Documentation

the highcharts java api

OVERVIEW

The HyJavaCharts Java library is a pure Java wrapper for the Highcharts Javascript library.
It allows your Java-based web applications to configure Highcharts charts using only Java methods. There is no need to write any Javascript.
The library will generate the required Javascript code that Highcharts needs to render the charts.

The Highchart class is the starting point for all charts.

To define the contents of a chart, you simply have to instantiate the Highchart class and use its setter methods to define the chart options and chart data as required. For details on the effects of each chart option refer to the Highcharts API reference.
The naming of the chart options in the HyJavaCharts library in most cases exactly follows the Highcharts Javascript API.

In the example below we are defining a bar chart.
Instantiate the Highchart class and then get a reference to ChartOptions. You then set the chart options as required.
The Java getter/setter naming convention for method names is also supported.

```
public class YourBarChart {

 public Highchart configure() {
  Highchart highChart = new Highchart();
  ChartOptions chartOptions = highChart.getChartOptions();

  chartOptions.chart().type(ChartType.BAR);
  chartOptions.title().text("Historic World Population by Region");
  chartOptions.subtitle().text("Source: Wikipedia.org");

  set all your required chart options......

  return highChart;
 }

}
```

JAVADOC API

In most cases the HyJavaCharts API exactly matches the Highcharts API. There is no need to learn another API. The HyJavaCharts API can be viewed online or refer to the API jar provided in the product download.

DEMO WEB APPLICATION

To give you a kick start using the library, the demo web application provides thorough examples of more than 80 different charts including Java source code for each example. Most of the Highcharts Demos from the Highcharts Demos page are included in this application.
The Java source code for the demo application including all chart examples is available from the Other Downloads page.

HOW TO USE THE LIBRARY

Java

Include the HyJavaCharts jar in your project.

Configure your chart and generate the Javascript code for the global and chart options.

```
Highchart hc = new YourBarChart().configure();
String globalOptionsJs = hc.globalOptionsToJs();
String chartOptionsJs = hc.chartOptionsToJs();
```

The YourBarChart class will look something like this:
(the Java getter/setter naming convention for method names is also supported)

```
public class YourBarChart {

 public Highchart configure() {
  Highchart highChart = new Highchart();
  ChartOptions chartOptions = highChart.getChartOptions();

  chartOptions.chart().type(ChartType.BAR);
  chartOptions.title().text("Historic World Population by Region");
  chartOptions.subtitle().text("Source: Wikipedia.org");

  set all your required chart options......

  return highChart;
 }

}
```

Javascript

Include required Javascript on your page as per Highcharts documentation.

```
Highcharts.setOptions(${globalOptionsJs});
Highcharts.chart('container', ${chartOptionsJs});
```

Your Java code will pass the globalOptionsJs and chartOptionsJs strings to the markup variables ${globalOptionsJs} and ${chartOptionsJs}.
The method used to pass the generated chart options from the java to the web page will vary depending on your java web framework. *(Refer to Java Web Frameworks below).*

Web Page

Your markup will look something like this:

```
<script src=https://code.highcharts.com/highcharts.js></script>

<div id="container" style="min-width: 310px; max-width: 800px; height: 400px; margin: 0 auto"></div>

<script>
Highcharts.setOptions(${globalOptionsJs});
Highcharts.chart('container', ${chartOptionsJs});
</script>
```

That is all that is required to get a working Bar chart example.

MAIN LIBRARY CLASSES

GlobalOptions Class

The GlobalOptions class maps directly to the Highcharts global options in the Highcharts API.
All Highchart global options will be configured using setter methods on this class.

ChartOptions Class

The ChartOptions class maps directly to the Highcharts chart options in the Highcharts API.
All Highchart chart options will be configured using setter methods on this class.

Highchart Class

The Highchart class wraps both the GlobalOptions and ChartOptions classes into one convenient class to make usage as simple as possible.
This class handles chart theming; the setting of global and chart options; generates Javascript for GlobalOptions and ChartOptions.
All Highchart global options can be configured by getting a reference to GlobalOptions from this class via getGlobalOptions().
All Highchart chart options can be configured by getting a reference to ChartOptions from this class via getChartOptions().
Usage of the Highchart class has been described in examples above.

HighchartOptions Class

The HighchartOptions class handles chart theming and generates Javascript for GlobalOptions and ChartOptions for the case where you choose to not use the Highchart class.

As an alternative to using the Highchart class, global and chart options can be configured separately by using the GlobalOptions and ChartOptions classes outside of the Highchart class. Global options can be configured once and will apply to multiple charts. Global options are optional.
Usage is described below.

Configure your chart and get the generated Javascript code for the global and chart options.

```
GlobalOptions globalOptions = new GlobalOptions();
globalOptions.global().useUTC(false);
set all your required global options......

ChartOptions barChartOptions = new YourBarChart().configure();

String globalOptionsJs = new HighchartOptions().toJs(globalOptions);
String chartOptionsJs = new HighchartOptions().toJs(barChartOptions);
```

The YourBarChart class will look something like this:

```
public class YourBarChart {

 public ChartOptions configure() {
  ChartOptions chartOptions = new ChartOptions();

  chartOptions.chart().type(ChartType.BAR);
  chartOptions.title().text("Historic World Population by Region");
  chartOptions.subtitle().text("Source: Wikipedia.org");

  set all your required chart options......

  return chartOptions;
 }

}
```

HighchartRenderer Class

The HyJavaCharts library performance is extremely fast right out of the box.
In cases where thousands of charts are generated, for example batch processes, some improvements can be found using HighchartRenderer.toJs(...) rather than the Highcharts.chartOptionsToJs() and Highcharts.globalOptionsToJs().
Instantiate the HighchartRenderer class once and then use it to generate the chart options Javascript for multiple charts.
Note that there are no improvements when generating single charts.

JAVA WEB FRAMEWORKS

The HyJavaCharts library is a pure Java library so can be used by any Java web framework.

As long as your framework can add the generated Javascript chart options to your markup at page build time then you can use this library.
Most (if not all) Java web frameworks will offer some method of adding javascript to a page.

We have included several example applications showing usage of the HyJavaCharts library with various frameworks. These include:-

- a Spring MVC demo web application;
- a JSF demo web application;
- a Struts2 demo web application;
- a GWT demo web application;
- a basic servlet demo web application;
- a JSP servlet demo web application;
- a Wicket demo web application;
- a Vaadin demo web application;
- a JavaFX demo application.

The Java source code for these example applications is available from the Other Downloads page.

JAVA CHART OPTION METHOD NAMES

You have the choice to use method names to set each chart option that closely match the Highcharts Javascript API or use the method names that follow the Java getter/setter naming convention. Its totally up to you to use the naming convention you are most comfortable with.

```
Javascript:
chart: { type: 'bar' }

Java:
chartOptions.chart().type(ChartType.BAR);
 OR
chartOptions.getChart().setType(ChartType.BAR);
```

OPTIONS WITH MULTIPLE DATA TYPES

Where a chart option has more than one data type, eg. Number or String the Java API will include one method per type.

```
chartOptions.pane().innerRadiusAsNumber(25);
chartOptions.pane().innerRadiusAsString("50%");
```

OPTION DATA TYPES

AnimationOptions Class

Maps to the Highcharts.AnimationOptionsObject data type.
Set the animation duration and easing options.

Color Class

Maps to the Highcharts Color or Highcharts.ColorString data type.
To set a color string value use: new Color("#55BF3B"); or new Color().setColor("#55BF3B");
To set a javascript color value use: new Color().setColorValue("jsvalue");
To set a RGB color value use: new Color(255, 255, 255); or new Color().setRGB(255, 255, 255);
To set a RGBA color value use: new Color(255, 255, 255, 0.5); or new Color().setRGBA(255, 255, 255, 0.5);
To set a linear or radial gradient color value use: new Color(gradient, stops); or new Color().setGradient(gradient, stops);
To brighten a color use: new Color("#55BF3B").brighten(0.2); or new Color(255, 255, 255).brighten(0.2); or new Color(255, 255, 255, 0.5).brighten(0.2);

140 Web color code strings are also defined in this class. Eg. new Color(Color.ALICEBLUE).

CSSObject Class

Maps to the Highcharts CSSObject or Highcharts.CSSObject data type.
To set a CSS javascript value use: new CSSObject("jsvalue"); or new CSSObject().setCssValue("jsvalue");
Set the backgroundColor, borderRadius, color, cursor, fontFamily, fontSize, fontStyle, fontWeight, lineWidth, opacity, pointerEvents, position, textAlign, textOutline, textOverflow, whiteSpace options.

Function Class

Maps to the Highcharts Function data type.
To set the body of a Function use: new Function("functionbodystring"); or new Function().setFunctionBody("functionbodystring");
To add parameters to the function use: addParameter("parameter");

General Class

To set a javascript value use: new General("jsvalue"); or new General().setJsValue("jsValue");

ShadowOptions Class

Maps to the Highcharts.ShadowOptionsObject data type.
Set the shadow color, offsetX, offsetY, opacity and width options.

SVGAttributes Class

Maps to the Highcharts.SVGAttributes data type.
Set the halo fill, stroke and stroke-width options.

ARRAYS

There are methods for each chart option array to help simplify usage where there is only 1 element in the array.
In the example the xAxisSingle() method will add a new XAxis object to the xAxis array and operate on that object.

```
XAxis xAxis = new XAxis();
xAxis.allowDecimals(false);
chartOptions.xAxis().add(xAxis);
 OR
chartOptions.xAxisSingle().allowDecimals(false);
```

THEMES

Include your chosen theme.js on your page as per Highcharts documentation.
If your Java code uses the chartOptions.colors() or chartOptions.getColors() methods you will also need to set the Theme in your Java prior to calling those methods.
eg. *highChart.setTheme(HighchartTheme.DARK_UNICA);*
All 10 standard Highcharts themes have been created in HighchartTheme.

If you have created your own theme you will need to create a Theme class as follows:
(1) create a new theme class implementing Theme.
(2) add the method getColors() to the class returning your theme colors.

```
public class YourNewTheme implements Theme {
  public List<Color> getColors() {
   return new ArrayList(Arrays.asList(
     new Color(Color.LIGHTSKYBLUE), new Color(Color.LIGHTGREEN), new Color(Color.LIGHTSALMON),
     new Color(Color.LIGHTCORAL), new Color(Color.LIGHTSTEELBLUE), new Color(Color.LIGHTGRAY),
     new Color(Color.LIGHTPINK), new Color(Color.LIGHTSEAGREEN), new Color(Color.LIGHTCYAN),
     new Color(Color.LIGHTBLUE), new Color(Color.LIGHTYELLOW), new Color(Color.LIGHTSLATEGRAY)
     ));
   }
 }
```

You can also use color strings ie. new Color("#87cefa").
To use the theme: *highChart.setTheme(new YourNewTheme());*

If you are using the ChartOptions class then set your theme with:
new HighchartOptions().setTheme(HighchartTheme.DARK_UNICA, chartOptions);
or
new HighchartOptions().setTheme(new YourNewTheme(), chartOptions);

CHART DATA SOURCES

Chart data sources can be databases, web services, files or any resource your application server has access to.
The Highcharts Data module allows access to CSV, HTML tables or remote CSV data by specifying a URL.

JAVA VERSION

The HyJavaCharts library has been compiled with Java 1.8. There are no external dependencies.

API ISSUES

Every endeavor has been made to ensure the HyJavaCharts Java API matches the Highcharts javascript API.
Please contact us if there is a chart option missing from the Java API or if an option data type does not match the Highcharts API.
We will publish an updated API as soon as possible.

Generic Chart Options

To allow the developer to continue until the Java API is updated, generic options that can have any property name and value are available on every Highcharts model class.
eg. If an option “allowAnimation” Boolean was missing from Chart in the API or the data type is incorrect you can still include the option as shown below.

```
chartOptions.chart().get_genericOption()
    .add(new GenericOption("allowAnimation", "true"));
or
chartOptions.chart().get_genericOption()
    .add(new GenericOption()
        .setPropertyName("allowAnimation")
        .setPropertyValue("true"));
```

SUPPORT

For any questions or issues on the HyJavaCharts API please contact our support team via the “Contact Us” area on the home page.
Please refer to the Highcharts API for usage of each chart option or contact Highcharts for support of the Highcharts product.