JavaScript Hello jQuery and Bootstrap

This project assumes that you have done the basic Say Hello JavaScript project. The basic project uses vanilla JavaScript to read the value of text fields, and set the value of a heading. It also reacts to button clicks.

In the real world, nobody uses basic JavaScript to build anything, it gets too complicated. A huge variety of libraries and patterns have been developed to make things ‘easier’.

One of the first hugely popular libraries is called jQuery. jQuery gives us a different way of modifying and reacting to the web page. This means the code will look quite different. But once you are comfortable with jQuery, we will use it for a large number of our future projects.

The other library we are going to use is called Bootstrap. This is a styling framework which will make our page look *much* better than it did before. We will need to assign *classes* to our HTML elements so that Bootstrap can pick them up.

The Documentation

Using a third party library is only sensible if it comes with good documentation. Reading documentation and applying it to your own problems is one of the core skills of a software developer.

The documentation for jQuery can be found here: http://jquery.com/
The documentation for Bootstrap can be found here: http://getbootstrap.com/

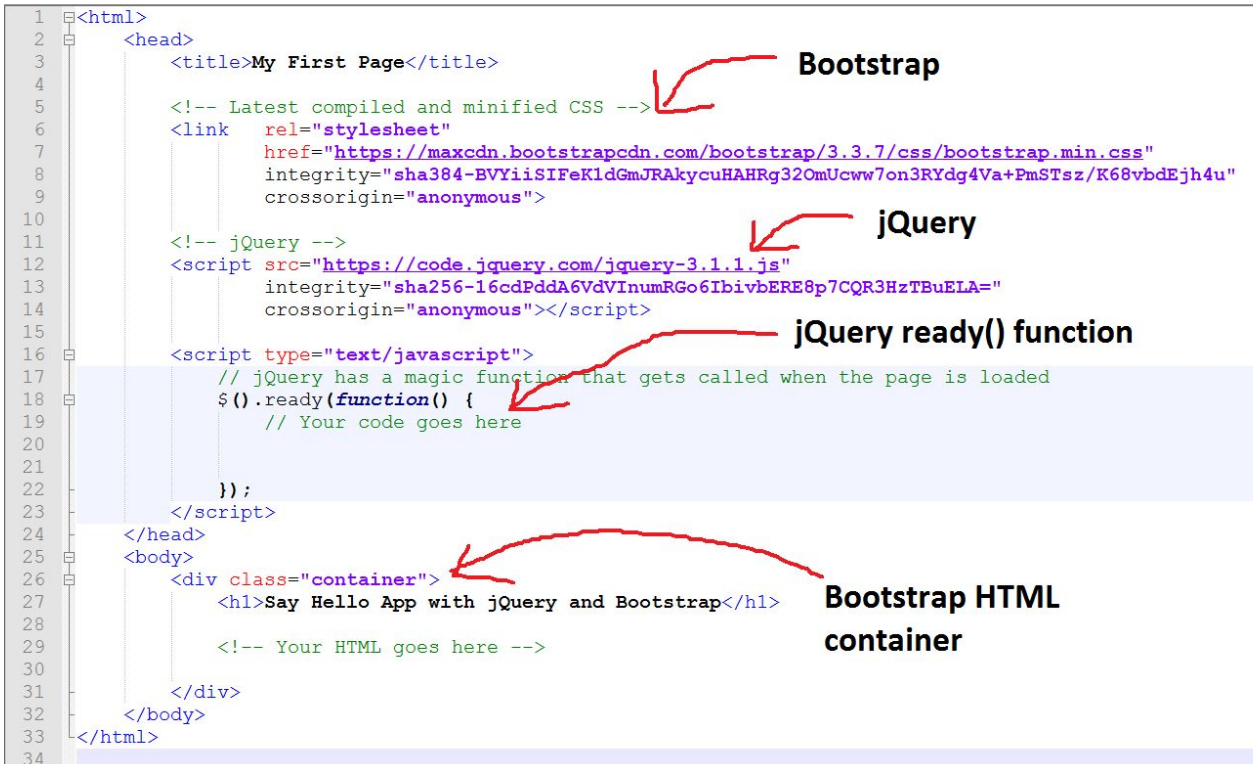
The Template Page

I have written a template page for a jQuery app styled by Bootstrap, and rather than type this in every time, I suggest you simply copy the template and work from there.

The template can be found in
KidsProjects/Exercises/JavaScript/Template_jQuery.html

Or if you are on the web, it can be found here
https://github.com/LonglevensCodeClub/KidsProjects/blob/master/Exercises/JavaScript/Template_jQuery.html

Here is a screenshot of the template page



```
<html>
    <head>
        <title>My First Page</title>

        <!-- Latest compiled and minified CSS -->
        <link   rel="stylesheet"
                href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css"
                integrity="sha384-BVYiiSIFeK1dGmJRAkycuHAHRg32OmUcww7on3RYdg4Va+PmSTsz/K68vbdEjh4u"
                crossorigin="anonymous">

        <!-- jQuery -->
        <script src="https://code.jquery.com/jquery-3.1.1.js"
                integrity="sha256-16cdPddA6VdVInumRGo6IbivbERE8p7CQR3HzTBuELA="
                crossorigin="anonymous"></script>

        <script type="text/javascript">
            // jQuery has a magic function that gets called when the page is loaded
            $().ready(function() {
                // Your code goes here


            });
        </script>
    </head>
    <body>
        <div class="container">
            <h1>Say Hello App with jQuery and Bootstrap</h1>

            <!-- Your HTML goes here -->

        </div>
    </body>
</html>
```

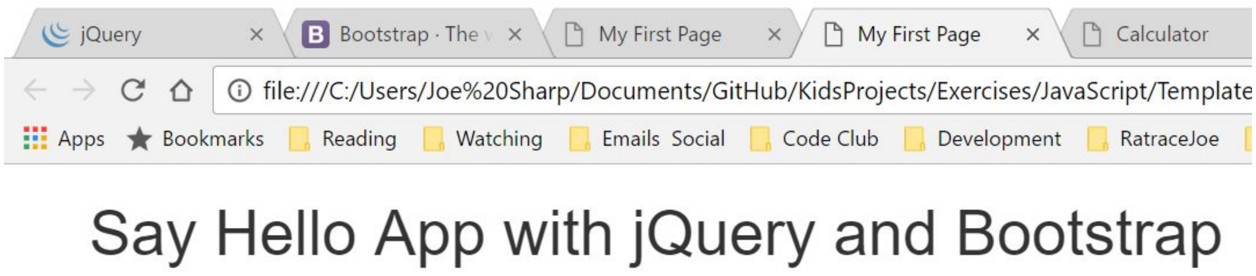
So there is a fair bit of code there, before we have event started! Where did it come from?

The two <script> tags are based on the documentation. Both jQuery and Bootstrap host versions of their libraries on Content Delivery Networks and we can simply *include* them in our page.

The jQuery ready() function is a magic function that jQuery calls when the page has been completely loaded. This means that our JavaScript can assume the entire page has been loaded before it tries to manipulate anything. Our code will all go inside this function.

The Bootstrap container is a bit of styled HTML that puts the page content in the middle of the page with a nice margin.

If you take a copy of this file into your folder under KidsProjects/Kids/<your name> then open it in Chromium, it should look like this.



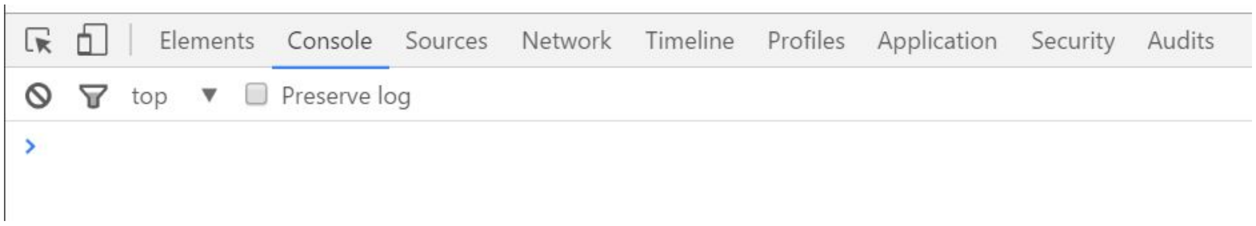
file:///C:/Users/Joe%20Sharp/Documents/GitHub/KidsProjects/Exercises/JavaScript/Template

Say Hello App with jQuery and Bootstrap

We are now going to rewrite our Say Hello application using these new libraries.

Open the Developer Console.

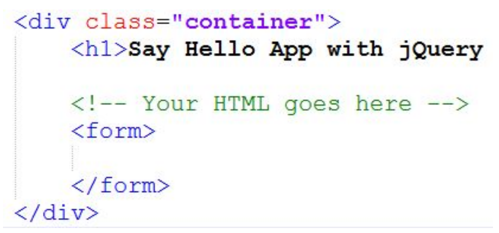
I am assuming that you are using Chromium on the Raspberry Pi (or Chrome on a normal desktop). Use the shortcut key of CTRL + SHIFT + I (capital i) to bring up the developer console. This takes a few seconds on the Pi.



I suggest you leave your page open with the developer console ready at **all times during development**. Every time you make a change, reload the page and check that things still work.

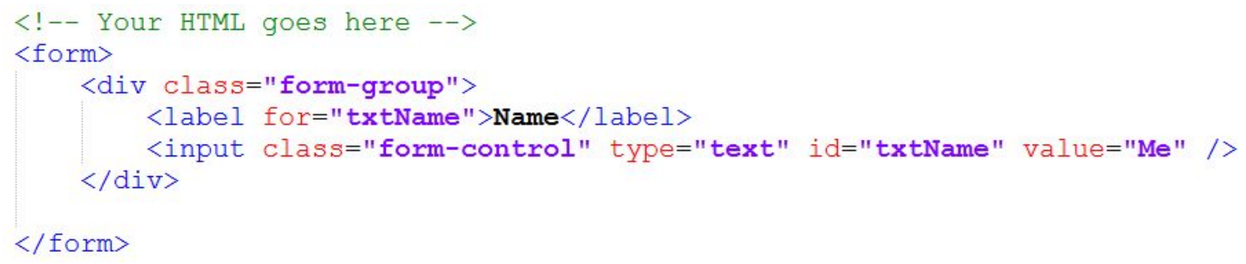
Write a Form with a Name Field

We will be writing a <form> within our HTML container. Add the tags as shown.



```
<div class="container">
    <h1>Say Hello App with jQuery

    <!-- Your HTML goes here -->
    <form>

    </form>
</div>
```

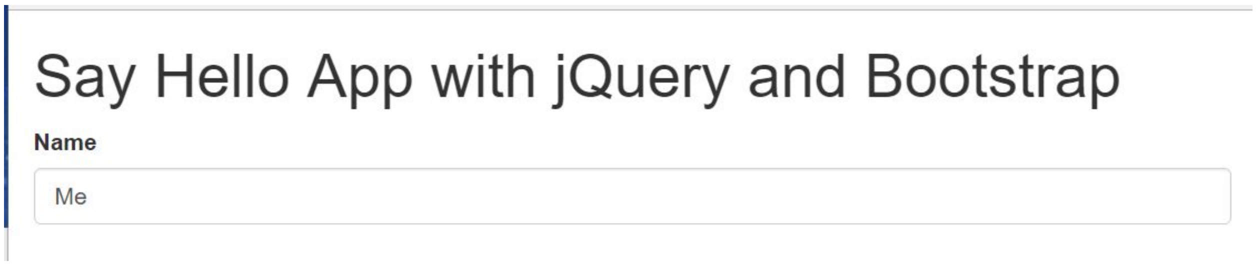
To add a field, we will be using the **form-group** Bootstrap class. The documentation for this can be found here: http://getbootstrap.com/css/#forms



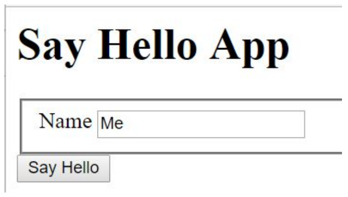
```
<!-- Your HTML goes here -->
<form>
    <div class="form-group">
        <label for="txtName">Name</label>
        <input class="form-control" type="text" id="txtName" value="Me" />
    </div>

</form>
```

We are using the HTML tag of <div> and applying the CSS class of **form-group** to it.
The <label> tag is used to help the user understand the purpose of the text field. The **for** attribute is used to associate the label with the correct input box using the **id**.
The <input> has a class of **form-control**. This is a Bootstrap class. It has an id of **txtName**. We will be using this id later to get a reference to the text field.

Save your code, reload the web page and it should look like this:



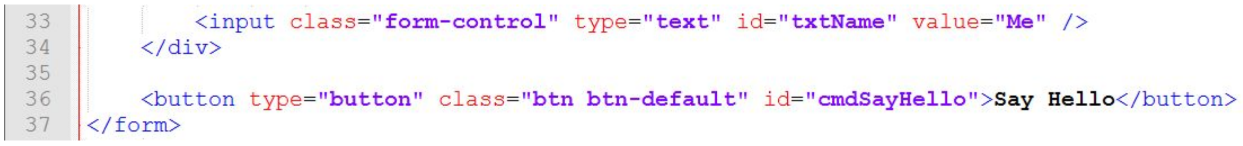
Hopefully you can see how this is an improvement over the vanilla JavaScript version, which looked like this:



Add the Say Hello Button

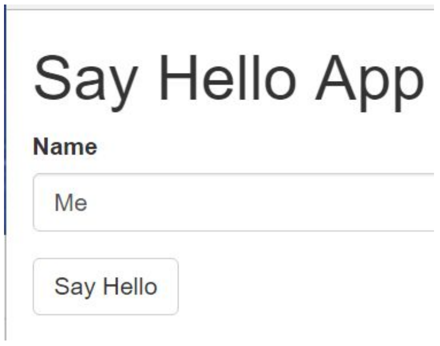
In this section we are going to add the button for the user to click. The Bootstrap documentation for buttons can be found here: http://getbootstrap.com/css/#buttons

We are going to use these examples to make our button look the way we want. Add the button code as shown on Line 36



```
33            <input class="form-control" type="text" id="txtName" value="Me" />
34        </div>
35
36        <button type="button" class="btn btn-default" id="cmdSayHello">Say Hello</button>
37    </form>
```

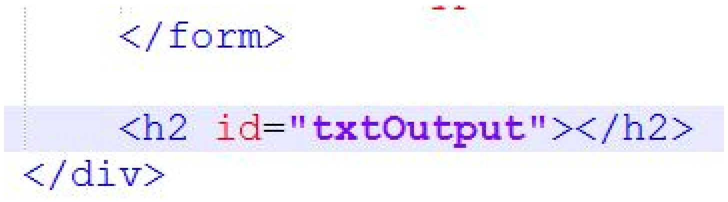
Save your code and reload the page, it should look like this:



We will now add the output heading.

Add the Output Heading

This is very simple, after the <form> tag, add the following <h2> tag. The important thing is to note the **id** you use for the heading. We will be using that **id** to get a reference to the heading using jQuery.



```
    </form>

    <h2 id="txtOutput"></h2>
</div>
```

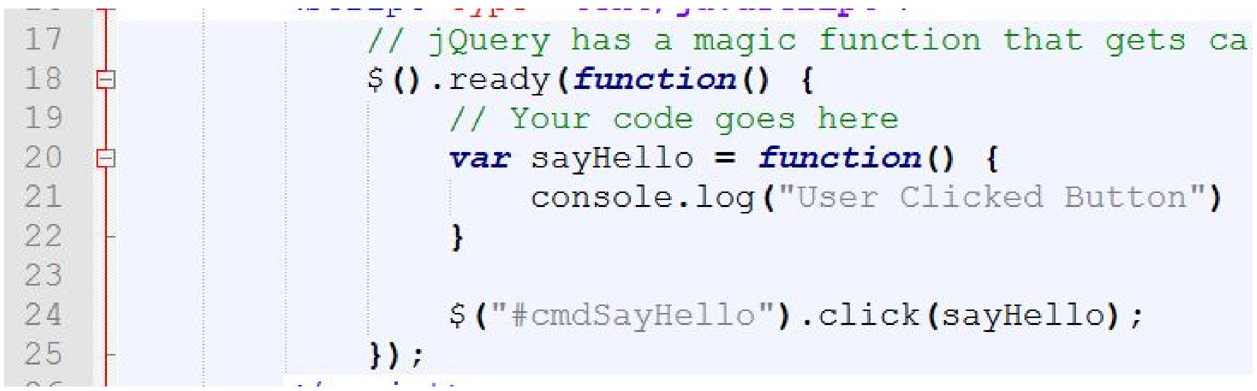
Save and reload the page, but at this point it will not look any different.

Now we are going to add a click handling function, this is where things get more exciting.

Add a Click Handler

Our button does not yet do anything, in the vanilla JavaScript project we added an **onclick** attribute to the button and wrote that function in our <script> tag. In jQuery we will do things slightly differently. The button HTML does not know anything about our click function, we will add it using code instead. This means the click handler is registered dynamically rather than statically.

Add the following code inside our jQuery ready function.



```
17      // jQuery has a magic function that gets ca
18      $().ready(function() {
19          // Your code goes here
20          var sayHello = function() {
21              console.log("User Clicked Button")
22          }
23
24          $("#cmdSayHello").click(sayHello);
25      });
```

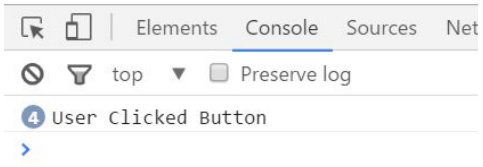
There is a lot going on here!

The function is using the console to simply print out a message when the user clicks a button. We can use this to test that we have assigned up the click handler properly.

To assign the click handler we use jQuery's magic selection syntax on Line 24. Note that we have to add the hash (#) symbol on the front of the id. The **id** used here must match the **id** of the HTML tag we are trying to reference. In this case it is the **id** of the **button**.

The click() function is part of jQuery's API. It tells jQuery that whenever we click on the element from our selection, we want it to call our sayHello() function.

Save your code and reload the page. Try clicking on the button and watch the **console** in the Developer Window. It should look like this:



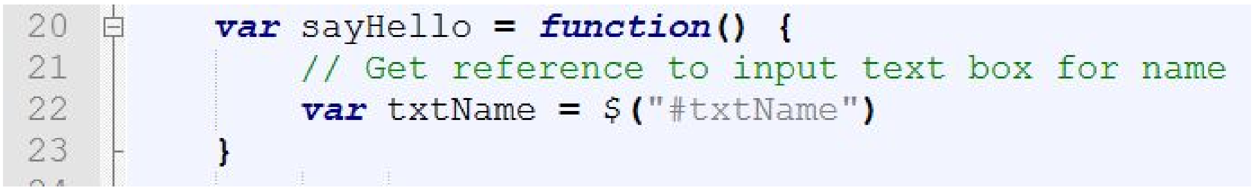
```
4 User Clicked Button
>
```

Fill out the sayHello() Function

In this section we will complete our application by getting hold of the text inside the **name** field, and then adding **Hello** to it and printing it out to our **output** heading.

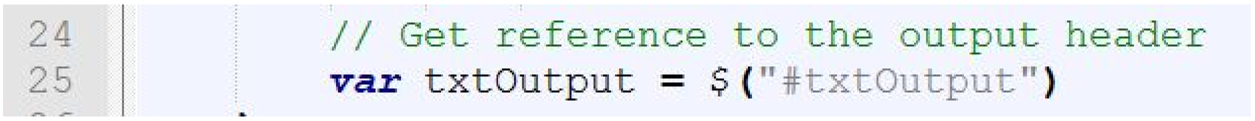
This code will all go inside the **sayHello()** function that we wrote in the last section.

Add the following line (Line 22) to get a reference to the **input name** field. Note that I have removed the console.log() from before.



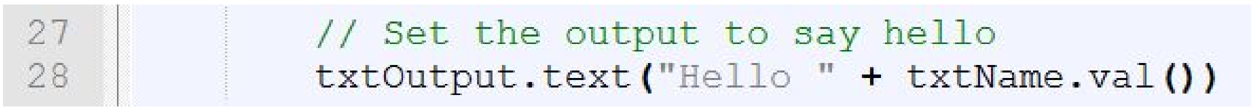
```
20      var sayHello = function() {
21          // Get reference to input text box for name
22          var txtName = $("#txtName")
23      }
```

Next we need to get a reference to the output header. This can be done as shown on line 25.



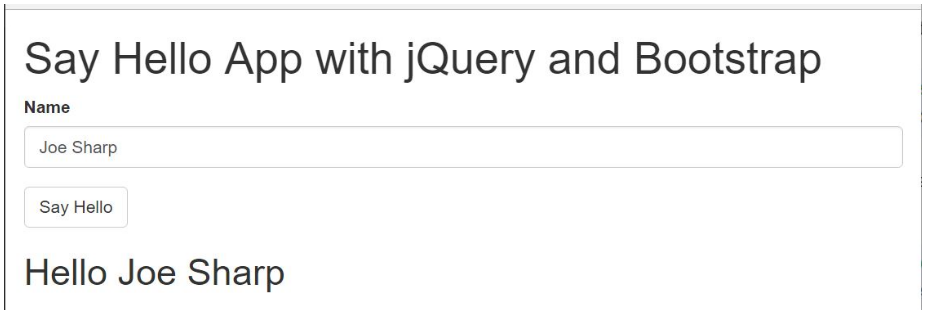
```
24          // Get reference to the output header
25          var txtOutput = $("#txtOutput")
```

The last line shows us setting the **text()** of the output field using the **val()** of the input field. **Val** is short for **Value**.



```
27          // Set the output to say hello
28          txtOutput.text("Hello " + txtName.val())
```

Save your code and reload the web page. Try putting your name in and clicking the button, you should find that the heading at the bottom is properly displayed as shown here:



If this does not work, check the console to see if there are any errors. Remember that **JavaScript is Case Sensitive!**

CONGRATULATIONS - YOU HAVE FINISHED THE SAY HELLO TO JQUERY PROJECT