JavaScript API Reference

Dashboard Classes

dashboard Class

Defines the dashboard's attributes, including layout, contained widgets, and filters.

Methods

Method	Arguments	Return Value	Description
refresh()		promise	Refreshes the dashboard.
addWidget (widget)	**widget** (widget) Widget model to be added.		Adds the defined widget to the dashboard. The widget is placed at the bottom of the first column in the dashboard.
addWidget (widget, layout)	**widget** (widget) Widget model to add. **layout** (widgetLayout) Layout object that defines the widget's placement.		Adds the given widget to the dashboard using the given layout definition.
datasources()		[datasource]	Returns an array of datasource objects used by the various widgets in the dashboard.
isEmpty()		boolean	Returns a response stating whether the dashboard includes widgets, or is empty.
destroy()			Releases the resources of the dashboard object and contained widgets.

Properties

Property	Type	Description
filters	dashboardFilters	Returns the dashboard's fields object.
widgets	dashboardwidgets	Returns the dashboard's widgets collection.
layout	dashboardLayout	Returns the dashboard's layout object.
style	dashboardStyle	Returns the dashboard's style object.
initialized	boolean	Indicates if the dashboard was initialized.
refreshing	boolean	Indicates if the dashboard is refreshing.
editing	boolean	Indicates if the dashboard is currently in edit state.

Events

Event	Event Arguments	Description
initialized	**dashboard** (dashboard) Dashboard instance.	Fired after the dashboard is initialized.
refreshstart	**dashboard** (dashboard) Dashboard instance. **totalQueries** (integer) Number of queries in the current run.	Fired after one or more widgets have started the refresh process.
refreshend	**dashboard** (dashboard) Dashboard instance. **completedQueries** Number of completed queries.	Fired when a refresh process ends for one or more widgets.
widgetrefreshed	**dashboard** (dashboard) Dashboard instance. **widget** (widget) Refreshed widget. **totalQueries** (integer) Number of queries in the current run. **completedQueries** (integer) Number of completed queries.	Fired when a widget refresh has ended.
widgetadded	**dashboard** (dashboard) Dashboard instance. **widget** (widget) Added widget. **layout** (widgetLayout) Layout options object.	Fired when a widget was added to the dashboard.
stylechanged	**dashboard** (dashboard) Dashboard instance.	Fired when the dashboard style is changed.
destroyed	**dashboard** (dashboard) Dashboard instance.	Fired when the dashboard is destroyed.
editstart	**dashboard** (dashboard) Dashboard instance.	Fired when entering edit mode.
editend	**dashboard** (dashboard) Dashboard instance.	Fired when in edit mode.
filterschanged	**dashboard** (dashboard) Dashboard instance. **type** (string) Change type. Available values: *add/update* and *remove*. **item** (jaql) Changed item.	Fired when the dashboard filter changes.
widgetinitialized	**widget** (widget) Widget instance.	Fired when the widget is initialized.
widgetbuildquery	**widget** (widget) Widget instance. **query** (object) JAQL query object.	Fired when executing the widget's native build query, and allows customization of the JAQL query object before execution.
widgetbeforequery	**widget** (widget)Widget instance. **query** (object) JAQL query object.	Fired before the query is executed.
widgetprocessresult	**widget** (widget) Widget instance. **query** (object) JAQL query object. **result** (queryResult) Processed result. **rawResult** (rawQueryResult) JAQL query result. **reason** (string) Query reason.	Fired after executing the widget's native result processing, and allows customization of the query result before being rendered.

widgetrender	**widget** (widget) Widget instance. **reason** (string) Rendering reason. **disableAnimations** (boolean) Indicates whether animations are enabled.	Fired when the widget is rendered.
widgetready	**widget** (widget) Widget instance.	Fired when the widget's rendering is complete and the widget is ready.
beforewidgetmenu	**widget** (widget) Widget instance. **items** (object array) Array of items to be displayed. **element** (dom object) The source element that triggers the event. **cancel** (boolean) Determines whether the menu is displayed.	Fired before a widget shown in a dashboard shows it's context menu.
widgetdestroyed	**widget** (widget) Widget instance.	Fired when the widget is destroyed and it's resources are released.

dashboardFilters Class

Defines filters for dashboards.

Methods

Method	Argument	Return Value	Description
item	**index** (integer) Index to fetch by.	**JAQL**	Returns the JAQL filter object by the given index.
item	**dimension** (string) dimensionality **level** (optional) (string) level name	**JAQL**	Returns the JAQL filter object by the given dimension and level.
update	**filter** (jaql) JAQL filter object. **options** (object) */refresh* true to refresh the dashboard. */save* true to post the the dashboard object to the server	-	Adds or updates the filter object matching the dimensions and level of the given filter.
remove	**index** (integer) Index to fetch by.	-	Removes the filter item at the given index.
remove	**dimension** (string) dimensionality **level (optional)** (string) level name	-	Removes the filter item by the given dimension and level.
reset	**filters** ([jaql]) filters with which to reset the collection	-	Removes all filters and resets the filters collection with the given filters array.
clear	-	-	Removes all items from the filters collection.

Properties

Property	Type	Description
length	(integer)	Returns the length of the dashboard filters collection.

dashboardwidgets Class

Returns widget details.

Methods

Method	Arguments	Return Value	Description
get	**index** (integer) Index to fetch by.	**widget**	Returns the widget by the given index.
get	**id** (string) Widget ID.	**widget**	Returns the widget by the given ID.
toArray		[**widget**]	Returns an array containing all widgets in the collection.

Properties

Property	Type	Description
length	(integer)	Returns the length of the widgets collection.

dashboardLayout Class

Defines the layout options for contained widgets in a dashboard.

Methods

Method	Arguments	Return Value	Description
$$get	**widget** (widget) Widget to get layout object by.	**widgetLayout**	Returns the widget layout object defined for the widget.

Properties

Property	Type	Description
columns	[column]	Returns the column objects array.

dashboardStyle Class

Defines style attributes of the dashboard including widget layouts and colors.

Methods

Method	Arguments	Return Value	Description
palette	-	[string]	Returns a hexadecimal colors array representing the dashboard palette.
minColor	-	string	Returns the color to be used for a color range min value.
maxColor	-	string	Returns the color to be used for a color range max value.
setPalette	array OR palette object, bool	-	Sets the current palette of the dashboard. Accepts either an array of colors or a palette object. Receives a Boolean parameter that indicates whether the existing widget colors (only single colors, not range / conditional) should be reset to the new palette.

col Object

Represents a column in a dashboard.

Properties

Properties	Type	Description
width	integer	The width as a percentage for the containing dashboard.

cells	[cell]	The nested cells array.
index	integer	The column index.

cell Object

Represents a cell in a column.

Properties

Property	Type	Description
subcells	[subcell]	The nested sub-cells array.

subcell Object

Represents a sub cell in a cell.

Properties

Property	Type	Description
elements	[element]	Nested elements array.
width	integer	The width as a percentage of the containing cell.
index	integer	The index of the sub-cells in it's parent cell's sub-cells array.

element Object

Represents an element hosting a widget.

Properties

Property	Type	Description
height	string	The element height in pixels.
minHeight	integer	The minimum height in pixels to which the element can shrink.
maxHeight	integer	The maximum height in pixels to which the element can expand.
minWidth	integer	The maximum width in pixels to which the element can expand.
maxWidth	integer	The maximum height in pixels to which the element can expand.
widgetid	string	The ID of the contained widget.

Widget Classes

widget Class

Defines widget behavior attributes.

Methods

Property	Arguments	Return Value	Description
refresh()		promise	Refreshes the widget.
redraw()			Requests a widget redraw.
datasources()		[**datasource**]	Returns an array of data source objects used by the various widgets in the dashboard.

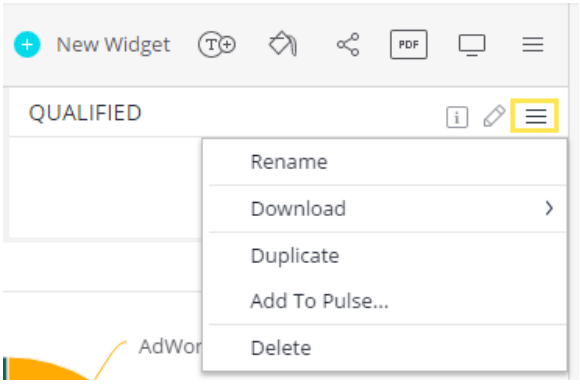
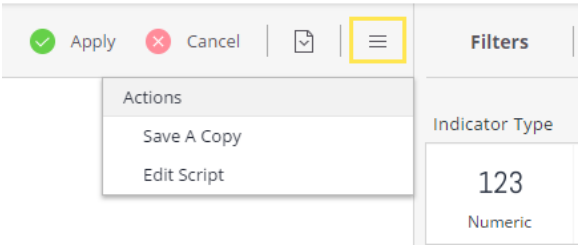
isEmpty()		boolean	Returns a response stating whether the dashboard is defined with widgets.
destroy()			Releases the resources of the dashboard object along with all it's contained widgets.

Properties

Property	Type	Description
manifest	object	Returns the widget's read-only manifest declaration.
datasource	datasource	Returns the widget's data source.
metadata	widgetMetadata	Returns the widget's meta-data object.
options	object	Returns the widget's options object.
style	object	Returns the widget style object.
initialized	boolean	States whether the widget was initialized.
refreshing	boolean	States whether the widget is refreshing.
queryResult	queryResult	Returns the last processed query result. Each widget has it's own processing logic that converts the raw, tabular query result and generates an object, which is more suitable for rendering by the widget.
rawQueryResult	rawQueryResult	Returns the last pre-processed query result.

Events

Property	Type	Description
initialized	**widget** (widget) Widget instance.	Fired when the widget is initialized.
domready	**widget** (widget) Widget instance.	Fired when the widget is rendered and added to the DOM.
refreshed	**widget** (widget) Widget instance.	Fired when a widget is refreshed.
buildquery	**widget** (widget) Widget instance. **query** (object) JAQL query object.	Fired when executing the widget's native build query and allows customization of the JAQL query object before execution.
beforequery	**widget** (widget) Widget instance. **query** (object) JAQL query object.	Fired before the query is executed.
querystart	**widget** (widget) Widget instance.	Fired when the widget's query starts.
queryend	**widget** (widget) Widget instance. **query** (object) JAQL query object. **rawResult** (rawQueryResult) JAQL query result. **reason** (string) Query reason.	Fired when the widget's query has ended.

processsresult	**widget** (widget) Widget instance. **query** (object) JAQL query object. **result** (queryResult) processed result. **rawResult** (rawQueryResult) JAQL query result. **reason** (string) Query reason.	Fired when executing the widget's native result processing, and allows customization of the query result before being rendered.
render	**widget** (widget) Widget instance. **reason** (string) Rendering reason. **disableAnimations** (boolean) Indicates whether animations should take place.	Fired when the widget is rendered.
ready	**widget** (widget) Widget instance.	Fired when the widget's rendering is over and the widget is ready.
readjust	**widget** (widget) Widget instance.	Fired when a layout change is applied to the widget.
beforewidgetindashboardmenu	**widget** (widget) widget instance **items** (object array) array of items to be displayed **element** (dom object) source element that triggers the event **cancel** (boolean) Determines whether the menu is displayed	Fired when clicking the widget's menu while viewing a dashboard. 
beforewidgetmenu	**widget** (widget) Widget instance. **items** (object array) Array of items to be displayed. **element** (dom object) Source element that triggers. the event. **cancel** (boolean) Determines whether the menu is displayed or not.	Fired when clicking the widget menu while in Edit Widget mode. 

beforeviewloaded	**widget** (widget) Widget instance. **element** (dom object) **Element container** **options** (object) /chart - final Highcharts options /scattermap - Object that contains the map instance and the array of markers.	Fired before the Highcharts object/map configuration is either updated or created. The event has two parameters, the first one is the widget model and the second is as described in the event arguments column.
beforedatapointtooltip	**widget** (widget) Widget instance. **context** (object) Default tooltip context. In charts there is context.pointScope for the point context. **template** (string) Angular template string of the default tooltip. Change it to replace the template. **cancel** (boolean) Determines whether the tooltip should be shown or not.	Fired for all widgets that render a tooltip. Allows hooking to the tooltip initialization flow, disables the default application tooltip, and creates a new tooltip instead, or overrides the default tooltip template. The event has two parameters; the first one is the widget model, and the second is as described in the event arguments column. The widgets that support the event are: scatter map, area map and all the charts. In maps it is not possible to use the default tooltip and add some more html to it, it is possible only to change it completely. For writing a property value from context, write in the template {{model.varPathFromContext}}. Example: For context.pointScope.total, the template is: ' {{model.pointScope.total}} '
destroyed	**widget** (widget) Widget instance.	Fired when the widget is destroyed and it's resources are released.
processcell	**widget** (widget) Widget instance. **panel** (datapanel) Data panel the cell is related to. **item** (item) The item that the cell is related to. **row** (object[]) The entire result set that the row of the cell belongs to. **cell** (object) The cell being processed.	Fired when a cell is being processed and enables customization of the cell attributes (text, data, color, size).

widgetMetadata Class

Returns metadata for widgets in the dashboard.

Methods

Method	Arguments	Return Value	Description
panel	index (integer)	widgetPanel	Gets a widget's metadata panel by index.
panel	name (string)	widgetPanel	Gets a widget's metadata panel by name.
panel	predicate (function)	widgetPanel	Gets the first widget panel that gets validated by the predicate.
items	predicate (function)	[jaql]	Gets the widget metadata items that get validated by the predicate.
filters		[jaql]	Gets the array of widget filters.

Properties

Property	Type	Description
panels	[widgetPanel]	Returns an array with the widget's panels.
ignore	object `{ dimensions: [ "sales.product", "sales.city" ], all: boolean}`	Returns the widget's ignore declaration that states which dashboard filters should be ignored by the widget. Adds dimension IDs to the dimensions array to explicitly state which dimensions should be ignored, or sets the *all* attribute to true to ignore the dashboard filters.

widgetPanel Class

Methods

Method	Arguments	Return Value	Description
item	index (integer)	JAQL	Gets an item by index
item	guid (string)	JAQL	Gets an item by it's GUID.
item	predicate (function)	JAQL	Gets the first item that gets validated by the predicate.
push	jaql		Adds the given JAQL item to the panel.
push	[jaql]		Adds the given JAQL items array to the panel.
isEmpty		boolean	Returns a response stating whether the panel is empty or not.

Properties

Property	Type	Description
items	[jaql]	Returns the panel items array.

widgetManifest Class

The manifest object used to register new widget types with Sisense.

Name	Type	Description
name	string	Widget name.
family	string	Widget family name.
styleEditorTemplate	string	Widget editor template URL.
iconSmall	string	Widget small icon URL. The icon should be a sprite file of two 24x24 icons.
hideNoResults	bool	Defines whether the widget renders the default no result state.
noResultImg	string	URL for an image representing the no result state of the widget.

noResultImgSmall	string	URL for a small image representing the no result state of the widget.
options Default options object that should be defined for a newly created widget. The options object shares both common attributes along with widget-defined attributes.		
options.selector	boolean	Indicates if the widget should be a selector widget by default. Default = false.
options.disallowSelector	boolean	Indicates if the widget can act as a selector. Default = true.
options.dashboardFiltersMode	string	Indicates if the dashboard filters filter or highlight the widget when the same field is set to both the dashboard filters and the widget. Default = filter.
options.triggersDomready	boolean	Indicates if the widget is triggering the DOM-ready event once it's DOM representation was added to the DOM in full.
sizing Default sizing that defines the various sizing attributes for the defined widget type.		
sizing.minHeight	integer	States the minimum height to which the widget can shrink.
sizing.maxHeight	integer	States the maximum height to which the widget can grow.
sizing.minWidth	integer	States the minimum width to which the widget can shrink.
sizing.maxWidth	integer	States the maximum width to which the widget can grow.
sizing.height	integer	States the default height set for a newly created widget of the defined type.
directive Directives used as handlers for custom DOM content. Sisense uses custom directives for widgets to have a proper Views layer that handles the widget DOM rendering.		
directive.desktop	string	Desktop directive name and acts as the default name when the target device is not defined.
directive.tablet	string	Tablet directive name, with the desktop attribute as default fallback.
directive.mobile	string	Mobile directive name, with the desktop attribute as default fallback.
style Each widget type is defined with it’s own style object.		
data Widget data manifest object.		
data.selection	[string]	States which of the widget's data panels to use. A widget can include multiple data panels with metadata. For example, a pivot includes row, column, value and filter panels.
data.defaultQueryResult	object	Defines the default query result with which the widget is served.

data.buildQuery(...)	**Arguments** **widget**: Widget model. **query**: Query object to load widget metadata to. **Returns** query object	Gets a basic query object and is responsible for building the query from the current state of the widget.
data.processResult(...)	**Arguments** **widget**: Widget model. **queryResult**: Data-set object. **Returns** Processed result object to be passed to the widget view.	Prepares the widget-specific query result from the given result data-table.
data.rankMetadata(...)	**Arguments** **items**: Items to rank. **type**: Widget type (for use with multi-type manifests). **subtype**: Widget sub-type (for use with multi-sub-type manifests). **Returns** -1 = Items are not supported. 0 = Items are supported. 1 = The widget is a preferred visualization of the items. 2 = The widget is the best suited visualization for the items.	Ranks the compatibility of the given metadata items with the widget.
data.populateMetadata(...)	**Arguments** **widget**: Widget model to populate items. **items**: Items to populate into the widget.	Implements the widget meta-data population of the items.
data.getMeasureFilteredItem(...)	**Arguments** **widget**: Widget model from where to get the item from. **Returns** The data panel item to be applied with the measure filter.	Gets the data panel item on which the measure filter should be applied.
data.**panels** States the data panels array with which the widget is		
panel.name	string	States the name of the panel.
panel.type	string	States the type of the panel.
panel.noDrag	boolean	Indicates if data items can be dragged in/out to/from the panel.
panel.visibility(...)	**Arguments** **widget**: Triggering widget. **Returns** boolean value indicating whether the panel should be shown in the widget editor.	Dynamically show/hides the panel.

panel.defaultFormatter(...)	**Arguments** **widget**: Triggering widget. **item**: Triggering item. **Returns** Default formatting object. `{ type: "range", steps: 9, rangeMode: "auto" }`	Generates the default format object to be attached by default to panel items.
panel.canMask(...)	**Arguments** **item**: Triggering item. **Returns** boolean value indicating whether the item can be masked.	Determines whether the item can be masked.
panel.color(...)	**Arguments** **data**: Cell data. **cell**: Triggering cell. **item**: Metadata item to which the given cell belongs. **widget**: Triggering widget. **Returns** string color value	Applies custom coloring to every result cell.
panel.size(...)	**Arguments** **data**: Cell data. **cell**: Triggering cell. **item**: Metadata item to which the given cell belongs. **widget**: Triggering widget. **minvalue**: Min value in the item's column. **maxvalue**: Max value in the item's column. **Returns** float number between 0 and 1	Applies custom sizing to every result cell.
panel.mask(...)	**Arguments** **data**: Cell data. **cell**: Triggering cell. **item**: Metadata item to which the given cell belongs. **widget**: Triggering widget. **Returns** string masked value	Applies custom masking to every result cell.
panel.itemAttributes	[string]	Array of supported formatting attributes for the panel's data items. Available values are color and size.
panel.canDisableItems	boolean	Defines whether items on the panel can be disabled.
panel.itemDisabledStateChanged(...)	**Arguments** **widget**: Triggering widget. **item**: Triggering item.	Triggered when an item was enabled or disabled.

panel.allowedColoringTypes(...)	**Arguments** **widget**: Triggering widget. **Returns** An object stating which coloring methods are supported. `{ color: boolean, range: boolean, condition: boolean}`	Determine which of the following coloring methods are supported by the given widget color, range, or condition.
panel.itemAdded(...)	**Arguments** **widget**: Widget triggering the callback. **item**: Added item.	Called when a new item is added to the panel.
panel.**metadata**		
metadata.types	[string]	Defines the JAQL types the panel can hold - *measures* and *dimensions* (I/E ["measures", "dimensions"] allows addition of both measures and dimensions.
metadata.maxitems	integer	Defines the maximal number of items the panel can hold.
metadata.mixed	boolean	Defines whether the panel can mix both Measures and Dimensions.
metadata.sortable(...)	**Arguments** **widget**: Triggering widget. **item**: Triggering item. **Returns** boolean	Determine whether a given item in the given widget instance can be sorted.

widgetLayout Class

Groups a set of widget-layout related attributes in a single object.

Properties

Property	Type	Description
col	column object	Ancestor column object.
colidx	integer	Column index.
cell	cell object	Ancestor cell object.
cellidx	integer	Cell index in its parent column cells collection.
subcell	subcell object	Ancestor sub-cell object.
subcellidx	integer	Sub-cell index in its parent cell sub-cells collection.
element	element object	Parent element object.
elementidx	integer	Element index in its parent sub-cell elements collection.

Indicator Object

The indicatorInstance object represents an Indicator widget.

You can modify the styling of an Indicator widget through the indicatorInstance object setOptions method.

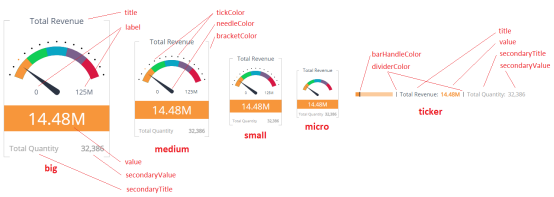
Note: This object is available from Sisense V6.7.1 and later.

Example:

```
widget.on('initialized', function(w) {
  var options = {
    title: {
      fontFamily: 'Helvetica, sans-serif',
      fontWeight: 'bold',
      fontSizes: {
        big: 40,
        medium: 30
      },
      color: '#29aba4'
    },
    value: {
      fontFamily: 'Helvetica, sans-serif',
      fontStyle: 'italic',
      color: '#e9e0d6'
    },
    bracketColor: '#f2ae72'
  };
  w.indicatorInstance.setOptions('numericBar', options);
});
```

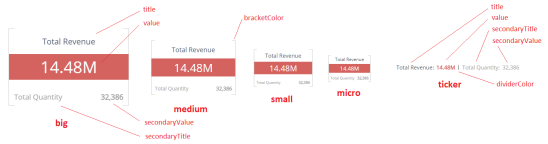
The following images display labels that define all the elements you can customize for each type of Indicator widget.

Gauge



Numeric Simple

Numeric Bar



Methods

Method	Arguments	Return Value	Description
setOptions	type options		The setOptions method defines styling for an Indicator widgets. **type**: Defines the type widget to be modified. The supported values are as follows: • gauge • numericSimple • numericBar • ticker **option**s: Options for specified type of indicator. See the Options Structure below for more information.

Options Structure

Property	Type	Details
These properties applies to all Indicator widget types		

title	object	Set of options to style the title section of an Indicator widget. The object has following properties:
		• **fontStyle** - {string} Specifies the font style. Possible values:
		  • normal
		  • italic
		  • oblique
		• **fontVariant** - {string} Specifies the font variant. Possible values:
		  • normal
		  • small-caps
		• **fontWeight** - {string | number} Specifies the font weight. Possible values:
		  • normal
		  • bold
		  • bolder
		  • lighter
		  • 100
		  • 200
		  • 300
		  • 400
		  • 500
		  • 600
		  • 700
		  • 800
		  • 900
		• **fontSizes** - {object} (only for main types) Specifies font sizes for different indicator sizes. The object has the following properties:
		  • big - {string | number} Specifies the font size for the big indicators, in pixels. You can specify it as a number or as a string (number plus 'px', 200px).
		  • medium - {string | number} Specifies the font size for the medium indicators, in pixels.
		  • small - {string | number} Specifies the font size for the small indicators, in pixels.
		  • micro - {string | number} Specifies the font size for the micro indicators, in pixels.
		• **fontSize** - {string | number} (only for ticker type) Specifies the font size, in pixels. You can specify it as a number or as a string (number plus 'px', 200px).
		• **fontFamily** - {string} Specifies the font family. You can also specify it as a set of the main font and fallback fonts divided by a comma.
		• **color** - {string} Specifies the color of the text.

value	object	Set of options to style the value section. The object has following properties: • fontStyle • fontVariant • fontWeight • fontSizes - (only for main types) • fontSize - (only for ticker type) • fontFamily • color
secondaryTitle	object	Set of options to style secondaryTitle section. The object has following properties: • fontStyle • fontVariant • fontWeight • fontSizes - (only for main types) • fontSize - (only for ticker type) • fontFamily • color
secondaryValue	object	Set of options to style secondaryValue section. The object has following properties: • fontStyle • fontVariant • fontWeight • fontSizes - (only for main types) • fontSize - (only for ticker type) • fontFamily • color
backgroundColor	string	Background color of the indicator widget.
Gauge Indicator widget specific properties		

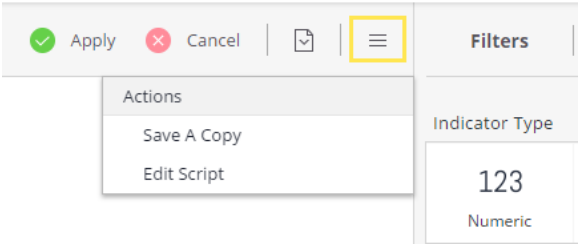
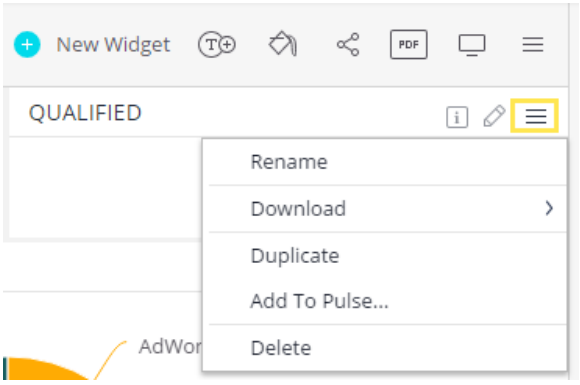
label	object	Set of options to style label sections. The object has following properties: • fontStyle • fontVariant • fontWeight • fontSizes • fontFamily • color
needleColor	string	Color of the needle.
tickColor	string	Color of the ticks.
bracketColor	string	Color of the brackets.
Numeric Simple Indicator widget specific properties		
borderColor	string	Color of the border.
Numeric Bar Indicator widget specific properties		
bracketColor	string	Color of the brackets.
Ticker Indicator widget specific properties. Note, that ticker properties will be used by all the indicator types' tickers		
dividerColor	string	Color of the divider.
barHandleColor (only for gauge type)	string	Color of the bar handle.

Global Events

Prism.On

Global events are available for registration *via prism.on(eventname)*.

Event Name	Event Arguments	Description
apploaded		Fired when the whole Sisense Web Application is loaded.

beforemenu	settings **name**: Menu name. **items**: Menu items to show. **widget**: Widget reference for widget menus. ui ...	Fired before every menu is opened and allows the removal of existing menu items and the addition of custom menu items. Available menu names to customize: • datapoint • dashboard • widget • widgetindashboard • widget-metadataitem • widget-metadataitem-sort • widget-metadataitem-type • databrowser-field • databrowser-formula • formula-metadataitem • dashboard-filter
beforewidgetmenu	**widget** (widget) Widget instance. **items** (object array) Array of items to be displayed. **element** (dom object) Source element that triggers. the event. **cancel** (boolean) Determines whether the menu is displayed or not.	Fired when clicking the widget menu while in Edit Widget mode. 
beforewidgetindashboardmenu	**widget** (widget) widget instance **items** (object array) array of items to be displayed **element** (dom object) source element that triggers the event **cancel** (boolean) Determines whether the menu is displayed	Fired when clicking the widget's menu while viewing a dashboard. 
homeloaded		Fired when the home page in the Sisense Web Application is loaded.
widgetloaded	**widget**: (**widget)** Widget instance.	Fired when a widget model was loaded to the widget editor.
unwidgetloaded	**widget**: (**widget)** widget instance.	Fired when a dashboard model was unloaded from the widget editor.
beforedashboardloaded	**dashboard**: Dashboard JSON.	Fired before a dashboard JSON is loaded into a dashboard model.
dashboard loaded	**dashboard**: (**dashboard)** Dashboard instance.	Fired when a dashboard model is loaded into the environment.
dashboard unloaded	**dashboard**: (**dashboard)** Dashboard instance.	Fired when a dashboard model is unloaded from the environment.

Widget instance.