

Introduction to JavaScript

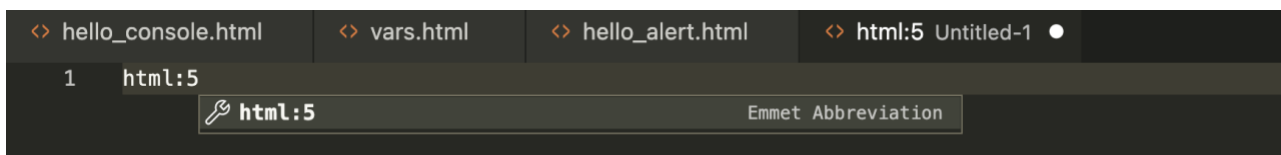
Introduction

“JavaScript is a programming language that makes websites interactive. HTML & CSS will let you make nice looking webpages, but they'll be static. JavaScript allows you to ask your user to input information, move elements around on the page, and for to make your website more modern and functional.” (Codecademy, 2013)

In this tutorial you will create some *very simple* programs in JavaScript and explore how JavaScript is incorporated into web pages.

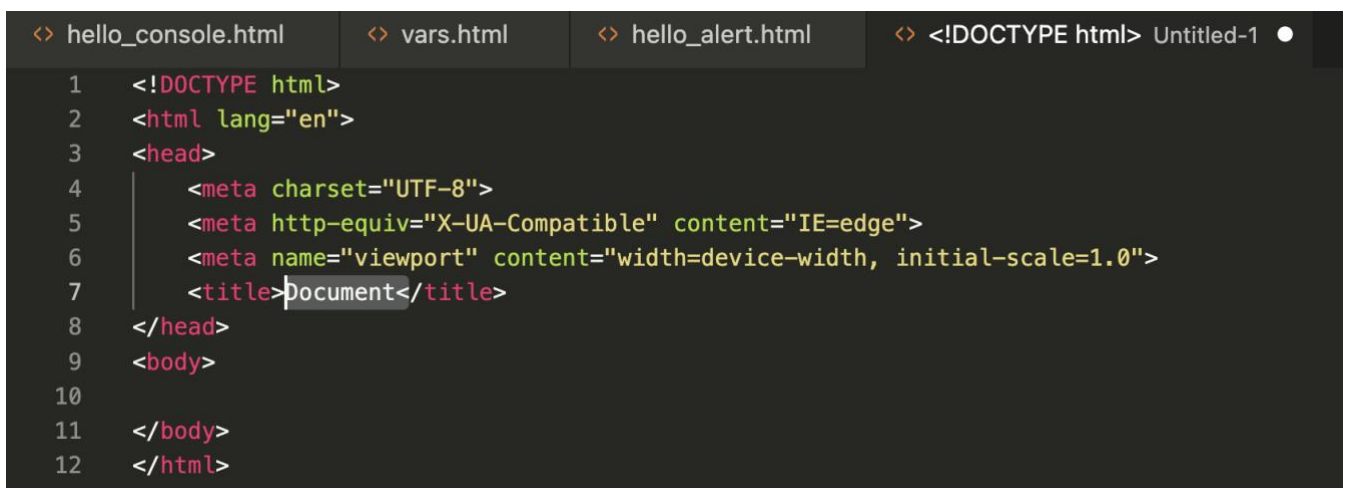
Recap: Creating a basic HTML file

You will be using VS Code (or another editor if you prefer) to create your JavaScript programs, the same editor that you used last week to create HTML5 webpages and CSS files. Open VS Code and create a new blank HTML file. Remember we can create our basic HTML 5 boiler plate by typing `html:5` and hitting enter.



The screenshot shows the VS Code interface with four tabs: `hello_console.html`, `vars.html`, `hello_alert.html`, and `html:5 Untitled-1`. The active tab is `html:5 Untitled-1`. In the editor, line 1 contains the text `html:5`. A tooltip is visible below the text, showing a magnifying glass icon, the text `html:5`, and the label `Emmet Abbreviation`.

Once we have our basic boiler plate, change the title tag on line 7 to something meaningful like ‘week 6 task 1’ for example.



The screenshot shows the VS Code interface with four tabs: `hello_console.html`, `vars.html`, `hello_alert.html`, and `<!DOCTYPE html> Untitled-1`. The active tab is `<!DOCTYPE html> Untitled-1`. The editor displays the following HTML boilerplate code:

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta http-equiv="X-UA-Compatible" content="IE=edge">
6   <meta name="viewport" content="width=device-width, initial-scale=1.0">
7   <title>document</title>
8 </head>
9 <body>
10
11 </body>
12 </html>
```

Remember that it is always good practice to save files regularly so before you type anything else save the file using an appropriate name such as `week6_task1` for example. Avoid spaces in file names, use underscore, hyphen or camelCase instead. Save this to an appropriately named folder structure in your G: drive - like `COMP1589/Week6`.

Task 1: Writing your first JavaScript

There are a number of ways of including JavaScript in a webpage. Today we will look at the simplest method, by inserting some JavaScript into the `body` of the webpage.

Following in the tradition of millions of other programmers, you are now going to write your first “Hello World” program (https://en.wikipedia.org/wiki/%22Hello,_World!%22_program).

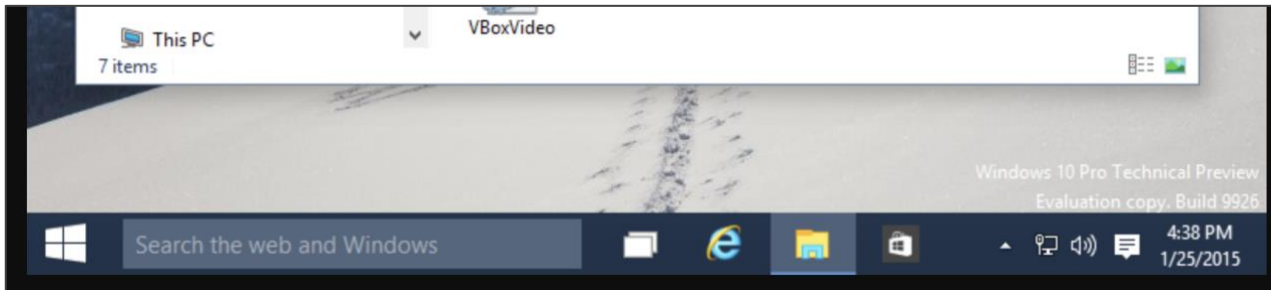
To do this, you will be utilising the `alert()` method, which is particularly useful in JavaScript as a way of outputting the result of code in an alert box on the screen (you may have come across alert boxes when you have forgotten to fill in a required box on a web form).

In the HTML page that you have just created add the following code in between the body tags
Line 8 opens our script tag, line 10 closes the script tag

Line 9 is our actual JavaScript code.

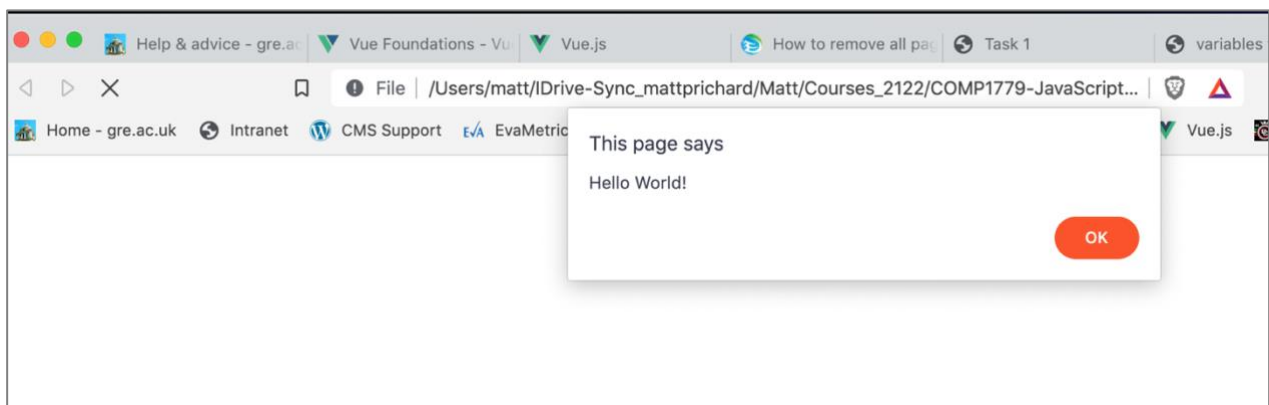
```
6     </head>
7     <body>
8     <script>
9         alert("Hello World!");
10    </script>
11
12    </body>
13    </html>
```

To run your code you now need to load the webpage in a browser. Therefore, it is important that you get into the habit of saving your webpage in VS Code and then opening the file in a browser. For this course we will be using Google Chrome but Firefox is fine too (avoid Edge or IE) to view our pages go to File Explorer/This PC and navigate to your file and open it with Google Chrome.



Depending on how your computer is setup you might have to right click on the file and choose “Open with ...” instead of double clicking on the file, which might open the file in Internet Explorer. *Please ask your tutor if you don’t know how to do this, as you will need to do this from now on.*

Once opened, you should see a browser window and an alert box saying “Hello World” like the following:



If you click on “OK”, or press return, the alert will disappear and you are left with a blank page.

Question to consider: How would you make the alert box say “*Hello Matt*” instead of “*Hello World*” ?

If you install the VS Code “open in browser” extension [I have linked to on Moodle](#) you can also open the code in your browser from within VS Code like I do in the lecture. This is a little easier but remember where your actual files are on your G drive and remember you can run them even if VS Code is closed by using the above instructions.

Task 2: Using `console.log`

Instead of using the `alert()` method, a more contemporary technique for outputting the results of a JavaScript program as you are creating it, is to use the JavaScript console that is included in the Developer Tools of browsers such as Google Chrome. In order to output something to the JavaScript console you use the `console.log()` method.

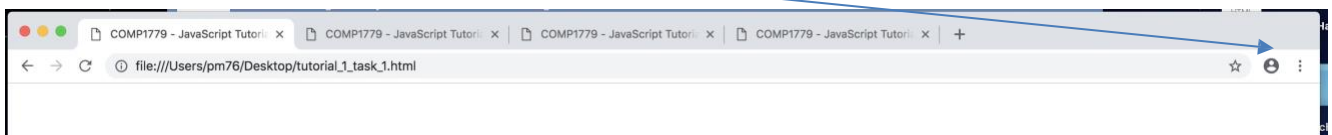
Save the file that you created for Task 1 as a new file (“File > Save As ...”) with an appropriate name such as `week6_task1.html`. Then, replace the line that contains:

```
alert("Hello World");
```

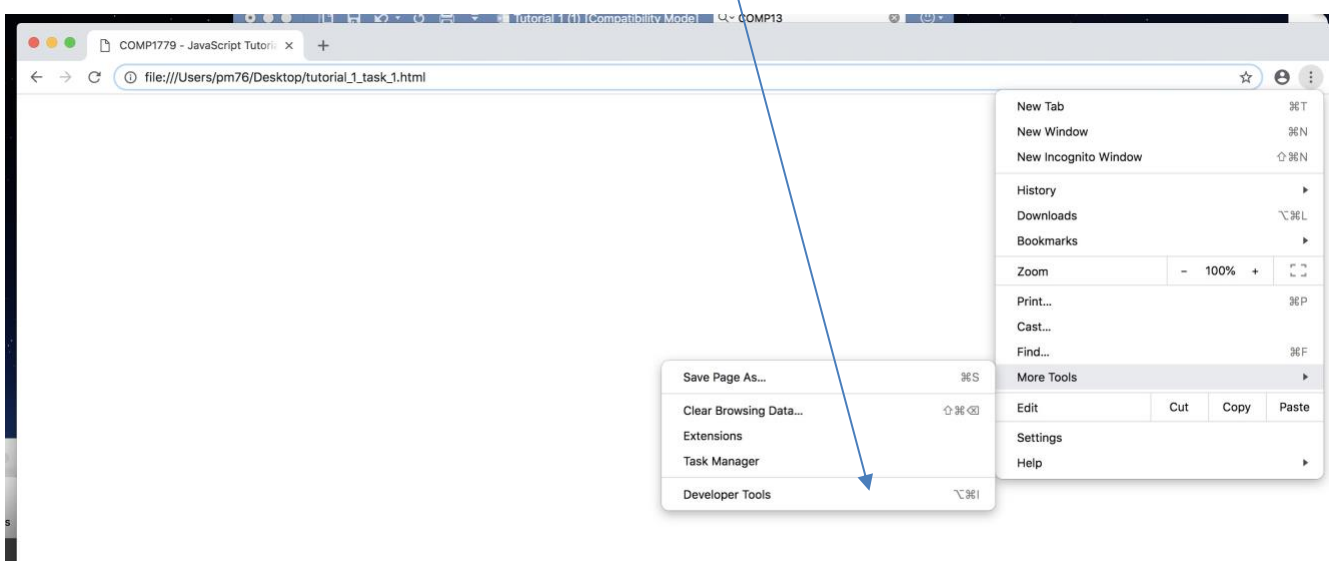
with the following:

```
console.log("Hello World")
```

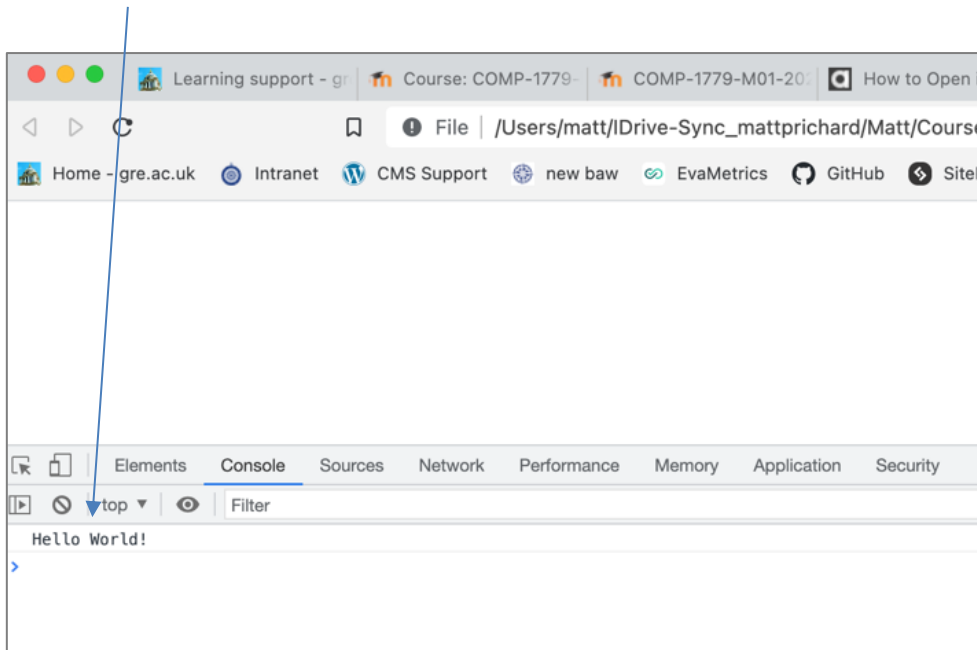
If you now try and open this using Google Chrome you should notice that the alert box does not appear, only the blank webpage. So where does the `console.log()` command appear? To view the JavaScript Console in Google Chrome on Windows 10, click the icon to the right of the URL bar that looks like 3 little dots:



- and then choose “More Tools > Developer Tools” or press **Ctrl + Shift + J**
- (**Cmd + Option + J on Mac**)
- <https://appuals.com/open-browser-console/> options for other browsers



You should then see the JavaScript console at the bottom of the window with the words “hello world” being shown (if you don’t, refresh the page).



From now on, **always make sure that the JavaScript console is open** when you run your webpages that contain JavaScript code.

Task 3: using InnerHTML

Save the file that you created for Task 2 as a new file (“File > Save As ...”) with an appropriate name such as `week6_task3.html`. Then, replace the line that contains:

```
console.log("Hello World")
```

With the following:

```
document.getElementById("text").innerHTML = "Hello World";
```

Remember to put this line

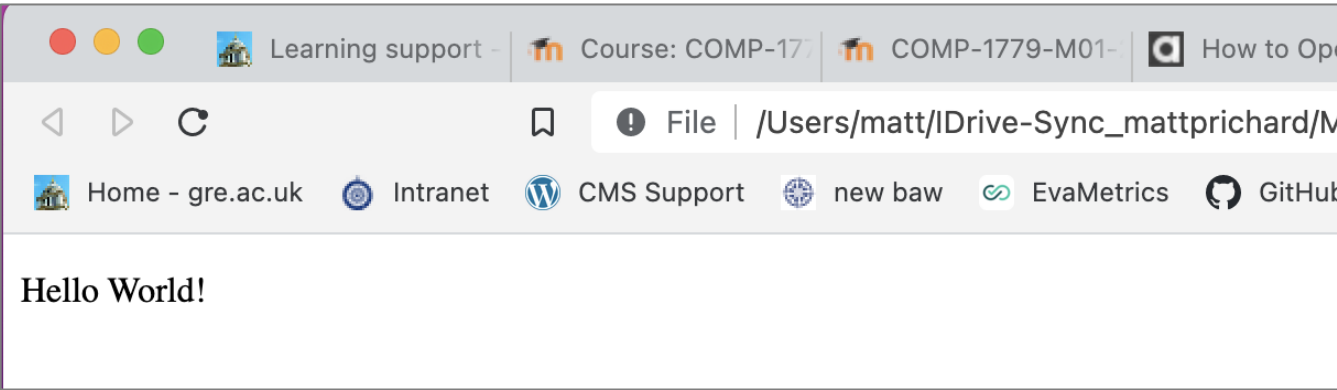
```
<p id="text"></p>
```

Just after your opening `<body>` tag **but before** the `<script>` tag.

```
</head>
<body>
  <p id="text"></p>
  <script>
    document.getElementById("text").innerHTML="Hello World!";
  </script>
</body>
</html>
```

We must provide somewhere for the script to output to before we run it. If we put the `<p>` tags **after** the script we will get an error.

You should see this in your browser.



Task 4: Creating Functions

Below is some code that will calculate the area of a rectangle. If we needed to calculate the area of another rectangle we would have to repeat parts of the code:

```
// Calculate the area of a rectangle
var aWidth = 10.5;
var aHeight = 3;
var message = "Area of this rectangle is ";
var area = aWidth * aHeight;
console.log(message + area);
```

This is an appropriate opportunity to create a function that can be reused calculate the area of any rectangle, dependent on values that we send to it. In a new file `week6_task4.html` add the following code into the body of your HTML page:

```
8  <body>
9      <script>
10         function calcRectangleArea(aWidth, aHeight){
11             var area = aWidth * aHeight;
12             return area;
13         }
14     var message = "Area of this rectangle is ";
15     var rectangleArea = calcRectangleArea(10.5,3);
16     console.log(message + rectangleArea);
17     rectangleArea = calcRectangleArea(4,10.2);
18     console.log(message + rectangleArea);
19
20     </script>
```

In this program, the function `calcRectangleArea()` accepts 2 parameters, `aWidth` and `aHeight`. It then multiplies the 2 numbers (`aWidth * aHeight`), storing the result in the `area` variable, and then returns the value of `area` to where the function was called from. In this case the result is returned and stored in the variable `rectangleArea`. The `calcRectangleArea()` function can be called a number of times, and sent different parameters each time, performing the same function.

As you were shown in the lecture, there are different types of function. The following code has the same output as the code above but achieves this in a slightly different way. It does not return a result to where it was called from but simply outputs the answer within the function itself. Type the following code into a new file and load the page in Google Chrome.

```
21 // Calculate the area of a rectangle
22 function calcRectangleArea(aWidth,aHeight,message){
23     var area = aWidth * aHeight;
24     console.log(message + area);
25 }
26 calcRectangleArea(10.5,3,"Area of the first rectangle is ");
27 calcRectangleArea(4,10.2,"Area of the second rectangle is ");
28
29
30 </script>
```

There are a number of ways of achieving the same result when programming. Either of these solutions is correct in this particular instance. As you gain more experience you will start to understand when to use these different types of function. For now, concentrate on trying to understand how each function works.

Task 5: Creating your own Functions

In order to demonstrate the use of functions you are now going to create a simple programme that converts Fahrenheit to Centigrade.

The maths for this calculation can be found at

<http://www.mathsisfun.com/temperature-conversion.html>

You will need to look through the interactive area calculator program on Moodle for the basic structure for this program.

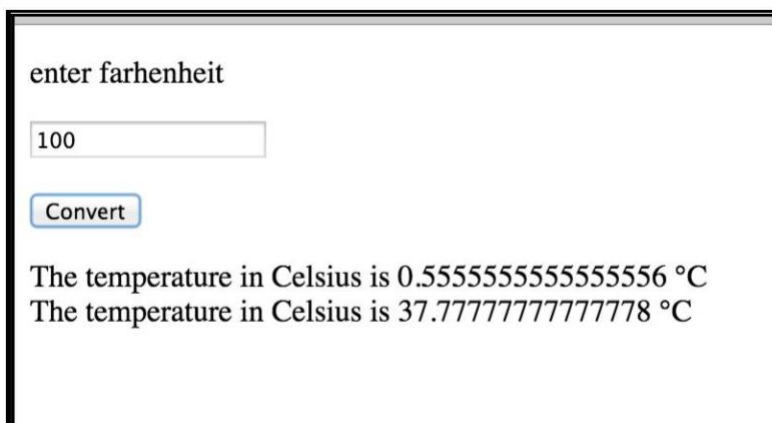
There are a few stipulations you must follow.

1. The user must be able to enter a numeric value for Fahrenheit into a text box
2. There needs to be a submit button.
3. There will be 2 functions
 - a. One will do the temperature conversion and return the result.
 - b. The second will be called when the button is clicked, it will get the value from the text box, pass it to the first function and output the result using innerHTML.

Make sure the user knows what the result means by use of string concatenation.

Make use of the += assignment operator so the user can display multiple results

You may have something that looks like this in the end – or better if you use some CSS.



The screenshot shows a web form with a title "enter farhenheit" (note the typo). Below the title is a text input field containing the number "100". Underneath the input field is a button labeled "Convert". Below the button, there are two lines of text output: "The temperature in Celsius is 0.555555555555556 °C" and "The temperature in Celsius is 37.7777777777778 °C".

For the keen.

Notice how my output has too many numbers after the decimal place.

Using the toFixed() method http://www.w3schools.com/jsref/jsref_tofixed.asp

I have set my result to 2 decimal places, see if you can apply the toFixed method too.

enter farhenheit

The temperature in Celsius is 0.56 °C

The temperature in Celsius is 60.00 °C

The temperature in Celsius is 37.78 °C

