



## SAP Crystal Reports JavaScript API Developer Guide

- SAP BusinessObjects Business Intelligence platform 4.1

2013-09-20

## Copyright

© 2013 SAP AG or an SAP affiliate company. All rights reserved. No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP AG. The information contained herein may be changed without prior notice. Some software products marketed by SAP AG and its distributors contain proprietary software components of other software vendors. National product specifications may vary. These materials are provided by SAP AG and its affiliated companies ("SAP Group") for informational purposes only, without representation or warranty of any kind, and SAP Group shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP Group products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty. SAP and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP AG in Germany and other countries. Please see <http://www.sap.com/corporate-en/legal/copyright/index.epx#trademark> for additional trademark information and notices.

2013-09-20

# Contents

<b>Chapter 1</b>	<b>About the SAP Crystal Reports JavaScript API .....</b>	<b>5</b>
<b>Chapter 2</b>	<b>Using the SAP Crystal Reports JavaScript API.....</b>	<b>7</b>
2.1	Create a report viewer and open a report.....	7
2.2	Show and hide viewer controls.....	9
2.3	Change the look and feel the JavaScript viewer.....	9
2.4	Event listeners.....	10
<b>Chapter 3</b>	<b>API reference.....</b>	<b>11</b>
3.1	SAP.CR.Viewer.ActionListener.....	11
3.1.1	onEvent.....	11
3.1.2	removeEvent.....	12
3.2	SAP.CR.Viewer.CanvasListener.....	13
3.2.1	onEvent.....	13
3.2.2	removeEvent.....	14
3.3	SAP.CR.Parameter.....	15
3.3.1	addValue.....	16
3.3.2	setReportName.....	17
3.4	SAP.CR.Parameter.RangeValue.....	17
3.4.1	setBeginValue.....	17
3.4.2	setEndValue.....	18
3.4.3	setLowerBound.....	19
3.4.4	setUpperBound.....	20
3.5	SAP.CR.Viewer.....	20
3.5.1	addActionListener.....	21
3.5.2	addCanvasListener.....	21
3.5.3	batchExecute.....	21
3.5.4	create.....	22
3.5.5	drilldown.....	22
3.5.6	getInstance.....	23
3.5.7	refresh.....	23
3.5.8	removeActionListener.....	23

3.5.9	removeCanvasListener.....	23
3.5.10	setDisplayBreadcrumb.....	24
3.5.11	setDisplayLeftPanel.....	24
3.5.12	setDisplayStatusBar.....	24
3.5.13	setDisplayToolbar.....	25
3.5.14	setHasLogo.....	25
3.5.15	setHasRefreshButton.....	25
3.5.16	setLogo.....	26
3.5.17	setPageNumber.....	26
3.5.18	setParameters.....	26
3.5.19	setPrintMode.....	27
3.5.20	setPromptOnRefresh.....	27
3.5.21	setReportMode.....	28
3.5.22	setReportSource.....	28
3.6	SAP.CR.Viewer.ThemeManager.....	29
3.6.1	setThemeColor.....	29
3.6.2	setThemeFont.....	29

<b>Appendix A</b>	<b>More Information.....</b>	<b>31</b>
-------------------	------------------------------	-----------

<b>Index</b>		<b>33</b>
--------------	--	-----------

## About the SAP Crystal Reports JavaScript API

With the SAP Crystal Reports JavaScript API, you can display Crystal reports content in an embedded web application without any client-side component installation. The JavaScript API lets you customize the report viewer, and add interactivity to your Crystal reports content. You can develop your web application in any language that uses JavaScript, because reports are generated in DHTML.

API features include:

- Event listeners for mouse clicks and mouse over events.
- Action listeners for printing and exporting events.
- Methods to customize color and font.
- Methods to show and hide viewer components.

The JavaScript API supports most OpenDocument parameters. If you previously used OpenDocument URLs to link to reports stored in the BI Platform repository, you can transition to the JavaScript API without losing functionality.



## Using the SAP Crystal Reports JavaScript API

The SAP Crystal Reports JavaScript API is included as part of the SAP BusinessObjects Business Intelligence Platform installation. The API can be accessed by referencing the `ViewerSeed.js` file located on the BIP server.

General workflow, shown in detail in this guide:

1. Find the location of the `ViewerSeed.js` file. In a typical BIP installation, the file is available at the following location: `http://localhost:8080/clientapi/CR/ViewerSeed.js`
2. Retrieve a logon token to use for BI platform authentication.
3. Add a reference to the `ViewerSeed.js` file to your web application.
4. Add the CR Embedded viewer to your web application.
5. Set the report source.

### Note:

If you are developing your application on your local computer instead of on deployed to a web server, the Adobe Flash Player may prevent the Flex prompting dialog from appearing. You can resolve this issue by deploying your application to a web server or designating your application content as trusted in the Flash Player Global Security Settings.

### 2.1 Create a report viewer and open a report

1. Set up `head`, `style` and `body` opening and closing tags.
2. Add a script tag within the `head` tags and use it to reference `ViewerSeed.js`.

```
<script src="http://localhost:8080/clientapi/CR/ViewerSeed.js">
```

3. Create a token variable, which will hold the logon token and set it to `null`.

```
var token = null;
```

4. Create a function called `init` that sets the value of `token` to the logon token and creates a new JavaScript viewer instance.

Refer to the Business Intelligence Platform RESTful Web Services Developer Guide for information on generating a logon token.

```
init(){
    token = "logonToken";
    SAP.CR.Viewer.create("viewerName", "container", onViewerInit);
}
```

5. Add a function called `onViewerInit` that sets the report source.

```
function onViewerInit(viewerInstance){
    viewerInstance.setReportSource('reportID', token);
}
```

The `reportID` is the numerical code assigned to a report.

You can discover the report ID programmatically or through the Central Management Console (CMC). When you right-click a report in the CMC, the report ID is listed with the report properties. For more information see the SAP Crystal Reports RESTful web services API.

6. Within the `body` tag set `onload` event handler equal to the `init` function.

```
<body onload="init()">
```

This calls the initialization function once the page is fully loaded, which causes any text or buttons coded into the body to load before the report viewer.

7. Within the `style` tag create a new class that defines the position and size of the JavaScript viewer. In the code below the viewer is given absolute positioning. Its width and height will be 75% of the size of the width and height of the page the viewer will be opened in. The position of the viewer in the page is specified by the `left` and `top` properties.

```
.viewerStyle
{
    position : absolute;
    left : 12.5%;
    top : 20.5%;
    width : 75%;
    height : 75%;
}
```

8. Within the body tags add code to set the style of the viewer.

```
<div id="container" class="viewerStyle"></div>
```

### Example: Create a report viewer and open a report

The following example opens a report with report ID 1234 in a report viewer given the name `crystalViewer`. The value set to the `token` variable in the `init` function is an example. The value you use must be generated by you. The viewer is given absolute positioning. Its width and height will be 75% of the size of the width and height of the page the viewer will be opened in. The position of the viewer in the page is specified by the `left` and `top` properties. The `onViewerFailure` function is optional.

```
<head>

<script src="http://computername/clientapi/CR/ViewerSeed.js">

<script>
    var token = null;

    function init(){
        token = "COMMANDCOM-LCM:6400@{3&2=5328,U3&p=40676
            .8926203819,Y7&4F=12,U3&63=secEnterprise
            ,0P&66=60,03&68=secEnterprise:Administrator
            ,0P&qe=100,U3&vz=IVD21LbMCB0eRiI4at
            z9sNL18Ux5anRBdYB9fFv5NrY,UP}";
        SAP.CR.Viewer.create("crystalViewer", 'viewerContainer1', onViewerInit, onViewerFailure);
    }

    function onViewerInit(){
        viewerInstance.setReportSource('1234', token);
    }
}
```



```
function onViewFailure(instance, error){
    alert(error);
}

</script>
</head>
<style>
    .viewerStyle
    {
        position : absolute;
        left : 12.5%;
        top : 20.5%;
        width : 75%;
        height : 75%;
    }
</style>
<body>
    <div id="viewerContainer1" class = "viewerStyle"></div>
</body>
```

---

## 2.2 Show and hide viewer controls

The JavaScript API allows you to show or hide the following browser controls in the JavaScript viewer:

- The left panel (used to find, set parameters and view the "Group Tree.")
- The toolbar.
- The status bar.
- The page navigation bar.

This is done by setting the control to `true` or `false`. By default all browser controls are set to `true`.

### Example:

The following code causes all control bars and panels to be displayed, except for the status bar.

```
var viewerInstance = SAP.CR.Viewer.getInstance("crystalViewer");

viewerInstance.setDisplayToolbar(true);
viewerInstance.setDisplayLeftPanel(true);
viewerInstance.setDisplayStatusbar(false);
viewerInstance.setDisplayBreadcrumb(true);
```

---

## 2.3 Change the look and feel the JavaScript viewer

The JavaScript API allows you to customize the color and font used in the JavaScript viewer.

**Example:**

The following example changes the color of the viewer to a light purple gradient and the font to Times. The first parameter in the `setThemeColor` method is the hexadecimal representation of a color. Use of `true` instead of `false` as the boolean value of the second parameter gives the viewer a solid color instead of a gradient.

```
SAP.CR.Viewer.ThemeManager.setThemeColor("#CDB7F9", false);
SAP.CR.Viewer.ThemeManager.setThemeFont("times");
```

---

## 2.4 Event listeners

The JavaScript API supports canvas and action event listeners. Event listeners detect when a certain event has occurred, and then trigger a response. Canvas events are mouse events (clicks and mouse-overs) that occur on the report canvas. Action listeners deal with action events, such as clicking a button or selecting an item from a list.

**Example:**

The following code causes report objects to get highlighted in green on a mouseover event. When the cursor leaves the report object the background color property is cleared. This means that the default background color of the object is not restored.

```
var canvasListener = new SAP.CR.Viewer.CanvasListener();
var viewerInstance = SAP.CR.Viewer.getInstance("crystalViewer");

canvasListener.onEvent(SAP.CR.Viewer.CanvasEvents.REPORT_ELEMENT_MOUSE_ENTER, function(args) {
    args.target.style.backgroundColor="green";
});

canvasListener.onEvent(SAP.CR.Viewer.CanvasEvents.REPORT_ELEMENT_MOUSE_LEAVE, function(args) {
    args.target.style.backgroundColor=" ";
});

viewerInstance.addCanvasListener(canvasListener);
```

---

**Related Topics**

- [SAP.CR.Viewer.CanvasListener](#)
- [SAP.CR.Viewer.ActionListener](#)

## API reference

This section provides a reference of the classes and methods in the JavaScript API.

### 3.1 SAP.CR.Viewer.ActionListener

The `ActionListener` class allows you to add custom functionality for viewer action events, for most items in the toolbar.

#### Syntax

```
var actionListener = new SAP.CR.Viewer.ActionListener();
```

#### 3.1.1 onEvent

The `onEvent` method registers an action listener for an event. Multiple listeners can be registered for the same event.

#### Syntax

```
var actionListener = new SAP.CR.Viewer.ActionListener();  
actionListener.onEvent(eventName, listener);
```

#### Parameters

- `eventName` - The name of the event.

Possible values:

Value	Description
<code>SAP.CR.Viewer.ActionEvents.EXPORT</code>	User clicks the export button.
<code>SAP.CR.Viewer.ActionEvents.PRINT</code>	User clicks the print button.
<code>SAP.CR.Viewer.ActionEvents.DRILL</code>	User drills down to the report from the group tree, breadcrumb, content or a chart.
<code>SAP.CR.Viewer.ActionEvents.GROUP_TREE_NAVIGATE</code>	User clicks on a group tree node that changes the page being viewed.
<code>SAP.CR.Viewer.ActionEvents.PROMPT</code>	Report prompts for database or parameters.
<code>SAP.CR.Viewer.ActionEvents.ERROR</code>	An error occurs in the viewer.

- `listener` - `[function(arg)]` Custom function that you define, that determines what happens when the event occurs. The parameter `arg` can be given any name except for JavaScript keyword names, and is created by the handler of the report viewer and passed to the function.

### 3.1.2 removeEvent

The `removeEvent` method removes listeners from an `ActionListener` object. The method can be used to remove a specific listener, with an associated event name or to remove all listeners with an associated event name.

#### Syntax

```
var actionListener = new SAP.CR.Viewer.ActionListener();
actionListener.removeEvent(eventName, listener);
```

#### Parameters

- `eventName` - The name of the event to be removed.

Possible values:

Value	Description
<code>SAP.CR.Viewer.ActionEvents.EXPORT</code>	User clicks the export button.
<code>SAP.CR.Viewer.ActionEvents.PRINT</code>	User clicks the print button.
<code>SAP.CR.Viewer.ActionEvents.DRILL</code>	User drills down to the report from the group tree, breadcrumb, content or a chart.
<code>SAP.CR.Viewer.ActionEvents.GROUP_TREE_NAVIGATE</code>	User clicks on a group tree node that changes the page being viewed.
<code>SAP.CR.Viewer.ActionEvents.PROMPT</code>	Report prompts for database or parameters.
<code>SAP.CR.Viewer.ActionEvents.ERROR</code>	An error occurs in the viewer.

- `listener` - [function(arg)] The specific listener to be removed. A custom function defined by the user when the event listener was created.

## 3.2 SAP.CR.Viewer.CanvasListener

The `CanvasListener` class allows you to add custom functionality for mouse events that occur on the report page.

### Syntax

```
var canvasListener = new SAP.CR.Viewer.CanvasListener();
```

### 3.2.1 onEvent

The `onEvent` method registers a canvas listener for an event. Multiple listeners can be registered for the same event.

### Syntax

```
var canvasListener = new SAP.CR.Viewer.CanvasListener();
canvasListener.onEvent(eventName, listener);
```

## Parameters

- `eventName` - The name of the event.

Possible values:

Value	Description
<code>SAP.CR.Viewer.CanvasEvents.REPORT_ELEMENT_MOUSE_ENTER</code>	Curser enters a report element.
<code>SAP.CR.Viewer.CanvasEvents.REPORT_ELEMENT_MOUSE_LEAVE</code>	Curser leaves a report element.
<code>SAP.CR.Viewer.CanvasEvents.REPORT_ELEMENT_CLICK</code>	User clicks a report element.
<code>SAP.CR.Viewer.CanvasEvents.REPORT_ELEMENT_RIGHT_CLICK</code>	User right-clicks a report element.
<code>SAP.CR.Viewer.CanvasEvents.REPORT_CANVAS_MOUSE_ENTER</code>	Curser enters the report canvas.
<code>SAP.CR.Viewer.CanvasEvents.REPORT_CANVAS_MOUSE_LEAVE</code>	Curser leaves the report canvas.
<code>SAP.CR.Viewer.CanvasEvents.REPORT_CANVAS_CLICK</code>	User clicks the report canvas.
<code>SAP.CR.Viewer.CanvasEvents.REPORT_CANVAS_RIGHT_CLICK</code>	User right-clicks the report canvas.

- `listener` - `[function(arg)]` Custom function that you define, that determines what happens when the event occurs. The parameter `arg` can be given any name except for JavaScript keyword names, and is created by the handler of the report viewer and passed to the function.

### 3.2.2 removeEvent

The `removeEvent` method removes listeners from a `CanvasListener` object. The method can be used to remove a specific listener, with an associated event name or to remove all listeners with an associated event name.

#### Syntax

```
var canvasListener = new SAP.CR.Viewer.CanvasListener();
canvasListener.removeEvent(eventName, listener);
```

## Parameters

- `eventName` - The name of the event to be removed.

Possible values:

Value	Description
<code>SAP.CR.Viewer.CanvasEvents.REPORT_ELEMENT_MOUSE_ENTER</code>	Cursor enters a report element.
<code>SAP.CR.Viewer.CanvasEvents.REPORT_ELEMENT_MOUSE_LEAVE</code>	Cursor leaves a report element.
<code>SAP.CR.Viewer.CanvasEvents.REPORT_ELEMENT_CLICK</code>	User clicks a report element.
<code>SAP.CR.Viewer.CanvasEvents.REPORT_ELEMENT_RIGHT_CLICK</code>	User right-clicks a report element.
<code>SAP.CR.Viewer.CanvasEvents.REPORT_CANVAS_MOUSE_ENTER</code>	Cursor enters the report canvas.
<code>SAP.CR.Viewer.CanvasEvents.REPORT_CANVAS_MOUSE_LEAVE</code>	Cursor leaves the report canvas.
<code>SAP.CR.Viewer.CanvasEvents.REPORT_CANVAS_CLICK</code>	User clicks the report canvas.
<code>SAP.CR.Viewer.CanvasEvents.REPORT_CANVAS_RIGHT_CLICK</code>	User right-clicks the report canvas.

- `listener` - [function(arg)] The specific listener to be removed. A custom function defined by the user when the event listener was created.

## 3.3 SAP.CR.Parameter

The `SAP.CR.Parameter` class allows you to create parameters and add them to a viewer instance. The class can also be used to set parameter values in a sub-report.

### Syntax

```
var param = new SAP.CR.Parameter(parameterName,parameterType);
```

### Parameters

- `parameterName` - [String] The name of the parameter in the report. This value is often pre-defined by the report.
- `parameterType` - The type of the parameter.

Possible values:

Value	Description
<code>SAP.CR.Parameter DataTypes.BOOLEAN</code>	Boolean value (true or false.)
<code>SAP.CR.Parameter DataTypes.CURRENCY</code>	Numerical value.
<code>SAP.CR.Parameter DataTypes.DATE_TIME</code>	Date and time specified using the JavaScript <code>Date</code> function.
<code>SAP.CR.Parameter DataTypes.DATE</code>	Date specified using the JavaScript <code>Date</code> function.
<code>SAP.CR.Parameter DataTypes.TIME</code>	Time specified using the JavaScript <code>Date</code> function.
<code>SAP.CR.Parameter DataTypes.NUMBER</code>	Numerical value.
<code>SAP.CR.Parameter DataTypes.STRING</code>	String value.

### 3.3.1 addValue

Sets the value of a parameter.

#### Syntax

```
var param = new SAP.CR.Viewer.Parameter(parameterName,parameterType);
param.addValue(val);
```

#### Parameters

- `val` - Parameter value. The type depends on `parameterType`. Can be a discreet or range value.



### 3.3.2 setReportName

Sets the parameters of a sub report. If you do not use the `setReportName` method the parameters are automatically set to the main report.

#### Syntax

```
var param = new SAP.CR.Viewer.Parameter(parameterName,parameterType);  
param.setReportName(subReportName);
```

#### Parameters

- `subReportName` - [String] Name of the sub report you are setting the parameters to.

#### Example:

The following example creates a new parameter of type `SAP.CR.DataTypes.NUMBER`, and adds it to the sub report with name `subReport1`.

```
var instance = SAP.CR.Viewer.getInstance("crystalViewer");  
  
var param = new SAP.CR.Parameter("subNumberParameter",SAP.CR.Parameter.DataTypes.NUMBER);  
param.setReportName("subReport1");  
param.addValue(10);  
instance.setParameters([param]);
```

---

## 3.4 SAP.CR.Parameter.RangeValue

The `SAP.CR.Parameter.Range` class is used to specify a range of values for a parameter. The `SAP.CR.Parameter.Range` object is added to an `SAP.CR.Parameter` object using the `addValue` method.

#### Syntax

```
var range = new SAP.CR.Parameter.RangeValue();
```

### 3.4.1 setBeginValue

Sets the first of the range of values.

#### Syntax

```
var range = new SAP.CR.Parameter.RangeValue();  
range.setBeginValue(val);
```

### Parameters

- `val` - The lower bound of the range of values.

Possible types for `val`:

Value	Description
<code>SAP.CR.Parameter .DataTypes.BOOLEAN</code>	Boolean value ( <code>true</code> or <code>false</code> .)
<code>SAP.CR.Parameter .DataTypes.CURRENCY</code>	Numerical value.
<code>SAP.CR.Parameter .DataTypes.DATE_TIME</code>	Date and time specified using the JavaScript <code>Date</code> function.
<code>SAP.CR.Parameter .DataTypes.DATE</code>	Date specified using the JavaScript <code>Date</code> function.
<code>SAP.CR.Parameter .DataTypes.TIME</code>	Time specified using the JavaScript <code>Date</code> function.
<code>SAP.CR.Parameter .DataTypes.NUMBER</code>	Numerical value.
<code>SAP.CR.Parameter .DataTypes.STRING</code>	String value.

The type must be the same as that of the parameter the range value will be added to.

### 3.4.2 `setEndValue`

Sets the last of the range of values.

#### Syntax

```
var range = new SAP.CR.Parameter.RangeValue();
range.setEndValue(val);
```

#### Parameters

- `val` - The upper bound of the range of values.

Possible types for val:

Value	Description
<code>SAP.CR.Parameter .DataTypes.BOOLEAN</code>	Boolean value (true or false.)
<code>SAP.CR.Parameter .DataTypes.CURRENCY</code>	Numerical value.
<code>SAP.CR.Parameter .DataTypes.DATE_TIME</code>	Date and time specified using the JavaScript <code>Date</code> function.
<code>SAP.CR.Parameter .DataTypes.DATE</code>	Date specified using the JavaScript <code>Date</code> function.
<code>SAP.CR.Parameter .DataTypes.TIME</code>	Time specified using the JavaScript <code>Date</code> function.
<code>SAP.CR.Parameter .DataTypes.NUMBER</code>	Numerical value.
<code>SAP.CR.Parameter .DataTypes.STRING</code>	String value.

The type must be the same as that of the parameter the range value will be added to.

### 3.4.3 setLowerBound

Sets the type of the lower bound of the range parameter.

#### Syntax

```
var range = new SAP.CR.Parameter.RangeValue();
range.setLowerBound(boundType);
```

#### Parameters

- `boundType` - Determines the type of the lower bound.

Possible values:

Value	Description
<code>SAP.CR.Parameter.Parameter</code> <code>.RangeBoundTypes.INCLUSIVE</code>	Range of values includes the begin value.
<code>SAP.CR.Parameter.Parameter</code> <code>.RangeBoundTypes.EXCLUSIVE</code>	Range of values does not include the begin value.
<code>SAP.CR.Parameter.Parameter</code> <code>.RangeBoundTypes.UNBOUNDED</code>	No lower bound.

### 3.4.4 setUpperBound

Sets the type of the upper bound of the range parameter.

#### Syntax

```
var range = new SAP.CR.Parameter.RangeValue();
range.setUpperBound(boundType);
```

#### Parameters

- `boundType` - Determines the type of the upper bound.

Possible values:

Value	Description
<code>SAP.CR.Parameter.Parameter</code> <code>.RangeBoundTypes.INCLUSIVE</code>	Range of values includes the end value.
<code>SAP.CR.Parameter.Parameter</code> <code>.RangeBoundTypes.EXCLUSIVE</code>	Range of values does not include the end value.
<code>SAP.CR.Parameter.Parameter</code> <code>.RangeBoundTypes.UNBOUNDED</code>	No upper bound.

## 3.5 SAP.CR.Viewer

The `SAP.CR.Viewer` class contains methods that allows you to create an instance of the report viewer, and to customize and add functionality to it.

### 3.5.1 addActionListener

Adds an action event listener to a viewer instance.

#### Syntax

```
var viewerInstance = SAP.CR.Viewer.getInstance(viewerName);
viewerInstance.addActionListener(aListener);
```

#### Parameters

- `aListener` - Name of the action listener you are adding. Must be an instance of `SAP.CR.ActionListener`.

### 3.5.2 addCanvasListener

Adds a canvas event listener to a viewer instance.

#### Syntax

```
var viewerInstance = SAP.CR.Viewer.getInstance(viewerName);
viewerInstance.addCanvasListener(cListener);
```

#### Parameters

- `cListener` - Name of the canvas listener you are adding. Must be an instance of `SAP.CR.CanvasListener`.

### 3.5.3 batchExecute

The `batchExecute` method ensures that API calls are executed synchronously.

#### Syntax

```
var viewerInstance = SAP.CR.Viewer.getInstance(viewerName);
viewerInstance.batchExecute(function());
```

#### Parameters

- `function` - [function()] Function defined by you that contains the API calls to be executed in the order they are to be executed.

**Example:**

The following example drills down on a group path and then sets the page number to 2. If the `batchExecute` method was not used, two asynchronous calls would be sent to the server at the same time. This would cause only one of the two API calls to occur.

```
var viewerInstance = SAP.CR.Viewer.getInstance("crystalViewer");

viewerInstance.batchExecute(function() {
    viewerInstance.drillDown(["0"]);
    viewerInstance.setPageNumber(2);
});
```

---

### 3.5.4 create

Creates a CR Embedded viewer instance.

**Syntax**

```
SAP.CR.Viewer.create(viewerName, containerID, initCB, failCB)
```

**Parameters**

- `containerID` - [String] ID of the element that will hold the viewer.
- `viewerName` - [String] Name of the viewer.
- `initCB` - [function(instance)] Function that initializes the viewer instance. It is not defined within the method call. The parameter `instance` is the viewer instance.
- `failCB` - [function(instance,error)] Function that is executed if something fails. It is not defined within the method call. This is an optional parameter. The parameter `instance` is the viewer instance.

### 3.5.5 drilldown

Drills down to a group.

**Syntax**

```
var viewerInstance = SAP.CR.Viewer.getInstance(viewerName);
viewerInstance.drilldown([groupPath]);
```

**Parameters**

- `groupPath` - An array of integers, which displays the path you have drilled down on.

### 3.5.6 getInstance

Gets a viewer instance based on the viewer name.

#### Syntax

```
var viewerInstance = SAP.CR.Viewer.getInstance(viewerName);
```

#### Parameters

- viewerName - [String] Name of the viewer you are getting.

### 3.5.7 refresh

Refreshes the report displayed in the JavaScript viewer. The method refreshes the data in the report being displayed and if necessary prompts for parameters or logon information.

#### Syntax

```
var viewerInstance = SAP.CR.Viewer.getInstance(viewerName);  
viewerInstance.refresh();
```

### 3.5.8 removeActionListener

Removes an action event listener from a viewer instance.

#### Syntax

```
var viewerInstance = SAP.CR.Viewer.getInstance(viewerName);  
viewerInstance.removeActionListener(aListener);
```

#### Parameters

- aListener - Name of the action listener you are removing.

### 3.5.9 removeCanvasListener

Removes a canvas event listener from a viewer instance.

**Syntax**

```
var instance = SAP.CR.Viewer.getInstance(viewerName);  
instance.removeCanvasListener(cListener);
```

**Parameters**

- cListener - Name of the canvas listener you are removing.

### 3.5.10 setDisplayBreadcrumb

Show or hide the breadcrumb.

**Syntax**

```
var viewerInstance = SAP.CR.Viewer.getInstance(viewerName);  
viewerInstance.setDisplayBreadcrumb(isBreadcrumb);
```

**Parameters**

- isBreadcrumb - [Boolean] True to show breadcrumb or false to hide breadcrumb.

### 3.5.11 setDisplayLeftPanel

Show or hide the left panel, which can be used as the find panel, to view the "Group Tree", or to view and edit parameters.

**Syntax**

```
var viewerInstance = SAP.CR.Viewer.getInstance(viewerName);  
viewerInstance.setDisplayLeftPanel(isLeftPanel);
```

**Parameters**

- isLeftPanel - [Boolean] Set to true to show panel or false to hide panel.

### 3.5.12 setDisplayStatusBar

Show or hide the status bar along the bottom of the report viewer.

**Syntax**

```
var viewerInstance = SAP.CR.Viewer.getInstance(viewerName);  
viewerInstance.setDisplayStatusBar(isStatusBar);
```

**Parameters**

- isStatusBar - [Boolean] Set to true to show status bar or false to hide status bar.



### 3.5.13 setDisplayToolbar

Show or hide the toolbar.

#### Syntax

```
var viewerInstance = SAP.CR.Viewer.getInstance(viewerName);  
viewerInstance.setDisplayToolbar(isToolbar);
```

#### Parameters

- `isToolbar` - [Boolean] True to show toolbar or false to hide toolbar.

### 3.5.14 setHasLogo

Shows or hides the logo in the top, right corner of the report viewer. By default the logo is shown.

#### Note:

Must be called before the viewer has initialized, or an exception is thrown.

#### Syntax

```
var viewerInstance = SAP.CR.Viewer.getInstance(viewerInstance);  
viewerInstance.setHasLogo(isLogo);
```

#### Parameters

- `isLogo` - [Boolean] Set to `true` to show the logo or to `false` to hide it.

### 3.5.15 setHasRefreshButton

Shows or hides the refresh button in the report viewer's toolbar. By default the refresh button is hidden.

#### Note:

Must be called before the viewer has initialized, or an exception is thrown.

#### Syntax

```
var viewerInstance = SAP.CR.Viewer.getInstance(viewerName);  
viewerInstance.setHasRefreshButton(isRefreshButton);
```

#### Parameters

- `isRefreshButton` - [Boolean] Set to `true` to show the refresh button or to `false` to hide it.

### 3.5.16 setLogo

Sets the image to be used as a logo in the top, right corner. Allows you to link a url to the logo and set a tooltip for when you mouseover the logo.

**Note:**

- Must be called before the viewer has initialized, or an exception is thrown.
- The image should be at most 100x125 pixels in size.

**Syntax**

```
var viewerInstance = SAP.CR.Viewer.getInstance(viewerName);
viewerInstance.setLogo(url, link, tooltip);
```

**Parameters**

- `url` - [String] Relative or absolute path to the image to be used as the logo.
- `link` - [String] The url that will be opened when the user clicks the logo.
- `tooltip` - [String] The tooltip that appears when you mouseover the logo. If it contains non-English characters it must be encoded.

### 3.5.17 setPageNumber

Sets the page number of the report page being viewed.

**Syntax**

```
var viewerInstance = SAP.CR.Viewer.getInstance(viewerName);
viewerInstance.setPageNumber(pageNumber);
```

**Parameters**

- `pageNumber` - [number] A positive number. If larger than the number of pages in the report the page number is set to the last page in the report.

### 3.5.18 setParameters

Sets report parameters. Multiple parameters can be set in one call of the method.

**Syntax**

```
var viewerInstance = SAP.CR.Viewer.getInstance(viewerName);
viewerInstance.setParameters([param]);
```

### Parameters

- `param` - An array of the parameter(s) being set to the viewer.

## 3.5.19 setPrintMode

Sets the print mode of the report viewer to either ACTIVEX or PDF.

### Note:

Must be called before the viewer has initialized, or an exception is thrown.

### Syntax

```
var viewerInstance = SAP.CR.Viewer.getInstance(viewerName);  
viewerInstance.setReportMode(mode);
```

### Parameters

- `mode` - The print mode.

Possible values:

Value	Description
<code>SAP.CR.PrintMode.ACTIVEX</code>	Print using the ActiveX printer. Only works in Internet Explorer.
<code>SAP.CR.PrintMode.PDF</code>	Print to PDF.

## 3.5.20 setPromptOnRefresh

Sets whether or not the viewer prompts for new parameter values after being refreshed. By default the viewer prompts for new parameter values.

### Note:

Must be called before the viewer has initialized, or an exception is thrown.

### Syntax

```
var viewerInstance = SAP.CR.Viewer.getInstance(viewerName);  
viewerInstance.setPromptOnRefresh(isPrompt);
```

### Parameters

- `isPrompt` - [Boolean] Set to `true` to prompt for new parameters or to `false` to use existing parameter values.

### 3.5.21 setReportMode

Sets the color of the background behind the report page and the alignment of the report pages in the report viewer.

**Note:**

Must be called before the viewer has initialized, or an exception is thrown.

**Syntax**

```
var viewerInstance = SAP.CR.Viewer.getInstance(viewerName);  
viewerInstance.setReportMode(mode);
```

**Parameters**

- `mode` - Sets the color and alignment of the report page in the report viewer.

Possible values:

Value	Description
<code>SAP.CR.Viewer .ReportMode.PRINT</code>	Sets the background of the report viewer to grey.
<code>SAP.CR.Viewer .ReportMode.WEB</code>	Sets the background of the report viewer to white. Report pages are not center aligned in the viewer.

### 3.5.22 setReportSource

Sets the report source based on a numerical object ID and an enterprise session. If needed you can also set the document locale.

**Syntax**

```
var viewerInstance = SAP.CR.Viewer.getInstance(viewerName);  
viewerInstance.setReportSource(infoObjectID, token, locale);
```

**Parameters**

- `infoObjectID` - [String] ID or report in info view/CMC.
- `token` - [String] Logon token. The token expires after a few minutes of inactivity and needs to be regenerated. Refer to the Business Intelligence Platform RESTful Web Services Developer Guide for how to generate a logon token.
- `locale` - [String] The document locale.

## 3.6 SAP.CR.Viewer.ThemeManager

The `ThemeManager` class allows you to set the color and font style used in the viewer.

### 3.6.1 setThemeColor

Sets the color of the viewer.

#### Syntax

```
SAP.CR.Viewer.ThemeManager.setThemeColor(hexColor,isSolid);
```

#### Parameters

- `hexColor` - [String] The hexadecimal representation of the color being used. Value always starts with the `#` symbol.
- `isSolid` - [Boolean] Set to `true` for a solid color or set to `false` for a gradient color.

### 3.6.2 setThemeFont

Sets the family font name used in the JavaScript viewer UI. This does not effect the font used in the report being viewed.

#### Syntax

```
SAP.CR.Viewer.ThemeManager.setThemeFont(font);
```

#### Parameters

- `font` - [String] The font to be used in the viewer. Any browser supported fonts can be used.



## More Information

Information Resource	Location
SAP BusinessObjects product information	<a href="http://www.sap.com">http://www.sap.com</a>
SAP Help Portal	<p>Navigate to <a href="http://help.sap.com/businessobjects">http://help.sap.com/businessobjects</a> and on the "SAP BusinessObjects Overview" side panel click <b>All Products</b>.</p> <p>You can access the most up-to-date documentation covering all SAP BusinessObjects products and their deployment at the SAP Help Portal. You can download PDF versions or installable HTML libraries.</p> <p>Certain guides are stored on the SAP Service Marketplace and are not available from the SAP Help Portal. These guides are listed on the Help Portal accompanied by a link to the SAP Service Marketplace. Customers with a maintenance agreement have an authorized user ID to access this site. To obtain an ID, contact your customer support representative.</p>
SAP Service Marketplace	<p><a href="http://service.sap.com/bosap-support">http://service.sap.com/bosap-support</a> &gt; Documentation</p> <ul style="list-style-type: none"> <li>• Installation guides: <a href="https://service.sap.com/bosap-instguides">https://service.sap.com/bosap-instguides</a></li> <li>• Release notes: <a href="http://service.sap.com/releasenotes">http://service.sap.com/releasenotes</a></li> </ul> <p>The SAP Service Marketplace stores certain installation guides, upgrade and migration guides, deployment guides, release notes and Supported Platforms documents. Customers with a maintenance agreement have an authorized user ID to access this site. Contact your customer support representative to obtain an ID. If you are redirected to the SAP Service Marketplace from the SAP Help Portal, use the menu in the navigation pane on the left to locate the category containing the documentation you want to access.</p>
Docupedia	<p><a href="https://cw.sdn.sap.com/cw/community/docupedia">https://cw.sdn.sap.com/cw/community/docupedia</a></p> <p>Docupedia provides additional documentation resources, a collaborative authoring environment, and an interactive feedback channel.</p>
Developer resources	<p><a href="https://boc.sdn.sap.com/">https://boc.sdn.sap.com/</a></p> <p><a href="https://www.sdn.sap.com/irj/sdn/businessobjects-sdklibrary">https://www.sdn.sap.com/irj/sdn/businessobjects-sdklibrary</a></p>

Information Resource	Location
SAP BusinessObjects articles on the SAP Community Network	<a href="https://www.sdn.sap.com/irj/boc/businessobjects-articles">https://www.sdn.sap.com/irj/boc/businessobjects-articles</a> These articles were formerly known as technical papers.
Notes	<a href="https://service.sap.com/notes">https://service.sap.com/notes</a> These notes were formerly known as Knowledge Base articles.
Forums on the SAP Community Network	<a href="https://www.sdn.sap.com/irj/scn/forums">https://www.sdn.sap.com/irj/scn/forums</a>
Training	<a href="http://www.sap.com/services/education">http://www.sap.com/services/education</a> From traditional classroom learning to targeted e-learning seminars, we can offer a training package to suit your learning needs and preferred learning style.
Online customer support	<a href="http://service.sap.com/bosap-support">http://service.sap.com/bosap-support</a> The SAP Support Portal contains information about Customer Support programs and services. It also has links to a wide range of technical information and downloads. Customers with a maintenance agreement have an authorized user ID to access this site. To obtain an ID, contact your customer support representative.
Consulting	<a href="http://www.sap.com/services/bysubject/businessobjectsconsulting">http://www.sap.com/services/bysubject/businessobjectsconsulting</a> Consultants can accompany you from the initial analysis stage to the delivery of your deployment project. Expertise is available in topics such as relational and multidimensional databases, connectivity, database design tools, and customized embedding technology.



# Index

## A

- ActionListener 11
  - onEvent 11
  - removeEvent 12
- addActionListener 21
- addCanvasListener 21
- addValue 16
- Ajax 5
- API 11

## B

- batchExecute 21
- browser controls 9

## C

- CanvasListener 13
  - onEvent 13
  - removeEvent 14
- color 9
- create 22

## D

- drilldown 22

## E

- event listener
  - action listener 10
  - canvas listener 10

## F

- font 9

## G

- getInstance 23

## I

- introduction 5

## O

- onEvent
  - ActionListener 11
  - CanvasListener 13

## P

- Parameter 15
- Parameter.RangeValue 17

## R

- refresh 23
- removeActionListener 23
- removeCanvasListener 23
- removeEvent
  - ActionListener 12
  - CanvasListener 14
- report
  - opening 7

## S

- SAP.CR.Parameter 15
  - addValue 16
  - setReportName 17
- SAP.CR.Parameter.RangeValue 17
  - setBeginValue 17
  - setEndValue 18
  - setLowerBound 19
  - setUpperBound 20
- SAP.CR.Viewer 21
  - addActionListener 21
  - addCanvasListener 21
  - batchExecute 21
  - create 22
  - drilldown 22
  - getInstance 23
  - refresh 23
  - removeActionListener 23
  - removeCanvasListener 23
  - setBreadcrumb 24
  - setDisplayLeftPanel 24
  - setDisplayStatusBar 24

- SAP.CR.Viewer (*continued*)

- setDisplayToolBar 25
  - setHasLogo 25
  - setHasRefreshButton' 25
  - setLogo 26
  - setPageNumber 26
  - setParameters 26
  - setPrintMode 27
  - setPromptOnRefresh 27
  - setReportMode 28
  - setReportSource 28
- SAP.CR.Viewer.ActionListener 11
- SAP.CR.Viewer.CanvasListener 13
- SAP.CR.Viewer.ThemeManager 29
  - setThemeColor 29
  - setThemeFont 29
- setBeginValue 17
- setBreadcrumb 24
- setDisplayLeftPanel 24
- setDisplayStatusBar 24
- setDisplayToolBar 25
- setEndValue 18
- setHasLogo 25
- setHasRefreshButton 25
- setLogo 26
- setLowerBound 19
- setPageNumber 26
- setParameters 26
- setPrintMode 27
- setPromptOnRefresh 27
- setReportMode 28
- setReportName 17
- setReportSource 28
- setThemeColor 29
- setThemeFont 29
- setUpperBound 20

## T

- ThemeManager 29

## V

- Viewer 21

