2015

Stimulsoft Reports.JS Quick Start Guide

Connecting library Stimulsoft Reports.JS

We made a library for working with Stimulsoft Reports on JavaScript, but we used the ideology of .NET, so all JavaScript code presented below uses .NET C# notation.

The library Stimulsoft Reports.JS provides to the developer the ability to quickly and easily integrate a powerful and flexible reporting system from Stimulsoft in yours web application written in any language.

To do this, connect the JavaScript library to the client application:

```
<script src="stimulsoft.reports.js"></script>
```

Additionally, to connect Stimulsoft Report Viewer you need to add CSS and JS:

```
<link href="stimulsoft.viewer.office2013.css" rel="stylesheet">
<script src="stimulsoft.viewer.js" type="text/javascript"></script>
```

Additionally, to connect Stimulsoft Report Designer you need to add CSS and JS:

```
<link href="stimulsoft.designer.office2013.white.blue.css" rel="stylesheet">
<script src="stimulsoft.designer.js" type="text/javascript"></script>
```

The appearance of the viewer and the designer can be modified using any of the available themes by simply connecting the desired CSS.

The list of CSS themes available for the viewer:

- stimulsoft.viewer.default.css
- stimulsoft.viewer.windows.xp.css
- stimulsoft.viewer.windows7.css
- stimulsoft.viewer.office2003.css
- stimulsoft.viewer.office2007.blue.css
- stimulsoft.viewer.office2007.silver.css
- stimulsoft.viewer.office2007.black.css
- stimulsoft.viewer.office2010.black.css
- stimulsoft.viewer.office2010.css
- stimulsoft.viewer.office2010.silver.css
- stimulsoft.viewer.office2013.css
- stimulsoft.viewer.office2013.white.carmine.css
- stimulsoft.viewer.office2013.white.green.css
- stimulsoft.viewer.office2013.white.orange.css
- stimulsoft.viewer.office2013.white.purple.css
- stimulsoft.viewer.office2013.white.teal.css
- stimulsoft.viewer.office2013.white.violet.css

The list of CSS themes available for the designer:

- stimulsoft.designer.office2013.white.blue.css
- stimulsoft.designer.office2013.white.carmine.css

- stimulsoft.designer.office2013.white.green.css
- stimulsoft.designer.office2013.white.orange.css
- stimulsoft.designer.office2013.white.purple.css
- stimulsoft.designer.office2013.white.teal.css
- stimulsoft.designer.office2013.white.violet.css
- stimulsoft.designer.office2013.lightgray.blue.css
- stimulsoft.designer.office2013.lightgray.carmine.css
- stimulsoft.designer.office2013.lightgray.green.css
- stimulsoft.designer.office2013.lightgray.orange.css
- stimulsoft.designer.office2013.lightgray.purple.css
- stimulsoft.designer.office2013.lightgray.teal.css
- stimulsoft.designer.office2013.lightgray.violet.css
- stimulsoft.designer.office2013.darkgray.blue.css
- stimulsoft.designer.office2013.darkgray.carmine.css
- stimulsoft.designer.office2013.darkgray.green.css
- stimulsoft.designer.office2013.darkgray.orange.css
- stimulsoft.designer.office2013.darkgray.purple.css
- stimulsoft.designer.office2013.darkgray.teal.css
- stimulsoft.designer.office2013.darkgray.violet.css
- stimulsoft.designer.office2013.verydarkgray.blue.css
- stimulsoft.designer.office2013.verydarkgray.carmine.css
- stimulsoft.designer.office2013.verydarkgray.green.css
- stimulsoft.designer.office2013.verydarkgray.orange.css
- stimulsoft.designer.office2013.verydarkgray.purple.css
- stimulsoft.designer.office2013.verydarkgray.teal.css
- stimulsoft.designer.office2013.verydarkgray.violet.css

The uncompressed version for customized settings for each CSS file is available.

Note: The HTML page must begin with a proper DOCTYPE definition to set the type of the current document in HTML5:

```
<!DOCTYPE html>
```

Due to limitations of the JavaScript language to work with external data, you must run a minimum server instance. This will provide the ability to open reports from files, use the external data in the report and to work with rendered reports.

Loading and Saving a Report

Two methods of the StiReport object are used to load a report – loadFile() and load(). They are used as follows:

- loadFile(filePath) – loads a report from the MRT file which path is specified in the filePath;
- load(str) – loads a report from the string str, that contains XML or JSON;
- load(data) – loads a report from the array data of the number[] type;
- load(xml) – loads a report from the XML of the XMLDocument type;
- load(json) – loads a report from the JS object.

For example, use the code below to load a report from file:

```
var report = new Stimulsoft.Report.StiReport();
report.loadFile("SimpleList.mrt");
```

The MRT format of Stimulsoft Reports is the JSON based description of reports. You can use MRT files, created in other Stimulsoft Reports designers in the JSON based description.

Use the code below to save a report to a string:

```
var report = new Stimulsoft.Report.StiReport();
var jsonString = report.saveToJsonString();
```

Use the code below to load a report from this string:

```
var report = new Stimulsoft.Report.StiReport();
report.load(jsonString);
```

Getting access to pages of a report

The report has a collection of pages, which can be accessed by the following way:

```
var report = new Stimulsoft.Report.StiReport();
report.loadFile("SimpleList.mrt");

for (var index = 0; index < report.pages.count; index++) {
        alert(report.pages.getByIndex(index).name);
}
```

Rendering a Report

The report includes presentation and data. To merge these entities and build a report using the “render()” method.

The following code renders a report:

```
var report = new Stimulsoft.Report.StiReport();
report.loadFile("SimpleList.mrt");
report.render();
```

Getting access to pages of a rendered report

After rendering the report, a collection of pages is created. It can be accessed by the following way:

```
var report = new Stimulsoft.Report.StiReport();
report.loadFile("SimpleList.mrt");
report.render();

for (var index = 0; index < report.renderedPages.count; index++) {
        alert(report.renderedPages.getByIndex(index).name);
}
```

Binding Data to a Report

To connect to the data you need to perform the “regData()” method of the report, passing it as a data source parameter. The data source can load data from XML files (with XSD scheme) and from JSON files.

For example, loading data from XML:

```
var dataSet = new Stimulsoft.System.Data.DataSet("SimpleDataSet");
dataSet.readXmlSchemaFile("Demo.xsd");
dataSet.readXmlFile("Demo.xml");

var report = new Stimulsoft.Report.StiReport();
report.regData(dataSet.dataSetName, "", dataSet);
```

You could just download JSON data source:

```
var dataSet = new Stimulsoft.System.Data.DataSet("SimpleDataSet");
dataSet.readJsonFile("Demo.json");

var report = new Stimulsoft.Report.StiReport();
report.regData(dataSet.dataSetName, "", dataSet);
```

The StiDataSet object has other methods for loading data:

- readJsonFile(fileName) – takes a path to JSON file as a parameter;
- readJson(string) – takes a string with JSON data as a parameter;
- readJson(data) – takes a byte array with JSON data as a parameter;
- readJson(obj) – takes a JavaScript object as a parameter;

- readXmlFile(fileName) – takes a path to XML file as a parameter;
- readXml(string) – takes a string with XML data as a parameter;
- readXml(data) – takes an byte array with XML data as a parameter;
- readXml(obj) – takes a JavaScript object as a parameter;

- readXmlSchemaFile(fileName) – takes a path to XML schema (XSD) file as a parameter;
- readXmlSchema(string) – takes a string with XML schema as a parameter;
- readXmlSchema(data) – takes an byte array with XML schema as a parameter;
- readXmlSchema(obj) – takes a JavaScript object as a parameter;

Note: Loading data schema is not required. If you want to use the scheme, you must load it before loading XML data.

To convert an array of bytes to string and back you can use the static functions – Stimulsoft.System.Text.Encoding.UTF8.getString(buffer) (returns a string derived from the byte array passed in the parameter) and Stimulsoft.System.Text.Encoding.UTF8.getBytes(str) (returns a byte array derived from the string passed in the parameter).

There is also another way of data management, which allows you to connect to different data sources and modify them in real time:

```
var report = new Stimulsoft.Report.StiReport();
var xmlDataBase = new Stimulsoft.Report.Dictionary.StiXmlDatabase("Demo",
"demo.xsd", "demo.xml");
```

```
report.loadFile("SimpleList.mrt");
report.dictionary.databases.clear();
report.dictionary.databases.add(xmlDataBase);
```

This way you can easily add the required amount of data sources of different types.

Connect to MySQL, MS SQL and Firebird databases

Since pure JavaScript does not allow access to a remote database, this functionality is implemented through the server-side code. For this, Stimulsoft Reports.JS supports two server-side technologies – PHP and Node.JS.

The database adapter is a layer between DBMS and the client-side script. It connects to the DBMS and receives the necessary data, converting them to the JSON format. The script running on the server (with help of adapter) provides exchange of JSON data between the client-side JavaScript and the server-side using a POST-request.

In order to use this mechanism in the client-side JavaScript must specify the URL of the host that handles POST-requests to the required adapter.

In case, if it is a PHP script it indicates a web page:

```
StiOptions.WebServer.url = "http://mywebsite.com/script.php";
```

If you have adapters for Node.JS, you need to specify the host and port, running the script:

```
StiOptions.WebServer.url = "http://mywebsite.com:9615";
```

If you use the report designer, then this line should be placed before the code to display the designer.

Call the function to render a report template:

```
report.renderAsync();
```

PHP and Node.JS scripts to handle POST-requests to the server and data transfer between the adapter and the client can be downloaded together with the Stimulsoft.Reports.JS assembly.

On the client-side, the connection string in any of the supported DBMS (MySQL, MS SQL or Firebird) must specify the required parameters, including the host, which is not a host that is running the DBMS, but the host on which the POST-requests are processed and the adapter is running:

```
Server=myserver; Database=sakila; uid=root; password=mypass;
```

As an example the following architecture, built on multiple servers:

1) a MySQL server **mysqlserver** that is running the database;
2) a server supporting Node.JS **adapterserver** that is running the script to handle POST-requests (app.js) and adapter's script (in this case, a script MySQLAdapter.js);
3) a HTTP server **httpserver** that is running the http server for distribution to your pages and the JavaScript client.

On the client-side, the next option runs a script that starts the designer:

```
StiOptions.WebServer.url = "http://adapterserver:9615";
```

In the form of creating a new connection the required parameters and the address of the MySQL server are specified:

```
Server=mysqlserver; Database=sakila; uid=root; password=mypass;
```

When all is done, then by pressing the Test button on the form should display "Connection was successful".

Synchronize data between DataStore and Dictionary

In the process of adding data to the Dictionary, there is a need to synchronize the data structure of the Dictionary with the added data source. This problem is easily solved by calling:

```
Stimulsoft.Report.Dictionary.syncronise()
```

The method adds a data structure (Tables and Columns, Relations and whole DataSources) in the dictionary, and solves the problem of the empty data source added to the dictionary using the method:

```
Stimulsoft.Report.regData()
```

If DataSources, Columns or Relations from the DataStore does not exist in Dictionary, then new elements will be added to the Dictionary.

Saving a Rendered Report

Rendered report can be saved into a JSON document for later viewing (data stored within a document):

```
var report = new Stimulsoft.Report.StiReport();
report.loadFile("SimpleList.mrt");
report.render();

var jsonString = report.saveDocumentToJsonString();
```

Report printing

Report printing is implemented by means of the browser, so the view of the dialog box may be different on variuos operating systems and browsers.

Printing is possible after rendering a report by the “print()” method:

```
var report = new Stimulsoft.Report.StiReport();
report.loadFile("SimpleList.mrt");
report.render();

report.print();
```

The optional parameter of Stimulsoft.Report.StiPagesRange allows you to specify the print range.

The parameters of the constructor of the Stimulsoft.Report.StiPagesRange object is:

- The range type (can have values Stimulsoft.Report.StiRangeType.All, Stimulsoft.Report.StiRangeType.Pages, Stimulsoft.Report.StiRangeType.CurrentPage);
- The range in the string representation (page numbers separated by commas may specify the range with a hyphen);
- The current page number.

Now you can flexibly configure report printing:

```
var pageRange = new
Stimulsoft.Report.StiPagesRange(Stimulsoft.Report.StiRangeType.CurrentPage,
"1,3-8", 5);
report.print(pageRange);
```

Running the Report Viewer

Stimulsoft Report Viewer is a tool to view reports that can be easily integrated into your Web application with a few lines of code.

Minimal server support required to start Stimulsoft Reports.JS is any Web server that can provide access to the files of any type.

To get started, you must connect CSS and scripts:

```
<link href="stimulsoft.viewer.office2013.css" rel="stylesheet">
<script src="stimulsoft.reports.js" type="text/javascript"></script>
<script src="stimulsoft.viewer.js" type="text/javascript"></script>
```

Then load the required report and insert it into the HTML page code to deploy Stimulsoft Report Viewer in the current DOM element:

```
<script type="text/javascript">
        var report = new Stimulsoft.Report.StiReport();
        report.loadFile("SimpleList.mrt");

        var viewer = new Stimulsoft.Viewer.StiViewer();
        viewer.report = report;
</script>
```

Stimulsoft Report Viewer object is created by the Stimulsoft.Viewer.StiViewer() constructor, which has three optional parameters.

The first parameter is a set of options Stimulsoft.Viewer.StiViewerOptions, divided into categories, appearance, toolbar and exports. All of them are listed in the table below.

The second parameter is the identifier of the viewer as a DOM object.

The third parameter is the flag of rendering the viewer in the moment of creation. If this flag is set to true, the viewer will be drawn in the same place of the DOM tree, which houses the code. If the flag is set to false, the viewer will be drawn where there is the “renderHtml()” method of the viewer.

For example:

Initialization of the viewer in the head of a page:

```
<script type="text/javascript" >
        var viewer = new Stimulsoft.Viewer.StiViewer(null, "StiViewer",
false);
</script>
```

And rendering of the viewer control in div:

```
<div>Page content</div>
<div>
        <script type="text/javascript">
                // Render the report viewer in this place
                viewer.renderHtml();
        </script>
</div>
```

Another way to display the viewer in a particular control is to use the optional method parameter – renderHtml(elementId), which indicates the ID of the HTML element:

Initialization of the viewer in the head of a page:

```
<script type="text/javascript" >
        var viewer = new Stimulsoft.Viewer.StiViewer(null, "StiViewer",
false);
        viewer.renderHtml("viewerContent");
</script>
```

This will display the viewer in the element:

```
<div id="viewerContent"></div>
```

Note: a viewer can be hidden by specifying the values of the property “visible”:

```
viewer.visible = false;
```

Stimulsoft Report Viewer events

In order to write interactive applications it is needed to respond to changes in the application. The event system which is represented in Stimulsoft Report Viewer can be used for this. The following events exist:

- **onBeginProcessData** is called before requesting the data for the report. The event handler argument “event” is an object with the next fields:
    - **event** – a string ID for the current even., By default it is “BeginProcessData”;
    - **command** – a string ID for the current command. Can have values "TestConnection" and "ExecuteQuery" to call the command to test a connection and data acquisition respectively;
    - **connectionString** – a connection string to the data source for a report;
    - **queryString** – a string with the SQL query to the database for data acquisition. Used only with the command = "ExecuteQuery";
    - **database** – a string name of the current database;
    - **preventDefault** – a flag to prevent further processing of the event. The default value is “false”;

```
viewer.onBeginProcessData = function (event) {
        event.connectionString = "server=real.server; uid=root;
password=actual_password; database=production_db;";
}
```

- **onEndProcessData** is called after retrieving data for the report. The event handler argument “event” is an object with the next fields:
    - **event** – a string ID for the current event. By default it is “EndProcessData”;
    - **result** – a result dataset in a specific JSON format. It has two collections - columns and rows, with descriptions of columns and rows of data sources;
    - **preventDefault** – a flag to prevent further processing of the event. The default value is “false”;

  This event can be used to add its own data sources generated "on the fly".

- **onPrintReport** is called before printing the report. The event handler argument “event” is an object with the next fields:
    - **event** – a string ID for the current event. By default it is “PrintReport”;
    - **fileName** – a printing file name;
    - **report** – a report for saving;
    - **preventDefault** – a flag to prevent further processing of the event. The default value is “true”;

```
viewer.onPrintReport = function (event) {
        console.log("printing");
}
```

- **onBeginExportReport** is called before export but after applying the export options. The event handler argument “event” is an object with the next fields:
    - **event** – a string ID for the current event. By default it is “BeginExportReport”;
    - **fileName** – an exporting file name;
    - **settings** – export report settings;

- **format** – export report format. Can have values Stimulsoft.Report.StiExportFormat.Pdf, Stimulsoft.Report.StiExportFormat.Excel2007, Stimulsoft.Report.StiExportFormat.Word2007, Stimulsoft.Report.StiExportFormat.Html, Stimulsoft.Report.StiExportFormat.Html5, Stimulsoft.Report.StiExportFormat.Document;
- **preventDefault** – a flag to prevent further processing of the event. The default value is "false";

```
viewer.onBeginExportReport = function (event) {
        switch (event.format) {
                case Stimulsoft.Report.StiExportFormat.Html:
                        event.settings.zoom = 2;  // Set zoom to 200%
                        break;
        }
        console.log("exporting");
}
```

- **onEndExportReport** is called after export report. The event handler argument "event" is an object with the next fields:
  - **event** – a string ID for the current event. By default it is "EndExportReport";
  - **fileName** – an exporting file name;
  - **format** – export report format. Can have values Stimulsoft.Report.StiExportFormat.Pdf, Stimulsoft.Report.StiExportFormat.Excel2007, Stimulsoft.Report.StiExportFormat.Word2007, Stimulsoft.Report.StiExportFormat.Html, Stimulsoft.Report.StiExportFormat.Html5, Stimulsoft.Report.StiExportFormat.Document;
  - **data** – export data to the string or byte array representation;
  - **preventDefault** – a flag to prevent further processing of the event. The default value is "false";

- **onEmailReport** is called before sending the report by email. The event handler argument "event" is an object with the next fields:
  - **event** – a string ID for the current event. By default it is "EmailReport";
  - **fileName** – an exporting file name;
  - **settings** – export report settings;
  - **format** – export report format. Can have values Stimulsoft.Report.StiExportFormat.Pdf, Stimulsoft.Report.StiExportFormat.Excel2007, Stimulsoft.Report.StiExportFormat.Word2007, Stimulsoft.Report.StiExportFormat.Html, Stimulsoft.Report.StiExportFormat.Html5, Stimulsoft.Report.StiExportFormat.Document;
  - **data** – export data to the string or byte array representation;
  - **preventDefault** – a flag to prevent further processing of the event. The default value is "false";

- **onDesignReport** is called by pressing the Design button. The event handler argument "event" is an object with the next fields:
  - **event** – a string ID for the current event, by default it is "DesignReport";
  - **fileName** – an exporting file name;
  - **report** – a report for designing;
  - **preventDefault** – a flag to prevent further processing of the event. The default value is "false";

```
viewer.onDesignReport = function (event) {
        console.log("designing");
}
```

Stimulsoft Report Viewer options

The following options are used to customize the appearance and behavior of the Stimulsoft Report Viewer:

Stimulsoft.Viewer.StiViewerOptions:

Option	Description	Default value
width	Specifies the width of the component in “px” or “%”	“100%”
height	Specifies the height of the component in “px” or “%”. Works only with options.appearance.scrollbarsMode = true;	String.empty

Stimulsoft.Viewer.StiViewerOptions.appearance:

Option	Description	Default value
appearance.backgroundColor	Changes the background color of the viewer	Stimulsoft.System.Drawing.Color.white
appearance.pageBorderColor	Sets the border color of report pages	Stimulsoft.System.Drawing.Color.gray
appearance.rightToLeft	Sets the Right to Left mode for viewer controls	false
appearance.fullScreenMode	Sets the full screen mode of the report viewer. If set to true, then the width and height are ignored	false
appearance.scrollbarsMode	Sets the mode of displaying the report viewer without scroll bars or with them	false
appearance.openLinksTarget	Target window to open the link contained in the report	“_self”
appearance.openExportedReportTarget	Target window to open the export file from the viewer	“_blank”
appearance.showTooltips	Displays tips for the viewer controls when hovering the mouse	True
appearance.pageAlignment	Sets the position of the report page in the viewer window. Stimulsoft.Viewer.StiContentAlignment.Left – the page is aligned by the left side of the viewer; Stimulsoft.Viewer.StiContentAlignment.Center – the page is aligned by the center; Stimulsoft.Viewer.StiContentAlignment.Right – the page will be aligned by the right side of the viewer.	Stimulsoft.Viewer.StiContentAlignment.Center
appearance.showPageShadow	Hides the report page shadow	true
appearance.bookmarksPrint	Prints report bookmarks on the printer (in addition to the report)	false

appearance.bookmarksTreeWidth	Sets the width of the bookmark panel in pixels	180
appearance.parametersPanelMaxHeight	Sets the maximum height of the parameter panel in pixels	300
appearance.parametersPanelColumnsCount	Sets the number of columns for showing report parameters	2
appearance.parametersPanelDateFormat	Sets the date format and time for variables of the corresponding type on the parameters panel	String.empty
appearance.interfaceType	Sets the type of the interface viewer. Can take the following values: Stimulsoft.Viewer.StiInterfaceType. Auto – the interface type of the report viewer is selected automatically depending on the device used; Stimulsoft.Viewer.StiInterfaceType. Mouse – forced use the interface to manage viewer using the mouse device; Stimulsoft.Viewer.StiInterfaceType. Touch – force use of the touch interface to manage viewer using the touch screen (mobile devices).	Stimulsoft.Viewer. StiInterfaceType .Auto
appearance.chartRenderType	Sets the type of drawing charts on the report page. It has the following values: Stimulsoft.Viewer.StiChartRenderType.AnimatedVector – charts are drawn in the vector mode with animation; Stimulsoft.Viewer.StiChartRenderType.Vector – charts are drawn as vector images; Stimulsoft.Viewer.StiChartRenderType.Image – charts are drawn as images.	Stimulsoft.Viewer.StiChartRenderType.AnimatedVector

Stimulsoft.Viewer.StiViewerOptions.toolbar:

Option	**Description**	**Default value**
toolbar.visible	Shows/hides the toolbar of the report viewer	true
toolbar.backgroundColor	Changes the toolbar color	Stimulsoft.System.Drawing.Color.empty
toolbar.borderColor	Changes the color of the toolbar borders	Stimulsoft.System.Drawing.Color.empty
toolbar.fontColor	Changes the font color for the toolbar and menu	Stimulsoft.System.Drawing.Color.empty
toolbar.fontFamily	Changes the font family for the toolbar and menu	”Arial”
toolbar.alignment	Sets the position of the toolbar	Stimulsoft.Viewer.StiC

	controls. Stimulsoft.Viewer.StiContentAlignment. Default – the items are placed depending on the value of the RightToLeft option; Stimulsoft.Viewer.StiContentAlignment. Left – the items are aligned by the left side of the toolbar; Stimulsoft.Viewer.StiContentAlignment. Center – the items are aligned by the center of the toolbar; Stimulsoft.Viewer.StiContentAlignment. Right – the items are aligned by the right side of the toolbar.	ontentAlignment.Default
toolbar.showButtonCaptions	Shows/hides the buttons of the viewer toolbar	true
toolbar.showPrintButton	Shows/hides the Print button on the toolbar of the report viewer	true
toolbar.showSaveButton	Shows/hides the Save button on the toolbar of the report viewer	true
toolbar.showSendEmailButton	Shows/hides the Send Email button on the toolbar of the report viewer	false
toolbar.showBookmarksButton	Shows/hides the Bookmarks button on the toolbar of the report viewer. If this button is hidden then the toolbar panel will not be shown even if bookmarks are present in the report.	true
toolbar.showParametersButton	Shows/hides the Parameters button on the toolbar of the report viewer. If this button is hidden then the toolbar panel will not be shown even if bookmarks are present in the report.	true
toolbar.showEditorButton	Shows/hides the Editor button on the report viewer toolbar. If the report does not have editable components, the button will be hidden regardless of the property value.	true
toolbar.showFullScreenButton	Shows/hides the Full Screen button on the toolbar of the report viewer	true
toolbar.showFirstPageButton	Shows/hides the First Page button on the toolbar of the report viewer	true
toolbar.showPreviousPageButton	Shows/hides the Previous Page button on the toolbar of the report viewer	true
toolbar.showCurrentPageControl	Shows/hides the indicator of the current report page	true
toolbar.showNextPageButton	Shows/hides the Next Page button on the toolbar of the report viewer	true
toolbar.showLastPageButton	Shows/hides the Last Page button on the toolbar of the report viewer	true
toolbar.showZoomButton	Shows/hides the button for selecting the report zoom	true

toolbar.showViewModeButton	Shows/hides the button for selecting viewing modes of report pages	true
toolbar.showDesignButton	Shows/hides the Design button on the toolbar of the report viewer	true
toolbar.showAboutButton	Shows/hides the About button on the toolbar of the report viewer	true
toolbar.viewMode	Sets the mode of showing report pages. Stimulsoft.Viewer.StiWebViewMode.OnePage – shows one page of the report selected in the toolbar of the viewer; Stimulsoft.Viewer.StiWebViewMode.WholeReport – displays all report pages at once.	Stimulsoft.Viewer.StiWebViewMode.OnePage
toolbar.zoom()	This method sets zoom of report pages. By default, it is 100 percent. Values vary from 10 to 500 percent. You can use the predefined values: Stimulsoft.Viewer.StiZoomMode.PageWidth – when you run the viewer, the zoom will be set to display the report by the page width. Stimulsoft.Viewer.StiZoomMode.PageHeight – when you run the viewer, the zoom will be set to display the report by the page height.	100
toolbar.menuAnimation	Disables animation when showing/hiding the menu of the report viewer	true
toolbar.showMenuMode	Sets the mode of showing the menu	Stimulsoft.Viewer.StiShowMenuMode.Click

Stimulsoft.Viewer.StiViewerOptions.exports:

Option	Description	Default value
exports.showExportDialog	Shows/hides the export parameters dialog. If the dialog window is disabled then exporting will be processed with the standard settings.	true
exports.showExportToDocument	Shows/hides the menu item Document File	true
exports.showExportToPdf	Shows/hides the menu item Adobe PDF File	false
exports.showExportToHtml	Shows/hides the menu item HMTL File	true
exports.showExportToWord2007	Shows/hides the menu item Microsoft Word 2007/2010 File.	false
exports.showExportToExcel2007	Shows/hides the menu item Microsoft Excel 2007/2010 File.	false

For example, run customized viewer with some options:

```
<script type="text/javascript">
    var report = new Stimulsoft.Report.StiReport();
    report.loadFile("SimpleList.mrt");

    var options = new Stimulsoft.Viewer.StiViewerOptions;
    options.width = "1000px";
    options.height = "1000px";

    options.appearance.scrollbarsMode = true;
    options.appearance.backgroundColor =
Stimulsoft.System.Drawing.Color.dodgerBlue;
    options.appearance.showTooltips = false;

    options.toolbar.showPrintButton = false;
    options.toolbar.showDesignButton = false;
    options.toolbar.showAboutButton = false;

    options.exports.showExportToPdf = true;
    options.exports.ShowExportToWord2007 = true;

    var viewer = new Stimulsoft.Viewer.StiViewer(options);
    viewer.report = report;
</script>
```

Running the Report Designer

Stimulsoft Report Designer is an application for working with reports. It can be easily integrated into any web page (requires any web server).

To get started, you must connect CSS and scripts:

```
<link href="stimulsoft.designer.office2013.css" rel="stylesheet">
<script src="stimulsoft.reports.js" type="text/javascript"></script>
<script src="stimulsoft.designer.js" type="text/javascript"></script>
```

Then load the required report and insert it into the HTML page code to deploy Stimulsoft Report Designer in the current DOM element:

```
var report = new Stimulsoft.Report.StiReport();
report.loadFile("SimpleList.mrt");

var designer = new Stimulsoft.Designer.StiDesigner();
designer.report = report;
```

Stimulsoft Report Designer is an object created by the Stimulsoft.Viewer.StiDesigner() constructor, which has three optional parameters.

The first parameter is a set of Stimulsoft.Designer.StiDesignerOptions divided into categories toolbar, bands, cross bands, components and dictionary. Some of them are listed in the table below.

The second parameter is an identifier of the designer as a DOM object.

The third parameter is a flag of rendering the designer in the moment of creation. If this flag is set to true, the designer will be drawn in the same place of the DOM tree, which houses the code. If the flag is set to false, the designer will be drawn where there is the “renderHtml()” method of the designer.

For example:

Initialization of the designer in the head of a page:

```
<script type="text/javascript">
	var designer = new Stimulsoft.Designer.StiDesigner(null,
"StiDesigner", false);
</script>
```

Rendering of the designer control in div:

```
<div>Page content</div>
<div>
	<script type="text/javascript">
		// Render the report designer in this place
		designer.renderHtml();
	</script>
</div>
```

Another way to display the designer in a particular control is using the optional method parameter renderHtml(elementId), which indicates the ID of the HTML element:

Initialization of the designer in the head of a page:

```
<script type="text/javascript" >
        var designer = new Stimulsoft.Designer.StiDesigner(null,
"StiDesigner", false);
        designer.renderHtml("designerContent");
</script>
```

This will display the designer in the element:

```
<div id="designerContent"></div>
```

Note: a designer can be hidden by specifying the values of the property “visible”:

```
designer.visible = false;
```

Stimulsoft Report Designer events

To write interactive applications need to respond to changes in the application. The event system is used for this. It is represented in Stimulsoft Report Designer with the following events:

- **onBeginProcessData** is called before requesting the data for the report. The event handler argument “event” is an object with the next fields:
    - **event** – a string ID for the current event, by default is “BeginProcessData”;
    - **command** – a string ID for the current command. Can have values "TestConnection" and "ExecuteQuery" to call the command for test connection and data acquisition respectively;
    - **connectionString** – a connection string to the data source for the report;
    - **queryString** – a string with the SQL query to the database for data acquisition, used only with the command = "ExecuteQuery";
    - **database** – a string name of the current database;
    - **preventDefault** – a flag to prevent further processing of the event. The default value is “false”;

```
designer.onBeginProcessData = function (event) {
        event.connectionString = "server=real.server; uid=root;
password=actual_password; database=production_db;";
}
```

- **onEndProcessData** is called after retrieving the data for the report. The event handler argument “event” is an object with the next fields:
    - **event** – a string ID for the current event. By default it is “EndProcessData”;
    - **result** – a result dataset in a specific JSON format. It has two collections: columns and rows, with descriptions of columns and rows of data sources;
    - **preventDefault** – a flag to prevent the further processing of the event. The default value is “false”;

  This event can be used to add its own data sources generated "on the fly" to the report.

- **onSaveReport** is called before saving the report. The event handler argument “event” is an object with the next fields:
    - **event** – a string ID for the current event. By default it is “`SaveReport`”;
    - **fileName** – a saving file name;
    - **report** – a report for saving;
    - **preventDefault** – a flag to prevent further processing of the event. The default value is “true”;

```
designer.onSaveReport = function (event) {
        var jsonStr = event.report.saveToJsonString();
        console.log("saving a report");
}
```

- **onOpenReport** is called before user click button for opening a report. The event handler argument “event” is an object with the next fields:
    - **event** – a string ID for current event. By default it is “OpenReport”;
    - **preventDefault** – a flag to prevent further processing of the event. The default value is “true”;

- **onCreateReport** is called after a new report is created. The event handler argument “event” it is an object with the next fields:
  - **event** – a string ID for the current event. By default it is “`CreateReport`”;
  - **report** – report for saving;

```
Designer.onCreateReport = function (event) {
        var dataSet = new Stimulsoft.System.Data.DataSet("Demo");
        dataSet.readJsonFile("/reports/Demo.json");
        event.report.regData("Demo", "Demo", dataSet);
        console.log("creating a new report");
}
```

- **onPreviewReport** is called before the launch of the preview when requesting data for the report. If the data source is set in the report template, then calling “report.regData()” with the same name as the data source has a lower priority, and data will be taken from the source specified in the report template. It is recommended to clean the collection of data sources by “report.dictionary. databases.clear()”. The event handler argument “event” is an object with the next fields:
  - **event** – a string ID for the current event. By default it is “`PreviewReport`”;
  - **fileName** – a report file name;
  - **report** – report for preview;
  - **preventDefault** – a flag to prevent further processing of the event. The default value is “false”;

```
designer.onPreviewReport = function (event) {
        switch (event.format) {
                case Stimulsoft.Report.StiExportFormat.Html:
                        event.settings.zoom = 2;  // Set zoom to 200%
                        break;
        }
        console.log("running the preview");
}
```

- **onExit** is called before closing the designer. The event handler argument “event” is an object with the next fields:
  - **event** – a string ID for the current event. By default it is “Exit”;

```
designer.onExit = function (event) {
        console.log("exiting");
}
```

Stimulsoft Report Designer options

The following options are used to customize the appearance and behavior of Stimulsoft Report Designer:

Stimulsoft.Designer.StiDesignerOptions:

Option	Description	Default value
Width	Specifies the width of the component in “px” or “%”	”100%”
Height	Specifies the height of the component in “px” or “%”	”800px”

Stimulsoft.Designer.StiDesignerOptions.appearance:

Option	Description	Default value
defaultUnit	A default value of unit in the designer. It has the following values: Stimulsoft.Report.StiReportUnitType.Centimeters, Stimulsoft.Report.StiReportUnitType. HundredthsOfInch, Stimulsoft.Report.StiReportUnitType. Inches, Stimulsoft.Report.StiReportUnitType. Millimeters	Stimulsoft.Report.StiReportUnitType. Centimeters
interfaceType	The type of the designer interface. It has the following values: Stimulsoft.Designer. StiInterfaceType.Auto, Stimulsoft.Designer. StiInterfaceType.Mouse, Stimulsoft.Designer. StiInterfaceType.Touch	Stimulsoft.Designer. StiInterfaceType.Auto
showAnimation	Show menu animation for designer	true
showSaveDialog	Displays the dialog to enter the name of the report when it is saved	true
showTooltips	Displays tips for the designer controls when hovering the mouse	true
showTooltipsHelp	Displays a link to the online documentation on the tips	true
appearance.fullScreenMode	Sets the full screen mode of the report designer. If set to true then width and height are ignored	false

Stimulsoft.Designer.StiDesignerOptions.toolbar:

Option	Description	Default value
showPageButton	Shows/hides the Page button in the toolbar of the designer	true
showPreviewButton	Shows/hides the Preview button in the toolbar of the designer	true
showSaveButton	Shows/hides the Save button in the toolbar of the designer	true
showAboutButton	Shows/hides the About button in the toolbar of the designer	true
showFileMenu	Shows/hides the file menu of the designer	true
showFileMenuNew	Shows/hides the New item in the file menu of the designer	true
showFileMenuOpen	Shows/hides the Open item in the file menu of the	true

	designer	
showFileMenuSave	Shows/hides the Save item in the file menu of the designer	true
showFileMenuClose	Shows/hides the Close item in the file menu of the designer	true
showFileMenuExit	Shows/hides the Exit item in the file menu of the designer	true
showFileMenuReportSetup	Shows/hides the Report Setup item in the file menu of the designer	true
showFileMenuOptions	Shows/hides the Options item in the file menu of the designer	true

Stimulsoft.Designer.StiDesignerOptions.bands:

Option	Description	Default value
showReportTitleBand	Shows/hides the ReportTitleBand item in the bands menu of the designer	true
showReportSummaryBand	Shows/hides the ReportSummaryBand item in the bands menu of the designer	true
showPageHeaderBand	Shows/hides the PageHeaderBand item in the bands menu of the designer	true
showPageFooterBand	Shows/hides the PageFooterBand item in the bands menu of the designer	true
showGroupHeaderBand	Shows/hides the GroupHeaderBand item in the bands menu of the designer	true
showHeaderBand	Shows/hides the HeaderBand item in the bands menu of the designer	true
showFooterBand	Shows/hides the FooterBand item in the bands menu of the designer	true
showColumnHeaderBand	Shows/hides the ColumnHeaderBand item in the bands menu of the designer	true
showColumnFooterBand	Shows/hides the ColumnFooterBand item in the bands menu of the designer	true
showDataBand	Shows/hides the DataBand item in the bands menu of the designer	true
showHierarchicalBand	Shows/hides the HierarchicalBand item in the bands menu of the designer	true
showChildBand	Shows/hides the ChildBand item in the bands menu of the designer	true
showEmptyBand	Shows/hides the EmptyBand item in the bands menu of the designer	true
showOverlayBand	Shows/hides the OverlayBand item in the bands menu of the designer	true
showTable	Shows/hides the Table item in the bands menu of the designer	true

Stimulsoft.Designer.StiDesignerOptions.crossBands:

Option	Description	Default value
showCrossTab	Shows/hides the CrossTab item in the cross bands menu of the designer	false
showCrossGroupHeaderBand	Shows/hides the CrossGroupHeaderBand item in the	false

	cross bands menu of the designer	
showCrossGroupFooterBand	Shows/hides the CrossGroupFooterBand item in the cross bands menu of the designer	false
showCrossHeaderBand	Shows/hides the CrossHeaderBand item in the cross bands menu of the designer	false
showCrossFooterBand	Shows/hides the CrossFooterBand item in the cross bands menu of the designer	false
showCrossDataBand	Shows/hides the CrossDataBand item in the cross bands menu of the designer	false

Stimulsoft.Designer.StiDesignerOptions.components:

Option	Description	Default value
showText	Shows/hides the Text item in the components menu of the designer	true
showTextInCells	Shows/hides the TextInCells item in the components menu of the designer	false
showRichText	Shows/hides the RichText item in the components menu of the designer	false
showImage	Shows/hides the Image item in the components menu of the designer	true
showBarCode	Shows/hides the BarCode item in the components menu of the designer	false
showShape	Shows/hides the Shape item in the components menu of the designer	true
showPanel	Shows/hides the Panel item in the components menu of the designer	true
showClone	Shows/hides the Clone item in the components menu of the designer	false
showCheckBox	Shows/hides the CheckBox item in the components menu of the designer	true
showSubReport	Shows/hides the SubReport item in the components menu of the designer	true
showZipCode	Shows/hides the ZipCode item in the components menu of the designer	false
showChart	Shows/hides the Chart item in the components menu of the designer	true

Stimulsoft.Designer.StiDesignerOptions.`dictionary`:

Almost all options described below can take the following values:

- Stimulsoft.Designer.StiDesignerPermissions.None - deny all;
- Stimulsoft.Designer.StiDesignerPermissions.Create – allows creating an item;
- Stimulsoft.Designer.StiDesignerPermissions.Delete – allows deleting an item;
- Stimulsoft.Designer.StiDesignerPermissions.Modify – allows modifying an item;
- Stimulsoft.Designer.StiDesignerPermissions.View – allows viewing an item;
- Stimulsoft.Designer.StiDesignerPermissions.ModifyView – allows modifying and viewing an item (Modify + View);
- Stimulsoft.Designer.StiDesignerPermissions.All – allows any action with the item (Create + Delete + Modify + View).

Option	**Description**	**Default value**
`showDictionary`	Shows/hides the dictionary in the designer	true
`dataSourcesPermissions`	A value of permissions for data sources in the designer	Stimulsoft.Designer.StiDesignerPermissions.All
`dataConnectionsPermissions`	A value of connections for data sources in the designer	Stimulsoft.Designer.StiDesignerPermissions.All
`dataColumnsPermissions`	A value of connections for columns in the designer	Stimulsoft.Designer.StiDesignerPermissions.All
`dataRelationsPermissions`	A value of connections for relations in the designer	Stimulsoft.Designer.StiDesignerPermissions.All
`businessObjectsPermissions`	A value of connections for business objects in the designer	Stimulsoft.Designer.StiDesignerPermissions.All
`variablesPermissions`	A value of connections for variables in the designer	Stimulsoft.Designer.StiDesignerPermissions.All

For example, run the designer on the full screen of a browser:

```
<script type="text/javascript">
    var report = new Stimulsoft.Report.StiReport();
    report.loadFile("SimpleList.mrt");

    var options = new Stimulsoft.Designer.StiDesignerOptions;

    options.appearance.fullScreenMode = true;

    var designer = new Stimulsoft.Designer.StiDesigner(options);
    designer.report = report;
</script>
```

Exporting a Rendered Report

A rendered report can be exported to several formats.

This example demonstrates the export to HTML:

```
var report = new Stimulsoft.Report.StiReport();
report.loadFile("SimpleList.mrt");
report.render();

var settings = new Stimulsoft.Report.Export.StiHtmlExportSettings();
var service = new Stimulsoft.Report.Export.StiHtmlExportService();

var textWriter = new Stimulsoft.System.IO.TextWriter();
var htmlTextWriter = new
Stimulsoft.Report.Export.StiHtmlTextWriter(textWriter);

service.exportTo(report, htmlTextWriter, settings);

var resultHtml = textWriter.getStringBuilder().toString();
```

It uses one of the export services, in this case it is Stimulsoft.Report.Export.StiHtmlExportService. The "exportTo()" method produces output in one of the parameters (htmlTextWriter) the flow of the report in HTML. After executing the code, "resultHtml" contains a string with the contents of the HTML report.

Options for exporting to HTML are defined in the Stimulsoft.Report.Export.StiHtmlExportSettings object and passed as a parameter to the "exportTo()" method of the Stimulsoft.Report.Export.StiHtmlExportService object. These options allow you to customize the look and the structure of the HTML document.

Table of properties of Stimulsoft.Report.Export.StiHtmlExportSettings:

Option	Description	Default value
htmlType	A type of the report file to be converted into. Can have the following values Stimulsoft.Report.StiHtmlType.Html, Stimulsoft.Report.StiHtmlType.Html5, Stimulsoft.Report.StiHtmlType.Mht	Stimulsoft.Report.StiHtmlType.Html
imageQuality	Quality of images which will be exported to the result file	0.75
imageResolution	Resolution of images in DPI which will be exported to the result file	100
imageFormat	Sets an image export format. Can have the following values Stimulsoft.Report.ImageFormat.Bmp, Stimulsoft.Report.ImageFormat.Emf, Stimulsoft.Report.ImageFormat.Exif, Stimulsoft.Report.ImageFormat.Gif, Stimulsoft.Report.ImageFormat.Guid, Stimulsoft.Report.ImageFormat.Icon, Stimulsoft.Report.ImageFormat.Jpeg, Stimulsoft.Report.ImageFormat.MemoryBmp, Stimulsoft.Report.ImageFormat.Png, Stimulsoft.Report.ImageFormat.Tiff,	null

	Stimulsoft.Report.ImageFormat.Wmf	
encoding	File encoding. Can have the following values Stimulsoft.System.Text.Encoding.ASCII, Stimulsoft.System.Text.Encoding.BigEndianUnicode, Stimulsoft.System.Text.Encoding.Default (set to Stimulsoft.System.Text.Encoding.Unicode), Stimulsoft.System.Text.Encoding.Unicode, Stimulsoft.System.Text.Encoding.UTF32, Stimulsoft.System.Text.Encoding.UTF7, Stimulsoft.System.Text.Encoding.UTF8	System.Text.Encoding.Default
zoom	Zoom factor. By default the value is set to 1.0 that is equal 100% in the export settings dialog	1
exportMode	Sets the mode of the document export using the div, span or table element. Can have the following values Stimulsoft.Report.Export.StiHtmlExportMode.Span, Stimulsoft.Report.Export.StiHtmlExportMode.Div, Stimulsoft.Report.Export.StiHtmlExportMode.Table	Stimulsoft.Report.Export.StiHtmlExportMode.Table
exportQuality	Export quality of the component size. Can have the following values Stimulsoft.Report.Export.StiHtmlExportQuality.High and Stimulsoft.Report.Export.StiHtmlExportQuality.Low	Stimulsoft.Report.Export.StiHtmlExportQuality.High
addPageBreaks	Adds page breaks	true
bookmarksTreeWidth	Bookmark column width, in pixels	150
exportBookmarksMode	An exporting mode of a document with bookmarks. Can have the following values Stimulsoft.Report.Export.StiHtmlExportBookmarksMode.BookmarksOnly, Stimulsoft.Report.Export.StiHtmlExportBookmarksMode.ReportOnly, Stimulsoft.Report.Export.StiHtmlExportBookmarksMode.All	Stimulsoft.Report.Export.StiHtmlExportBookmarksMode.All
useStylesTable	Uses the Styles Table; if false then the style table is empty and all properties of each component will be described directly in the style of this component	true
removeEmptySpaceAtBottom	Indicates that table styles will be used in the result HTML file. The HTML5 and MHT export mode is not supported.	true
pageHorAlignment	The horizontal alignment of pages. The HTML5 and MHT export mode is not supported. Can have the following values Stimulsoft.Base.Drawing.StiHorAlignment.Left, Stimulsoft.Base.Drawing.StiHorAlignment.Center, Stimulsoft.Base.Drawing.StiHorAlignment.Right	Stimulsoft.Base.Drawing.StiHorAlignment.Left
compressToArchive	Provides the ability (when exporting to HTML) to get the zip file after conversion. If this flag set to is true, the report processing occurs first, and then all the files and folders will be packed in a zip archive	false

useEmbeddedImages	Provides the ability to embed images directly into the HTML file. In this case, it is necessary to consider that the correct displaying of this file depends on the browser being used. Not all browsers support the option to view the HTML file with embedded pictures	false
continuousPages	Indicates that all report pages will be shown as vertical ribbon. The HTML and MHT export mode is not supported	true
chartType	A type of the chart in the HTML exports. Can have the following values Stimulsoft.Report.Export.StiHtmlChartType.Image, Stimulsoft.Report.Export.StiHtmlChartType.Vector, Stimulsoft.Report.Export.StiHtmlChartType.AnimatedVector	Stimulsoft.Report.Export.StiHtmlChartType.AnimatedVector
openLinksTarget	A target (string) to open links from the exported report	null

This example demonstrates the export to PDF:

```
var report = new Stimulsoft.Report.StiReport();
report.loadFile("SimpleList.mrt");
report.render();

var settings = new Stimulsoft.Report.Export.StiPdfExportSettings();
var service = new Stimulsoft.Report.Export.StiPdfExportService();

var stream = new Stimulsoft.System.IO.MemoryStream();

service.exportTo(report, stream, settings);

var data = stream.toArray();

Object.saveAs(data, "SimpleList.pdf", "application/pdf");
```

It uses the export service Stimulsoft.Report.Export.StiPdfExportService. The “exportTo()” method produces output in one of the parameters (stream) the flow of the report in PDF. After executing the code, “data” contains a byte array with the contents of the PDF report.

Options for exporting to PDF are defined in the Stimulsoft.Report.Export.StiPdfExportService object and passed as a parameter to the “exportTo()” method of the Stimulsoft.Report.Export.StiPdfExportService object. These options allow you to customize the look and the structure of the PDF document.

Table of properties of Stimulsoft.Report.Export.StiPdfExportService:

Option	**Description**	**Default value**
imageQuality	Quality of images which will be exported to the result file	0.75
imageResolution	Resolution of images in DPI which will be exported to the result file	100
imageResolutionMode	Image resolution mode. Can have the following values Stimulsoft.Report.Export.StiImageResolutionMode.Exactly, Stimulsoft.Report.Export.StiImageResolutionMode.NoMoreThan,	Stimulsoft.Report.Export.StiImageResolutionMode.Exactly

	Stimulsoft.Report.Export.StiImageResolutionMode.Auto	
embeddedFonts	Indicates that fonts which used in report will be included in PDF file	false
standardPdfFonts	Indicates that only standard PDF fonts will be used in result PDF file	false
compressed	Indicates that result file will be used compression	false
useUnicode	Indicates that Unicode symbols must be used in result PDF file	false