Breaking changes in RadGridView for WinForms Q1 2008

Contents

Introduction 2
What has been changed 2
Detailed list of all changes 3
General changes 3
Renamed classes 3
Changes in RadGridView control 3
IGridView interface 4
GridTableElement 4
GridViewTemplate 4
GridViewInfo 5
GridViewColumn 5
GridViewDataColumn 5
GridViewRowInfo 5
GridViewGroupRowInfo 5
GridViewDetailsRowInfo 5
GridNewRowElement 6
GridCellElement 6
GridDataCellElement 6

Introduction

To make the performance of the grid top notch we had to reengineer the product and inevitably to implement some breaking changes. In the refactoring process we simplified and improved the API, and tagged as obsolete all properties and events which impact compatibility. Although we did our best to keep backwards compatibility, there are some events and properties which have been thrown out, because of design improvements. We doubt that many of our customers are using these in their current implementation, but nevertheless, these are changes which break compatibility.

What has been changed

We have gathered a lot of information from our customers on how we can improve RadGridView in a way that it incorporates both powerful and friendly API. In the course of the re-factoring process we redesigned the way data and UI communicate. The idea was to reduce the middle layer and entirely eliminated its performance cost.
By extracting some common APIs into new interfaces (and breaking compatibility), the data layer and UI layer presently live in complete symbiosis, communicating in the most efficient way. The Data Layer takes care of creating and initializing the logical objects that the UI layer uses to spread over its visuals. We use lazy instantiation where possible to ensure optimal swiftness in scenarios demanding both small and large amounts of data are to be visualized. For example, the grouping and aggregation objects are calculated at the time just prior to data being displayed on the screen. All operations like updating, deleting, and inserting new records of data have minimum impact over performance, especially in cases when the grid is grouped and sorted.
Another of the most delicate areas concerned grid rows and columns. We received numerous reports describing troubles accessing RadGridView rows with the need to retrieve/store values of grid cells. We found it curious that most developers feel more comfortable editing data using the API rather than through their app's data-layer, but nevertheless this is the approach we decided to follow.
The main trouble was that when first designing RadGridView's APIs we thought that the Rows property should resemble the visual representation of the data. The hierarchically structured Rows consisted of both calculated rows (like group headers), and rows that contain the actual data. Well, many of our customers found this model very difficult to work with, so we made three very important changes, which also break compatibility:

- Rows are no longer hierarchical, even when the grid is grouped;
- The Rows collection itself contains only GridViewDataRowInfo objects. This way it is much easier to access various features this data-record wrapper offers;
- And the most important one – Rows now consists of actual data objects only (wrapped up in GridViewDataRowInfo's, of course). For those of you who still need to access various rows product of data-calculations (group headers, summaries, etc) we have introduced an easily accessible hierarchical structure!

We are confident that these changes drastically improve the usability of the grid and present a new way of working with data. The new setup has been designed with developers in mind and we think that they will find the new approach much more reasonable and easy to use.

Another part that was changed significantly relates to grouping and filtering APIs. For example, when grouping was used, the API did not keep in synch the visual appearance of the data in the grid with the grid's (grouping) header. These problems have been addressed.

Detailed list of all changes

General changes

- The **Rows** collection now contains only the data rows (e.g. **GridViewDataRowInfo**).
- The **Columns** collection now contains only the data columns (e.g. **GridViewDataColumn**). Properties like Sort, Filter, FieldName (prev. DataField) are now much easier to access
- You can access the group rows through the **Groups** property of **RadGridView**
- The system and pinned rows are now accessed through the **SystemRows** and **PinnedRows** properties of **GridViewInfo**
- All row elements of the rows that are currently visible on the screen can be accessed through the **VisualRows** collection of the **GridElement**.
- All grid collections are now observable collections[1] and can be attached directly to their **CollectionChanged** events
- A brand new set of APIs for managing filtering has been introduced: FilterExpression, predicates, etc. Filters are assignable to each grid column, immediately affecting grid's UI
- **GridViewComboBoxColumn** inherits **GridViewLookUpColumn**

Renamed classes

- **GridViewBooleanColumn → GridViewCheckBoxColumn**
- **GridViewHeaderRowInfo → GridViewGroupRowInfo**
- **GridHeaderRowElement → GridTableHeaderRowElement**
- **FilterEditorElement → GridFilterRowElement**

Changes in RadGridView control

Removed

- **OnColumnsChanged**
- **OnFilterChanging**
- **OnFilterChanged**
- **OnSortChanging**
- **OnSortChanged**

In Q1 2008 use the events in the corresponding observable collections, e.g. **CollectionChanged** of the **Columns** collection.

Changed

- **ViewColumnsChanged** - type changed to **NotifyCollectionChangedEventHandler**
- **SortChanged** - type changed to **GridViewCollectionChangedEventHandler**
- **SortChanging** - type changed to **GridViewCollectionChangingEventHandler**
- **FilterChanged** - type changed to **GridViewCollectionChangedEventHandler**
- **FilterChanging** - type changed to **GridViewCollectionChangingEventHandler**
- **SelectedCells** - type changed to **GridViewSelectedCellsCollection**
- **SelectedRows** - type changed to **GridViewSelectedRowsCollection**
- **CurrentCell** - type changed to **GridDataCellElement**
- **EditingElement** - type changed to **IInputEditor**
- **LocalizationSettings** is obsolete now. You should use the **RadGridLocalizationProvider.CurrentProvider** property instead

[1] Generally speaking, the observable collection is a new type of collection introduced in WPF. Our observable collections are similar and as of Q1 2008 support events, i.e. all collections in RadGridView now support event notifications when something is changed.

- **RowDeleting/RowDeleted** events have been replaced by RadGridView **RowsChanging/RowsChanged** events
- **RowSelecting/RowSelected** events are replaced by RadGridView.**SelectionChanged** event
- **ActiveEditorCell** property is replaced by **CurrentCell** property.

IGridView interface

Removed

- **InvalidateColumns()**
- **InvalidateRowInfos()**
- **ResetCurrentRow()**
- **ColumnManager** is no longer needed. Use the **EditorManager** property of the **RadGridViw** control
- **UpdateGrouping()** – calling this method is no longer needed

Changed

The **Update** method has been changed and now takes **GridUINotifyAction** parameter that specifies which part of the grid needs updating.

GridTableElement

Removed

- **ScrollBarThemeName** – now scrollbars are styled through the Visual Style Builder

GridViewTemplate

Removed

- **ViewInfos** is no longer supported
- **AllowMultiColumnSorting** – the multi column sorting is always allowed now
- **SummaryDisplayStyle, SummaryItems –** use the **SummaryRowsTop**, **SummaryRowsBottom** and **SummaryRowGroupHeaders** collections of **GridViewTemplate** instead
- **FilterMenus, FiltersOperationMenu** are obsolete – use the **RadGridLocalizationProvider.CurrentProvider** property in order to localize menus
- **ColumnsCollectionChanged** – use the **CollectionChanged** event of the **Columns** collection instead
- **MoveColumnAfter** - use the **Move** method of the **Columns** collection instead
- **OnColumnUserMoved, OnColumnWidthUserChanged, OnColumnsCollectionChanged, OnGroupingChanged, OnSortChanged, OnSortChanging** – use the corresponding event in the **GroupByExpressions** and **SortExpressions** collections
- **ColumnUserMoved** event is replaced by **ColumnIndexChanged** event

Changed

- **OwnerGrid** property is obsolete, replaced by **Owner** property
- **ColumnAutoWidth** property is obsolete
- **AllowColumnRemove** property is obsolete
- **ColumnWidthUserChanged** is replaced by **ColumnWitdthChanged**
- **AddToGroupExpressions** method is replaced by **GroupByExpressions**.**InsertUnique** method;
- **AddGroupExpression** method is replaced by **GroupByExpressions.Add** method;
- **ClearGroupExpressions** method is replaced by **GroupByExpressions.Clear** method;
- **SwitchGroupExpressions** method is replaced by **GroupByExpressions.Move** method;

- **RemoveGroupExpressions** method is replaced by **GroupByExpressions.RemoveAt** method;
- **UpdateGroupPanel** method is replaced by RadGridView**.UpdateGroupPanel** method;
- All events firing when a particular property changes are replaced by **NotifyPropertyChanged** event with appropriate arguments.

GridViewInfo

Removed

- **Initialized** is no longer supported
- **SelectedCells, SelectedRows, CurrentRowChanging , CurrentRowChanged, OnCurrentRowChanged, OnCurrentRowChanging** are all moved to the RadGridView control
- **FilterExpression, FiltersOperation, OnFilterUpdate** – use the FilterExpressions collection in GridViewTemplate instead
- **SortExpression** – use the SortExpressions collection in GridViewTemplate instead
- **UpdateView, UpdateFiltering, UpdateSorting** - no longer needed

Changed

- **Rows** - changed type to **GridViewRowCollection**

GridViewColumn

Removed

- **AutoSize** - no longer supported

GridViewDataColumn

- **DataFiled** property is replaced by **FieldName** property
- **SortExpression** property is obsolete
- **DataTextFormatString** property is replaced by **FormatString** property
- **DataEditFormatString** property is obsolete, replaced by FormatString property
- **FilterChanging/FilterChanged** events are obsolete
- **GetSortExpression** method is obsolete.

GridViewRowInfo

Removed

- **ParentRow, GroupLevel** - moved to **GridViewDataRowInfo**
- **Rows** - no longer supported. If this is a **GridViewDataRowInfo** row, use the **ChildRow** property. If this is a **GridViewGroupRowInfo** use the **Group** property.

GridViewGroupRowInfo

Changed

- **Rows** – you can obtain child rows now through the **Group.Rows** property
- **Text** – the group header text is accessed now through the **GetSummary** method

GridViewDetailsRowInfo

Removed

- **Relation** - no longer supported

GridNewRowElement

Removed

- **AddNewRow()** - no longer supported

GridCellElement

Removed

- **Column** – use the **ColumnInfo** property instead
- **GridViewInfo** – use the **ViewInfo** property instead
- **IsCurrentRow** – no longer needed
- **Row** – no longer supported
- **Value** – moved to **GridDataCellElement**
- **SetCellValue()** – use the **SetContent** method instead
- **SetCellValueCore()** – use the **SetContentCore** method instead
- **InvokeContextMenu()** – use the methods of the **IContextMenuProvider** interface instead

GridDataCellElement

Removed

- **BeginEdit()** – use the **BeginEdit** method of the **RadGridView** control instead
- **EndEdit()** – use the **EndEdit** method of the **RadGridView** control instead