customizers are used.

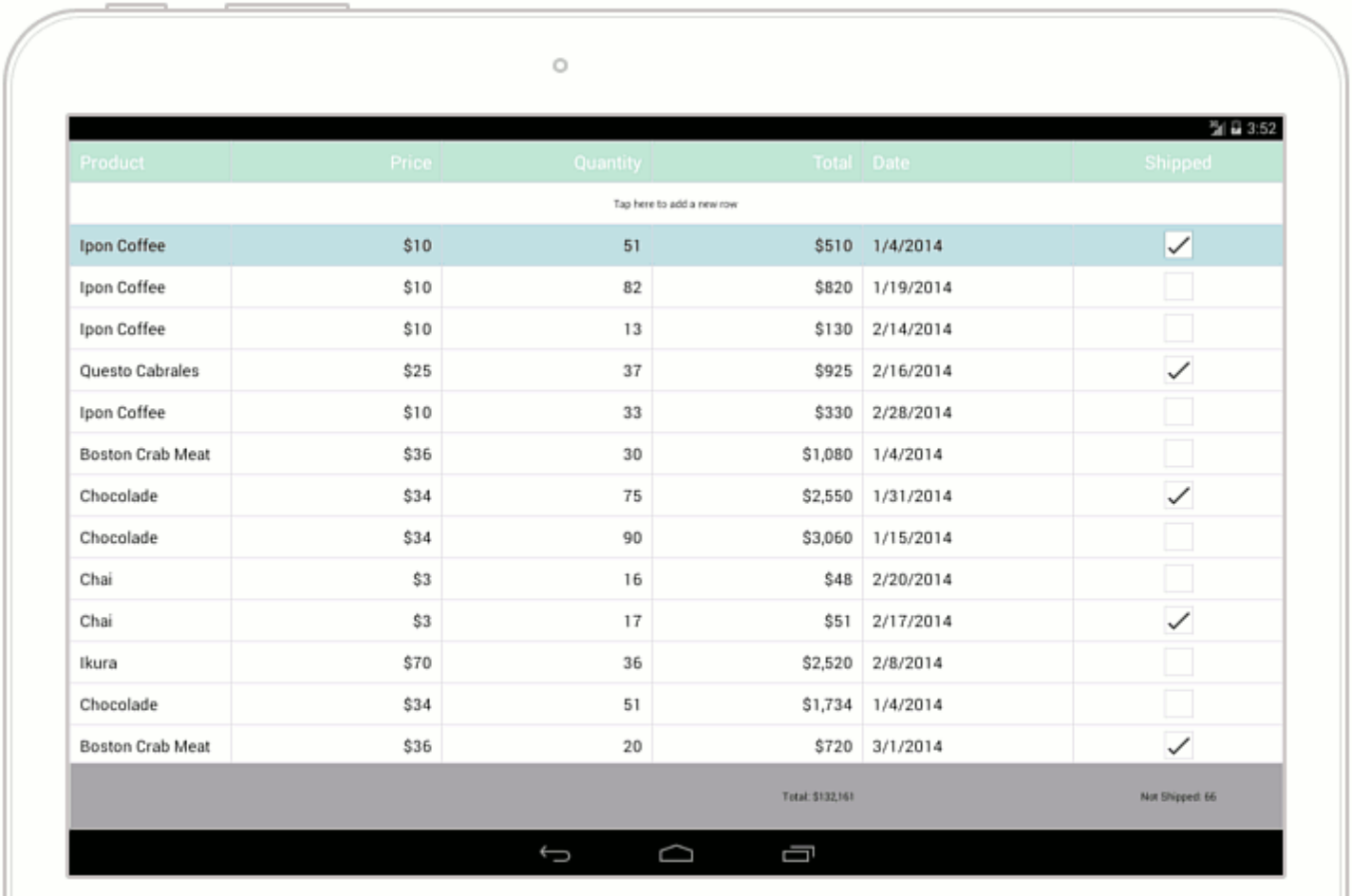
- ThemeBase.HeaderCustomizer
- ThemeBase.CellCustomizer
- ThemeBase.TotalSummaryCustomizer
- ThemeBase.NewItemRowCustomizer

| Important | |
|---|---|
| To apply the customization, call the RefreshTheme method. | |

This example customizes the Light theme as it is shown in the image below.

© 2018 Developer Express Inc.


Carrier
8:48 AM
100%
Product
Price
Quantity
Total
Date
Shipped
Tap here to add a new row
Boston Crab Meat $36 73 $2,628 1/17/2014
Chocolade $34 9 $306 2/10/2014
Chocolade $34 29 $986 1/2/2014
Ipon Coffee $10 10 $100 1/19/2014
Ravioli Angelo $18 10 $180 1/24/2014
Chai $3 69 $207 2/2/2014
Tofu $50 3 $150 1/7/2014
Chocolade $34 92 $3,128 2/14/2014
Ikura $70 14 $980 1/13/2014
Chocolade $34 68 $2,312 2/15/2014
Ravioli Angelo $18 53 $954 2/8/2014
Ipon Coffee $10 67 $670 1/21/2014
Boston Crab Meat $36 93 $3,348 1/13/2014
Total: $130,168
Not Shipped: 66


3:52
Product
Price
Quantity
Total
Date
Shipped
Tap here to add a new row
Ipon Coffee $10 51 $510 1/4/2014
Ipon Coffee $10 82 $820 1/19/2014
Ipon Coffee $10 13 $130 2/14/2014
Questo Cabrales $25 37 $925 2/16/2014
Ipon Coffee $10 33 $330 2/28/2014
Boston Crab Meat $36 30 $1,080 1/4/2014
Chocolade $34 75 $2,550 1/31/2014
Chocolade $34 90 $3,060 1/15/2014
Chai $3 16 $48 2/20/2014
Chai $3 17 $51 2/17/2014
Ikura $70 36 $2,520 2/8/2014
Chocolade $34 51 $1,734 1/4/2014
Boston Crab Meat $36 20 $720 3/1/2014
Total: $132,161
Not Shipped: 66

**C#**

```
 (MainPage.xaml.cs)
using System;
using System.Collections.Generic;
using Xamarin.Forms;
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.DataGrid.Theme;
namespace ThemeCustomizing {
    public partial class MainPage : ContentPage {
        int count = 0;
        public MainPage() {
            InitializeComponent();
            TestOrdersRepository model = new TestOrdersRepository();
            BindingContext = model;
            ThemeManager.ThemeName = Themes.Light;
            // Header customization.
            ThemeManager.Theme.HeaderCustomizer.BackgroundColor = Color.FromRgb(187, 228, 208);
            ThemeFontAttributes myFont = new ThemeFontAttributes("Verdana",
                                         ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Large),
                                         FontAttributes.None, Color.White);
            ThemeManager.Theme.HeaderCustomizer.Font = myFont;
            // Cell customization.
            ThemeManager.Theme.CellCustomizer.SelectionColor = Color.FromRgb(186, 220, 225);
            ThemeFontAttributes myFont1 = new ThemeFontAttributes("Verdana",
                                         ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Medium),
                                         FontAttributes.None, Color.Black);
            ThemeManager.Theme.CellCustomizer.Font = myFont1;
            // Various customization.
            ThemeManager.Theme.TotalSumaryCustomizer.BackgroundColor = Color.FromRgb(163, 162, 168);
            ThemeFontAttributes myFont2 = new ThemeFontAttributes("Verdana",
                                 ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Default),
                                 FontAttributes.None, Color.Black);
            ThemeManager.Theme.TotalSummaryCustomizer.Font = myFont2;
            ThemeManager.Theme.NewItemRowCustomizer.Font = myFont2;
            ThemeManager.RefreshTheme();
        }
        void OnCalculateCustomSummary(object sender, CustomSummaryEventArgs e) {
            if (e.FieldName.ToString() == "Shipped")
            if (e.IsTotalSummary) {
                if (e.SummaryProcess == CustomSummaryProcess.Calculate) {
                    if (!(bool)e.FieldValue)
                        count++;
                    e.TotalValue = count;
                }
            }
        }
    }
}
```

**See Also**

ThemeManager Class
ThemeManager Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# ThemeName Property

Gets or sets the currently applied theme's name.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public static string ThemeName { get; set; }
```

### Property Value

Specifies the name of the theme. The available theme names are stored in the Themes class. You can either use its fields or specify the desired name manually.

### Remarks

By default, GridControl provides you with two predefined themes - **Light** and **Dark**. To apply the required theme to a grid, use the code below.

**C#**

```
using Xamarin.Forms;
using DevExpress.Mobile.DataGrid.Theme;
     // ...
    ThemeManager.ThemeName = Themes.Dark;
    // or
    ThemeManager.ThemeName = "Dark";
    // ...
```

For more information, see the How to: Customize Themes document.

### See Also

ThemeManager Class
ThemeManager Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# ThemeManager Events

Allows manipulating a GridControl's themes - apply them to a grid and customize the applied theme's appearance.

The following tables list the members exposed by the ThemeManager type.

**Public Events**

| | Name | Description |
|---|---|---|
| | ThemeChanged | Occurs after you either apply a new theme to a grid or call the RefreshTheme method. |

**See Also**
ThemeManager Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# ThemeChanged Event

Occurs after you either apply a new theme to a grid or call the RefreshTheme method.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public static event EventHandler ThemeChanged
```

### Event Data

The event handler receives an argument of type System.EventArgs containing data related to this event.

### Remarks

By default, GridControl provides you with two predefined themes - **Light** and **Dark**. You can apply either of them to the control and then customize the theme appearance (if needed). For more information on working with grid themes, see examples in the Themes section.

### See Also

ThemeManager Class
ThemeManager Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# ThemeManager Methods

Allows manipulating a GridControl's themes - apply them to a grid and customize the applied theme's appearance.

The following tables list the members exposed by the ThemeManager type.

**Public Methods**

| | Name | Description |
|---|---|---|
|  | AddThemeChangedHandler | Adds a weak reference to a delegate that will be used to handle the ThemeChanged event. |
|  | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | ReferenceEquals | (Inherited from **System.Object**) |
| | RefreshTheme | Refreshes the theme that is currently applied to a GridControl. |
| | ToString | (Inherited from **System.Object**) |

**See Also**
ThemeManager Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# AddThemeChangedHandler Method

Adds a weak reference to a delegate that will be used to handle the ThemeChanged event.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public static void AddThemeChangedHandler(
   IThemeChangingHandler handler
)
```

### Parameters

*handler*
The handler that specifies the added delegate.

### Remarks

The added reference still allows the object to be garbage collected.

### See Also

ThemeManager Class
ThemeManager Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# RefreshTheme Method

Refreshes the theme that is currently applied to a GridControl.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public static void RefreshTheme()
```

### Remarks

By default, the **GridControl** class provides you with two predefined themes - **Light** and **Dark** - which can be assigned to the ThemeName property.

After you apply the desired theme to a grid, you can change theme appearance via its customizers. You use these customizers to modify background colors, fonts and so on. To apply the customization, call the **RefreshTheme** method at the end of your code. It will refresh the theme layout, so all customizations are correctly applied to a grid.

### Example

| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T297553. | |

This example demonstrates how to customize a theme used to paint a GridControl.

To apply a theme to a **GridControl**, assign its name to the ThemeName property. The Themes class stores available theme names.

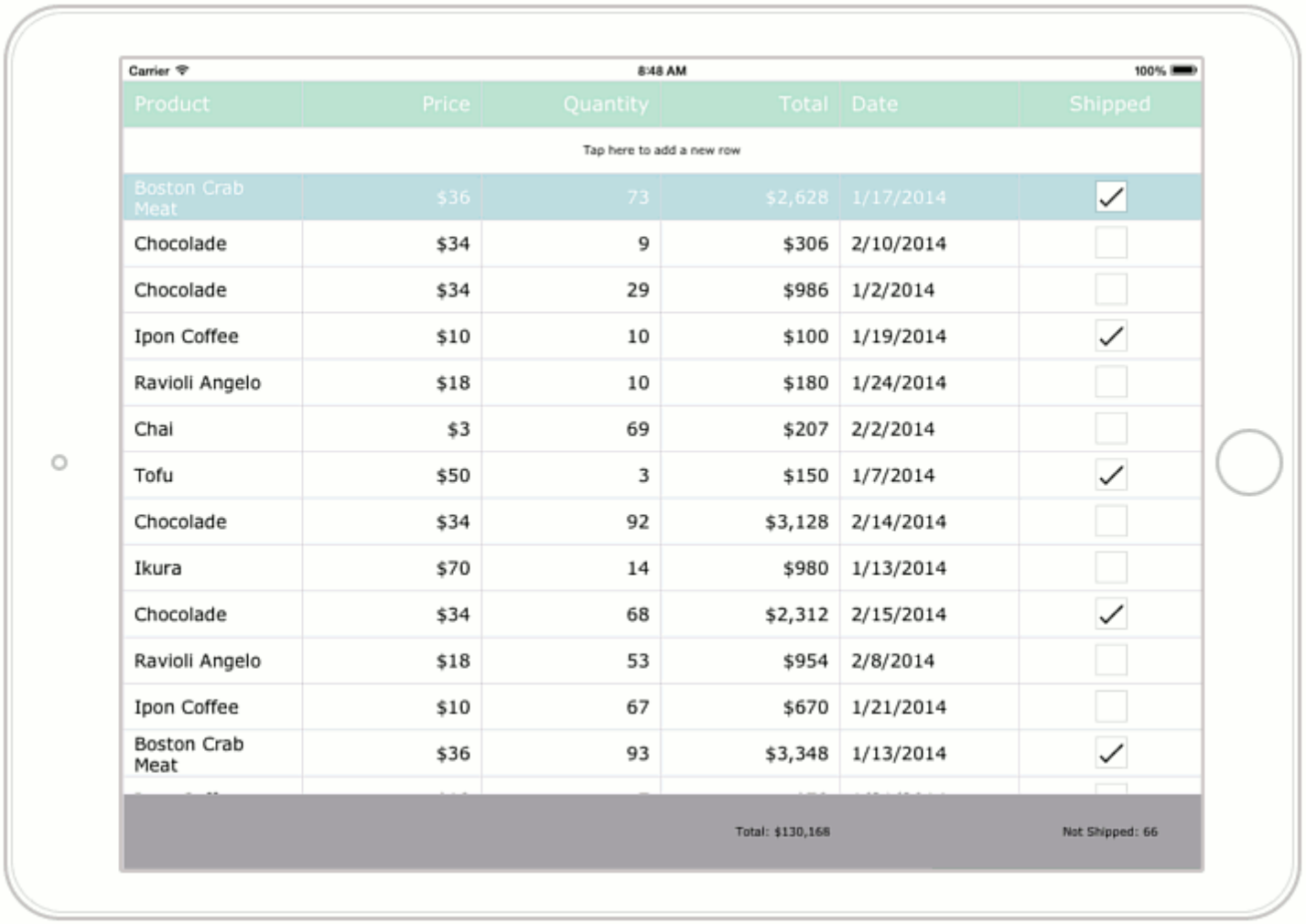
The ThemeBase class's properties provide access to different customizers, which you can adjust to change appearance settings (for example, font attributes, border and background colors, etc.) of the grid's corresponding visual elements (such as data cells, group rows, filter panel, etc.). In this example, the following customizers are used.

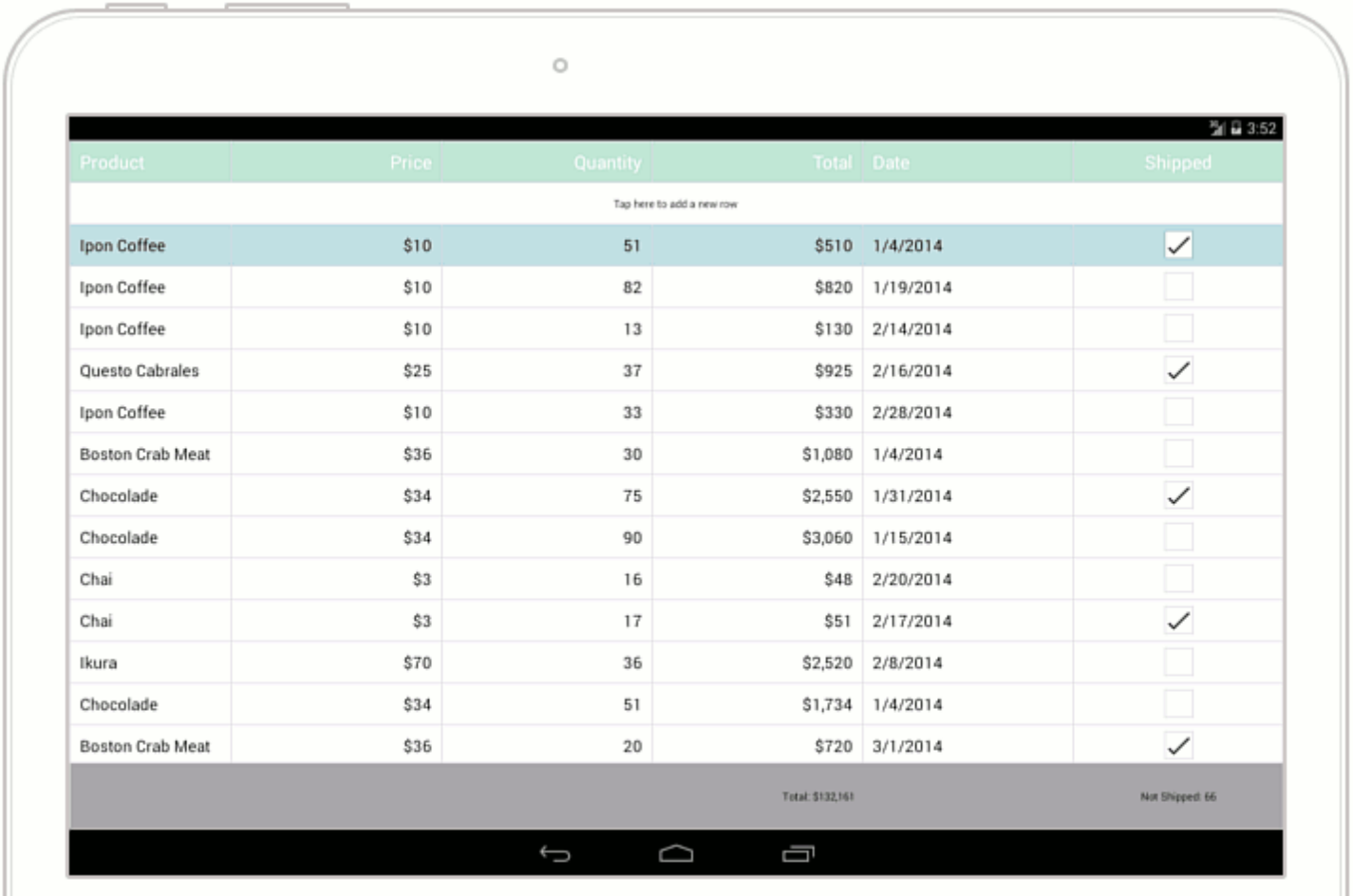
- ThemeBase.HeaderCustomizer
- ThemeBase.CellCustomizer
- ThemeBase.TotalSummaryCustomizer
- ThemeBase.NewItemRowCustomizer

| Important | |
|---|---|
| To apply the customization, call the RefreshTheme method. | |

This example customizes the Light theme as it is shown in the image below.

© 2018 Developer Express Inc.


Carrier
8:48 AM
100%
Product
Price
Quantity
Total
Date
Shipped
Tap here to add a new row
Boston Crab Meat $36 73 $2,628 1/17/2014
Chocolade $34 9 $306 2/10/2014
Chocolade $34 29 $986 1/2/2014
Ipon Coffee $10 10 $100 1/19/2014
Ravioli Angelo $18 10 $180 1/24/2014
Chai $3 69 $207 2/2/2014
Tofu $50 3 $150 1/7/2014
Chocolade $34 92 $3,128 2/14/2014
Ikura $70 14 $980 1/13/2014
Chocolade $34 68 $2,312 2/15/2014
Ravioli Angelo $18 53 $954 2/8/2014
Ipon Coffee $10 67 $670 1/21/2014
Boston Crab Meat $36 93 $3,348 1/13/2014
Total: $130,168
Not Shipped: 66


3:52
Product
Price
Quantity
Total
Date
Shipped
Tap here to add a new row
Ipon Coffee $10 51 $510 1/4/2014
Ipon Coffee $10 82 $820 1/19/2014
Ipon Coffee $10 13 $130 2/14/2014
Questo Cabrales $25 37 $925 2/16/2014
Ipon Coffee $10 33 $330 2/28/2014
Boston Crab Meat $36 30 $1,080 1/4/2014
Chocolade $34 75 $2,550 1/31/2014
Chocolade $34 90 $3,060 1/15/2014
Chai $3 16 $48 2/20/2014
Chai $3 17 $51 2/17/2014
Ikura $70 36 $2,520 2/8/2014
Chocolade $34 51 $1,734 1/4/2014
Boston Crab Meat $36 20 $720 3/1/2014
Total: $132,161
Not Shipped: 66

C#

```
 (MainPage.xaml.cs)
using System;
using System.Collections.Generic;
using Xamarin.Forms;
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.DataGrid.Theme;
namespace ThemeCustomizing {
    public partial class MainPage : ContentPage {
        int count = 0;
        public MainPage() {
            InitializeComponent();
            TestOrdersRepository model = new TestOrdersRepository();
            BindingContext = model;
            ThemeManager.ThemeName = Themes.Light;
            // Header customization.
            ThemeManager.Theme.HeaderCustomizer.BackgroundColor = Color.FromRgb(187, 228, 208);
            ThemeFontAttributes myFont = new ThemeFontAttributes("Verdana",
                                         ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Large),
                                         FontAttributes.None, Color.White);
            ThemeManager.Theme.HeaderCustomizer.Font = myFont;
            // Cell customization.
            ThemeManager.Theme.CellCustomizer.SelectionColor = Color.FromRgb(186, 220, 225);
            ThemeFontAttributes myFont1 = new ThemeFontAttributes("Verdana",
                                         ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Medium),
                                         FontAttributes.None, Color.Black);
            ThemeManager.Theme.CellCustomizer.Font = myFont1;
            // Various customization.
            ThemeManager.Theme.TotalSumaryCustomizer.BackgroundColor = Color.FromRgb(163, 162, 168);
            ThemeFontAttributes myFont2 = new ThemeFontAttributes("Verdana",
                                 ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Default),
                                 FontAttributes.None, Color.Black);
            ThemeManager.Theme.TotalSummaryCustomizer.Font = myFont2;
            ThemeManager.Theme.NewItemRowCustomizer.Font = myFont2;
            ThemeManager.RefreshTheme();
        }
        void OnCalculateCustomSummary(object sender, CustomSummaryEventArgs e) {
            if (e.FieldName.ToString() == "Shipped")
            if (e.IsTotalSummary) {
                if (e.SummaryProcess == CustomSummaryProcess.Calculate) {
                    if (!(bool)e.FieldValue)
                        count++;
                    e.TotalValue = count;
                }
            }
        }
    }
}
```

**See Also**

ThemeManager Class
ThemeManager Members
DevExpress.Mobile.DataGrid.Theme Namespace
How to: Apply Themes
How to: Customize Themes

© 2018 Developer Express Inc.

# Themes Class

Contains the available GridControl theme names.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

### Syntax

**C#**

```
public class Themes : object
```

### Remarks

The Themes class provides fields containing theme names (e.g. **Light**, **Dark**), which correspond to the predefined grid themes. To enable any of these themes, set the appropriate field value to the ThemeManager.ThemeName property.

The **Default** theme depends on the current mobile platform. For iOS applications, the **Default** theme is **Light**; for Android applications - **Dark**.

### Example

| Show Me | |
|---|---|
| A complete sample project is available in the DevExpress Code Examples database at http://www.devexpress.com/example=T297553. | |

This example demonstrates how to customize a theme used to paint a GridControl.

To apply a theme to a **GridControl**, assign its name to the ThemeManager.ThemeName property. The Themes class stores available theme names.

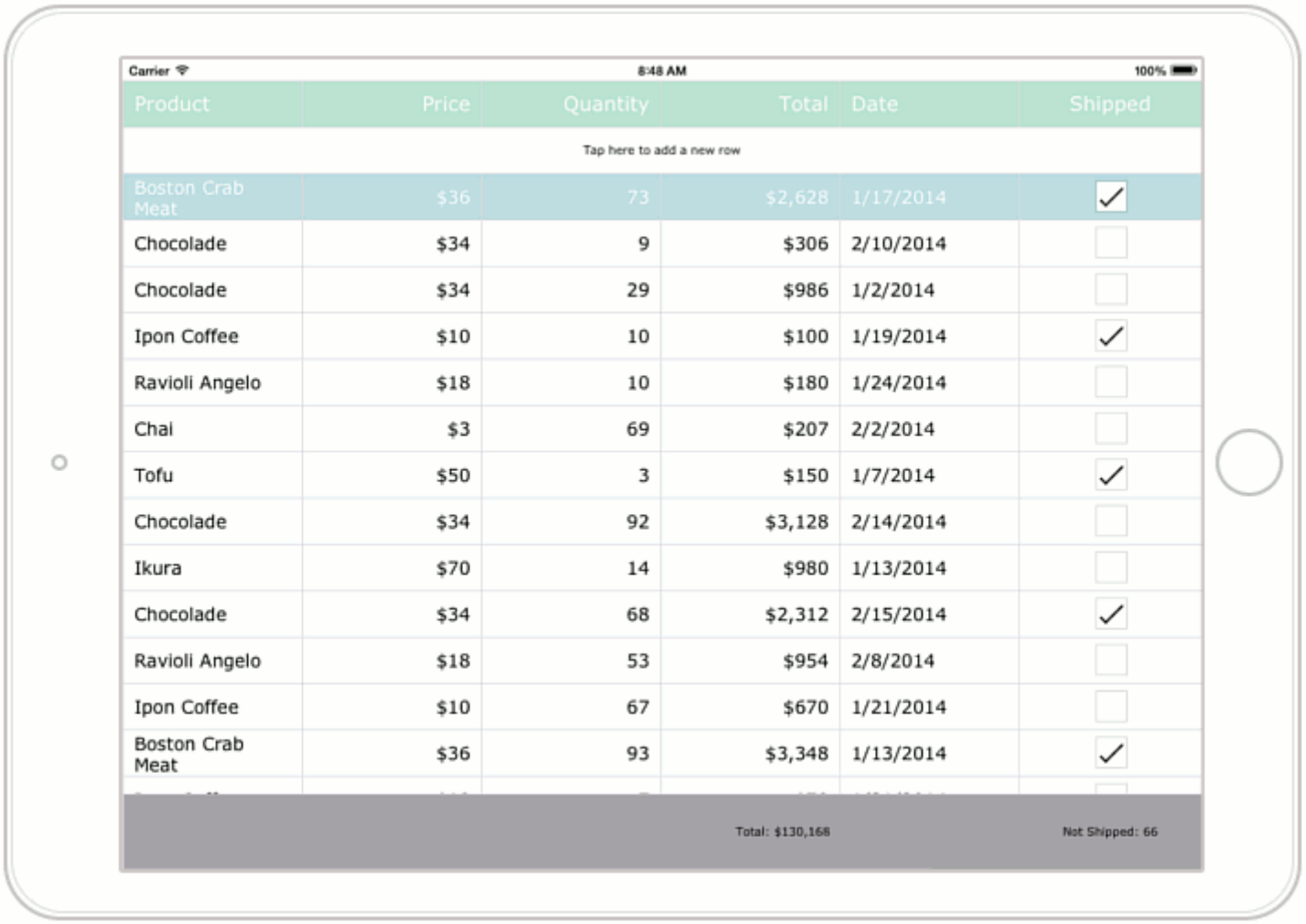
The ThemeBase class's properties provide access to different customizers, which you can adjust to change appearance settings (for example, font attributes, border and background colors, etc.) of the grid's corresponding visual elements (such as data cells, group rows, filter panel, etc.). In this example, the following customizers are used.

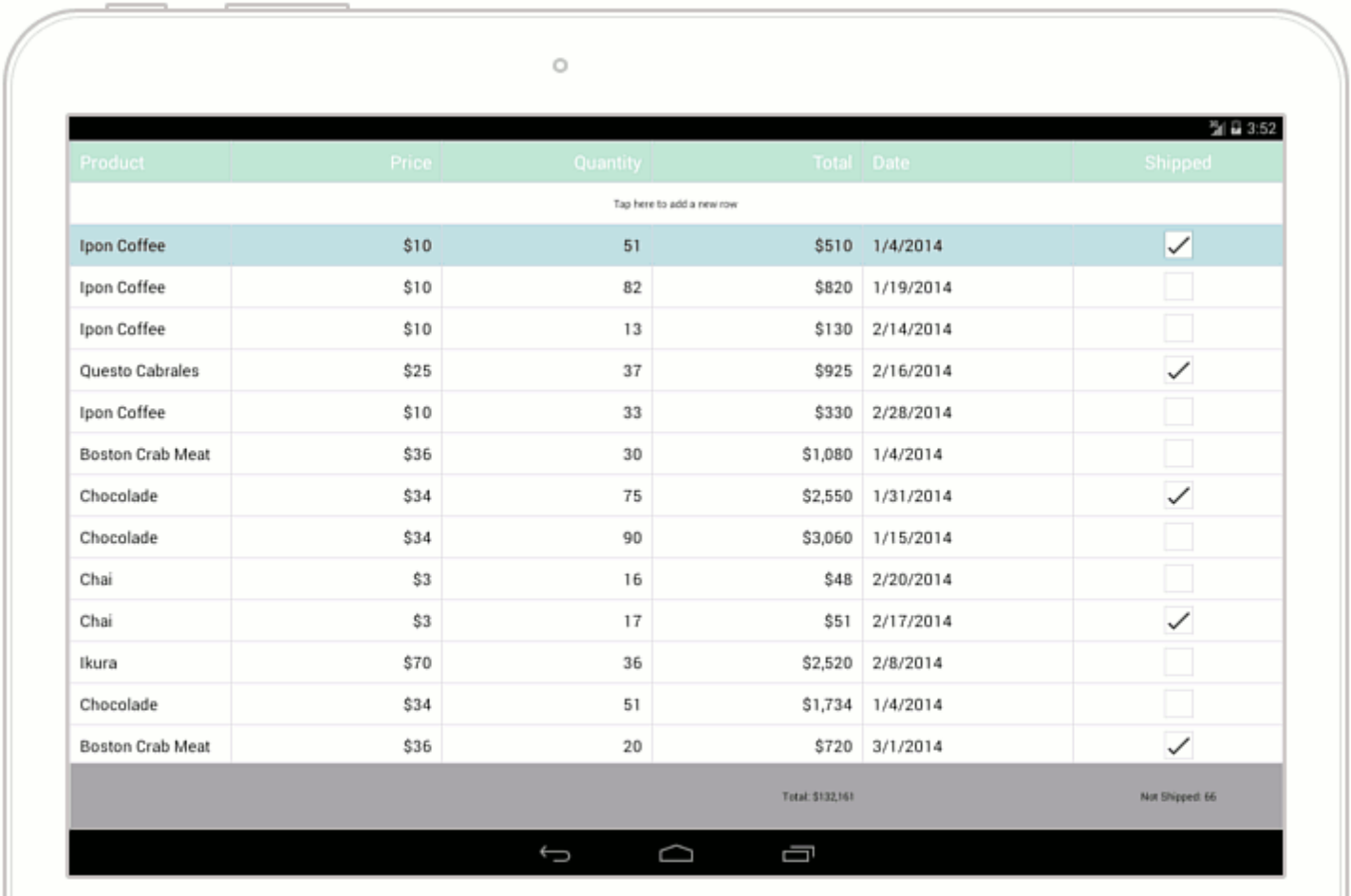
- ThemeBase.HeaderCustomizer
- ThemeBase.CellCustomizer
- ThemeBase.TotalSummaryCustomizer
- ThemeBase.NewItemRowCustomizer

| Important | |
|---|---|
| To apply the customization, call the ThemeManager.RefreshTheme method. | |

This example customizes the Light theme as it is shown in the image below.

© 2018 Developer Express Inc.


Carrier
8:48 AM
100%
Product
Price
Quantity
Total
Date
Shipped
Tap here to add a new row
Boston Crab Meat $36 73 $2,628 1/17/2014
Chocolade $34 9 $306 2/10/2014
Chocolade $34 29 $986 1/2/2014
Ipon Coffee $10 10 $100 1/19/2014
Ravioli Angelo $18 10 $180 1/24/2014
Chai $3 69 $207 2/2/2014
Tofu $50 3 $150 1/7/2014
Chocolade $34 92 $3,128 2/14/2014
Ikura $70 14 $980 1/13/2014
Chocolade $34 68 $2,312 2/15/2014
Ravioli Angelo $18 53 $954 2/8/2014
Ipon Coffee $10 67 $670 1/21/2014
Boston Crab Meat $36 93 $3,348 1/13/2014
Total: $130,168
Not Shipped: 66


3:52
Product
Price
Quantity
Total
Date
Shipped
Tap here to add a new row
Ipon Coffee $10 51 $510 1/4/2014
Ipon Coffee $10 82 $820 1/19/2014
Ipon Coffee $10 13 $130 2/14/2014
Questo Cabrales $25 37 $925 2/16/2014
Ipon Coffee $10 33 $330 2/28/2014
Boston Crab Meat $36 30 $1,080 1/4/2014
Chocolade $34 75 $2,550 1/31/2014
Chocolade $34 90 $3,060 1/15/2014
Chai $3 16 $48 2/20/2014
Chai $3 17 $51 2/17/2014
Ikura $70 36 $2,520 2/8/2014
Chocolade $34 51 $1,734 1/4/2014
Boston Crab Meat $36 20 $720 3/1/2014
Total: $132,161
Not Shipped: 66

C#

```
 (MainPage.xaml.cs)
using System;
using System.Collections.Generic;
using Xamarin.Forms;
using DevExpress.Mobile.DataGrid;
using DevExpress.Mobile.DataGrid.Theme;
namespace ThemeCustomizing {
    public partial class MainPage : ContentPage {
        int count = 0;
        public MainPage() {
            InitializeComponent();
            TestOrdersRepository model = new TestOrdersRepository();
            BindingContext = model;
            ThemeManager.ThemeName = Themes.Light;
            // Header customization.
            ThemeManager.Theme.HeaderCustomizer.BackgroundColor = Color.FromRgb(187, 228, 208);
            ThemeFontAttributes myFont = new ThemeFontAttributes("Verdana",
                                         ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Large),
                                         FontAttributes.None, Color.White);
            ThemeManager.Theme.HeaderCustomizer.Font = myFont;
            // Cell customization.
            ThemeManager.Theme.CellCustomizer.SelectionColor = Color.FromRgb(186, 220, 225);
            ThemeFontAttributes myFont1 = new ThemeFontAttributes("Verdana",
                                         ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Medium),
                                         FontAttributes.None, Color.Black);
            ThemeManager.Theme.CellCustomizer.Font = myFont1;
            // Various customization.
            ThemeManager.Theme.TotalSumaryCustomizer.BackgroundColor = Color.FromRgb(163, 162, 168);
            ThemeFontAttributes myFont2 = new ThemeFontAttributes("Verdana",
                                 ThemeFontAttributes.FontSizeFromNamedSize(NamedSize.Default),
                                 FontAttributes.None, Color.Black);
            ThemeManager.Theme.TotalSummaryCustomizer.Font = myFont2;
            ThemeManager.Theme.NewItemRowCustomizer.Font = myFont2;
            ThemeManager.RefreshTheme();
        }
        void OnCalculateCustomSummary(object sender, CustomSummaryEventArgs e) {
            if (e.FieldName.ToString() == "Shipped")
            if (e.IsTotalSummary) {
                if (e.SummaryProcess == CustomSummaryProcess.Calculate) {
                    if (!(bool)e.FieldValue)
                        count++;
                    e.TotalValue = count;
                }
            }
        }
    }
}
```

**Inheritance Hierarchy**

System.Object
  **Themes**

**See Also**
Themes Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# Themes Members

Contains the available GridControl theme names.

The following tables list the members exposed by the Themes type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | Themes | Initializes a new instance of the Themes class. |

### Public Methods

| | Name | Description |
|---|---|---|
| | Equals | (Inherited from **System.Object**) |
| S | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | ToString | (Inherited from **System.Object**) |

### Public Fields

| | Name | Description |
|---|---|---|
| S | Dark | Indicates the **Dark** theme. |
| S | Default | By default, for iOS indicates the **Light** theme. For Android, indicates the **Dark** theme. |
| S | Light | Indicates the **Light** theme. |

**See Also**
Themes Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# Themes Constructor

Initializes a new instance of the Themes class.

**Namespace:** DevExpress.Mobile.DataGrid.Theme
**Assembly:** DevExpress.Mobile.Grid.v18.1.dll

**Syntax**
**C#**

```
public Themes()
```

**See Also**
Themes Class
Themes Members
DevExpress.Mobile.DataGrid.Theme Namespace

© 2018 Developer Express Inc.

# DevExpress.Mobile.Export

Contains base classes required for data export to XLS, XLSX and CSV formats.

**Classes**

| | Class | Description |
|---|---|---|
| | CsvExportOptions | Contains options which define how grid data is exported to CSV format. |
| | DataAwareExportOptionsBase | Contains options which define how data shaping settings are exported from the grid to XLS and XLSX formats. |
| | ExportOptionsBase | The base class for objects which define specific export options. |
| | TextExportOptionsBase | The base class for objects which define export options specific to text-based formats. |
| | XlsExportOptions | Contains options which define how grid data is exported to XLS format. |
| | XlsExportOptionsBase | The base class for objects which define export options specific to XLS and XLSX formats. |
| | XlsxExportOptions | Contains options which define how grid data is exported to XLSX format. |

© 2018 Developer Express Inc.

# CsvExportOptions Class

Contains options which define how grid data is exported to CSV format.

**Namespace:** DevExpress.Mobile.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public class CsvExportOptions : TextExportOptionsBase, IDataAwareExportOptions
```

### Remarks

To export data from the GridControl to CSV format, use the GridControl.ExportToExcel method with the ExportTarget.Csv value passed as the *format* parameter.

To specify CSV export settings, create an instance of the **CsvExportOptions** class and assign it to the GridControl.OptionsExportCsv property.

### Inheritance Hierarchy

System.Object
  Xamarin.Forms.BindableObject
    ExportOptionsBase
      TextExportOptionsBase
        **CsvExportOptions**

### See Also

CsvExportOptions Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# CsvExportOptions Members

Contains options which define how grid data is exported to CSV format.

The following tables list the members exposed by the CsvExportOptions type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | CsvExportOptions | Overloaded. Initializes a new instance of the CsvExportOptions class with default settings. |

### Public Properties

| | Name | Description |
|---|---|---|
| | BindingContext | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Encoding | Gets or sets the encoding of the text-based file to which grid's data is exported. (Inherited from **TextExportOptionsBase**) |
| | Separator | Gets or sets the symbol(s) to separate text elements when grid data is exported to a text-based file. (Inherited from **TextExportOptionsBase**) |
| | WritePreamble | Gets or sets whether to write a prefix that is the sequence of bytes that specifies the encoding used. |

### Public Methods

| | Name | Description |
|---|---|---|
| | Assign | Copies all settings from the options object passed as a parameter. |
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ToString | (Inherited from **System.Object**) |

### Public Fields

| | Name | Description |
|---|---|---|
| | BindingContextProperty | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SeparatorProperty | Identifies the TextExportOptionsBase.Separator bindable property. (Inherited from **TextExportOptionsBase**) |
| | WritePreambleProperty | Identifies the WritePreamble bindable property. |

### Public Events

| | Name | Description |
|---|---|---|
| | BindingContextChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | CustomizeCell | When exporting data from the grid to CSV format, this event allows you to customize a cell in the output document. |
| | CustomizeDataArea | When exporting to CSV format, this event allows you to customize data in the output document. |
| | CustomizeFooter | When exporting to CSV format, this event allows you to add a footer to the output document. |
| | CustomizeHeader | When exporting to CSV format, this event allows you to add a header to the output document. |
| | CustomizeSheet | When exporting to CSV format, this event allows you to customize the output document's settings. |
| | ExportProgress | Fires repeatedly while the data is being exported. |
| | PropertyChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanging | (Inherited from **Xamarin.Forms.BindableObject**) |

**See Also**
CsvExportOptions Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# CsvExportOptions Constructor

Initializes a new instance of the CsvExportOptions class with default settings.

**Overload List**

| Name | Description |
|---|---|
| CsvExportOptions() | Initializes a new instance of the CsvExportOptions class with default settings. |
| CsvExportOptions(string separator, Encoding encoding) | Initializes a new instance of the CsvExportOptions class with the specified separator and encoding settings. |

**See Also**
CsvExportOptions Class
CsvExportOptions Members
DevExpress.Mobile.Export Namespace
CsvExportOptions Overload List

© 2018 Developer Express Inc.

# CsvExportOptions Constructor

Initializes a new instance of the CsvExportOptions class with default settings.

**Namespace:** DevExpress.Mobile.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

**Syntax**
**C#**

```
public CsvExportOptions()
```

**See Also**
CsvExportOptions Class
CsvExportOptions Members
DevExpress.Mobile.Export Namespace
CsvExportOptions Overload List

© 2018 Developer Express Inc.

# CsvExportOptions Constructor (string, Encoding)

Initializes a new instance of the CsvExportOptions class with the specified separator and encoding settings.

**Namespace:** DevExpress.Mobile.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public CsvExportOptions(
   string separator,
   Encoding encoding
)
```

### Parameters

*separator*
A System.String value specifying the symbols that are used to separate the text elements in the created CSV file. This value is assigned to the TextExportOptionsBase.Separator property.
*encoding*
A System.Text.Encoding descendant specifying the encoding of the created CSV document. This value is assigned to the TextExportOptionsBase.Encoding property.

### See Also

CsvExportOptions Class
CsvExportOptions Members
DevExpress.Mobile.Export Namespace
CsvExportOptions Overload List

© 2018 Developer Express Inc.

# CsvExportOptions Properties

Contains options which define how grid data is exported to CSV format.

The following tables list the members exposed by the CsvExportOptions type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | CsvExportOptions | Overloaded. Initializes a new instance of the CsvExportOptions class with default settings. |

### Public Properties

| | Name | Description |
|---|---|---|
| | BindingContext | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Encoding | Gets or sets the encoding of the text-based file to which grid's data is exported. (Inherited from **TextExportOptionsBase**) |
| | Separator | Gets or sets the symbol(s) to separate text elements when grid data is exported to a text-based file. (Inherited from **TextExportOptionsBase**) |
| | WritePreamble | Gets or sets whether to write a prefix that is the sequence of bytes that specifies the encoding used. |

### Public Methods

| | Name | Description |
|---|---|---|
| | Assign | Copies all settings from the options object passed as a parameter. |
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ToString | (Inherited from **System.Object**) |

### Public Fields

| | Name | Description |
|---|---|---|
| | BindingContextProperty | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SeparatorProperty | Identifies the TextExportOptionsBase.Separator bindable property. (Inherited from **TextExportOptionsBase**) |
| | WritePreambleProperty | Identifies the WritePreamble bindable property. |

### Public Events

| | Name | Description |
|---|---|---|
| | BindingContextChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | CustomizeCell | When exporting data from the grid to CSV format, this event allows you to customize a cell in the output document. |
| | CustomizeDataArea | When exporting to CSV format, this event allows you to customize data in the output document. |
| | CustomizeFooter | When exporting to CSV format, this event allows you to add a footer to the output document. |
| | CustomizeHeader | When exporting to CSV format, this event allows you to add a header to the output document. |
| | CustomizeSheet | When exporting to CSV format, this event allows you to customize the output document's settings. |
| | ExportProgress | Fires repeatedly while the data is being exported. |
| | PropertyChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanging | (Inherited from **Xamarin.Forms.BindableObject**) |

**See Also**
CsvExportOptions Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# WritePreamble Property

Gets or sets whether to write a prefix that is the sequence of bytes that specifies the encoding used.

**Namespace:** DevExpress.Mobile.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public bool WritePreamble { get; set; }
```

### Property Value

**true** to write a byte array containing a sequence of bytes that specifies the encoding used; otherwise, **false**.

### See Also

CsvExportOptions Class
CsvExportOptions Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# CsvExportOptions Events

Contains options which define how grid data is exported to CSV format.

The following tables list the members exposed by the CsvExportOptions type.

**Public Events**

| | Name | Description |
|---|---|---|
| | BindingContextChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | CustomizeCell | When exporting data from the grid to CSV format, this event allows you to customize a cell in the output document. |
| | CustomizeDataArea | When exporting to CSV format, this event allows you to customize data in the output document. |
| | CustomizeFooter | When exporting to CSV format, this event allows you to add a footer to the output document. |
| | CustomizeHeader | When exporting to CSV format, this event allows you to add a header to the output document. |
| | CustomizeSheet | When exporting to CSV format, this event allows you to customize the output document's settings. |
| | ExportProgress | Fires repeatedly while the data is being exported. |
| | PropertyChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanging | (Inherited from **Xamarin.Forms.BindableObject**) |

**See Also**
CsvExportOptions Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# CustomizeCell Event

When exporting data from the grid to CSV format, this event allows you to customize a cell in the output document.

**Namespace:** DevExpress.Mobile.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public event CustomizeCellEventHandler CustomizeCell
```

### Event Data

The event handler receives an argument of type CustomizeCellEventArgs containing data related to this event.

The following **CustomizeCellEventArgs** properties provide information specific to this event.

| Property | Description |
|---|---|
| ColumnFieldName | Gets or sets the data source field name to which the current cell corresponds. |
| DataSourceOwner | |
| DataSourceRowIndex | |
| DocumentRow | |
| Formatting | Gets or sets the object that defines the cell's formatting settings (font, alignment, background color, format string, etc.) |
| Handled | |
| Hyperlink | Specifies the hyperlink, which belongs to the currently processed cell. |
| RowHandle | |
| Value | Specifies the value of the cell currently being processed. |

### Remarks

The **CustomizeCell** event allows you to customize cells in the exported document by applying the following modifications.

- Substitute a cell value (CustomizeCellEventArgs.Value).
- Insert a hyperlink in a cell (CustomizeCellEventArgs.Hyperlink).
- Customize the cell's format, font and background color (CustomizeCellEventArgsBase.Formatting).

To apply changes made in your **CustomizeCell** event handler, set the event's **Handled** parameter to **true**. Otherwise, a cell will be exported using the default settings.

### See Also

CsvExportOptions Class
CsvExportOptions Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# CustomizeDataArea Event

When exporting to CSV format, this event allows you to customize data in the output document.

**Namespace:** DevExpress.Mobile.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public event CustomizeDataAreaEventHandler CustomizeDataArea
```

### Event Data

The event handler receives an argument of type CustomizeDataAreaEventArgs containing data related to this event.
The following **CustomizeDataAreaEventArgs** properties provide information specific to this event.

| Property | Description |
|---|---|
| Context | Specifies the context in which a data area is customized. |
| DataSourceOwner | |
| DataSourceRowIndex | |
| DocumentRow | |
| Handled | |
| RowHandle | |

### Remarks

The **CustomizeDataArea** event is raised after data is exported.

### See Also

CsvExportOptions Class
CsvExportOptions Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# CustomizeFooter Event

When exporting to CSV format, this event allows you to add a footer to the output document.

**Namespace:** DevExpress.Mobile.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public event CustomizeFooterEventHandler CustomizeFooter
```

### Event Data

The event handler receives an argument of type CustomizeFooterEventArgs containing data related to this event.
The following **CustomizeFooterEventArgs** properties provide information specific to this event.

| Property | Description |
|---|---|
| Context | Specifies the context in which a footer is customized. |

### Remarks

The **CustomizeFooter** event is raised after data is exported. You can handle it to add new footer rows to the output document.

### See Also

CsvExportOptions Class
CsvExportOptions Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# CustomizeHeader Event

When exporting to CSV format, this event allows you to add a header to the output document.

**Namespace:** DevExpress.Mobile.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public event CustomizeHeaderEventHandler CustomizeHeader
```

### Event Data

The event handler receives an argument of type CustomizeHeaderEventArgs containing data related to this event.
The following **CustomizeHeaderEventArgs** properties provide information specific to this event.

| Property | Description |
|---|---|
| Context | Specifies the context in which a header is customized. |
| DataSourceOwner | |
| DataSourceRowIndex | |
| DocumentRow | |
| Handled | |
| RowHandle | |

### Remarks

The **CustomizeHeader** event is raised before data is exported. You can handle it to add new header rows to the output document.

### See Also

CsvExportOptions Class
CsvExportOptions Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# CustomizeSheet Event

When exporting to CSV format, this event allows you to customize the output document's settings.

**Namespace:** DevExpress.Mobile.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public event CustomizeSheetEventHandler CustomizeSheet
```

### Event Data

The event handler receives an argument of type CustomizeSheetEventArgs containing data related to this event.

The following **CustomizeSheetEventArgs** properties provide information specific to this event.

| Property | Description |
|---|---|
| Context | Specifies the context in which an output document is customized. |

### Remarks

The **CustomizeSheet** event allows you to do the following:

- Add the AutoFilter button to cells in a specified range (**CustomizeSheetEventArgs.Context.AddAutoFilter**).
- Anchor the output document's header to the top and specify its fixed height (**CustomizeSheetEventArgs.Context.SetFixedHeader**).

### See Also

CsvExportOptions Class
CsvExportOptions Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# ExportProgress Event

Fires repeatedly while the data is being exported.

**Namespace:** DevExpress.Mobile.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public event ExportProgressCallback ExportProgress
```

### Event Data

The event handler receives an argument of type System.ComponentModel.ProgressChangedEventArgs containing data related to this event.

### See Also

CsvExportOptions Class
CsvExportOptions Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# CsvExportOptions Methods

Contains options which define how grid data is exported to CSV format.

The following tables list the members exposed by the CsvExportOptions type.

**Public Methods**

| | Name | Description |
|---|---|---|
| | Assign | Copies all settings from the options object passed as a parameter. |
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ToString | (Inherited from **System.Object**) |

**See Also**
CsvExportOptions Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# Assign Method

Copies all settings from the options object passed as a parameter.

**Namespace:** DevExpress.Mobile.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public override void Assign(
   ExportOptionsBase source
)
```

### Parameters

*source*
A CsvExportOptions object (which is the ExportOptionsBase descendant) whose settings are assigned to the current object.

### Remarks

The **Assign** method allows the settings of the specified CsvExportOptions object to be copied to the current CsvExportOptions object.

### See Also

CsvExportOptions Class
CsvExportOptions Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# DataAwareExportOptionsBase Class

Contains options which define how data shaping settings are exported from the grid to XLS and XLSX formats.

**Namespace:** DevExpress.Mobile.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public abstract class DataAwareExportOptionsBase : XlsExportOptionsBase, IDataAwareExportOptions
```

### Remarks

The **DataAwareExportOptionsBase** class serves as a base for classes that define XLS or XLSX-specific export options (XlsExportOptions and XlsxExportOptions) and provides the following properties to specify data shaping in the output document.

- AllowGrouping, GroupState
- AllowSortingAndFiltering
- ShowGroupSummaries
- ShowTotalSummaries
- ShowColumnHeaders

### Inheritance Hierarchy

System.Object
  Xamarin.Forms.BindableObject
    ExportOptionsBase
      XlsExportOptionsBase
        **DataAwareExportOptionsBase**
          Derived classes

### See Also

DataAwareExportOptionsBase Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# DataAwareExportOptionsBase Members

Contains options which define how data shaping settings are exported from the grid to XLS and XLSX formats.

The following tables list the members exposed by the DataAwareExportOptionsBase type.

### Public Properties

| | Name | Description |
|---|---|---|
| | AllowGrouping | Gets or sets whether data groups are exported from the source grid to the output document. |
| | AllowLookupValues | Gets or sets whether lookup values of picker columns are exported. |
| | AllowSortingAndFiltering | Gets or sets whether the sorting and filtering functionality is enabled for columns in the exported document. |
| | BindingContext | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ExportHyperlinks | Gets or sets a value indicating whether hyperlinks should be exported to XLS or XLSX. (Inherited from **XlsExportOptionsBase**) |
| | GroupState | Gets or sets whether data groups are collapsed or expanded in the exported document. |
| | RawDataMode | Specifies whether a grid's actual data should be exported to XLS or XLSX (excluding all non-relevant elements, such as images, graphics, font and appearance settings). (Inherited from **XlsExportOptionsBase**) |
| | SheetName | Gets or sets the name of the sheet in the created XLS or XLSX file to which a document is exported. (Inherited from **XlsExportOptionsBase**) |
| | ShowColumnHeaders | Gets or sets whether column headers are visible in the exported document. |
| | ShowGridLines | Gets or sets a value indicating whether grid lines should be visible in the resulting XLS or XLSX file. (Inherited from **XlsExportOptionsBase**) |
| | ShowGroupSummaries | Gets or sets whether group summaries are enabled in the exported document. |
| | ShowTotalSummaries | Gets or sets whether total summaries are enabled in the exported document. |

### Public Methods

| | Name | Description |
|---|---|---|
| | Assign | Copies all settings from the options object passed as a parameter. |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ToString | (Inherited from **System.Object**) |

**Public Fields**

| | Name | Description |
|---|---|---|
| S | AllowGroupingProperty | Identifies the AllowGrouping bindable property. |
| S | AllowLookupValuesProperty | Identifies the AllowLookupValues bindable property. |
| S | AllowSortingAndFilteringProperty | Identifies the AllowSortingAndFiltering bindable property. |
| S | BindingContextProperty | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ExportHyperlinksProperty | Identifies the XlsExportOptionsBase.ExportHyperlinks bindable property. (Inherited from **XlsExportOptionsBase**) |
| S | GroupStateProperty | Identifies the GroupState bindable property. |
| S | RawDataModeProperty | Identifies the XlsExportOptionsBase.RawDataMode bindable property. (Inherited from **XlsExportOptionsBase**) |
| S | SheetNameProperty | Identifies the XlsExportOptionsBase.SheetName bindable property. (Inherited from **XlsExportOptionsBase**) |
| S | ShowColumnHeadersProperty | Identifies the ShowColumnHeaders bindable property. |
| S | ShowGridLinesProperty | Identifies the XlsExportOptionsBase.ShowGridLines |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | | bindable property. (Inherited from **XlsExportOptionsBase**) |
| S | ShowGroupSummariesProperty | Identifies the ShowGroupSummaries bindable property. |
| S | ShowPageTitleProperty | Identifies the ShowPageTitleProperty bindable property. |
| S | ShowTotalSummariesProperty | Identifies the ShowTotalSummaries bindable property. |

**Public Events**

| | Name | Description |
|---|---|---|
| | BindingContextChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | CustomizeCell | When exporting data from the grid to XLS or XLSX format, this event allows you to customize a cell in the output document. |
| | CustomizeDataArea | When exporting to XLS or XLSX format, this event allows you to customize data in the output document. |
| | CustomizeFooter | When exporting to XLS or XLSX format, this event allows you to add a footer to the output document. |
| | CustomizeHeader | When exporting to XLS or XLSX format, this event allows you to add a header to the output document. |
| | CustomizeSheet | When exporting to XLS or XLSX format, this event allows you to customize the output document's settings. |
| | ExportProgress | Fires repeatedly while the data is being exported. |
| | PropertyChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanging | (Inherited from **Xamarin.Forms.BindableObject**) |

**See Also**
DataAwareExportOptionsBase Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# DataAwareExportOptionsBase Properties

Contains options which define how data shaping settings are exported from the grid to XLS and XLSX formats.

The following tables list the members exposed by the DataAwareExportOptionsBase type.

**Public Properties**

| | Name | Description |
|---|---|---|
| | AllowGrouping | Gets or sets whether data groups are exported from the source grid to the output document. |
| | AllowLookupValues | Gets or sets whether lookup values of picker columns are exported. |
| | AllowSortingAndFiltering | Gets or sets whether the sorting and filtering functionality is enabled for columns in the exported document. |
| | BindingContext | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ExportHyperlinks | Gets or sets a value indicating whether hyperlinks should be exported to XLS or XLSX. (Inherited from **XlsExportOptionsBase**) |
| | GroupState | Gets or sets whether data groups are collapsed or expanded in the exported document. |
| | RawDataMode | Specifies whether a grid's actual data should be exported to XLS or XLSX (excluding all non-relevant elements, such as images, graphics, font and appearance settings). (Inherited from **XlsExportOptionsBase**) |
| | SheetName | Gets or sets the name of the sheet in the created XLS or XLSX file to which a document is exported. (Inherited from **XlsExportOptionsBase**) |
| | ShowColumnHeaders | Gets or sets whether column headers are visible in the exported document. |
| | ShowGridLines | Gets or sets a value indicating whether grid lines should be visible in the resulting XLS or XLSX file. (Inherited from **XlsExportOptionsBase**) |
| | ShowGroupSummaries | Gets or sets whether group summaries are enabled in the exported document. |
| | ShowTotalSummaries | Gets or sets whether total summaries are enabled in the exported document. |

**Public Methods**

| | Name | Description |
|---|---|---|
| | Assign | Copies all settings from the options object passed as a parameter. |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ToString | (Inherited from **System.Object**) |

**Public Fields**

| | Name | Description |
|---|---|---|
| S | AllowGroupingProperty | Identifies the AllowGrouping bindable property. |
| S | AllowLookupValuesProperty | Identifies the AllowLookupValues bindable property. |
| S | AllowSortingAndFilteringProperty | Identifies the AllowSortingAndFiltering bindable property. |
| S | BindingContextProperty | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ExportHyperlinksProperty | Identifies the XlsExportOptionsBase.ExportHyperlinks bindable property. (Inherited from **XlsExportOptionsBase**) |
| S | GroupStateProperty | Identifies the GroupState bindable property. |
| S | RawDataModeProperty | Identifies the XlsExportOptionsBase.RawDataMode bindable property. (Inherited from **XlsExportOptionsBase**) |
| S | SheetNameProperty | Identifies the XlsExportOptionsBase.SheetName bindable property. (Inherited from **XlsExportOptionsBase**) |
| S | ShowColumnHeadersProperty | Identifies the ShowColumnHeaders bindable property. |
| S | ShowGridLinesProperty | Identifies the XlsExportOptionsBase.ShowGridLines |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | | bindable property. (Inherited from **XlsExportOptionsBase**) |
| S | ShowGroupSummariesProperty | Identifies the ShowGroupSummaries bindable property. |
| S | ShowPageTitleProperty | Identifies the ShowPageTitleProperty bindable property. |
| S | ShowTotalSummariesProperty | Identifies the ShowTotalSummaries bindable property. |

**Public Events**

| | Name | Description |
|---|---|---|
| | BindingContextChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | CustomizeCell | When exporting data from the grid to XLS or XLSX format, this event allows you to customize a cell in the output document. |
| | CustomizeDataArea | When exporting to XLS or XLSX format, this event allows you to customize data in the output document. |
| | CustomizeFooter | When exporting to XLS or XLSX format, this event allows you to add a footer to the output document. |
| | CustomizeHeader | When exporting to XLS or XLSX format, this event allows you to add a header to the output document. |
| | CustomizeSheet | When exporting to XLS or XLSX format, this event allows you to customize the output document's settings. |
| | ExportProgress | Fires repeatedly while the data is being exported. |
| | PropertyChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanging | (Inherited from **Xamarin.Forms.BindableObject**) |

**See Also**
DataAwareExportOptionsBase Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# AllowGrouping Property

Gets or sets whether data groups are exported from the source grid to the output document.

**Namespace:** DevExpress.Mobile.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public DefaultBoolean AllowGrouping { get; set; }
```

### Property Value

A value that specifies whether data groups are exported from the source grid to the output document.

### See Also

DataAwareExportOptionsBase Class
DataAwareExportOptionsBase Members
DevExpress.Mobile.Export Namespace
GroupState

© 2018 Developer Express Inc.

# AllowLookupValues Property

Gets or sets whether lookup values of picker columns are exported.

**Namespace:** DevExpress.Mobile.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public DefaultBoolean AllowLookupValues { get; set; }
```

### Property Value

A value that specifies whether lookup values of picker columns are exported.

### See Also

DataAwareExportOptionsBase Class
DataAwareExportOptionsBase Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# AllowSortingAndFiltering Property

Gets or sets whether the sorting and filtering functionality is enabled for columns in the exported document.

**Namespace:** DevExpress.Mobile.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public DefaultBoolean AllowSortingAndFiltering { get; set; }
```

### Property Value

A value that specifies whether the sorting and filtering functionality is enabled for columns in the exported document.

### See Also

DataAwareExportOptionsBase Class
DataAwareExportOptionsBase Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# GroupState Property

Gets or sets whether data groups are collapsed or expanded in the exported document.

**Namespace:** DevExpress.Mobile.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public GroupState GroupState { get; set; }
```

### Property Value

A value that specifies whether data groups are collapsed or expanded in the exported document.

### See Also

DataAwareExportOptionsBase Class
DataAwareExportOptionsBase Members
DevExpress.Mobile.Export Namespace
AllowGrouping

© 2018 Developer Express Inc.

# ShowColumnHeaders Property

Gets or sets whether column headers are visible in the exported document.

**Namespace:** DevExpress.Mobile.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public DefaultBoolean ShowColumnHeaders { get; set; }
```

### Property Value

A value that specifies whether column headers are visible in the exported document.

### See Also

DataAwareExportOptionsBase Class
DataAwareExportOptionsBase Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# ShowGroupSummaries Property

Gets or sets whether group summaries are enabled in the exported document.

**Namespace:** DevExpress.Mobile.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public DefaultBoolean ShowGroupSummaries { get; set; }
```

### Property Value

A value that specifies whether group summaries are enabled in the exported document.

### See Also

DataAwareExportOptionsBase Class
DataAwareExportOptionsBase Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# ShowTotalSummaries Property

Gets or sets whether total summaries are enabled in the exported document.

**Namespace:** DevExpress.Mobile.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public DefaultBoolean ShowTotalSummaries { get; set; }
```

### Property Value

A value that specifies whether total summaries are enabled in the exported document.

### See Also

DataAwareExportOptionsBase Class
DataAwareExportOptionsBase Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# DataAwareExportOptionsBase Events

Contains options which define how data shaping settings are exported from the grid to XLS and XLSX formats.

The following tables list the members exposed by the DataAwareExportOptionsBase type.

**Public Events**

| | Name | Description |
|---|---|---|
| | BindingContextChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | CustomizeCell | When exporting data from the grid to XLS or XLSX format, this event allows you to customize a cell in the output document. |
| | CustomizeDataArea | When exporting to XLS or XLSX format, this event allows you to customize data in the output document. |
| | CustomizeFooter | When exporting to XLS or XLSX format, this event allows you to add a footer to the output document. |
| | CustomizeHeader | When exporting to XLS or XLSX format, this event allows you to add a header to the output document. |
| | CustomizeSheet | When exporting to XLS or XLSX format, this event allows you to customize the output document's settings. |
| | ExportProgress | Fires repeatedly while the data is being exported. |
| | PropertyChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanging | (Inherited from **Xamarin.Forms.BindableObject**) |

**See Also**

DataAwareExportOptionsBase Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# CustomizeCell Event

When exporting data from the grid to XLS or XLSX format, this event allows you to customize a cell in the output document.

**Namespace:** DevExpress.Mobile.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public event CustomizeCellEventHandler CustomizeCell
```

### Event Data

The event handler receives an argument of type CustomizeCellEventArgs containing data related to this event.

The following **CustomizeCellEventArgs** properties provide information specific to this event.

| Property | Description |
|---|---|
| ColumnFieldName | Gets or sets the data source field name to which the current cell corresponds. |
| DataSourceOwner | |
| DataSourceRowIndex | |
| DocumentRow | |
| Formatting | Gets or sets the object that defines the cell's formatting settings (font, alignment, background color, format string, etc.) |
| Handled | |
| Hyperlink | Specifies the hyperlink, which belongs to the currently processed cell. |
| RowHandle | |
| Value | Specifies the value of the cell currently being processed. |

### Remarks

The **CustomizeCell** event allows you to customize cells in the exported document by applying the following modifications.

- Substitute a cell value (CustomizeCellEventArgs.Value).
- Insert a hyperlink in a cell (CustomizeCellEventArgs.Hyperlink).
- Customize the cell's format, font and background color (CustomizeCellEventArgs.Formatting).

To apply changes made in your **CustomizeCell** event handler, set the event's **Handled** parameter to **true**. Otherwise, a cell will be exported using the default settings.

### See Also

DataAwareExportOptionsBase Class
DataAwareExportOptionsBase Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# CustomizeDataArea Event

When exporting to XLS or XLSX format, this event allows you to customize data in the output document.

**Namespace:** DevExpress.Mobile.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public event CustomizeDataAreaEventHandler CustomizeDataArea
```

### Event Data

The event handler receives an argument of type CustomizeDataAreaEventArgs containing data related to this event.
The following **CustomizeDataAreaEventArgs** properties provide information specific to this event.

| Property | Description |
|---|---|
| Context | Specifies the context in which a data area is customized. |
| DataSourceOwner | |
| DataSourceRowIndex | |
| DocumentRow | |
| Handled | |
| RowHandle | |

### Remarks

The **CustomizeDataArea** event is raised after data is exported.

### See Also

DataAwareExportOptionsBase Class
DataAwareExportOptionsBase Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# CustomizeFooter Event

When exporting to XLS or XLSX format, this event allows you to add a footer to the output document.

**Namespace:** DevExpress.Mobile.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public event CustomizeFooterEventHandler CustomizeFooter
```

### Event Data

The event handler receives an argument of type CustomizeFooterEventArgs containing data related to this event.
The following **CustomizeFooterEventArgs** properties provide information specific to this event.

| Property | Description |
|---|---|
| Context | Specifies the context in which a footer is customized. |

### Remarks

The **CustomizeFooter** event is raised after data is exported. You can handle it to add new footer rows to the output document.

### See Also

DataAwareExportOptionsBase Class
DataAwareExportOptionsBase Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# CustomizeHeader Event

When exporting to XLS or XLSX format, this event allows you to add a header to the output document.

**Namespace:** DevExpress.Mobile.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public event CustomizeHeaderEventHandler CustomizeHeader
```

### Event Data

The event handler receives an argument of type CustomizeHeaderEventArgs containing data related to this event.

The following **CustomizeHeaderEventArgs** properties provide information specific to this event.

| Property | Description |
|---|---|
| Context | Specifies the context in which a header is customized. |
| DataSourceOwner | |
| DataSourceRowIndex | |
| DocumentRow | |
| Handled | |
| RowHandle | |

### Remarks

The **CustomizeHeader** event is raised before data is exported. You can handle it to add new header rows to the output document.

### See Also

DataAwareExportOptionsBase Class
DataAwareExportOptionsBase Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# CustomizeSheet Event

When exporting to XLS or XLSX format, this event allows you to customize the output document's settings.

**Namespace:** DevExpress.Mobile.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public event CustomizeSheetEventHandler CustomizeSheet
```

### Event Data

The event handler receives an argument of type CustomizeSheetEventArgs containing data related to this event.

The following **CustomizeSheetEventArgs** properties provide information specific to this event.

| Property | Description |
|---|---|
| Context | Specifies the context in which an output document is customized. |

### Remarks

The **CustomizeSheet** event allows you to do the following:

- Add the AutoFilter button to cells in a specified range (**CustomizeSheetEventArgs.Context.AddAutoFilter**).
- Anchor the output document's header to the top and specify its fixed height (**CustomizeSheetEventArgs.Context.SetFixedHeader**).

### See Also

DataAwareExportOptionsBase Class
DataAwareExportOptionsBase Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# ExportProgress Event

Fires repeatedly while the data is being exported.

**Namespace:** DevExpress.Mobile.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public event ExportProgressCallback ExportProgress
```

### Event Data

The event handler receives an argument of type System.ComponentModel.ProgressChangedEventArgs containing data related to this event.

### See Also

DataAwareExportOptionsBase Class
DataAwareExportOptionsBase Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# DataAwareExportOptionsBase Methods

Contains options which define how data shaping settings are exported from the grid to XLS and XLSX formats.

The following tables list the members exposed by the DataAwareExportOptionsBase type.

**Public Methods**

| | Name | Description |
|---|---|---|
| | Assign | Copies all settings from the options object passed as a parameter. |
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ToString | (Inherited from **System.Object**) |

**See Also**

DataAwareExportOptionsBase Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# Assign Method

Copies all settings from the options object passed as a parameter.

**Namespace:** DevExpress.Mobile.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public override void Assign(
   ExportOptionsBase source
)
```

### Parameters

*source*
A DataAwareExportOptionsBase object (which is the ExportOptionsBase descendant) whose settings are assigned to the current object.

### Remarks

The **Assign** method allows the settings of the specified DataAwareExportOptionsBase object to be copied to the current DataAwareExportOptionsBase object.

### See Also

DataAwareExportOptionsBase Class
DataAwareExportOptionsBase Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# ExportOptionsBase Class

The base class for objects which define specific export options.

**Namespace:** DevExpress.Mobile.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public abstract class ExportOptionsBase : BindableObject, ICloneable
```

### Remarks

The **ExportOptionsBase** class serves as a base for classes that define specific export options (e.g., TextExportOptionsBase, XlsExportOptionsBase, etc.).

### Inheritance Hierarchy

System.Object
  Xamarin.Forms.BindableObject
    **ExportOptionsBase**
      Derived classes

### See Also

ExportOptionsBase Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# ExportOptionsBase Members

The base class for objects which define specific export options.

The following tables list the members exposed by the ExportOptionsBase type.

### Public Properties

| | Name | Description |
|---|---|---|
| | BindingContext | (Inherited from **Xamarin.Forms.BindableObject**) |

### Public Methods

| | Name | Description |
|---|---|---|
| | Assign | Copies all settings from the options object passed as a parameter. |
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ToString | (Inherited from **System.Object**) |

### Public Fields

| | Name | Description |
|---|---|---|
| S | BindingContextProperty | (Inherited from **Xamarin.Forms.BindableObject**) |

### Public Events

| | Name | Description |
|---|---|---|
| | BindingContextChanged | (Inherited from **Xamarin.Forms.BindableObject**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | PropertyChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanging | (Inherited from **Xamarin.Forms.BindableObject**) |

**See Also**
ExportOptionsBase Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# ExportOptionsBase Methods

The base class for objects which define specific export options.

The following tables list the members exposed by the ExportOptionsBase type.

**Public Methods**

| | Name | Description |
|---|---|---|
| | Assign | Copies all settings from the options object passed as a parameter. |
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ToString | (Inherited from **System.Object**) |

**See Also**
ExportOptionsBase Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# Assign Method

Copies all settings from the options object passed as a parameter.

**Namespace:** DevExpress.Mobile.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public abstract void Assign(
   ExportOptionsBase source
)
```

### Parameters

*source*
An ExportOptionsBase object whose settings are assigned to the current object.

### Remarks

The **Assign** method allows the settings of the specified ExportOptionsBase object to be copied to the current ExportOptionsBase object.

### See Also

ExportOptionsBase Class
ExportOptionsBase Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# TextExportOptionsBase Class

The base class for objects which define export options specific to text-based formats.

**Namespace:** DevExpress.Mobile.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public abstract class TextExportOptionsBase : ExportOptionsBase
```

### Inheritance Hierarchy

System.Object
  Xamarin.Forms.BindableObject
    ExportOptionsBase
      **TextExportOptionsBase**
        Derived classes

### See Also

TextExportOptionsBase Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# TextExportOptionsBase Members

The base class for objects which define export options specific to text-based formats.

The following tables list the members exposed by the TextExportOptionsBase type.

### Public Properties

| | Name | Description |
|---|---|---|
| | BindingContext | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Encoding | Gets or sets the encoding of the text-based file to which grid's data is exported. |
| | Separator | Gets or sets the symbol(s) to separate text elements when grid data is exported to a text-based file. |

### Public Methods

| | Name | Description |
|---|---|---|
| | Assign | Copies all settings from the options object passed as a parameter. |
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ReferenceEquals | (Inherited from **System.Object**) |
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ToString | (Inherited from **System.Object**) |

### Public Fields

| | Name | Description |
|---|---|---|
| | BindingContextProperty | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SeparatorProperty | Identifies the Separator bindable property. |

© 2018 Developer Express Inc.

**Public Events**

| | Name | Description |
|---|---|---|
| | BindingContextChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanging | (Inherited from **Xamarin.Forms.BindableObject**) |

**See Also**
TextExportOptionsBase Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# TextExportOptionsBase Properties

The base class for objects which define export options specific to text-based formats.

The following tables list the members exposed by the TextExportOptionsBase type.

### Public Properties

| | Name | Description |
|---|---|---|
| | BindingContext | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Encoding | Gets or sets the encoding of the text-based file to which grid's data is exported. |
| | Separator | Gets or sets the symbol(s) to separate text elements when grid data is exported to a text-based file. |

### Public Methods

| | Name | Description |
|---|---|---|
| | Assign | Copies all settings from the options object passed as a parameter. |
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ReferenceEquals | (Inherited from **System.Object**) |
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ToString | (Inherited from **System.Object**) |

### Public Fields

| | Name | Description |
|---|---|---|
| | BindingContextProperty | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SeparatorProperty | Identifies the Separator bindable property. |

© 2018 Developer Express Inc.

**Public Events**

| | Name | Description |
|---|---|---|
| | BindingContextChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanging | (Inherited from **Xamarin.Forms.BindableObject**) |

**See Also**
TextExportOptionsBase Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# Encoding Property

Gets or sets the encoding of the text-based file to which grid's data is exported.

**Namespace:** DevExpress.Mobile.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public Encoding Encoding { get; set; }
```

### Property Value

A System.Text.Encoding class descendant specifying the character encoding.

### See Also

TextExportOptionsBase Class
TextExportOptionsBase Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# Separator Property

Gets or sets the symbol(s) to separate text elements when grid data is exported to a text-based file.

**Namespace:** DevExpress.Mobile.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public string Separator { get; set; }
```

### Property Value

A string used as a separator.

### See Also

TextExportOptionsBase Class
TextExportOptionsBase Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# TextExportOptionsBase Methods

The base class for objects which define export options specific to text-based formats.

The following tables list the members exposed by the TextExportOptionsBase type.

**Public Methods**

| | Name | Description |
|---|---|---|
| | Assign | Copies all settings from the options object passed as a parameter. |
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Equals | (Inherited from **System.Object**) |
| S | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ToString | (Inherited from **System.Object**) |

**See Also**
TextExportOptionsBase Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# Assign Method

Copies all settings from the options object passed as a parameter.

**Namespace:** DevExpress.Mobile.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public override void Assign(
   ExportOptionsBase source
)
```

### Parameters

*source*
A TextExportOptionsBase object (which is the ExportOptionsBase descendant) whose settings are assigned to the current object.

### Remarks

The **Assign** method allows the settings of the specified TextExportOptionsBase object to be copied to the current TextExportOptionsBase object.

### See Also

TextExportOptionsBase Class
TextExportOptionsBase Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# XlsExportOptions Class

Contains options which define how grid data is exported to XLS format.

**Namespace:** DevExpress.Mobile.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public class XlsExportOptions : DataAwareExportOptionsBase
```

### Remarks

To export data from the GridControl to XLS format, use the GridControl.ExportToExcel method with the ExportTarget.Xls value passed as the *format* parameter.

To specify XLS export settings, create an instance of the **XlsExportOptions** class and assign it to the GridControl.OptionsExportXls property.

### Inheritance Hierarchy

System.Object
  Xamarin.Forms.BindableObject
    ExportOptionsBase
      XlsExportOptionsBase
        DataAwareExportOptionsBase
          **XlsExportOptions**

### See Also

XlsExportOptions Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# XlsExportOptions Members

Contains options which define how grid data is exported to XLS format.

The following tables list the members exposed by the XlsExportOptions type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | XlsExportOptions | Initializes a new instance of the XlsExportOptions class with default settings. |

### Public Properties

| | Name | Description |
|---|---|---|
| | AllowGrouping | Gets or sets whether data groups are exported from the source grid to the output document. (Inherited from **DataAwareExportOptionsBase**) |
| | AllowLookupValues | Gets or sets whether lookup values of picker columns are exported. (Inherited from **DataAwareExportOptionsBase**) |
| | AllowSortingAndFiltering | Gets or sets whether the sorting and filtering functionality is enabled for columns in the exported document. (Inherited from **DataAwareExportOptionsBase**) |
| | BindingContext | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ExportHyperlinks | Gets or sets a value indicating whether hyperlinks should be exported to XLS or XLSX. (Inherited from **XlsExportOptionsBase**) |
| | GroupState | Gets or sets whether data groups are collapsed or expanded in the exported document. (Inherited from **DataAwareExportOptionsBase**) |
| | RawDataMode | Specifies whether a grid's actual data should be exported to XLS or XLSX (excluding all non-relevant elements, such as images, graphics, font and appearance settings). (Inherited from **XlsExportOptionsBase**) |
| | SheetName | Gets or sets the name of the sheet in the created XLS or XLSX file to which a document is exported. (Inherited from **XlsExportOptionsBase**) |
| | ShowColumnHeaders | Gets or sets whether column headers are visible in the exported document. (Inherited from **DataAwareExportOptionsBase**) |
| | ShowGridLines | Gets or sets a value indicating whether grid lines should be visible in the |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | | resulting XLS or XLSX file. (Inherited from **XlsExportOptionsBase**) |
| | ShowGroupSummaries | Gets or sets whether group summaries are enabled in the exported document. (Inherited from **DataAwareExportOptionsBase**) |
| | ShowTotalSummaries | Gets or sets whether total summaries are enabled in the exported document. (Inherited from **DataAwareExportOptionsBase**) |

**Public Methods**

| | Name | Description |
|---|---|---|
| | Assign | Copies all settings from the options object passed as a parameter. (Inherited from **DataAwareExportOptionsBase**) |
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ReferenceEquals | (Inherited from **System.Object**) |
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ToString | (Inherited from **System.Object**) |

**Public Fields**

| | Name | Description |
|---|---|---|
| | AllowGroupingProperty | Identifies the DataAwareExportOptionsBase.AllowGrouping bindable property. (Inherited from **DataAwareExportOptionsBase**) |
| | AllowLookupValuesProperty | Identifies the DataAwareExportOptionsBase.AllowLookupValues bindable property. (Inherited from **DataAwareExportOptionsBase**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | AllowSortingAndFilteringProperty | Identifies the DataAwareExportOptionsBase.AllowSortingAndFiltering bindable property. (Inherited from **DataAwareExportOptionsBase**) |
| | BindingContextProperty | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ExportHyperlinksProperty | Identifies the XlsExportOptionsBase.ExportHyperlinks bindable property. (Inherited from **XlsExportOptionsBase**) |
| | GroupStateProperty | Identifies the DataAwareExportOptionsBase.GroupState bindable property. (Inherited from **DataAwareExportOptionsBase**) |
| | RawDataModeProperty | Identifies the XlsExportOptionsBase.RawDataMode bindable property. (Inherited from **XlsExportOptionsBase**) |
| | SheetNameProperty | Identifies the XlsExportOptionsBase.SheetName bindable property. (Inherited from **XlsExportOptionsBase**) |
| | ShowColumnHeadersProperty | Identifies the DataAwareExportOptionsBase.ShowColumnHeaders bindable property. (Inherited from **DataAwareExportOptionsBase**) |
| | ShowGridLinesProperty | Identifies the XlsExportOptionsBase.ShowGridLines bindable property. (Inherited from **XlsExportOptionsBase**) |
| | ShowGroupSummariesProperty | Identifies the DataAwareExportOptionsBase.ShowGroupSummaries bindable property. (Inherited from **DataAwareExportOptionsBase**) |
| | ShowPageTitleProperty | Identifies the DataAwareExportOptionsBase.ShowPageTitleProperty bindable property. (Inherited from **DataAwareExportOptionsBase**) |
| | ShowTotalSummariesProperty | Identifies the DataAwareExportOptionsBase.ShowTotalSummaries bindable property. (Inherited from **DataAwareExportOptionsBase**) |

**Public Events**

| | Name | Description |
|---|---|---|
| | BindingContextChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | CustomizeCell | When exporting data from the grid to XLS or XLSX format, this event allows |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | | you to customize a cell in the output document. (Inherited from **DataAwareExportOptionsBase**) |
| | CustomizeDataArea | When exporting to XLS or XLSX format, this event allows you to customize data in the output document. (Inherited from **DataAwareExportOptionsBase**) |
| | CustomizeFooter | When exporting to XLS or XLSX format, this event allows you to add a footer to the output document. (Inherited from **DataAwareExportOptionsBase**) |
| | CustomizeHeader | When exporting to XLS or XLSX format, this event allows you to add a header to the output document. (Inherited from **DataAwareExportOptionsBase**) |
| | CustomizeSheet | When exporting to XLS or XLSX format, this event allows you to customize the output document's settings. (Inherited from **DataAwareExportOptionsBase**) |
| | ExportProgress | Fires repeatedly while the data is being exported. (Inherited from **DataAwareExportOptionsBase**) |
| | PropertyChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanging | (Inherited from **Xamarin.Forms.BindableObject**) |

**See Also**
XlsExportOptions Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# XlsExportOptions Constructor

Initializes a new instance of the XlsExportOptions class with default settings.

**Namespace:** DevExpress.Mobile.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

**Syntax**
**C#**

```
public XlsExportOptions()
```

**See Also**
XlsExportOptions Class
XlsExportOptions Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# XlsExportOptionsBase Class

The base class for objects which define export options specific to XLS and XLSX formats.

**Namespace:** DevExpress.Mobile.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public abstract class XlsExportOptionsBase : ExportOptionsBase
```

### Remarks

The **XlsExportOptionsBase** class provides the following properties, which define how a document is exported to XLS and XLSX formats.

- ExportHyperlinks
- SheetName
- ShowGridLines
- RawDataMode

### Inheritance Hierarchy

System.Object
  Xamarin.Forms.BindableObject
    ExportOptionsBase
      **XlsExportOptionsBase**
        Derived classes

### See Also

XlsExportOptionsBase Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# XlsExportOptionsBase Members

The base class for objects which define export options specific to XLS and XLSX formats.

The following tables list the members exposed by the XlsExportOptionsBase type.

### Public Properties

| | Name | Description |
|---|---|---|
| | BindingContext | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ExportHyperlinks | Gets or sets a value indicating whether hyperlinks should be exported to XLS or XLSX. |
| | RawDataMode | Specifies whether a grid's actual data should be exported to XLS or XLSX (excluding all non-relevant elements, such as images, graphics, font and appearance settings). |
| | SheetName | Gets or sets the name of the sheet in the created XLS or XLSX file to which a document is exported. |
| | ShowGridLines | Gets or sets a value indicating whether grid lines should be visible in the resulting XLS or XLSX file. |

### Public Methods

| | Name | Description |
|---|---|---|
| | Assign | Copies all settings from the options object passed as a parameter. |
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Equals | (Inherited from **System.Object**) |
| S | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ToString | (Inherited from **System.Object**) |

© 2018 Developer Express Inc.

**Public Fields**

| | Name | Description |
|---|---|---|
| | BindingContextProperty | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ExportHyperlinksProperty | Identifies the ExportHyperlinks bindable property. |
| | RawDataModeProperty | Identifies the RawDataMode bindable property. |
| | SheetNameProperty | Identifies the SheetName bindable property. |
| | ShowGridLinesProperty | Identifies the ShowGridLines bindable property. |

**Public Events**

| | Name | Description |
|---|---|---|
| | BindingContextChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanging | (Inherited from **Xamarin.Forms.BindableObject**) |

**See Also**
XlsExportOptionsBase Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# XlsExportOptionsBase Properties

The base class for objects which define export options specific to XLS and XLSX formats.

The following tables list the members exposed by the XlsExportOptionsBase type.

### Public Properties

| | Name | Description |
|---|---|---|
| | BindingContext | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ExportHyperlinks | Gets or sets a value indicating whether hyperlinks should be exported to XLS or XLSX. |
| | RawDataMode | Specifies whether a grid's actual data should be exported to XLS or XLSX (excluding all non-relevant elements, such as images, graphics, font and appearance settings). |
| | SheetName | Gets or sets the name of the sheet in the created XLS or XLSX file to which a document is exported. |
| | ShowGridLines | Gets or sets a value indicating whether grid lines should be visible in the resulting XLS or XLSX file. |

### Public Methods

| | Name | Description |
|---|---|---|
| | Assign | Copies all settings from the options object passed as a parameter. |
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ReferenceEquals | (Inherited from **System.Object**) |
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ToString | (Inherited from **System.Object**) |

© 2018 Developer Express Inc.

**Public Fields**

| | Name | Description |
|---|---|---|
| | BindingContextProperty | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ExportHyperlinksProperty | Identifies the ExportHyperlinks bindable property. |
| | RawDataModeProperty | Identifies the RawDataMode bindable property. |
| | SheetNameProperty | Identifies the SheetName bindable property. |
| | ShowGridLinesProperty | Identifies the ShowGridLines bindable property. |

**Public Events**

| | Name | Description |
|---|---|---|
| | BindingContextChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | PropertyChanging | (Inherited from **Xamarin.Forms.BindableObject**) |

**See Also**
XlsExportOptionsBase Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# ExportHyperlinks Property

Gets or sets a value indicating whether hyperlinks should be exported to XLS or XLSX.

**Namespace:** DevExpress.Mobile.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public bool ExportHyperlinks { get; set; }
```

### Property Value

**true**, to export hyperlinks; otherwise, **false**.

### See Also

XlsExportOptionsBase Class
XlsExportOptionsBase Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# RawDataMode Property

Specifies whether a grid's actual data should be exported to XLS or XLSX (excluding all non-relevant elements, such as images, graphics, font and appearance settings).

**Namespace:** DevExpress.Mobile.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public bool RawDataMode { get; set; }
```

### Property Value

**true**, to enable the raw data export mode; otherwise **false**.

### See Also

XlsExportOptionsBase Class
XlsExportOptionsBase Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# SheetName Property

Gets or sets the name of the sheet in the created XLS or XLSX file to which a document is exported.

**Namespace:** DevExpress.Mobile.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public string SheetName { get; set; }
```

### Property Value

A System.String specifying the sheet name set in the resulting XLS or XLSX file.

### See Also

XlsExportOptionsBase Class
XlsExportOptionsBase Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# ShowGridLines Property

Gets or sets a value indicating whether grid lines should be visible in the resulting XLS or XLSX file.

**Namespace:** DevExpress.Mobile.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public bool ShowGridLines { get; set; }
```

### Property Value

**true**, to show the grid lines in the resulting XLS or XLSX file; otherwise, **false**.

### See Also

XlsExportOptionsBase Class
XlsExportOptionsBase Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# XlsExportOptionsBase Methods

The base class for objects which define export options specific to XLS and XLSX formats.

The following tables list the members exposed by the XlsExportOptionsBase type.

**Public Methods**

| | Name | Description |
|---|---|---|
| | Assign | Copies all settings from the options object passed as a parameter. |
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Equals | (Inherited from **System.Object**) |
| S | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| S | ReferenceEquals | (Inherited from **System.Object**) |
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ToString | (Inherited from **System.Object**) |

**See Also**
XlsExportOptionsBase Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# Assign Method

Copies all settings from the options object passed as a parameter.

**Namespace:** DevExpress.Mobile.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public override void Assign(
   ExportOptionsBase source
)
```

### Parameters

*source*
A XlsExportOptionsBase object (which is the ExportOptionsBase descendant) whose settings are assigned to the current object.

### Remarks

The **Assign** method allows the settings of the specified XlsExportOptionsBase object to be copied to the current XlsExportOptionsBase object.

### See Also

XlsExportOptionsBase Class
XlsExportOptionsBase Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# XlsxExportOptions Class

Contains options which define how grid data is exported to XLSX format.

**Namespace:** DevExpress.Mobile.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public class XlsxExportOptions : DataAwareExportOptionsBase
```

### Remarks

To export data from the GridControl to XLSX format, use the GridControl.ExportToExcel method with the ExportTarget.Xlsx value passed as the *format* parameter.

To specify XLSX export settings, create an instance of the **XlsxExportOptions** class and assign it to the GridControl.OptionsExportXlsx property.

### Inheritance Hierarchy

System.Object
  Xamarin.Forms.BindableObject
    ExportOptionsBase
      XlsExportOptionsBase
        DataAwareExportOptionsBase
          **XlsxExportOptions**

### See Also

XlsxExportOptions Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# XlsxExportOptions Members

Contains options which define how grid data is exported to XLSX format.

The following tables list the members exposed by the XlsxExportOptions type.

### Public Constructors

| | Name | Description |
|---|---|---|
| | XlsxExportOptions | Initializes a new instance of the XlsxExportOptions class with default settings. |

### Public Properties

| | Name | Description |
|---|---|---|
| | AllowGrouping | Gets or sets whether data groups are exported from the source grid to the output document. (Inherited from **DataAwareExportOptionsBase**) |
| | AllowLookupValues | Gets or sets whether lookup values of picker columns are exported. (Inherited from **DataAwareExportOptionsBase**) |
| | AllowSortingAndFiltering | Gets or sets whether the sorting and filtering functionality is enabled for columns in the exported document. (Inherited from **DataAwareExportOptionsBase**) |
| | BindingContext | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ExportHyperlinks | Gets or sets a value indicating whether hyperlinks should be exported to XLS or XLSX. (Inherited from **XlsExportOptionsBase**) |
| | GroupState | Gets or sets whether data groups are collapsed or expanded in the exported document. (Inherited from **DataAwareExportOptionsBase**) |
| | RawDataMode | Specifies whether a grid's actual data should be exported to XLS or XLSX (excluding all non-relevant elements, such as images, graphics, font and appearance settings). (Inherited from **XlsExportOptionsBase**) |
| | SheetName | Gets or sets the name of the sheet in the created XLS or XLSX file to which a document is exported. (Inherited from **XlsExportOptionsBase**) |
| | ShowColumnHeaders | Gets or sets whether column headers are visible in the exported document. (Inherited from **DataAwareExportOptionsBase**) |
| | ShowGridLines | Gets or sets a value indicating whether grid lines should be visible in the |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | | resulting XLS or XLSX file. (Inherited from **XlsExportOptionsBase**) |
| | ShowGroupSummaries | Gets or sets whether group summaries are enabled in the exported document. (Inherited from **DataAwareExportOptionsBase**) |
| | ShowTotalSummaries | Gets or sets whether total summaries are enabled in the exported document. (Inherited from **DataAwareExportOptionsBase**) |

**Public Methods**

| | Name | Description |
|---|---|---|
| | Assign | Copies all settings from the options object passed as a parameter. (Inherited from **DataAwareExportOptionsBase**) |
| | ClearValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | Equals | (Inherited from **System.Object**) |
| | Equals | (Inherited from **System.Object**) |
| | GetHashCode | (Inherited from **System.Object**) |
| | GetType | (Inherited from **System.Object**) |
| | GetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ReferenceEquals | (Inherited from **System.Object**) |
| | RemoveBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetBinding | (Inherited from **Xamarin.Forms.BindableObject**) |
| | SetValue | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ToString | (Inherited from **System.Object**) |

**Public Fields**

| | Name | Description |
|---|---|---|
| | AllowGroupingProperty | Identifies the DataAwareExportOptionsBase.AllowGrouping bindable property. (Inherited from **DataAwareExportOptionsBase**) |
| | AllowLookupValuesProperty | Identifies the DataAwareExportOptionsBase.AllowLookupValues bindable property. (Inherited from **DataAwareExportOptionsBase**) |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | AllowSortingAndFilteringProperty | Identifies the DataAwareExportOptionsBase.AllowSortingAndFiltering bindable property. (Inherited from **DataAwareExportOptionsBase**) |
| | BindingContextProperty | (Inherited from **Xamarin.Forms.BindableObject**) |
| | ExportHyperlinksProperty | Identifies the XlsExportOptionsBase.ExportHyperlinks bindable property. (Inherited from **XlsExportOptionsBase**) |
| | GroupStateProperty | Identifies the DataAwareExportOptionsBase.GroupState bindable property. (Inherited from **DataAwareExportOptionsBase**) |
| | RawDataModeProperty | Identifies the XlsExportOptionsBase.RawDataMode bindable property. (Inherited from **XlsExportOptionsBase**) |
| | SheetNameProperty | Identifies the XlsExportOptionsBase.SheetName bindable property. (Inherited from **XlsExportOptionsBase**) |
| | ShowColumnHeadersProperty | Identifies the DataAwareExportOptionsBase.ShowColumnHeaders bindable property. (Inherited from **DataAwareExportOptionsBase**) |
| | ShowGridLinesProperty | Identifies the XlsExportOptionsBase.ShowGridLines bindable property. (Inherited from **XlsExportOptionsBase**) |
| | ShowGroupSummariesProperty | Identifies the DataAwareExportOptionsBase.ShowGroupSummaries bindable property. (Inherited from **DataAwareExportOptionsBase**) |
| | ShowPageTitleProperty | Identifies the DataAwareExportOptionsBase.ShowPageTitleProperty bindable property. (Inherited from **DataAwareExportOptionsBase**) |
| | ShowTotalSummariesProperty | Identifies the DataAwareExportOptionsBase.ShowTotalSummaries bindable property. (Inherited from **DataAwareExportOptionsBase**) |

**Public Events**

| | Name | Description |
|---|---|---|
| | BindingContextChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| | CustomizeCell | When exporting data from the grid to XLS or XLSX format, this event allows |

© 2018 Developer Express Inc.

| | | |
|---|---|---|
| | | you to customize a cell in the output document. (Inherited from **DataAwareExportOptionsBase**) |
| ⚡ | CustomizeDataArea | When exporting to XLS or XLSX format, this event allows you to customize data in the output document. (Inherited from **DataAwareExportOptionsBase**) |
| ⚡ | CustomizeFooter | When exporting to XLS or XLSX format, this event allows you to add a footer to the output document. (Inherited from **DataAwareExportOptionsBase**) |
| ⚡ | CustomizeHeader | When exporting to XLS or XLSX format, this event allows you to add a header to the output document. (Inherited from **DataAwareExportOptionsBase**) |
| ⚡ | CustomizeSheet | When exporting to XLS or XLSX format, this event allows you to customize the output document's settings. (Inherited from **DataAwareExportOptionsBase**) |
| ⚡ | ExportProgress | Fires repeatedly while the data is being exported. (Inherited from **DataAwareExportOptionsBase**) |
| ⚡ | PropertyChanged | (Inherited from **Xamarin.Forms.BindableObject**) |
| ⚡ | PropertyChanging | (Inherited from **Xamarin.Forms.BindableObject**) |

**See Also**
XlsxExportOptions Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# XlsxExportOptions Constructor

Initializes a new instance of the XlsxExportOptions class with default settings.

**Namespace:** DevExpress.Mobile.Export
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

**Syntax**
**C#**

```
public XlsxExportOptions()
```

**See Also**
XlsxExportOptions Class
XlsxExportOptions Members
DevExpress.Mobile.Export Namespace

© 2018 Developer Express Inc.

# DevExpress.Utils

Contains utility classes used by DevExpress controls and components.

**Enumerations**

| | Enumeration | Description |
|---|---|---|
| | DefaultBoolean | Lists values which specify the validity of a condition (indicates whether the condition is true or false). |

© 2018 Developer Express Inc.

# DefaultBoolean Enumeration

Lists values which specify the validity of a condition (indicates whether the condition is true or false).

**Namespace:** DevExpress.Utils
**Assembly:** DevExpress.Mobile.Core.v18.1.dll

### Syntax

**C#**

```
public enum DefaultBoolean
```

### Members

| Name | Description |
|---|---|
| Default | The value is determined by the current object's parent object setting (e.g., a control setting). |
| False | Corresponds to a Boolean value of **false**. |
| True | Corresponds to a Boolean value of **true**. |

### Remarks

For example, the **DefaultBoolean** enumeration values are used by the GridColumn.AllowSort and GridColumn.AllowGroup properties.

### See Also

DevExpress.Utils Namespace

© 2018 Developer Express Inc.