**A452 Java Script**

**William Wood**

**7474**

**For June 2017**

**Task 1:**

**My plan for completing task 1:**

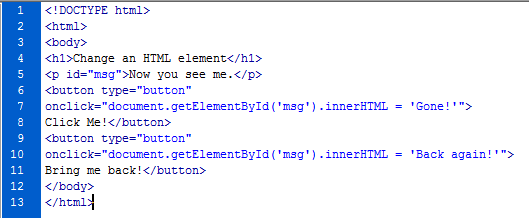
I am going to start by reading through the task. Then I’m going to copy the code into Dreamweaver and run it to make sure it works. If there are any errors, then I’m going take remedial action and run it again. I’m also going to explain how I ran the script and explain each line.

**Success criteria:**

In order for this task to be successful:

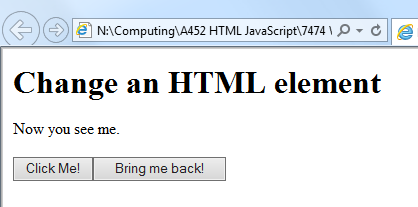
* I will copy the code in to my IDE (Macromedia Dreamweaver) without error.
* I must make sure that the code displays the right text on the initial run.
* I must make sure that when the "Click Me!" button is pressed the "Now you see me." paragraph changes to "Gone!".
* I must make sure that when the "Bring me back!" button is pressed the paragraph changes to "Back again"
* I must make sure that my code works in the web browsers Google Chrome and Internet Explorer.

**A screenshot of script that I have entered:**

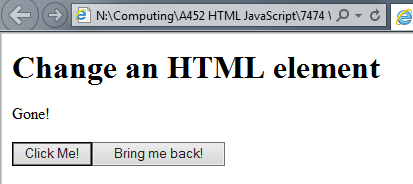


**Evidence of the code running:**

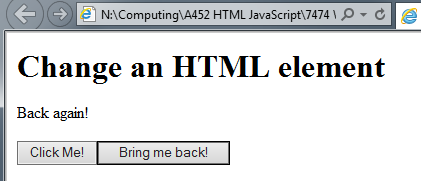
Initial run:



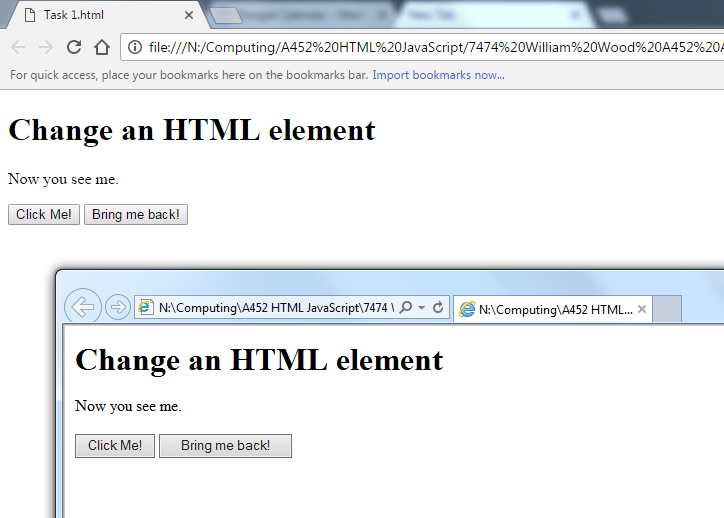
When the “Click Me!” button is clicked, the text changes from “Now you see me.” to “Gone!”:



When the “Bring me back!” button is clicked, the text changes from “Gone!” or “Now you see me.” to “Back again!”:



**Testing of the code in different browsers:**



This image shows the code runs correctly in the 2 browsers, Google Chrome and Internet Explorer.

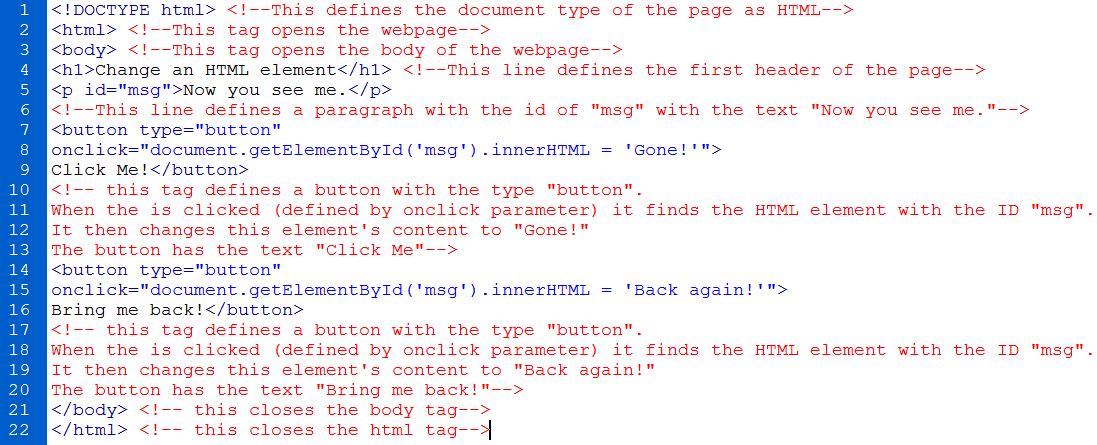
**Evidence of remedial action taken to correct errors in the code:**

The code ran perfectly first time so I didn’t need to take any remedial action.

**An explanation of how you ran the code (Add any sources of additional information you may have researched):**

I copied the code into Dreamweaver and pressed F12 to open the code in the web browser. The web browser fetched the page from the file then because the JavaScript is waiting for an event (onClick) it does not do anything until either buttons are pressed. The HTML runs one line at a time so the header is loaded first, then the heading, then the paragraph, the buttons and finally the JavaScript.

(Source: [http://www.makeuseof.com/tag/what-is-JavaScript-how-works/](http://www.makeuseof.com/tag/what-is-javascript-how-works/) )

**Annotation of the code**

I annotated all my HTML code using the comment tags **<!--** and **-->**. This makes it easy for other people to see what each line of the code does in detail. This also helps with troubleshooting as you can clearly see what each tag does and how it works.

**Evaluation against the success criteria:**

|  |  |  |
| --- | --- | --- |
| **Criteria Point** | **How the Criteria Point Has Been Met** | **Evidence** |
| I will copy the code in to my IDE (Macromedia Dreamweaver) without error. | I copied the code into Dreamweaver and the code ran perfectly first time with no errors |  |
| I must make sure that the code displays the right text on the initial run. | When I ran the code, there were no errors and the code displayed the correct text and buttons on the webpage. |  |
| I must make sure that when the "Click Me!" button is pressed the "Now you see me." paragraph changes to "Gone!". | The JavaScript buttons work when they are clicked (shown in YELLOW) and the text changes to "Gone!" (shown in RED) |  |
| I must make sure that when the "Bring me back!" button is pressed the paragraph changes to "Back again" | The JavaScript buttons work when they are clicked (shown in YELLOW) and the text changes to "Back Again" (shown in RED) |  |
| I must make sure that my code works in the web browsers Google Chrome and Internet Explorer. | When I ran the code in the two different web browsers, I found that it worked perfectly |  |

**Task 2:**

**My plan for completing task 2:**

I am going to start by reading through the task and planning how I am going to complete the task. I’m then going to make a success criteria and decide what my code needs to do. I’m then going to code the task, testing each phase as I complete them. When I have finished my code, I’m going to evaluate it against the success criteria, correcting my code if it does not meet all the criteria points. I’m then going to annotate every line of my code, explaining what each line of it does.

**Success criteria:**

For this task to be successful I should:

* Create an array with the items: Printer, Tablet, Router, Floppy Disk, Monitor.
* Create a paragraph and assign an ID to it
* Sort the array in alphabetical order
* Display how many items are in the array
* Make sure that the code works in all browsers

**A screenshot of my array:**

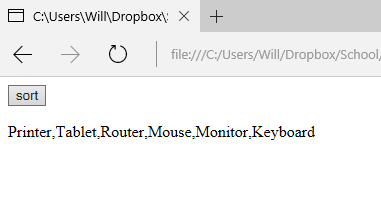
****

**A screenshot of the code:**

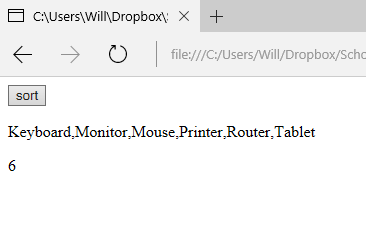
****

**Evidence of the code running:**

Initial run:

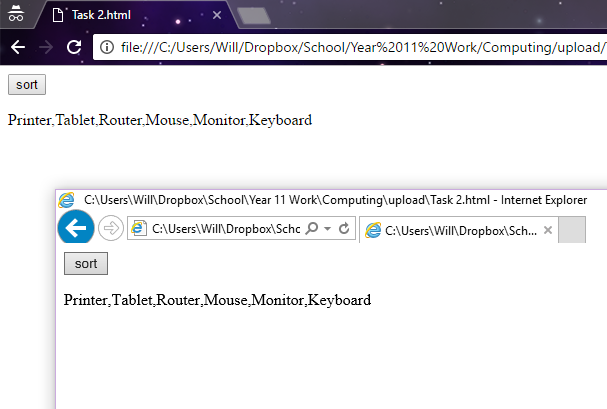


When the “sort” button is pressed, the array is sorted into alphabetical order and the number of items in the array is displayed:

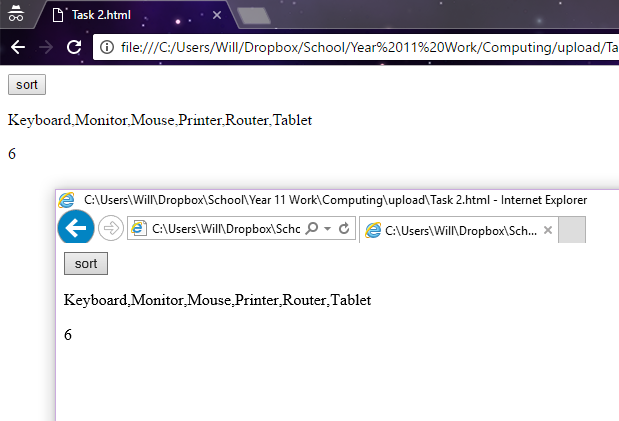


**Testing the code in different browsers:**

Initial run:



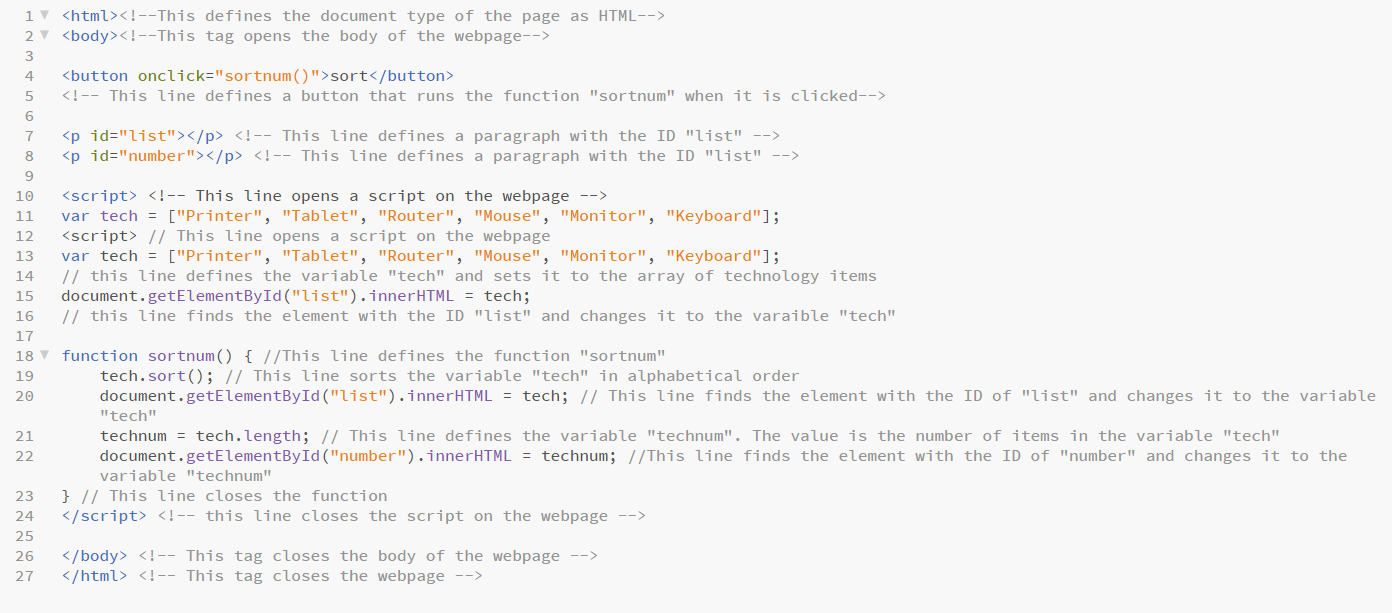
After the “sort” button is pressed:



**Evidence of remedial action taken to correct errors in the code:**

My code ran without error in the initial run so no remedial action needed to be taken.

**Annotation of my code:**

****

I annotated all my HTML code using the comment tags **<!--** and **-->**. I also annotated all my JavaScript code with “//”. This makes it easy for other people to see what each line of the code does in detail. This also helps with troubleshooting as you can clearly see what each tag does and how it works.

**Evaluation against success criteria:**

|  |  |  |
| --- | --- | --- |
| **Criteria Point** | **How the Criteria Point Has Been Met** | **Evidence** |
| Create an array with the items: Printer, Tablet, Router, Floppy Disk, Monitor. | I have created an array with all the required items |  |
| Create a paragraph and assign an ID to it | I have created two paragraphs, both with different IDs |  |
| Sort the array in alphabetical order | I have created a JavaScript function that sorts the array into alphabetical order when a button is pressed |  |
| Display how many items are in the array | I have created a JavaScript function that displays the number of items in the array when the button is pressed. |  |
| Make sure that the code works in all browsers | I have tested my code in two different web browsers; Google Chrome and Internet Explorer |  |

**Task 3**

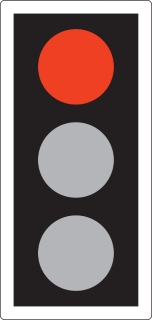
**My plan for completing task 3:**

I am going to start by making a list of assets that I need to gather to complete the task. I'm then going to create a success criteria to evaluate my code against later. I will research the best place to store the assets and why. I will write a script that changes the traffic lights when a button is clicked. I will test my code and evaluate it against the success criteria.

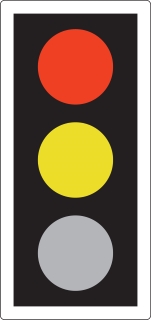
**Asset list:**

I will need:

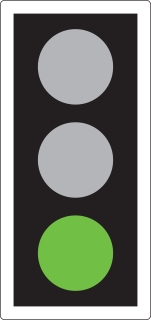
-Red traffic light:



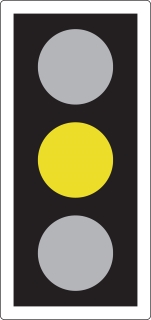
-Red amber traffic light:



-Green traffic light



-Amber traffic light:



I decided to get my assets from the UK Government website (<https://www.gov.uk/guidance/the-highway-code/light-signals-controlling-traffic>) as the artwork is designed clearly and is licenced under the Open Government Licence (<https://www.nationalarchives.gov.uk/doc/open-government-licence/version/3/>) so the images can be used freely any copyright issues.

The filenames are shortened versions of the colours, for example the red traffic light is called tl-red.jpg. This makes finding the correct assets easy and simple to type in.

I have stored the images in the JPEG format because it has very broad compatibility within web browsers.

**Success criteria:**

In order to be successful in this task, I must:

* Create a traffic light sequence in JavaScript that runs on the click of a button
* Test to make sure the code works as expected and take remedial action if needed
* Do research on where the best place to store assets is
* Explain and annotate every line of my code

**Screenshot of my code:**

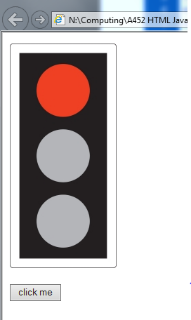


**A screenshot of my array that stores the traffic light assets:**

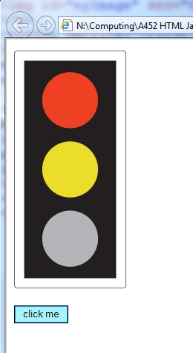


**Testing of the code:**

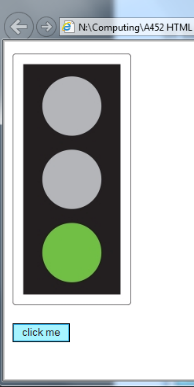
Initial run:



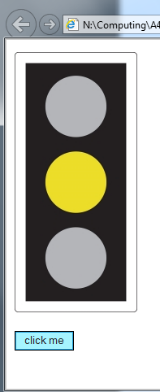
First button press:



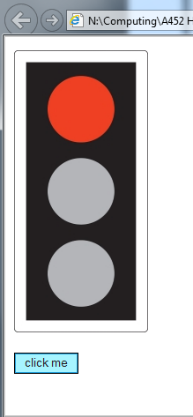
Second button press:



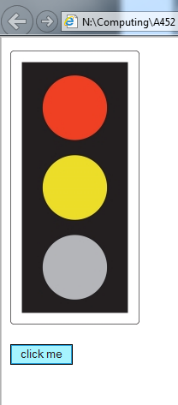
Third button press:



Fourth button press:



Fifth button press (to test the loop):



These screenshots show no issues with the order of the traffic lights and the loop also works with no issues.

**Remedial Action Taken:**

I did not have to take any remedial action with my code because it worked on the initial run and no errors were found.

**Annotation of my code:**

****

I annotated all my HTML code using the comment tags **<!--** and **-->**. I also annotated all my JavaScript code with “//”. This makes it easy for other people to see what each line of the code does in detail. This also helps with troubleshooting as you can clearly see what each tag does and how it works.

**Evaluation against the success criteria:**

|  |  |  |
| --- | --- | --- |
| **Criteria Point** | **How have I met the point?** | **Evidence** |
| -Create a traffic light sequence in JavaScript that runs on the click of a button | I have created a working webpage where the traffic lights advance on a click of a button. This is shown when the YELLOW button is pressed the traffic light advances. |  |
| -Test to make sure the code works as expected and take remedial action if needed | There were no issues with my code so no remedial action needed to be taken. Testing is shown above. | **See above** |
| -Do research on where the best place to store assets is. | I did research showing where to gather assets. I decided to gather them from gov.uk because they were copyright free | **N/A** |
| -Explain and annotate every line of my code. | I have annotated every line of my code to explain what each one does. | **See above** |

**Task 4:**

**My plan for completing task 4:**I am going to reuse my code from task 3 for this task, removing the button and adding a time statement as well. This will ensure that the code is functional without any errors. I will then evaluate my code against the success criteria and fix or improve any areas that do not meet the criteria.

**Success Criteria:**  
In order to be successful in this task:  
-Remove unnecessary code from task 3  
-Add a timing statement to advance the lights automatically.  
-Test the code to make sure that there are no issues.

**Screenshot of my code:**

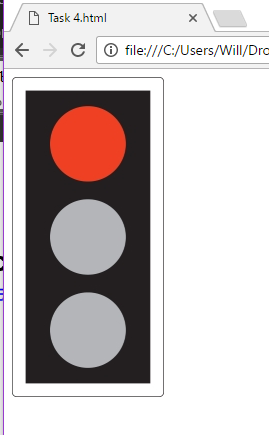
****

**A screenshot of the timing JavaScript:**

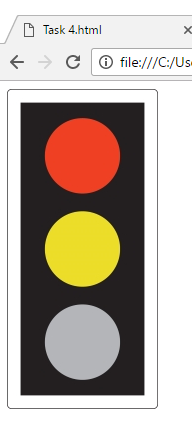
****

**Testing of the code:**

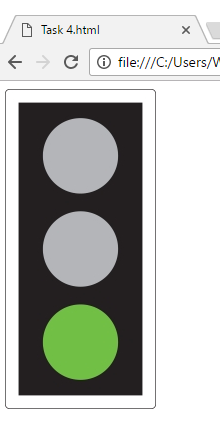
First traffic light:

****

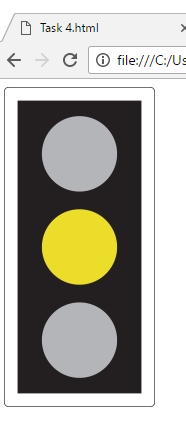
Second traffic light:



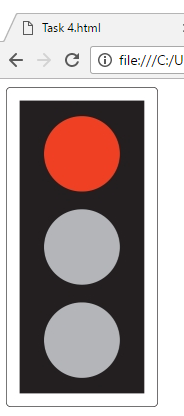
Third traffic light:



Fourth traffic light:



Back to the first traffic light (to show that the loop is working):

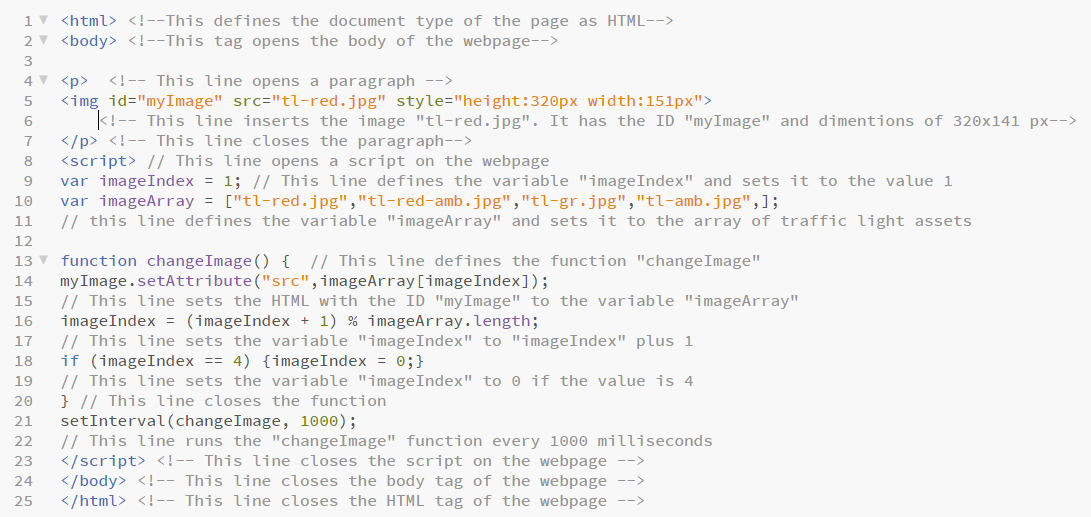


**Remedial action taken:**

I had to change the timing of the traffic lights because they were too fast at first. To do this I changed the “changeImage” value to 1000 instead of 100. This is shown below:

****

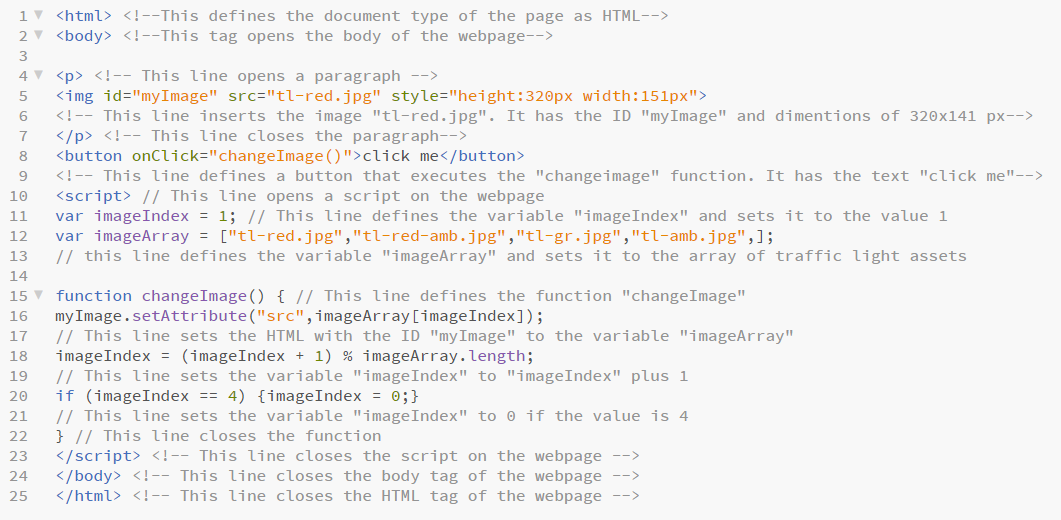
**Annotation of my code:**

****

I annotated all my HTML code using the comment tags **<!--** and **-->**. I also annotated all my JavaScript code with “//”. This makes it easy for other people to see what each line of the code does in detail. This also helps with troubleshooting as you can clearly see what each tag does and how it works.

**Evaluation against the success criteria:**

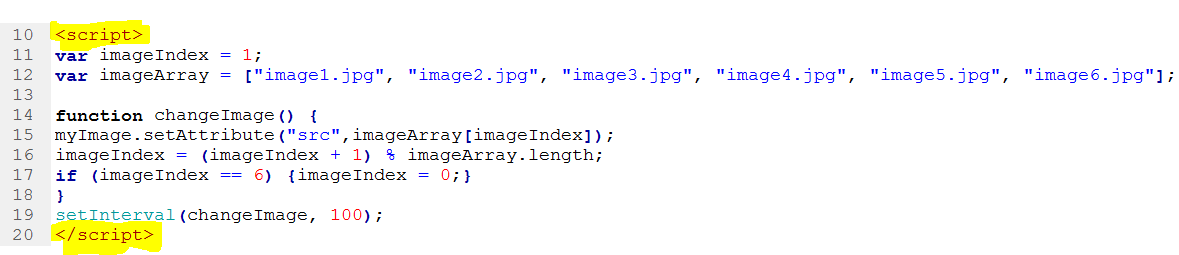
|  |  |  |
| --- | --- | --- |
| **Criteria Point** | **How have I met the point?** | **Evidence** |
| Remove unnecessary code from task 3 | I removed the button because it was no longer needed. The lines highlighted in RED are the ones that I removed | **See below** |
| Add a timing statement to advance the light automatically | I added a “setInterval” statement that runs the function “changeImage” every 1000 milliseconds |  |
| Test the code to make sure that there are no issues | I tested my code above and fixed the timing of the traffic lights. | **See above** |

****

**Task 5:**

JavaScript is an object-oriented computer programming language commonly used to create interactive effects within web browsers. [Oxford Dictionary]

An embedded script is JavaScript code in an HTML file. In my code, I have embedded JavaScript between the HTML <script> and </script> tags. Below is an example from my task 4 code.



**Advantages of embedded JavaScript:**  
You can embed the script in the head or body tags of the HTML code and keep the webpage in a single file. It is also faster because the page loads the webpage as one object rather than requiring multiple files.

**Disadvantages of embedded JavaScript:**   
It is harder to maintain and it can be slightly slower for the browser to load, because it has to compile and execute the JavaScript as well as the HTML webpage.

**Externally saved scripts**:

Externally saved scripts are a different file in the HTML with the extension **.js**. They are then embedded into the HTML (shown below). [w3schools]

<!DOCTYPE html>  
<html>  
<body>  
<script src="myScript.js"></script>  
</body>  
</html>

**Advantages of external scripts**  
It separates HTML and code making the JavaScript easier to maintain. The JavaScript file remains cached by the browser, speeding up the page load times. It also makes the code smaller, making it easier to spot mistakes. [Stackoverflow discussion]

**Disadvantages of external scripts**  
The browser has to make an extra HTTP request to get the JavaScript code. The code can be downloaded using the url of the file and this makes it easier for other people to steal your code.

**Discuss the benefits of both ways to use JavaScript:**

All scripts should be external, for maintenance and performance. Performance is better because the code is separate so it can be cached by browsers. If there are special characters in the JavaScript (like < or >) it creates problems with the HTML. It is better to use embedded code if you are only doing a few lines of JavaScript that are specific to the page then it is more convenient than doing an external file. [stackoverflow discussion]

In my tasks, I have used embedded scripts because I am only using a few lines of JavaScript and the disadvantages of an external script would have outweighed the advantages for using one.