

# DMIT 1530

Week 1  
CSS Refresher

HTML on its own is ...  
functional, but not great.

HTML is a glorified labelling system.  
All it does is tell a browser what things are.

**CSS** (Cascading Style Sheets) is a bunch of rules that tells the browser how all of those things should **look** and how they should be **laid out**.

# CSS Basics

cf. CSS acids

There are a few ways we can **include** CSS in our code. However, only one of them isn't trash.

We'll be using **external stylesheets** for this course.

External stylesheets are a **separate file** that we point to in the `<head>` of our HTML.

It's cleaner, easier to **maintain**, and allows us to control the look and feel of **multiple pages** at once.

But what about the CSS itself?

In CSS, we need to **select it to affect it**. All we're doing is selecting something we want to change and then applying some rules to it.



```
p { color: red; }
```

```
/* This selects every paragraph  
element in the document and makes  
the text colour red. */
```

## selector

The thing we want to affect with our rule.

## property

The characteristic about the selector we want to change.

## value

What we want to set the property to.

## declaration

Together, the property and its value make a declaration.  
You need to include both things for your rule to work.

selector

property

value

**p** { *color*: red; }



declaration

## Syntax Matters

After your selector, everything needs to be inside opening and closing { curly braces }.

The property and value must be separated by a colon.

Every declaration must end in a semicolon.

# Selector Types

It turns out that I have a lot of types,  
but others are a little more selective.

## multiple element selectors

Just like the name implies, these bad boys **select more than one thing at once**. This can keep you from repeating the same rules over and over again.

All you need to do is add a **comma** between your selectors.

```
h1, h2 {  
    font-family: 'Avenir', sans-serif;  
}
```

## descendant selectors

Descendant selectors let you target a **nested element**. Starting with the outermost element, work all the way down to the innermost element.

If you add a **space** between your elements, your browser will assume that the element is a child of the previous one.

```
header nav ul li a {  
    text-decoration: none;  
}
```

## class selectors

Class selectors target an element with a specified **class** applied to it. In the example below, only paragraphs with a class called 'intro' will be bolded.

```
p.intro {  
    font-weight: 800;  
}
```



## pseudo-class selectors

Pseudo-class selectors will affect an element only when a certain **condition has been met**. For example, this condition can be something that the user does.

```
a:hover {  
    text-decoration: underline;  
}
```

## nth-child( ) pseudo-class selector

The nth-child pseudo-class selector targets an element based upon the **order** that it's in. In the example below, the third list item will turn blue.

```
li:nth-child(3) {  
    color: blue;  
}
```

# The Cascade

I really feel for students who haven't had me before and aren't prepared for the barrage of dad jokes.

What is the **cascade** part of Cascading Style Sheets?

Some designers like to think of it like a waterfall, where everything at the bottom takes precedence.

The cascade means that when there are **conflicting rules**, the rule written last will be rendered last.

The **last rule written** is the **last rule standing**.

However, the **specificity** of a selector can override the cascade.

(weakest)

element selector

p, h1, h2 ...

descendant selector

ul li a

class selector

p.intro

id selector

#jumbotron

(strongest)

!important

p { color: red !important; }

I like to think of all of this as a  
Battle Royale.