Exercise 7 – CSS

Objective

Apply styling to some existing pages in order to demonstrate our understanding of the material covered so far.

Exercise Instructions

Part 1 – Creating a Navigation Bar

In the starter folder, you will find 4 pre-existing pages. We want to create a navigation element which we can use across these 4 pages to enable our users to move between them.

1. Open index.html from the starter folder and orient yourself to the code.
2. Create a main element to contain all of the content currently within the body.
3. Add a nav element above the main element and within it create an unordered list which contains individual list items with links to the following pages:
   i) Home (index.html)
   ii) Courses
   iii) Course Registration
   iv) The Team
4. Wrap the nav in a header element and include the white QA logo before the nav.
5. All the pieces are in place, have a look in the browser – the need for CSS is clear.

Part 2 – Styling the page

In order to style our HTML, we need to use CSS. We're going to use an embedded stylesheet here.

1. Insert style tags in the head of our document. This is where all your CSS will go.
2. First things first: if we want to use % based units on our heights, we need to set our body's height. Using the CSS you've learnt, select the body element and give it a height of 100%.
3. Select the header element and give it a background colour of #005BAB.
4. View the changes in your browser. See that white border around the header? Let's get rid of that – set the margin of the body element to 0.
5. Use the CSS descendent selector to get hold of the QA logo and set its display to block, float it left and set its height to 3em. What does 1em equate to?

6. Let’s centre our nav element: set its width to 60% and its left and right margins to auto.
7. Now we need to deal with our links which are contained in the unordered list. Firstly – we don’t want this to display as a bulleted list, so let’s turn that off by setting the list-style to none. Once you’ve done that, we can make some minor cosmetic adjustments by setting the following properties:

Margin	0
Padding	1em 0
Text-align	centre

8. List items are block-level elements and so display one per line. We want them to be next to each other, so set their display property to inline. We also want them to be slightly apart, so set their margin property to 2%.
9. Now to deal with the links themselves. It’s hard to see them on a blue background – so change the text to white by setting the colour property. We don’t want the underline, so turn that off by setting the text-decoration to none. We want the text to be in UPPERCASE, so use text-transform to achieve that.
10. Now when we hover over the links, we only have the change in cursor to indicate that these are links. This is not ideal. Using the appropriate pseudo-class, set the text-decoration to ‘underline’ when we move the mouse over each link.
11. Save your work and view it in your browser.

Part 3 – Styling the site

We've managed to style one page of our site, to do the same to each page would be tiresome and difficult to maintain. We're now going to implement this styling as an external stylesheet and add our navigation to each page of the site.

1. Create a new file in the starter folder called style.css.
2. Cut and paste all of the styling from between our style tags into style.css.
3. Import style.css into index.html and test that the page looks exactly the same as it did with the embedded stylesheet.
4. Create the same link to style.css in each of the HTML pages in the starter folder.
5. Check each page, you may find that the styles have 'broken' some of the layouts. We need to adjust our selectors to ensure we are only applying styles to the intended elements.
6. On the registration.html page our rules for unordered lists and list item elements are interfering with the intended layout. Adjust styles.css, so that you are only selecting ul and li elements descended from a nav element.
7. Copy and paste the embedded styles from registration.html into styles.css. It's unlikely this is how we want ul and li items to display by default, so amend the selectors, so they are only selected if descended from a form.
8. Insert (and where one already exists, replace) the whole header from index.html into each of the other HTML pages.

We now have a consistent header across all pages and one stylesheet for us to maintain.

Part 4 – Additional Changes

We can now start thinking about the base style of our website.

1. Currently, the website uses the browser's default font for the text which probably doesn't look quite how we'd like. Change the font to Verdana with sans-serif as the back-up.
2. We lost the headline images from some of our pages at the end there, so let's get those put in for each page. Add an h1 element, just before the close of our header and give a suitable heading to each page (you can just move the h1 elements that are already present in index.html and courses.html).
3. In our stylesheet, we need to select the h1 element and apply the following properties:

padding	4em 1em
color	white
background-size	cover
margin-top	0

4. We want each header to have a different element on each page, so we can apply that through the use of classes. In our stylesheet, create four classes with the below names which set the background-image to the corresponding URL.

keyboard	images/keyboard.jpg
team	images/theTeam.jpg
trainer	images/trainer.jpg
course	images/trainingCourses.jpg

5. Assign each of the h1 elements you just created to one of those classes and then view the different pages in your browser.
6. Finally, if you were to run our pages through an html5 outliner you'd find we have an untitled section. This is created by the nav element. We need to add a heading to it, but we don't necessarily want it to be displayed – so go ahead and add a heading to each nav and then using the appropriate selector in the stylesheet set its display property to none.