**Simple Dashboard**



[kannankeril](https://www.codeproject.com/script/Membership/View.aspx?mid=839621)

6 Jul 2013[CPOL](http://www.codeproject.com/info/cpol10.aspx" \o "The Code Project Open License (CPOL))

Simple dashboard utilizing an elegant HTML+CSS+JavaScript template from TechGYO and the Highcharts library.

* [Download HighchartsWebFinal-noexe.zip - 901.7 KB](https://www.codeproject.com/KB/web-image/616156/HighchartsWebFinal-noexe.zip)
* [Download HighchartsWebFinal.zip - 1.1 MB](https://www.codeproject.com/KB/web-image/616156/HighchartsWebFinal.zip)
* [Download Simple Dashboard (VS2010 version)](https://www.codeproject.com/KB/web-image/616156/Simple_Dashboard__VS2010_.zip)

**Introduction**

A dashboard is an easy to read, single-page graphical presentation of current status within an organization. With the increasing popularity of BI solutions and web applications as a delivery platform, there has been an increasing demand for easy to implement web portals that operate as functional dashboards.

A typical dashboard can be separated out into three functional layers, the dashboard UI layer, charting components or tabular representations of data and a data retrieval layer. The UI layer comprises the overall look & feel of the dashboard. It determines the structure of a dashboard in terms of header, footer and menu system components. It provides a subsystem of panels that can house charting components or tabular summaries of organizational data. The data retrieval layer is not unique to dashboards, it is an essential part of most non-trivial web applications and as is treated as out-of-scope for this article. We will focus on implementing the UI layer and a charting system that can be replicated for various types of dashboards.

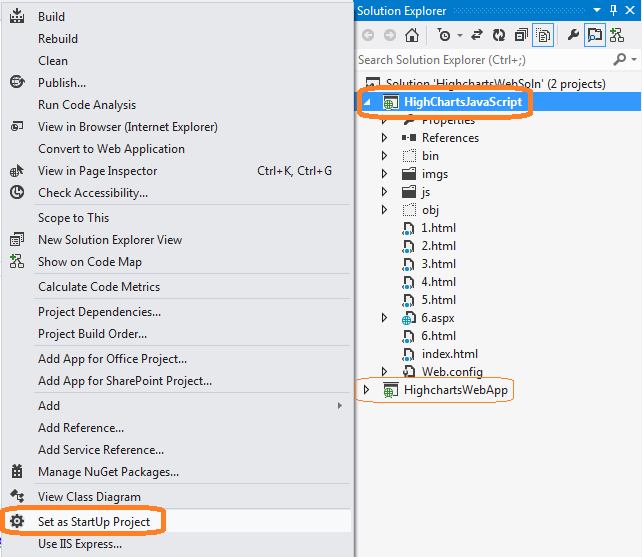
The attached solution contains two projects. One project (HighchartsWebApp) demonstrates the use of the ASP.NET control to create the dashboard. It is a minimalistic demo for those of us desiring a mostly C# based dashboard solution. The second and primary project (HighChartsJavaScript) uses JavaScript to fully leverage capabilities of the charting component (Highcharts) that I selected for this demo. I have implemented both projects to provide alternate paths to our goal of creating a generic dashboard.

The UI layer as implemented is a combination of JavaScript, CSS and HTML that was designed by the people at [TechGYO](http://techgyo.com/index.php/free-metro-ui-html-template-by-techgyo/" \o "Metro UI page at TechGYO). I loved the simplicity of the design and decided to leverage it to create this Dashboard. My goal was to provide an example that could be used as a starting point. This project is targeted to beginners and has not been stress-tested / live-tested at the time of publication. I might use the feedback in the comments to provide future updates as time permits.

Please [download the attached file (](https://www.codeproject.com/KB/web-image/616156/HighchartsWebFinal.zip" \o "Download zip file containing demo code)*[HighchartsWebFinal.zip](https://www.codeproject.com/KB/web-image/616156/HighchartsWebFinal.zip" \o "Download zip file containing demo code)*[)](https://www.codeproject.com/KB/web-image/616156/HighchartsWebFinal.zip" \o "Download zip file containing demo code), unzip and open the Visual Studio 2012 solution so you can refer to the code while reading the rest of this article.

You can activate the JavaScript version by right-clicking the project ‘HighChartsJavaScript’ within Solution Explorer and selecting ‘**Set As StartUp Project**’. This project implements both the UI and charting components using JavaScript.

You can alternately run the second project by right-clicking the ‘HigchartsWebApp‘ project within Solution Explorer and selecting ‘**Set As StartUp Project**’. This project differs in that the charting component is entirely wrapped within an ASP.NET control and all manipulation of the charts can be done using C# with no reliance on JavaScript. Once you have made your choice, press ‘F5’ to activate the demo.



I have put in less effort into the HigchartsWebApp project as I expect future development to concentrate on developing the JavaScript version of the charting code. This version does use some C# code to retrieve and populate data but the charting controls are defined & executed entirely in JavaScript. If you would like to get more information on using a purely server-side implementation like the ‘HighchartsWebApp’ project, you can check the website for [Highcharts.Net](http://highcharts.codeplex.com/" \o "Highcharts.Net at Codeplex.com), an open source project (<http://highcharts.codeplex.com/>).

**Dashboard - Operation**



The screenshot above shows an annotated view of the initial dashboard screen. Note, the last (green background) section has been intentionally left empty. Clicking on this section brings up the equivalent of a drill down/popup report (*see next screenshot*).



Note, the header changes to reflect a title for the drill down report. The Metro UI themed arrow appears to allow navigation back to the main dashboard.

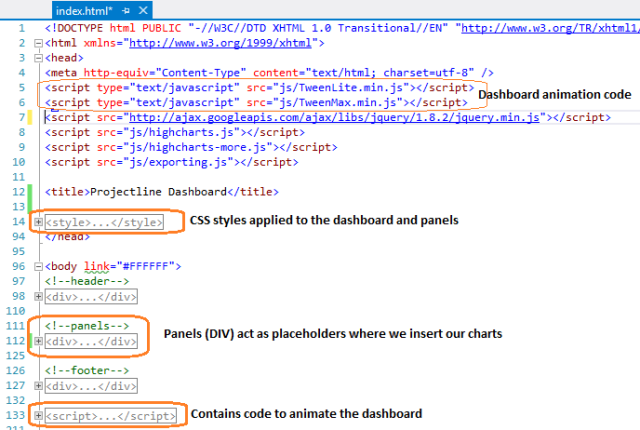
Assuming you have the solution open in VS2012, the above information should be sufficient to guide most of you in getting the project to execute, you could even start modifying it for your individual requirements. For those wanting a deeper dive, let us now look into the code so we can get a clearer vision of its operation.

As mentioned earlier, the attached solution contains two projects, the discussion from this point on focuses on the JavaScript version (‘HighChartsJavaScript’). Our objective is to utilize a charting library like Highcharts to create a dashboard. Additionally, we would like to determine how we can dynamically integrate our data into the charts without having to hard-code it as part of the JavaScript code. Data retrieval in most cases is so much easier in C#. The ideal scenario would allows us to use C# code to retrieve data, pass it seamlessly to the JavaScript library for rendering into charts for our dashboard.

I have broken down this review into three parts matching the three functional layers discussed above. Part 1 deals with the setup of the dashboard. It examines the HTML, CSS and JavaScript code that enables the look, feel and animation of the dashboard UI. Part 2 will look into the JavaScript code that creates a chart. Part 3 will demonstrate how we can use C# to merge sample application data with the chart code to enable us to integrate our data with the Highcharts library.

**Part 1: Dashboard**

The dashboard code as previously mentioned is based on a free template released by the folks at [TechGYO](http://techgyo.com/index.php/free-metro-ui-html-template-by-techgyo/). The majority of the code is contained in the file ‘*index.html*’, the adjacent screen snippet shows a collapsed view of this file with highlights & labels calling out various functional areas within the code. Note, the code has been collapsed in Visual Studio to provide this bird’s-eye view.



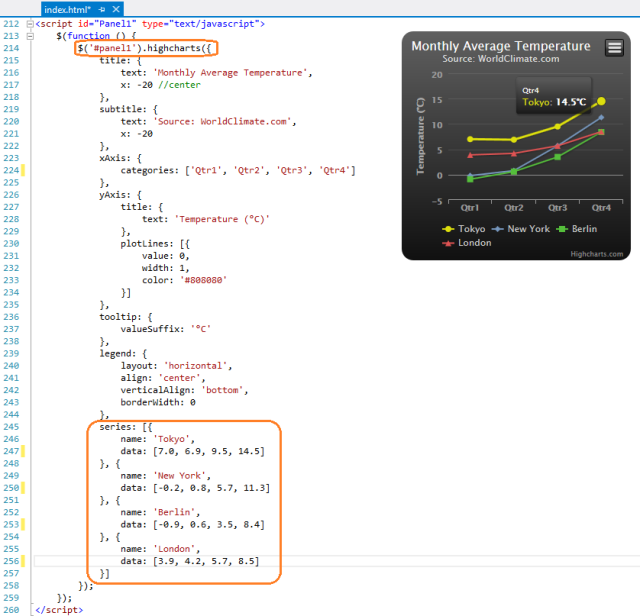
If you examine the code that draws each dashboard panel (lines 112 – 125), you will notice that it consists of a hyperlink enclosing an empty div and not much else.

Dashboard - Popup

This is because all the work of populating the panel is done within JavaScript code blocks that we will discuss in ‘Part 2: Drawing the charts’. The hyperlinks themselves are mostly inert and used only in case of panel 6 (*the green panel*) to demo drilldown/popup functionality.

**Part 2: Drawing the Charts**

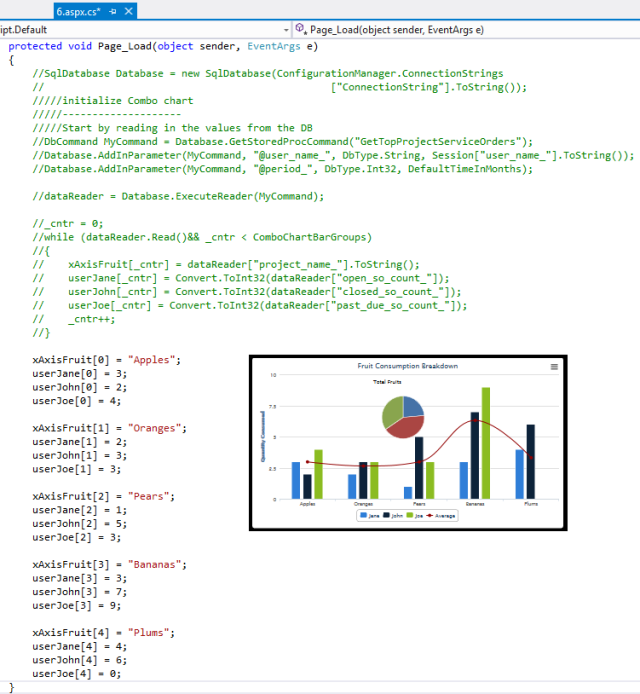
As mentioned in the previous section, each dashboard panel is initially empty and populated by individual JavaScript code blocks. The snippet below shows one such block alongside the chart rendered.



This script defines a JavaScript object. Line 214 ensures that the object (Linechart) is placed inside Panel 1. Lines 245 – 257 define the series data being displayed on the chart. Note, Highchart objects are defined using JSON. If you are unfamiliar with JSON, you should look up one of the many tutorials available on the web. A good starting reference is the one at [http://www.w3schools.com/json/](http://www.w3schools.com/json/" \o "W3Schools - JSON Tutorial).

**Part 3: Merging Application Data with Chart Code**

You may have realized that the example until this point is of limited utility as the data is hard-coded into the page. Most use case scenarios involve reading data from a data source and dynamically populating the chart. This section seeks to get you started on that path. In the interest of brevity and focus, I do not plan to cover how data is to be extracted from your data source. I assume you are able to write the code to read your application data. For what it’s worth, I have included some commented code that I used for my application. The item to note in the code below is that we are using some variables (*arrays defined at the class-level*) to store the values that will be the input for our chart.



You can then bind your application data into JavaScript code using data-binding expressions.



That completes this brief introduction to developing a dashboard. You can modify the JavaScript code to use any other JavaScript-based charting library. You are not limited to charts alone, the content of each dashboard panel can be pretty much anything that is valid within a web page. The simplicity of the template will hopefully ensure a short and gentle learning curve. Feel free to throw out ideas on where you would like to take this dashboard, perhaps we can explore those in a future update.

**Dependencies**

1. Visual Studio 2012 (<http://www.microsoft.com/visualstudio/eng/downloads>)
2. Highcharts Charting Library ([http://www.highcharts.com/products/highcharts](http://www.highcharts.com/products/highcharts" \o "Highcharts product page))
3. Highcharts.Net ([http://highcharts.codeplex.com/](http://highcharts.codeplex.com/" \o "Codeplex home of Highcharts.Net open source project)) – not affiliated with Highcharts Charting Library
4. Metro UI Template by TechGYO.com ([http://techgyo.com/index.php/free-metro-ui-html-template-by-techgyo/](http://techgyo.com/index.php/free-metro-ui-html-template-by-techgyo/" \o "TechGYO article describing Metro UI template))

**Dashboard Implementation Reference Websites**

1. The Geckoboard Blog ([https://demo.geckoboard.com/dashboard/B6782E562794C2F2/](https://demo.geckoboard.com/dashboard/B6782E562794C2F2/" \o "The Geckoboard Blog))
2. jSlate ([https://github.com/rasmusbergpalm/jslate/graphs](https://github.com/rasmusbergpalm/jslate/graphs" \o "jSlate))
3. D3 ([https://github.com/mbostock/d3/wiki/Gallery](https://github.com/mbostock/d3/wiki/Gallery" \o "D3))
4. Vaadin Dashboard ([https://vaadin.com/demo](https://vaadin.com/demo" \o "Vaadin Dashboard))

**History**

* 04 July 2013 - Initial draft
* 06 Oct 2014 - Added VS2010 version