Extra notes - Web Usage - Basic Usage - Part 2

- Dr Nick Hayward

A brief introduction to the basics of web usage, including CSS and HTML.

Contents

Intro We may initially test basic HTML and CSS usage with a simple web page.

In Part 2 of the example, we may consider the following updates

- extra HTML elements
- CSS options for style and usage
- …

Add extra HTML We may now update the example HTML page, `index.html` , with a *quick links* menu.

For example, we may add some links for the menu to the `<body>` of the HTML document.

```
<body>
        <h3>Welcome to our website</h3>
        <p>This is the homepage for our test site.</p>

        <!-- menu for quick links -->
        <ul>
            <li>About</li>
            <li>Catalogue</li>
            <li>Index</li>
        </ul>
</body>
```

Update CSS for list We may then, of course, update the CSS to style this new `quick links` menu.

For example, we'll add a new *ruleset* for the `<ul>` unordered list.

```
ul {
    width: 150px;
    border: 1px solid #0396A6;
}
```

This initial ruleset adds a fixed width for the `<ul>` list, and styles this box with a coloured border.

update default styles We may also update the default styles for a `<li>` list item element in HTML. The default browser CSS styles this element with a round bullet.

Instead, we'll remove this bullet in the custom CSS for this page.

```
li {
    list-style-type: none;
}
```

In effect, we can modify the default styles and behaviours defined initially by modern web browsers.

Add a CSS class We may define a `class` attribute on a given HTML element, which acts as a specific target and *selector*.

In effect, we may use a `class` attribute in the HTML to enable a defined subset of elements. We may, for example, add a class to a single `<p>` or `<li>` element.

The CSS ruleset for the class may then add specific styling for the HTML.

For example, we may add a class to the `<ul>` *quick links* menu. This would allow us to customise the styling of this specific list beyond a ruleset for `<ul>` elements.

```
<ul class="quick-links">
    <li>About</li>
    <li>Catalogue</li>
    <li>Index</li>
</ul>
```

We may then add a ruleset for this class to our CSS file,

```
.quick-links {
    background-color: #A7BDD9;
}
```

The `<ul>` element will now be styled using the ruleset for `ul` and the class `quick-links` in the CSS file. We may, however, also choose to override properties defined in the initial `ul` ruleset with specific values in the `quick-links` ruleset.

For example, we may choose a different colour for the border of this list grouping.

style elements with class Whilst the current example uses the class `quick-links` for a single, specific `<ul>` list grouping, we may choose to apply a class for a group of elements.

For example, we may define a class for all elements that need to be styled for a specific background colour, font size, and text indent. Wherever we apply this class correctly, the ruleset may be used for this required styling.

In our current *quick links* menu, we may choose to define the first list item with a larger font size and, perhaps, text underline. The defined class may then be added to each element where we require these style properties.

For example.

```
<li class="first-item">About</li>
```

and the CSS ruleset

```
.first-item {
    font-size: 18px;
    text-decoration: underline;
}
```

As noted, we may choose to add this class to other HTML elements and style with the above properties.

style specific elements However, we may also choose to be specific in our use of this class relative to a given HTML element.

For example, we may target only `li` elements with the specific class of `first-item`.

```
li.first-item {
    font-size: 18px;
    text-decoration: underline;
}
```

We have now restricted the class `first-item`, and may not apply it to other HTML elements. If we needed to style a paragraph with this class, we would have to update CSS selectors as well.

For example,

```
li.first-item, p.first-item {
    font-size: 18px;
    text-decoration: underline;
}
```

Style relative to position We may also select and style elements using their position in a HTML document. This position might be relative to a specific selected element or the hierarchy of the document itself.

For example, we might update our HTML for the *quick links* menu to include anchor elements for URL links.

```
<ul class="quick-links">
    <li class="first-item">About</li>
    <li><a href="">Catalogue</a></li>
    <li><a href="">Index</a></li>
</ul>
```

We may then select and style these links relative to their position in the document.

In this example, we may define a ruleset for these specific anchor elements relative to their position as a *descendant* of a `<li>` element.

```
li a {
    color: #F2CF63;
  padding: 15px;
  text-decoration: none;
}
```

We may also select elements using a *combinator* for selectors. For example, we might add a paragraph of content after the *quick links* menu in our current HTML page.

```
<ul class="quick-links">
    <li class="first-item">About</li>
    <li><a href="">Catalogue</a></li>
    <li><a href="">Index</a></li>
</ul>
<p>Add some content for the current page...</p>
```

This hierarchical relationship, adjacent sibling, may be defined as a selector for a ruleset in the current CSS,

```css
ul + p {
    margin: 30px;
}
```

With this type of selection, we may easily modify and style elements based upon their position in the document's hierarchy.

Style relative to state An element's *state* may also be used to select and style using CSS.

For example, we commonly determine a link's state, such as *visited* or *hover*, and then define an appropriate style using a *pseudoclass*.

For the links in our *quick links* menu, we may add a pseudoclass for *hover*.

```css
li a:hover {
    color: #ffffff;
    text-decoration: underline;
}
```

In effect, as the *state* of a `<li>` list item `<a>` anchor element is updated, changed to *hover*, we may apply the properties defined in the above ruleset. We change the text's colour to white, and then underline the text.

As the user moves the cursor from the element, the state is updated, removing the *hover*, and the style is removed.

References

- MDN - CSS
- W3 CSS
- W3 Schools - CSS