**Assignment 2**

s00165159, s00152164, s00165383

Brian McGowan, Pearse Gorman, Kate O’Neill

Code Repository

https://github.com/mcgowanb/client-side-scripting-202-2

Task Runner

The task runner in this project was gulp and it was used as if to prepare the project for deployment to a live environment.

All CSS and JS files are minified and inserted into a dist/{type}.min folder respectively so they can be used again if needed.

HTML files are copied to the dist folder along with the JSON files, which are minified in the process.

Additionally, all CSS and JS files respectively are combined into a single min.js or min.css file and minifed for a production release under the dist folder.

Running the default gulp task will execute all of this.

jqBootstrapValidation, getAddress() & GoogleMaps API

This page of the assignment included a from validator, with appropriate validation messages. One of the inputs required on the form is the users address. getAddress was used to give the option to autofill these fields by entering a postcode. Entering the postcode also brings up a marker of the appropriate location on the map. The main issue encountered with these api’s was to get them all working together, i.e. linking the postcode lookup for the form to the postcode lookup for the map. This page made use of callback functions as covered in class.

HighCharts

HighCharts consumes data from the openweathermap api and plots it on the screen.

Historical & forecast data for the last 7 days is returned as json via an api call. This data is parsed and the relevant information is stored into separate arrays of time and value datasets for highcharts. These datasets are then added to highcharts as series of data with different plots and rendered on the screen as two different lines. I’ve also leveraged the moment library to enable conversion of timestamps to simple date periods for display. The chart will enable users to zoom and see more detailed data points if desired.

Additionally, a separate api request is made for current weather information and this is displayed in the top three boxes on the page. I’ve also implemented some fonts and icons to dynamically change depending on the weather condition ie, rain, sun etc….

jQuery Image Magnify and jFlow Image Slider

The jFlow image slider allows for many images within a div with a particular id to be placed in an image slider. The slider can be controlled using the next and previous icons or can simply run automatically. There are many other features to the slider that can be changed such as the speed of the transition between images, the delay on each image and the size of the slider itself.

jQuery image magnify enables a magnifying effect on images. As part of my page I gave each of the images in the slider the same class which allowed for them to be magnified once they were clicked. The magnification amount and the length of time to carry out the magnification could all be set, as well as the opacity of the original image when the magnification had taken place.

Image Power Zoomer v1.2

I tried to have another magnification plugin that would further magnify the image once it was enlarged by jQuery Image Magnify. However, I had difficulty getting the two to work together.