CSS 1

Programming with Web Technologies

THE UNIVERSITY OF AUCKLAND
NEW ZEALAND
THE UNIVERSITY OF WAIKATO
Te Whare Wānanga o Waikato
Auckland ICT Graduate School

CSS (Cascading Style Sheets)

CSS is a language for describing presentational characteristics of elements in a document (commonly an HTML web page)

The idea is to separate description of document content/structure from description of layout/style. HTML code should define content and structure, while CSS code should define layout and style

A file containing CSS code is commonly referred to as a **stylesheet**

CSS

3 main ways of using CSS:

- Inline
  - Using a style attribute
- Internal
  - In the head of the document
  - Using class attribute and element selectors
- External
  - In a separate CSS file
  - Using class attribute and element selectors

Inline CSS

Uses the HTML **attribute** `style`. Contains semicolon separated `property:value` pairs within double quotes

```
<tag style="property: value [; property: value]*"></tag>
```

For example:

```
<p style="font-family: sans-serif; ">text</p>
<p style="font-family: sans-serif; font-size: 12pt">text</p>
```

Internal CSS

Uses the HTML **tag** <style>, which appears in the head of the document. A selector is inserted that determines what elements will be affected, then a block of property:value pairs is placed inside braces

```
<style type="text/css">
  selector {
      property: value; [property: value;]*
  }
  ...
</style>
```

External CSS

Uses the HTML **tag** <link>, which appears in the head of the document. This links to an external file that only contains CSS and includes it in your document

```
<link href="mystyle.css" rel="stylesheet" type="text/css">
```

The `href` can be any valid local, relative or external URL. When we are linking to a CSS file, the `rel`ation will always be `stylesheet`, and the `type` will always be `text/css`

All Together

```
<!doctype html>
  <html lang="en">
  <head>
    <meta charset="utf-8">
    <link href="mystyle.css" rel="stylesheet" type="text/css>
    <style type="text/css>
      #custom { color: green; border: 2px solid black; }
      a:visited { color: blue; }
    </style>
  <body>
    <p style="font-family: serif">Some content here</p>
    <a href="another.html">Some cool link</a>
    <div id="custom">
      <div id="external_target"></div>
    </div>
  </body>
</html>
```

Benefits of Each Approach

Inline

- Quick changes, useful for simple once-off styles
- No extra files or tags

Internal

- Removes duplication within pages
- Allows for styles that apply to multiple elements

External

- Removes duplication across pages
- Allows for caching, stylesheet only needs to be downloaded once

Basic Styling

As with tags in HTML, there are a huge number of CSS properties, and it would not be practical to cover them all

There are however, a number of common properties that you will use regularly that we will look at

- Colors
- Borders
- Fonts

CSS Colors

Colors can be represented in a number of ways using CSS

Named colors use common color names such as black, red, goldenrod, orangered, etc to represent colors. There are 140 of these that are supported

RGB Hexadecimal allows you to specify Red, Green and Blue 'channels' as Hexadecimal numbers, prefixed by a #. `#DAA520` = Goldenrod

RGB or HSL **color functions** allow colors to be calculated within RGB or HSL spaces. `hsl(296, 59%, 28%)`, `rgba(120, 55, 60, 0.3)`

Using Colors

Colors can be used in a number of different places, but the most common uses are setting the foreground and background colors of an element

The `color` property sets the foreground (text) color

```
color: goldenrod;
color: rgb(255, 255, 60);
```

The `background-color` property sets the background color

```
background-color: #487;
background-color: hsl(122, 75%, 25%)
```

Borders

Borders surround an element on all 4 sides, and can be set in a variety of colors, styles and thicknesses. These properties can be set individually, or at the same time; for all 4 sides at once, or for one side at a time

`border-style` is the most important border property, as by default it is set to `none`, so no other border changes will show up

```
border-style: solid;
border-left-style: dashed;
border-bottom-style: inset;
```

Borders

border-color is used for setting the color of the border. You can use any of the color representations shown earlier

```
border-color: green;
border-right-color: #5F6
```

border-width is used for setting the width of the border

```
border-width: 5px;
border-top-width: 17px;
```

Borders

If you were to set all of your borders at the same time using the `border-*` properties, you would need 3 lines of CSS. If you were to set each side individually with the `border-direction-*` properties, you would need 12 lines

The `border` property allows us to set the width, style and color of all borders, or a single border, in a 1 line

```
border: 15px solid red;
border-left: 2px inset green;
```

Fonts

Fonts can have a number of aspects of their appearances adjusted with CSS, including the size, weight and style of the font, as well as the font face itself

The five properties used to control fonts are:

`font-family` – The “font” to be used
`font-size` – The size of the font
`font-style` – Settings like italics
`font-weight` – Settings like bold
`font-variant` – Settings like drop-caps

Font Family

The `font-family` property is used to select the font face. There are 5 basic options that are provided by browsers, but you can also specify fonts that may be installed on the client system, or that are defined as web fonts

The 5 basic options are shown here

The `font-family` property can specify several comma separated fonts, known as a font stack, that is evaluated left to right until a suitable font that the client has is found

```
font-family: Times new roman, Georgia, serif;
```

Other Font Options

```
font-size: [medium | xx-small | x-small | small | large |
x-large | xx-large | smaller | larger | number];

font-style: [normal | italic | oblique];

font-weight: [normal | bold | bolder | lighter | number];

font-variant: [normal | small-caps];
```

Web Fonts

For most web applications, the 5 basic fonts are not enough, and more variety is needed. However, other fonts can't be guaranteed to work as the client may not support them

Having some users experience a site in a different way does not present a consistent user experience and should be avoided

Prior to HTML5 and CSS3, using custom fonts was difficult. Now however, it can easily be done in a few lines of CSS

Web Fonts

The `@font-face` rule allows for a new font to be defined. This rule will contain a `font-family` property that gives a name to the new font, and a `src` property that provides a link to where the font file can be found

```
@font-face {
    font-family: 'Fontasia';
    src: url('/path/to/Fontasia.ttf');
}
```

Once declared with font-face, you can use your new font in a font stack using the name provided in the `font-family` property (Fontasia in this case)

CSS Selectors

When defining CSS rule in an internal or external stylesheet, we need some way of indicating which attributes should be styled. We can do this using a selector. CSS3 supports many different types of selector, a few of which we will look at today.

When writing rule in a stylesheet, we write them like this

```
selector {
    property: value;
    ...
}
```

Type Selector

A type selector matches all elements of a named HTML tag. The selector consists simply of the name of the HTML tag to which it should apply

```
h1 { color: blue; }
```
All `h1` elements will have blue text

```
td { font-weight: bold; }
```
Text inside all `td` elements will be bold

This is useful for setting default properties for tags, as it will apply to all tags of that type

ID and Class Attributes

Tags in HTML can contain many different attributes, a number of which we have seen so far in this course. There are 2 attributes that nearly all tags support: `id` and `class`

Recall from HTML2, `id` defined a unique identifier that could be used as a target for anchor bookmarks. The uniqueness of `id` is important and can be used for more than just anchors

The `class` attribute is similar, but does not need to be unique, and can contain more than one value separated by spaces

Class Selectors

A `class` selector matches all HTML tags that have a value in their `class` attribute that matches that specified in the selector. To indicate we are referring to a class, we prefix the class name with a period

```
.correct { background-color: green; }
```
All elements with the class "correct" will have a green background

```
.important { border-style: solid; }
```
All elements that have "important" in their class attributes will have solid borders

ID Selectors

An `id` selector matches the HTML element that has the same value in its `id` attribute as in the selector. To indicate we are referring to an id, we prefix the id name with a hash

```
#reset { font-family: serif; }
```

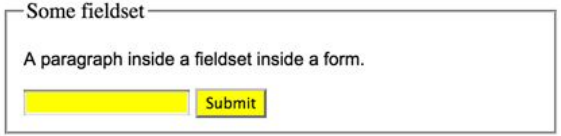
The element with the id "reset" will use a serifed font for any text

Attribute Selectors

We know that HTML elements can have attributes, and since this information is present in our markup it makes sense that we can use these values as part of selectors

An example of where this is needed would be when styling text input boxes. Recall that text boxes are <input> tags, but so are checkboxes, radio buttons etc. This means that the following rule would change all <input> elements

```
input { background-color: yellow; }
```



Attribute Selectors

Attribute selectors allow us to add information to type selectors so that only tags that match that type **and** have particular attributes will be affected

Attribute selectors take the form

```
tag[attribute="value"] {}
```

So for the previous example, we could write the following


```
input[type="text"] { background-color: yellow; }
```

Document Object Model

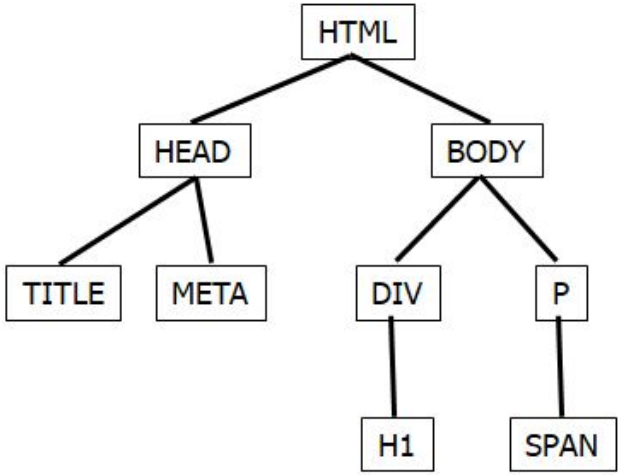
The Document Object Model (DOM) is a tree like representation of a web page. When a web browser loads a web page, it parses the HTML that makes up a page, and constructs a corresponding representation in the form of an internal tree data structure

There is a little more to it than this, but we will discuss it in more detail in the future

This important thing to take away from this, is that this DOM tree can be used by CSS and JavaScript code to identify and manipulate elements

DOM - Example Tree

```
<!doctype html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>
      Inline CSS
    </title>
  </head>
  <body>
    <div>
      <h1>heading</h1>
    </div>
    <p>A <span>new</span> paragraph.</p>
  </body>
</html>
```



DIV is a **descendant** of BODY and HTML
DIV is a **child** of BODY
DIV is the **first-child** of BODY
DIV is a **parent** of H1
DIV and P are **siblings**

Child Selector

Using what we know about the DOM and the relationships that elements have within the tree, we can come up with selectors that match elements based on their positions in the tree relative to another

The first of these is the child selector, which specifies a parent and child element to match, separated by a right-tag

```
thead > td { color: orange; }
```

Will match any `td` elements that are direct children (ie, one level below) of a `thead` element.

Descendant Selector

The descendant selector is similar to the child selector, but it matches elements that are anywhere below the matched element in the DOM tree, rather than just first level children. The syntax is similar too, only a space is used rather than a right tag to separate the ascendant and descendant

```
.article p { background-color: red; }
```

Will match any `p` element that is below elements matched by the class name `article` in the DOM tree

Grouping Selectors

When programming, we seek to reduce duplication wherever possible - including in CSS. To this end, CSS allows us to specify multiple comma-separated selectors for one block of CSS

```
p, input[type="text"], .abc, #unique {
    border: 1px inset orangered;
}
```

All paragraphs, text inputs, elements with class `abc` and the element with id `unique` will have a 1 pixel wide, inset orangered border

Divs and Spans

Most tags have some form of default styling, with the notable exceptions of `div` and `span`. Because they have no default appearance, this makes them prime candidates for assigning styles via `class` and `id` selectors

`div` is a block level element, and can be used to group together other block level elements and apply a style to multiple items in one go

`span` is an inline element, and can be used to style either text, or other inline elements

CSS Box Model

In HTML, every element is created inside of a box. That box may be contained inside another box, or may contain boxes within itself. The browser places these boxes appropriately on the page

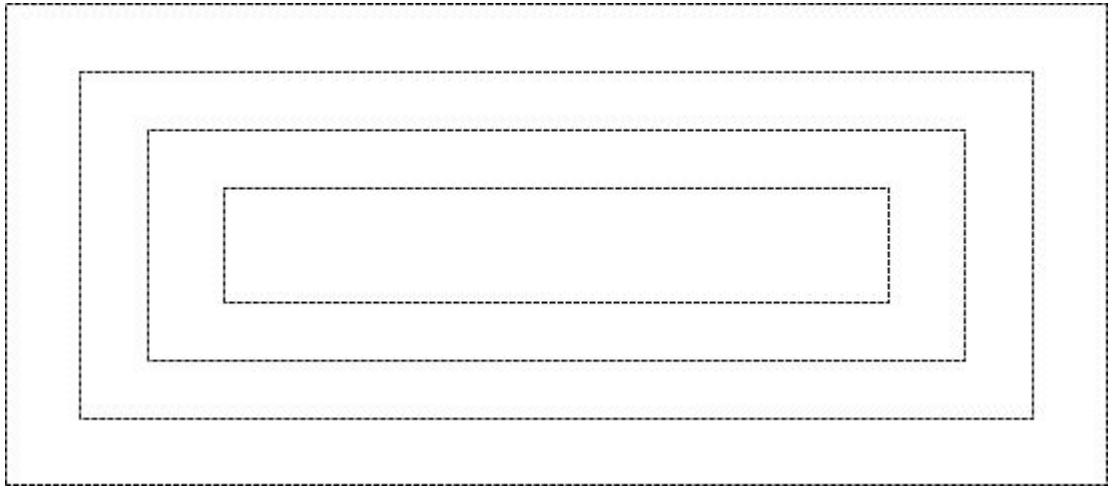
This can be tricky to visualize, so you can force a web page to reveal its boxes by inserting a new CSS rule into a page using the inspection tool

```
* { border: 1px solid black; }
```

This will temporarily force the page to show borders around all elements

CSS Box Model

The box for each individual element has this structure

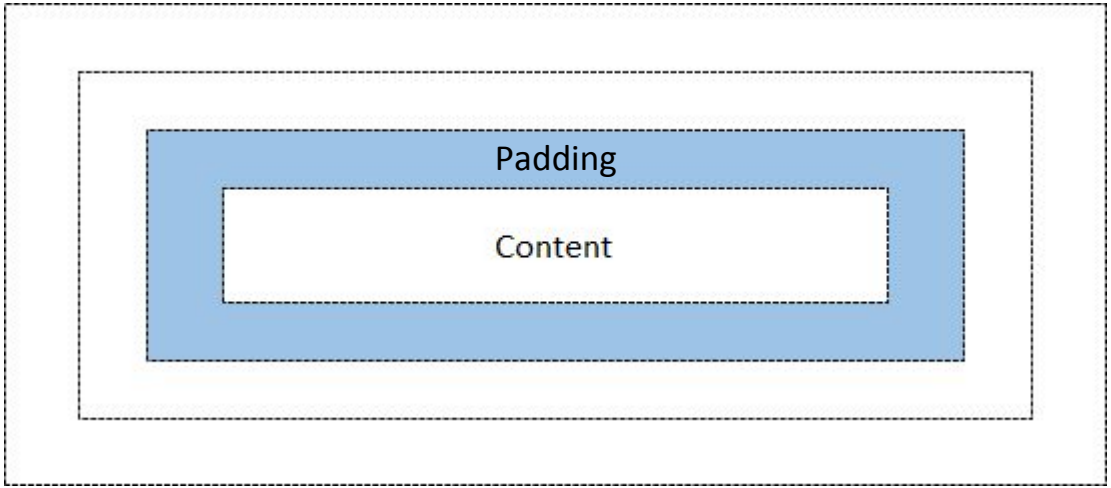


CSS Box Model

In the center is content. This will be the text, image, input control, etc that makes up this element

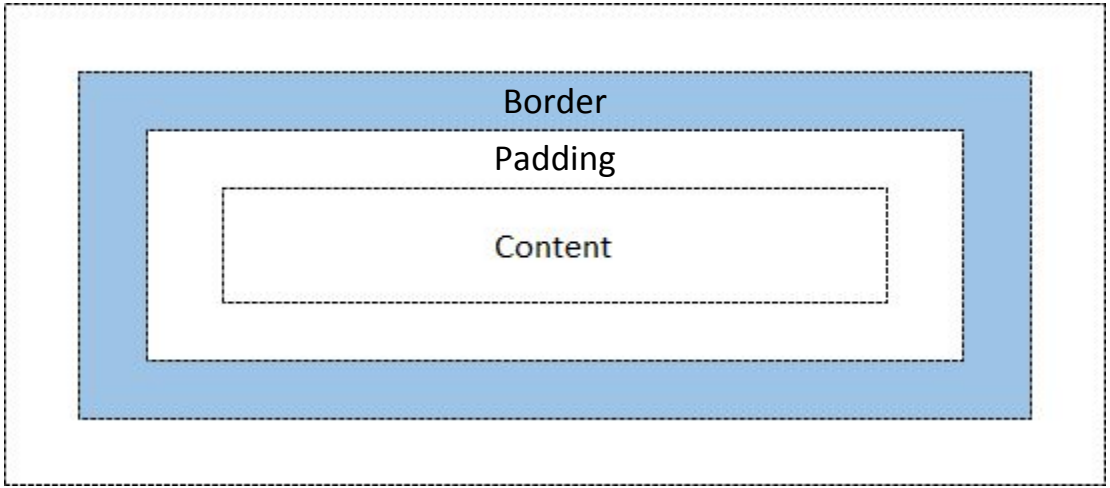
CSS Box Model

Around the content is the padding. This is space inside the element, between the content and the border. This will be the same color as the `background-color` of the element



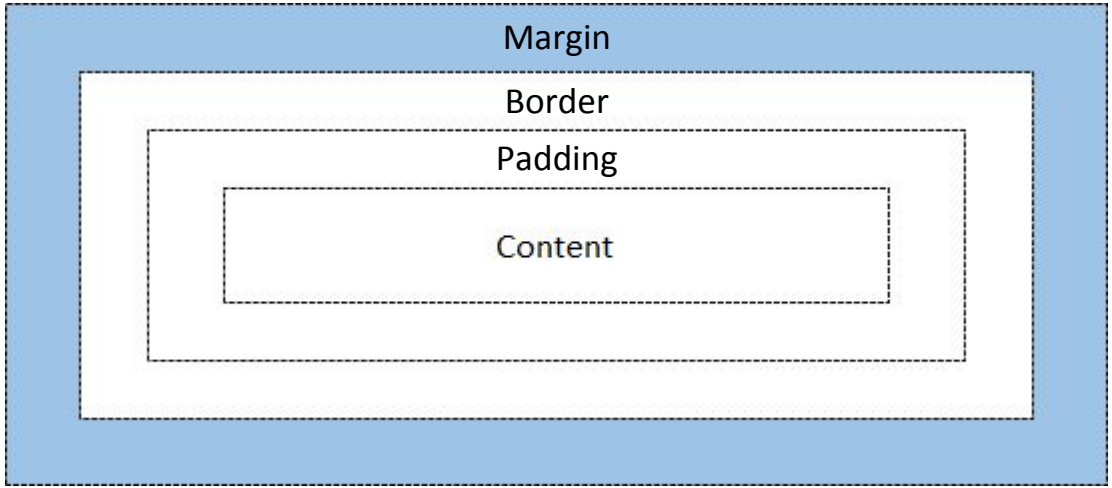
CSS Box Model

Next is the border. This surrounds the content and the padding. The color of the border is defined by the `border-color` property



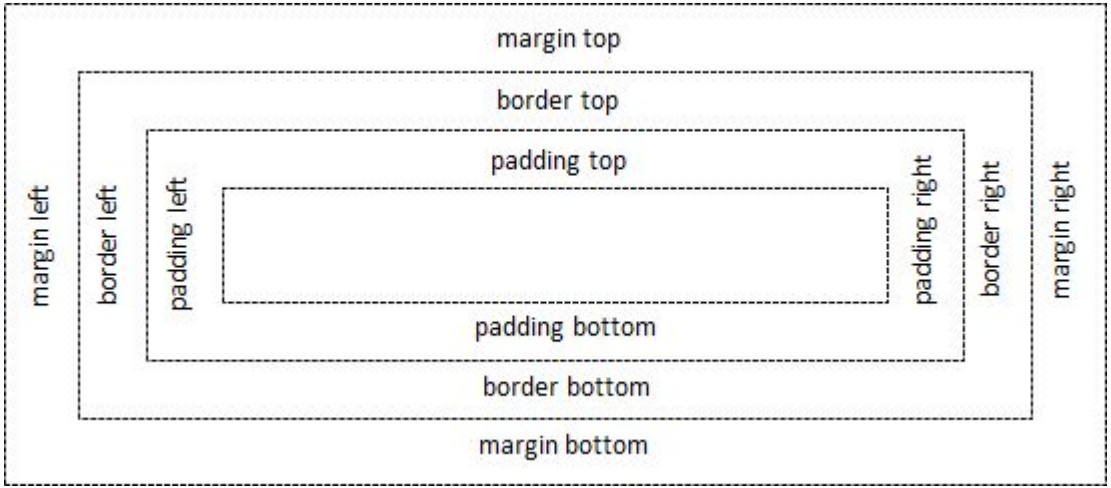
CSS Box Model

Finally, the margin is outside the border. The margin is always transparent



CSS Box Model

Just like with the border properties, the width of the margins and padding can be set either for all sides at once, or once side at a time.



Visualizing the Box Model

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut tristique, ex nec interdum sollicitudin, nunc eros lobortis leo, eu scelerisque diam metus eget ante. Quisque at nulla tempus, eleifend tortor ac, laoreet purus. Nam venenatis feugiat nisi, ut euismod quam lobortis ut. Proin quis maximus odio. Duis condimentum ultricies tristique. Sed tempus arcu eget sem porta, venenatis imperdiet ex pellentesque. Pellentesque suscipit rhoncus ante, sed dignissim urna dapibus sit amet. Interdum et malesuada fames ac ante ipsum primis in faucibus. Phasellus scelerisque dui at nulla

```
#img1 {
  background-color: green;
  padding         : 20px;
  border-color    : black;
  border-width    : 20px;
  border-style    : solid;
  margin          : 20px;
}
```

auctor, ac tempus sapien dignissim. Curabitur tincidunt metus in augue accumsan ullamcorper. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Curabitur vehicula tincidunt euismod. Cras feugiat mi at nisi semper, eu mollis quam porttitor. Aliquam erat volutpat. Suspendisse vitae leo in justo fermentum egestas. Sed ut urna maximus, maximus urna at, ultrices lectus. Maecenas ullamcorper, nunc sit amet pharetra aliquet, augue nunc hendrerit diam, vel tempor nisl turpis dapibus erat. Nam bibendum, nulla eleifend ultricies volutpat, mauris augue aliquet nulla, sit amet bibendum sem tellus sed erat. Donec euismod nisi vitae sem euismod, at tempor ipsum tincidunt. Sed scelerisque, elit ut ornare venenatis, nisi metus egestas arcu, a mattis risus erat ut augue. Nunc placerat convallis mauris eu convallis. Sed aliquam sapien quis lectus facilisis, eu porttitor est semper.

Visualizing the Box Model

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut tristique, ex nec interdum sollicitudin, nunc eros lobortis leo, eu scelerisque diam metus eget ante. Quisque at nulla tempus, eleifend tortor ac, laoreet purus. Nam venenatis feugiat nisi, ut euismod quam lobortis ut. Proin quis maximus odio. Duis condimentum ultricies tristique. Sed tempus arcu eget sem porta, venenatis imperdiet ex pellentesque. Pellentesque suscipit rhoncus ante, sed dignissim urna dapibus sit amet. Interdum et malesuada fames ac ante ipsum primis in faucibus. Phasellus scelerisque dui at nulla

```
#img1 {
  background-color: green;
  padding      : 20px 40px 20px 70px;
  border-color: black;
  border-width: 20px 50px 10px 0px;
  border-style: solid;
  margin       : 10px 80px 60px 30px;
}
```

auctor, ac tempus sapien dignissim. Curabitur tincidunt metus in augue accumsan ullamcorper. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Curabitur vehicula tincidunt euismod. Cras feugiat mi at nisi semper, eu mollis quam porttitor. Aliquam erat volutpat. Suspendisse vitae leo in justo fermentum egestas. Sed ut urna maximus, maximus urna at, ultrices lectus. Maecenas ullamcorper, nunc sit amet pharetra aliquet, augue nunc hendrerit diam, vel tempor nisl turpis dapibus erat. Nam bibendum, nulla eleifend ultricies volutpat, mauris augue aliquet nulla, sit amet bibendum sem tellus sed erat. Donec euismod nisi vitae sem euismod, at tempor ipsum tincidunt. Sed scelerisque, elit ut ornare venenatis, nisi metus egestas arcu, a mattis risus erat ut augue. Nunc placerat convallis mauris eu convallis. Sed aliquam sapien quis lectus facilisis, eu porttitor est semper.

CSS Units

Absolute units are useful assuming you know the characteristics of the device you are displaying on

in	Inches
cm	Centimetres
mm	Millimetres
pt	Points (1/72nd of an inch)
pc	Picas (1/6th of an inch, 12 points)

CSS Units

Relative units are useful if you want your elements to be sized relative to some other length property in the document. Relative units scale better than absolute units when displaying across different devices

px	Pixels. Depends upon the resolution of the viewing device
em	Relative to the font-size of the element (2em means 2 times the size of the current font)
ex	x-height, relative to the size of a lower-case "x" in the font being used

References

- https://www.w3schools.com/colors/
- https://www.w3schools.com/css/css_font.asp
- https://www.w3schools.com/cssref/css_selectors.asp
- https://www.w3schools.com/js/js_htmldom.asp
- https://www.w3schools.com/css/css_combinators.asp
- https://www.w3schools.com/tags/tag_div.asp
- https://www.w3schools.com/css/css_boxmodel.asp
- https://www.w3schools.com/CSSref/css_units.asp