Table of Contents

Modifying default behavior and presentation of queries

To modify the default behavior and presentation of queries you need to go to a particular query that you want to modify and click on the “Scripts” property in the list of properties of the query. You can define “initialization” script or “render” script or both (see “Architecture of the Client-side Code”)

The following objects are exposed to the initialization script:

1. “config” object - this object represents Kendo UI configuration of the main widget implementing the query (see the table below)
2. “markup” object – HTML markup prepared by the controller
3. “parser” object – the controller itself
4. “widgets” – an array of all widget configurations for the query

Query presentation type	Kendo UI widget	Kendo UI reference
Standard	Grid	http://docs.telerik.com/kendo-ui/api/javascript/ui/grid
Custom (Custom Data Template, “ListvView widget”)	List View	http://docs.telerik.com/kendo-ui/api/javascript/ui/listview
Custom (Custom Data Template, “Scroll View widget	http://docs.telerik.com/kendo-ui/api/javascript/mobile/ui/scrollview	
Calendar/Scheduler	Scheduler	http://docs.telerik.com/kendo-ui/api/javascript/ui/scheduler
Chart	Chart	http://docs.telerik.com/kendo-ui/api/javascript/dataviz/ui/chart
Gantt	Gantt	http://docs.telerik.com/kendo-ui/api/javascript/ui/gantt
Timeline	Timeline	http://docs.telerik.com/kendo-ui/api/javascript/ui/timeline
Tree	TreeList	http://docs.telerik.com/kendo-ui/api/javascript/ui/treelist

It is usually unnecessary to modify the markup, but you are welcome to modify the configuration of the widget.

For example, the following script will change the alignment of the column that corresponds to the “Status” attribue:

```
for (var i = 0; i < config.columns.length; ++ i)
{
    if (config.columns[i].field == "Status")
    {
        config.columns[i].attributes = { alignment: "right" }
    }
}
```

The following script will make the grid “groupable”, i.e allow dragging the columns to a special area in order to group the grid by this column:

```
config.groupable = true;
```

The “parser” object represents the controller and allows you to access certain properties of the

system that you may need. The type of this object depends on the type of the query representation and is provided in the table below:

Query presentation type	Aware IM Javascript object type	Source code
Standard	AwareApp_QueryLayoutParser	AwareIM/Tomcat/webapps/AwareIM/aware_ext/parsers/queryLayoutParser.js
Custom (without scroll view)	AwareApp_CustomLayoutParser	AwareIM/Tomcat/webapps/AwareIM/aware_ext/parsers/customLayoutParser.js
Custom (with scroll view)	AwareApp_ScrollViewParser	AwareIM/Tomcat/webapps/AwareIM/aware_ext/parsers/scrollViewParser.js
Chart	AwareApp_ChartParser	AwareIM/Tomcat/webapps/AwareIM/aware_ext/parsers/chartParser.js
Calendar/Scheduler	AwareApp_CalendarParser	AwareIM/Tomcat/webapps/AwareIM/aware_ext/parsers/calendarParser.js
Gantt	AwareApp_GanttParser	AwareIM/Tomcat/webapps/AwareIM/aware_ext/parsers/ganttParser.js

The “widgets” object representing an array of all widgets that the query has can be used to modify other widgets – for example, toolbars generated to represent query operations (if these are defined for the query).

Each member of the “widgets” array has the following properties:

- type (the type of the Kendo UI widget)
- id (the unique id in the markup of the query that the widget uses)
- config (the Kendo UI configuration of the widget)

So to modify a toolbar, for example (http://docs.telerik.com/kendo-ui/api/javascript/ui/toolbar) and stop it being resizable you would find the toolbar in the array and modify its “config” property like this:

```
for (var i = 0; i < widgets.length; ++ i)
{
    if (widgets[i].type == "toolbar")
        widgets[i].config.resizable = false;
}
```

The second script (“render” script) that you can define in the Scripts dialog allows you to modify the widget representing the query after it has been drawn. Here you would use methods of the corresponding Kendo UI object, rather than configuration options. Objects available for your Javascript here are:

- “widget” object
- “parser” object

The “widget” object represents the widget that has been drawn.

The “parser” object is the controller object described above.