# **DataGrid**

### **4.1 Data Source**

* **4.1.1 Simple Array** – Uses a local JavaScript array as the data source.
  + **Property**: dataSource – Binds the grid to an array.
  + **Method**: refresh() – Reloads the data.
  + **Event**: onContentReady – Triggered when the grid content is ready.
* **4.1.2 Ajax Request** – Fetches data from an API or server using AJAX.
  + **Property**: remoteOperations – Enables server-side data handling.
  + **Method**: reload() – Reloads data from the server.
  + **Event**: onDataErrorOccurred – Triggered if an error occurs while loading data.

### **4.2 Paging and Scrolling**

* **4.2.1 Record Paging** – Divides records into pages with a set page size.
  + **Property**: paging.enabled – Enables or disables paging.
  + **Method**: pageIndex() – Gets or sets the current page index.
* **4.2.2 Virtual Scrolling** – Loads only visible records for performance.
  + **Property**: scrolling.mode – Sets scrolling mode ("virtual" for virtual scrolling).
  + **Method**: refresh() – Reloads the data and refreshes the UI.
* **4.2.3 Infinite Scrolling** – Dynamically loads more records as you scroll.
  + **Property**: scrolling.mode – Set to "infinite" for infinite scrolling.
  + **Method**: load() – Manually loads the next set of records.

### **4.3 Editing**

* **4.3.1 Row Editing & Events** – Edits entire rows with save/cancel actions.
  + **Property**: editing.mode – Defines editing mode ("row" for row editing).
  + **Method**: editRow(rowIndex) – Puts a specific row into edit mode.
  + **Event**: onRowUpdating – Triggered before a row update.
* **4.3.2 Batch Editing** – Allows multiple edits before saving changes.
  + **Property**: editing.mode – Set to "batch" for batch editing.
  + **Method**: saveEditData() – Saves all changes.
  + **Event**: onSaving – Triggered before changes are saved.
* **4.3.3 Cell Editing** – Edits individual cells without affecting the row.
  + **Property**: editing.mode – Set to "cell" for cell editing.
  + **Method**: editCell(rowIndex, dataField) – Puts a specific cell into edit mode.
  + **Event**: onCellPrepared – Triggered after a cell is prepared.
* **4.3.4 Form Editing** – Opens a form to edit row data.
  + **Property**: editing.mode – Set to "form" for form-based editing.
  + **Method**: editRow(rowIndex) – Opens the form for editing.
* **4.3.5 Popup Editing** – Edits rows in a modal popup.
  + **Property**: editing.mode – Set to "popup" for popup editing.
  + **Method**: editRow(rowIndex) – Opens a popup for editing.
  + **Event**: onEditorPreparing – Triggered before the popup editor is created.
* **4.3.6 Data Validation** – Ensures input follows rules (e.g., required, max length).
  + **Property**: columns.validationRules – Defines validation rules for a column.
  + **Method**: validate() – Validates all fields.
  + **Event**: onRowValidating – Triggered before row validation.
* **4.3.7 Cascading Lookups** – Dynamically filters lookup fields based on selection.
  + **Property**: lookup.dataSource – Defines the lookup data source.
  + **Event**: onEditorPreparing – Adjusts lookup options dynamically.

### **4.4 Grouping**

* **4.4.1 Record Grouping** – Groups records based on a field.
  + **Property**: grouping.autoExpandAll – Expands or collapses all groups.
  + **Method**: expandAll() – Expands all groups.
  + **Event**: onRowPrepared – Triggered when a group row is prepared.

### **4.5 Filtering**

* **4.5.1 Filtering** – Allows users to filter records dynamically.
  + **Property**: filterRow.visible – Enables or disables filtering.
  + **Method**: clearFilter() – Clears the applied filter.
  + **Event**: onEditorPreparing – Customizes filter options.
* **4.5.2 Filter Panel** – Displays applied filters in a panel.
  + **Property**: filterPanel.visible – Shows or hides the filter panel.
  + **Method**: applyFilter() – Applies the current filter settings.
  + **Event**: onOptionChanged – Triggered when filter settings change.

### **4.6 Sorting**

* **4.6.1 Multiple Sorting** – Sorts records by multiple columns.
  + **Property**: sorting.mode – Set to "multiple" to allow multi-column sorting.
  + **Method**: columnOption(column, "sortOrder", "asc") – Sets sorting for a column.
  + **Event**: onOptionChanged – Triggered when sorting options change.

### **4.7 Selection**

* **4.7.1 Row Selection** – Allows selection of single rows.
  + **Property**: selection.mode – Set to "single" for single row selection.
  + **Method**: selectRows(keys, true) – Selects specific rows by key.
  + **Event**: onSelectionChanged – Triggered when selection changes.
* **4.7.2 Multiple Record Selection Modes** – Enables selecting multiple rows.
  + **Property**: selection.mode – Set to "multiple" for multiple selection.
  + **Method**: clearSelection() – Clears all selected rows.
  + **Event**: onSelectionChanged – Triggered when multiple selections change.

### **4.8 Columns**

#### **4.8.1 Column Customization**

* **Properties:**
  + columnAutoWidth: true (Adjusts column widths automatically)
  + columnResizingMode: "nextColumn" (Resizes columns proportionally)
* **Methods:**
  + columnOption("Name", "visible", false); (Hides a column dynamically)
* **Events:**
  + onOptionChanged (Triggered when column options change)

#### **4.8.2 Columns based on a Data Source**

* **Properties:**
  + dataSource: marvelCharacters (Binds data source dynamically)
* **Methods:**
  + refresh() (Reloads the grid data)
* **Events:**
  + onContentReady (Fires when the grid content is fully loaded)

#### **4.8.3 Multi-Row Headers**

* **Properties:**
  + columns: [{ caption: "Names", columns: [...] }] (Defines grouped headers)
* **Events:**
  + onCellPrepared (Can modify header cell appearance dynamically)

#### **4.8.4 Column Resizing**

* **Properties:**
  + allowColumnResizing: true
  + columnMinWidth: 10
* **Methods:**
  + columnOption("Affiliation", "width", 150); (Changes width dynamically)

#### **4.8.5 Command Column Customization**

* **Properties:**
  + type: "buttons" (Adds action buttons)
* **Methods:**
  + columnOption("Buttons", "visible", false);
* **Events:**
  + onCellClick (Handles button clicks)

### **4.9 State Persistence**

* **Properties:**
  + stateStoring: { enabled: true, type: "localStorage", storageKey: "gridState" }
* **Methods:**
  + state(null); (Clears saved state)
* **Events:**
  + onOptionChanged (Handles changes in state persistence)

### **4.10 Appearance**

#### **4.10.1 Appearance**

* **Properties:**
  + showBorders: true
* **Events:**
  + onRowPrepared (Applies custom row styling)

### **4.11 Template**

#### **4.11.1 Column Template**

* **Properties:**
  + columns: [{ dataField: "Abilities", cellTemplate: function (container, options) { $(container).text("⭐ " + options.value); } }]
* **Events:**
  + onCellPrepared (Allows cell customization)

#### **4.11.2 Row Template**

* **Properties:**
  + rowTemplate: function (rowElement, rowData) { rowElement.css("background-color", "lightgray"); }
* **Events:**
  + onRowPrepared (Custom row rendering)

#### **4.11.3 Cell Customization**

* **Properties:**
  + customizeText (Formats displayed text)
* **Events:**
  + onCellClick (Handles click events on cells)

#### **4.11.4 Toolbar Customization**

* **Properties:**
  + toolbar: [{ location: "after", widget: "dxButton", options: { text: "Export", onClick: function () { alert("Export Clicked"); } } }]
* **Events:**
  + onToolbarPreparing (Modifies toolbar dynamically)

### **4.12 Data Summaries**

#### **4.12.1 Grid Summaries**

* **Properties:**
  + summary: { totalItems: [{ column: "First\_Appearance", summaryType: "count" }] }

#### **4.12.2 Group Summaries**

* **Properties:**
  + summary: { groupItems: [{ column: "Affiliation", summaryType: "count" }] }

#### **4.12.3 Custom Summaries**

* **Properties:**
  + summary: { totalItems: [{ column: "First\_Appearance", summaryType: "custom", customizeText: function (data) { return "Total: " + data.value; } }] }

### **4.13 Master-Detail**

#### **4.13.1 Master-Detail View**

* **Properties:**
  + masterDetail: { enabled: true, template: function (container, options) { $("<div>").text("Details: " + options.data.Name).appendTo(container); } }
* **Events:**
  + onRowExpanding (Handles row expansion)

### **4.14 Export**

* **Properties:**
  + export.enabled → Enables/disables export functionality.
  + export.fileName → Sets the default exported file name.
  + export.allowExportSelectedData → Allows exporting selected rows only.
* **Methods:**
  + .exportToExcel(selectionOnly) → Exports grid data to Excel.
* **Events:**
  + onExporting → Fires before the export starts.
  + onExported → Fires after export completes.

### **4.15 Adaptability**

#### **4.15.1 Grid Adaptability Overview**

* **Properties:**
  + columnHidingEnabled → Enables automatic column hiding.
  + adaptivityEnabled → Enables adaptive layout for grids.
* **Events:**
  + onAdaptiveDetailRowPreparing → Handles the preparation of the adaptive row.

#### **4.15.2 Grid Columns Hiding Priority**

* **Properties:**
  + columns[].hidingPriority → Defines the priority for hiding columns (lower numbers hide first).
* **Example:**
* $("#gridContainer").dxDataGrid({  
   columnHidingEnabled: true,  
   columns: [  
   { dataField: "ID", hidingPriority: 3 },  
   { dataField: "Name", hidingPriority: 1 },  
   { dataField: "Department", hidingPriority: 2 },  
   ],  
  });