Extra notes - Markup Languages

- Dr Nick Hayward

CSS - Intro

A brief introduction to CSS.

Contents

Intro

A **Cascading Style Sheet**, or CSS, allows us to define stylistic characteristics for our HTML. In effect, it helps us define how our HTML is displayed and rendered. The colours used, font sizes, borders, padding, margins, links, and so on.

CSS can be added to a `<style>` element in the `<head>` of our HTML document, embedded as inline styles per element, or stored in external files.

All visual browsers, thankfully, now support CSS.

CSS is not intended to act as a replacement for encoding semantic and stylistic characteristics with elements, but instead to behave as an additional layer that allows us to directly affect the presentational layout and qualities of a web page.

Stylesheet `<link>`

We can add a link to our CSS stylesheet using the `<style>` element, which is then stored in the `<head>` element of the document,

```
<link rel="stylesheet" href="style.css" />
```

This allows us to write a stylesheet once, and then reference those styles in any HTML document throughout our site or application. If we make a change in this stylesheet, such as updating a `color` or `font-size`, where applicable this change will replicate throughout our site wherever the stylesheet is referenced. For most client-side development, this is still considered the standard way to add and maintain our CSS styles for a website or web application.

Embed styles

Another option is to embed the CSS directly within the `<head>` section of our HTML page using the `<style>` element. We then simply add standard CSS within this element.

It does have its limitations, as changes, if repeated across multiple pages, will also need to be changed multiple times. For multiple pages within a site, it lacks any grace of design and introduces an unnecessary replication of styling.

Inline styles

We can also embed styles per element, using what is known as **inline** styles. This simply means we can add a `style` attribute to our given element, and then add CSS styles as the value of the attribute. The benefit of this option

is that it allows us to customise a single element without using explicit selectors, although it can introduce further issues with maintenance and hard-coded redundancy.

However, **inline** styles have started to become more widely used with the surge in popularity of JavaScript libraries such as React. The main difference with this recent implementation of inline styles is the use of JavaScript to dynamically assign styles per element. However, such implementation is still dynamic and abstracted, just an alternative layer to the final CSS.

Pros of CSS

There are many benefits of CSS, including the inherent option and ability to abstract styles from content. In effect, we are isolating the styles and aesthetics of our design from the semantic markup and content.

Another inherent benefit lies in the cross-platform support offered for many aspects of CSS, and its rendering in most of today's popular browsers. It used to be a major headache ensuring correct rendering in different browsers, in some respects this can still be the case, but imagine if we had to style and design for each browser separately. CSS allows us to style once, in theory, and then apply to our HTML and render in different browsers. There are still caveats to this noble goal, but it's getting better with the support of various CSS frameworks.

CSS also allows us to support many different categories of device, not only major web browsers but mobile devices, screen readers, TVs, video game consoles, and so on. General support for accessibility features is a particularly useful aspect of CSS. We can specify spacing, alignment, positioning, fonts, and so on to help us create a tailored experience for accessible rendering and viewing.

Cons of CSS

However, as with most things, there are issues that seem to plague CSS and its implementation. Whilst browser support is improving, and underlying rendering engines are generally becoming more standards compliant, we still experience issues as designers with rendering quirks for certain styles. Rounded borders, square borders, padding, margins, and so on. Each may create issues depending upon rendering context, device, user agent, and so on.

Other issues can include, for example

- *everything is global* - CSS matches required selectors against the whole DOM. We try to avoid this using **naming strategies**, which can be awkward and difficult to maintain
- *it's a mess* - CSS can become an unruly mess, very quickly. This is often due to the nature of CSS itself, and therefore we're often tentative or simply afraid to delete anything for fear of the underlying consequences. CSS has a habit of growing large, also very quickly.

References

- MDN - CSS
- Perishable Press - Barebones Web Templates
- W3 CSS
- W3 Schools - CSS