HTML & Basic CSS

Web Development & Consulting Club
Workshop One, Session One

THE UNIVERSITY OF AUCKLAND
Te Whare Wānanga o Tāmaki Makaurau
NEW ZEALAND

THE UNIVERSITY OF WAIKATO
Te Whare Wānanga o Waikato

Auckland
ICT Graduate School

HTML

- **H**yper**T**ext **M**arkup **L**anguage, used to describe the content and structure of documents on the web (web pages)
- The first version of HTML
  - text-only documents
  - define parts of a document content semantically (e.g. a title, paragraphs, headings, lists)
  - define links as anchors
- Now it is more complex – more than just text in web pages
  - more complex document structures
  - enhanced user interaction with pages

HTML

- An HTML document consists of two types of information
  - The content of the document
  - Markup that describes the content
- Markup is in the form of element tags
  - Content is (generally) surrounded by a start tag and an end tag
  - A tag is a letter or word, surrounded by angle brackets
    - `<b>`, `<body>`, `<table>`
  - A closing tag additionally has a forward-slash after the first angle bracket
    - `</b>`, `</body>`, `</table>`
  - Excepts for a few exceptions which will be discussed as needed, opening tags should have a corresponding closing tag

HTML Example

A simple HTML document

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>
            My First HTML page
        </title>
    </head>
    <body>
        <p>This is the content of my first HTML page</p>
    </body>
</html>
```

HTML Example

A simple HTML document

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>
            My First HTML page
        </title>
    </head>
    <body>
        <p>This is the content of my first HTML page</p>
    </body>
</html>
```

The <!DOCTYPE> declaration is the very first thing in the HTML document. It is an instruction to the web browser about what version of HTML the page was written in

This tag does not require a matching closing tag

HTML Example

A simple HTML document

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>
            My First HTML page
        </title>
    </head>
    <body>
        <p>This is the content of my first HTML page</p>
    </body>
</html>
```

The content of the HTML document needs to be enclosed by the `html` tag. All other tags will be nested inside this

HTML Example

A simple HTML document

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>
            My First HTML page
        </title>
    </head>
    <body>
        <p>This is the content of my first HTML page</p>
    </body>
</html>
```

A container for all informative tags. Tags inside the head of the document do not appear directly on the page, but can influence how elements are displayed

HTML Example

A simple HTML document

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>
            My First HTML page
        </title>
    </head>
    <body>
        <p>This is the content of my first HTML page</p>
    </body>
</html>
```

`meta` tags contain information about the HTML document. This can include a description of the page, the author and keywords. This information can be used by search engines to help index your page and by the browser to determine how to display the content

This tag does not require a matching closing tag

HTML Example

A simple HTML document

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>
            My First HTML page
        </title>
    </head>
    <body>
        <p>This is the content of my first HTML page</p>
    </body>
</html>
```

Start and End point of the text that is the title of the document. This text will appear as the tab and window name when the page is loaded in a browser

HTML Example

A simple HTML document

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>
            My First HTML page
        </title>
    </head>
    <body>
        <p>This is the content of my first HTML page</p>
    </body>
</html>
```

A container for the tags that make up the document. Tags inside the body of the document appear directly on the page

HTML Example

A simple HTML document

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>
            My First HTML page
        </title>
    </head>
    <body>
        <p>This is the content of my first HTML page</p>
    </body>
</html>
```

← A paragraph of text

HTML Element Types

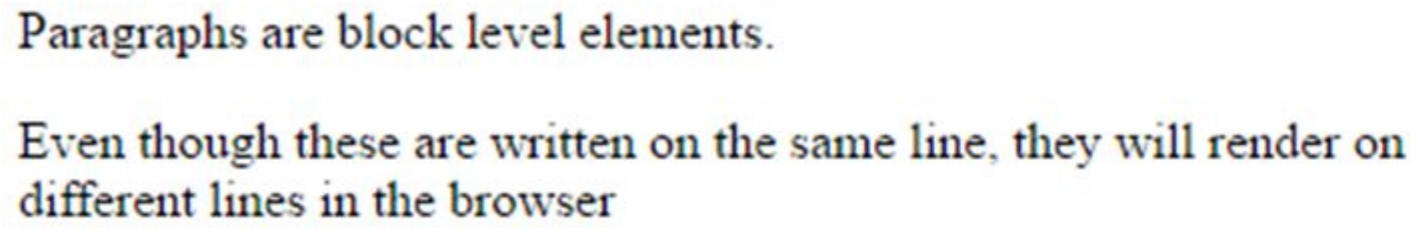
There are two types of elements which can be used for content in HTML

- Block level elements
    - Always starts on a new line
    - Expands to fill the available width (stretches out to the left and right as far as it can)

- Inline elements
    - Does not start on a new line
    - Only takes up as much width as necessary

HTML Elements - Block Level

```
<p>Paragraphs are block level elements.</p><p>Even though
these are written on the same line, they will render on
different lines in the browser</p>
```

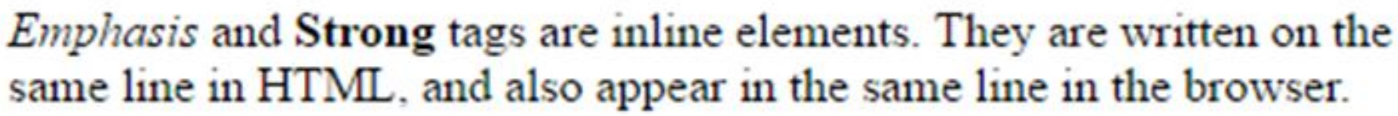


Paragraphs are block level elements.

Even though these are written on the same line, they will render on different lines in the browser

HTML Elements - Inline

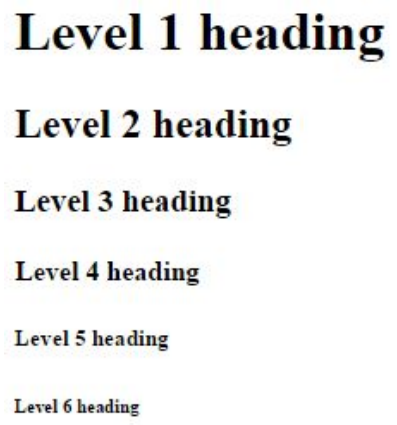
```
<p><em>Emphasis</em> and <strong>Strong</strong> tags are
inline elements. They are written on the same line in HTML,
and also appear in the same line in the browser.</p>
```


Emphasis and **Strong** tags are inline elements. They are written on the same line in HTML, and also appear in the same line in the browser.

Headings

- HTML supports six levels of heading, which can be used to organize documents into sections
- <h1> is the most important, and <h6> is the least important
- Search engines and other tools using heading importance to index web pages, so use them appropriately

```
<h1>Level 1 heading</h1>
<h2>Level 2 heading</h2>
<h3>Level 3 heading</h3>
<h4>Level 4 heading</h4>
<h5>Level 5 heading</h5>
<h6>Level 6 heading</h6>
```


Level 1 heading
Level 2 heading
Level 3 heading
Level 4 heading
Level 5 heading
Level 6 heading

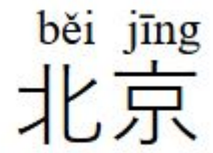
Text in HTML

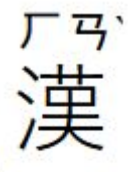
Text written in the `<body>` of an HTML document will be rendered into the browser directly, however it is good practice to surround text with appropriate tags to convey meaning/intent

- `<p>` - Indicates the text enclosed is a paragraph. Default styling will force a line break between paragraphs
- `<pre>` - Preformatted text. Text will be displayed in the browser exactly as it is typed in the HTML, with whitespace preserved
- `<code>` - Source code, by default displayed using a monospaced font

Text in HTML

- `<blockquote>` - A blockquote originating from a different source, by default displayed using a monospaced font
- `<q>` - A short quote, displayed inline
- `<cite>` - An inline citation of the title of an article, movie, book, etc
- `<address>` - An address, displayed in italics by default
- `<ruby>` - A ruby annotation, a pronunciation guide for Japanese or Chinese text to aid pronunciation of unfamiliar words





And many more, ranging from general to extremely special cases

Lists

HTML5 supports several varieties of list

- Unordered lists
  - Bullet points. Used where order doesn’t matter
- Ordered lists
  - Numeric list. Used where order matters
- Description lists
  - Term, Description pairs. Used when you want to define or describe terms

Unordered List

HTML Markup

```
<ul>
  <li>Entry 1</li>
  <li>Entry 2</li>
  <li>Entry A</li>
  <li>Entry B</li>
</ul>
```

Rendered Result

- Entry 1
- Entry 2
- Entry A
- Entry B

Ordered List

HTML Markup

```
<ol>
  <li>Entry 1</li>
  <li>Entry 2</li>
  <li>Entry A</li>
  <li>Entry B</li>
</ol>
```

Rendered Result

1. Entry 1
2. Entry 2
3. Entry A
4. Entry B

More useful tags

- `<hr>` - Indicates a "Thematic break" in the document content, often shown as a horizontal line across the page. Does not need a closing tag
- `<br>` - Forces a newline in some text, which is often used in representing addresses. Does not need a closing tag
- `<!--    -->` - A comment. Anything written between the `<!--` and `-->` markers will not be displayed on the page

HTML Entities

- Some characters have special meaning in HTML; such as angle brackets and quotes; or can't be typed with a typical keyboard
- If we need to display one of these, we can use HTML entities

Result	Description	Entity Name
	Non-breaking space	`&nbsp;`
>	Greater than	`&gt;`
<	Less than	`&lt;`
&	Ampersand	`&amp;`

Result	Description	Entity Name
"	Double quote	`&quot;`
'	Single quote	`&apos;`
©	Copyright Symbol	`&copy;`
™	Trademark Symbol	`&trade;`

More entities here

Tables

- Often we are given data that makes sense to present in a tabular fashion
- Tables are very flexible and have been used for page layout in HTML document for many years
  - No longer recommended as it distracts from the purpose of a table – displaying tabular data
- Trend with HTML5 is to avoid structures like tables for layout, and instead leave that to CSS
  - More on this in future lectures
- Tables make use of a collection of HTML tags

Tables

- `<table>` - Defines an HTML table, and groups the components of the table together
- `<tr>` - A row in the table that contains a number of cells
- `<td>` - A data cell in the table
- `<th>` - A header cell in the table
- `<thead>` - Table header, represents the top-most row of a table
- `<tbody>` - Table body, the main content of a table
- `<tfoot>` - Table footer, represents the bottom-most row of a table
- `<caption>` - A caption to attach or associate with the table, must appear immediately after the opening `<table>` tag

A basic table

```html
<table>
    <tr>
        <td>col1row1</td>
        <td>col2row1</td>
        <td>col3row1</td>
        <td>col4row1</td>
    </tr>
    <tr>
        <td>col1row2</td>
        <td>col2row2</td>
        <td>col3row2</td>
        <td>col4row2</td>
    </tr>
</table>
```

col1row1	col2row1	col3row1	col4row1
col1row2	col2row2	col3row2	col4row2

Basic table observations

The table will have as many rows as there are `<tr>` elements

The table will have as many columns as the largest number of `<td>` elements inside a `<tr>`

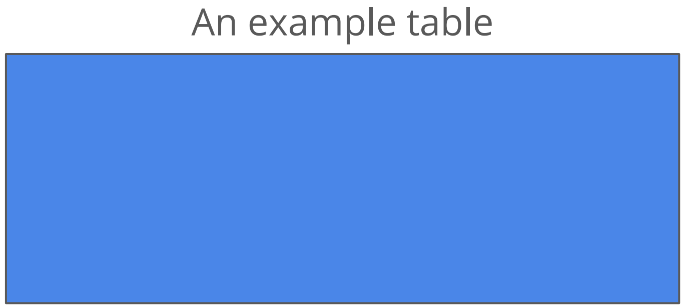
- If one row has five cells, and any others have 3 cells each, the table will have five columns
- Cells are placed left-to-right

Intermediate table

```
<table>



</table>
```

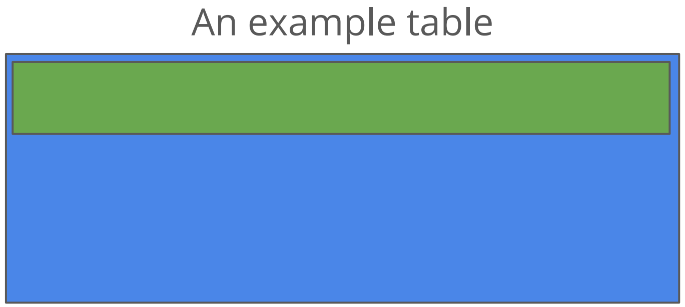
Intermediate table

```
<table>
    <caption>An example table</caption>







</table>
```



Intermediate table

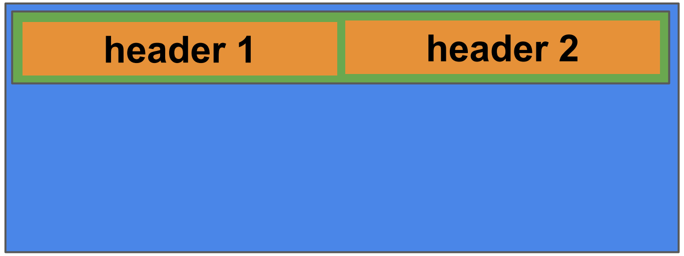
```
<table>
    <caption>An example table</caption>
    <thead>
        <tr>


        </tr>
    </thead>









</table>
```



Intermediate table

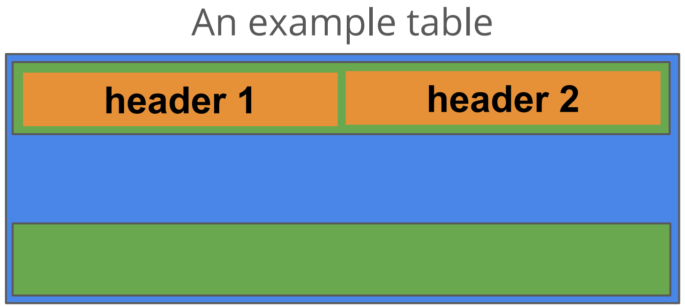
```html
<table>
    <caption>An example table</caption>
    <thead>
        <tr>
            <th>header 1</th>
            <th>header 2</th>
        </tr>
    </thead>









</table>
```

An example table



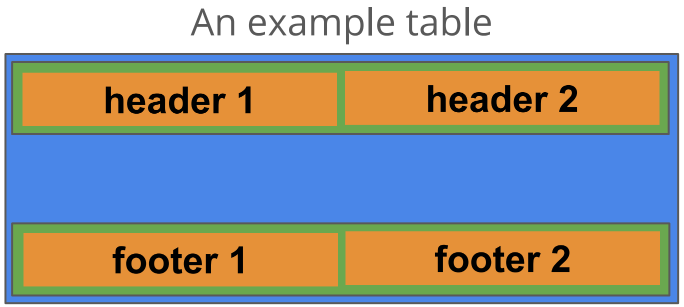
Intermediate table

```
<table>
    <caption>An example table</caption>
    <thead>
        <tr>
            <th>header 1</th>
            <th>header 2</th>
        </tr>
    </thead>
    <tfoot>
        <tr>


        </tr>
    </tfoot>





</table>
```



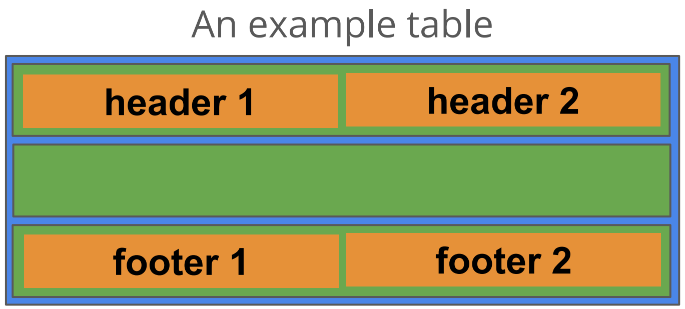
Intermediate table

```
<table>
    <caption>An example table</caption>
    <thead>
        <tr>
            <th>header 1</th>
            <th>header 2</th>
        </tr>
    </thead>
    <tfoot>
        <tr>
            <td>footer 1</td>
            <td>footer 2</td>
        </tr>
    </tfoot>






</table>
```



Intermediate table

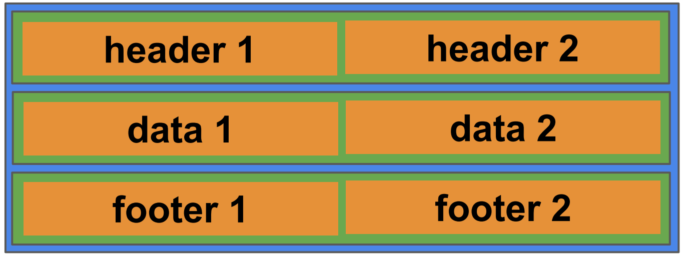
```
<table>
    <caption>An example table</caption>
    <thead>
        <tr>
            <th>header 1</th>
            <th>header 2</th>
        </tr>
    </thead>
    <tfoot>
        <tr>
            <td>footer 1</td>
            <td>footer 2</td>
        </tr>
    </tfoot>
    <tbody>
        <tr>


        </tr>
    </tbody>
</table>
```



Intermediate table

```
<table>
    <caption>An example table</caption>
    <thead>
        <tr>
            <th>header 1</th>
            <th>header 2</th>
        </tr>
    </thead>
    <tfoot>
        <tr>
            <td>footer 1</td>
            <td>footer 2</td>
        </tr>
    </tfoot>
    <tbody>
        <tr>
            <td>data 1</td>
            <td>data 2</td>
        </tr>
    </tbody>
</table>
```

An example table



header 1	header 2
data 1	data 2
footer 1	footer 2

Intermediate table observations

- The order of `<tbody>`, `<thead>` and `<tfoot>` inside the `<table>` does not matter
  - `<thead>` will always be rendered as the top-most row in the table, above the `<tbody>`
  - `<tfoot>` will always be rendered as the bottom-most row in the table, below the `<tbody>`

- The order of `<tr>` elements does matter. They will be displayed in the order in which they are declared in their respective sections

HTML attributes

Attributes are name/value pairs that are seen inside the opening tags of HTML elements. We have seen some of these already

```
<html lang="en">

<meta charset="UTF-8">
```

All HTML elements can have attributes, some just have more than others

HTML attributes

Attributes are used to provide additional information about an element and come in 4 major varieties

- Required – The tag needs these attributes in order to function correctly
- Optional – The tag does not require these attributes to function, but useful functionality can be added using them
- Standard – Available to most tags, provide general functionality
- Event – Will discuss these in future lectures

HTML attributes

- Tables
  - `colspan="2"` – Makes a `<td>` span across multiple columns (merging cells horizontally)
  - `rowspan="2"` – Makes a `<td>` span across multiple rows (merging cells vertically)
- Ordered lists
  - `reversed` – Makes the list numbers count down rather than up (this attribute does not need a value)
  - `type="i"` – Pick the variety of marker to use in the list (roman numerals, alphabet, etc)
- A large variety of attributes exist, and many apply to specific tags
  - Many and difficult to remember options
  - To find the available attributes for a tag, use the MDN tag reference
    - Locate the tag you are interested in and check the "Attributes" section

Anchors

Also known as links, or hyperlinks, anchors are what allow us to navigate easily from page to page

Generally look something like this, but can be styled to look different

Can link to external pages, or to sections within the same page

Anchor tags

`<a>` - The anchor tag. Marks the content surrounded by the tags as a link. The content can be text or even an image.

Anchors have a required attribute `href` which indicates the address of the resource they link to.

```
<a href="http://www.google.com">Google</a>
```

Google

href attribute - external

Absolute Address

http://www.fakewebsite.com/res/2017.html

Equivalent to specifying a street address in long-form

130 Victoria St

Hamilton

New Zealand

Indicates exactly where to find the page and how to get there

href attribute - external

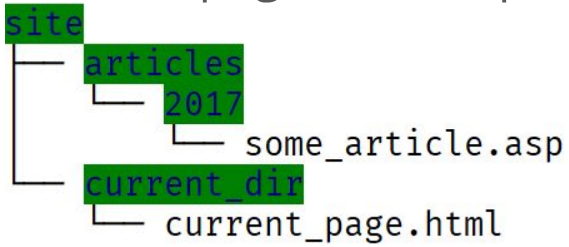
Relative Address

../articles/2017/some_article.asp

Equivalent to specifying a nearby street address

Go back up the road and take the next right, you will find it at number 17.

Indicates exactly where to find the page with respect to the current page



```
site
├── articles
│   └── 2017
│       └── some_article.asp
└── current_dir
    └── current_page.html
```

Images

`<img>` - The image element. This tag does not need to be closed

Images have a required attribute `src` which, like the `href` attribute used with anchors, indicates where the image can be found.

```
<img src="http://waikato.ac.nz/coa.gif">
```

THE UNIVERSITY OF
WAIKATO
Te Whare Wānanga o Waikato

Intro to CSS

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**

External CSS files

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`

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
- Margins & padding

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

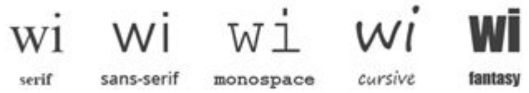
Fonts can have a number of aspects of their appearances adjusted with CSS, including the size, weight and style of the font, as well as the font face itself

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];
```

CSS Selectors

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`

`id` is 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

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

Child selector

We can come up with selectors that match elements based on their positions in the DOM tree (element hierarchy) 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. For example:

```
<div class=”article”><section><p>
```

Pseudo class selectors

In CSS1 we looked at a number of selectors, but all of them had one thing in common - they all related to information that was given in the HTML structure, such as the element type, id, class and attributes

But what about characteristics that aren't represented directly in the markup?

- A link that hasn't been visited
- A link that has been visited
- The mouse cursor is hovering over an element
- Whether an input has focus

Pseudo class selectors can be used to select elements in these special states

Pseudo class selectors

There are a large number of pseudo classes available, far too many to cover here, but a complete list can be found on MDN

Pseudo class selectors take the form

```
selector:pseudo-class { property: value; }
```

Where `selector` is any valid CSS selector, and `pseudo-class` is a pseudo class appropriate for the element selected

Pseudo class selectors

```
a:visited { }
```

Matches an anchor element that has been visited

```
div.important:hover { }
```

Matches a `div` element with the class important, that is being hovered over

```
input[type="text"]:focus { }
```

Matches a text `input` field that the client has clicked into

```
p:nth-child(5) { }
```

Matches all `p` elements that are the 5th child of their parent

CSS Box Model

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

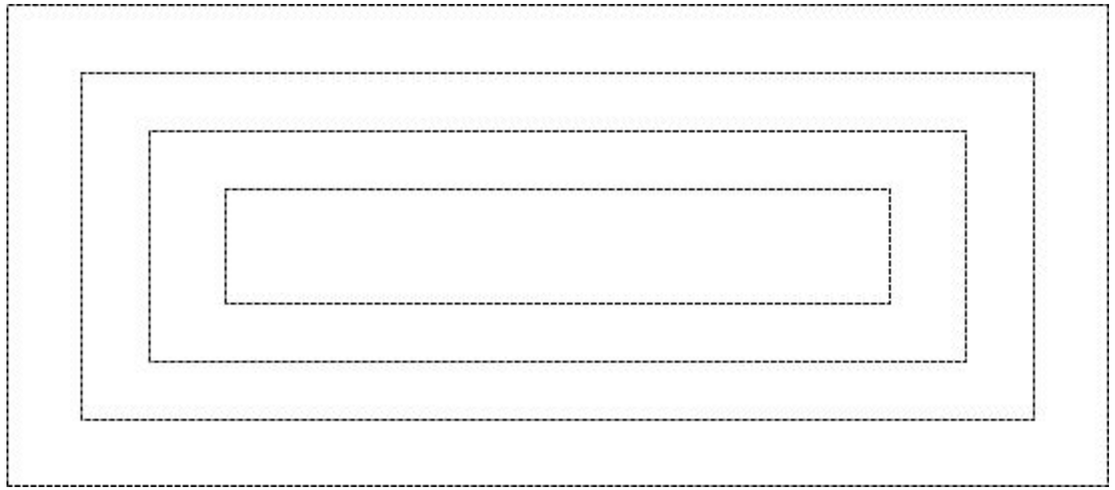
This can be tricky to visualize, so you can force a web page to reveal its boxes by inserting a new CSS rule into a page using the inspection tool

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

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

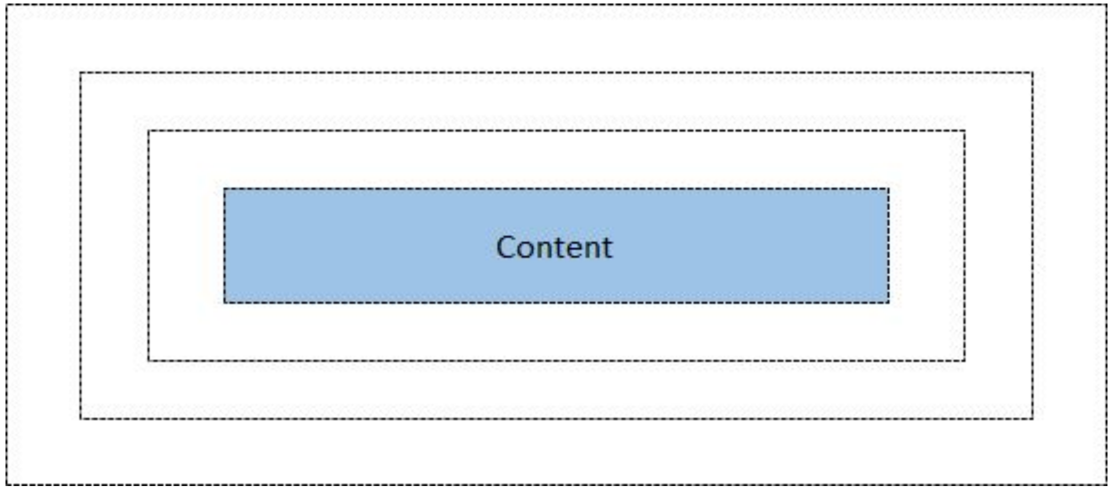
CSS box model

The box for each individual element has this structure



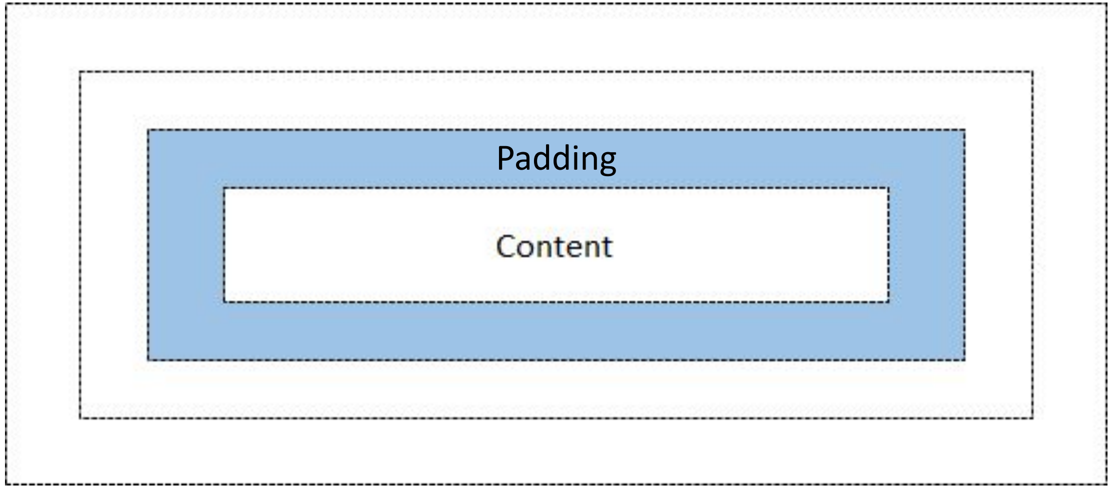
CSS box model

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



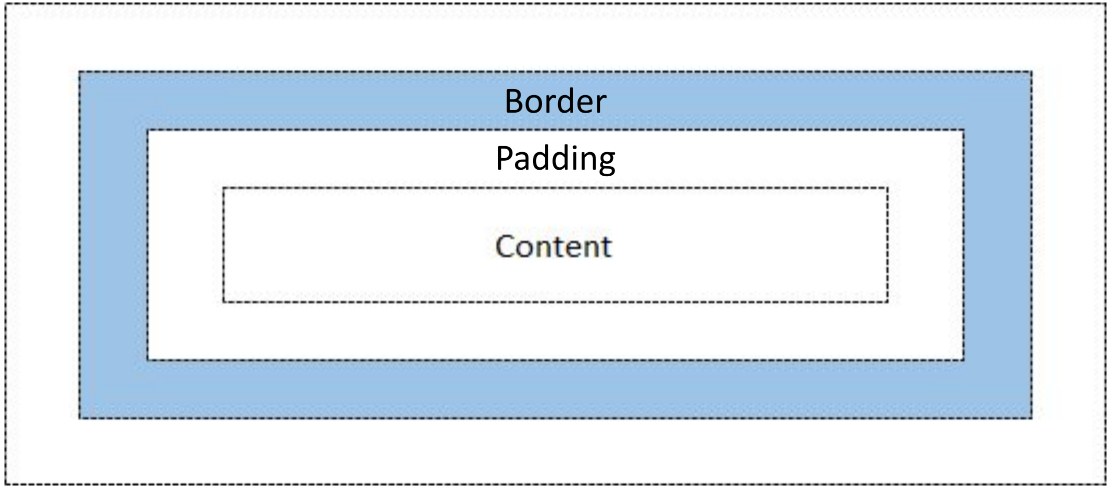
CSS box model

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



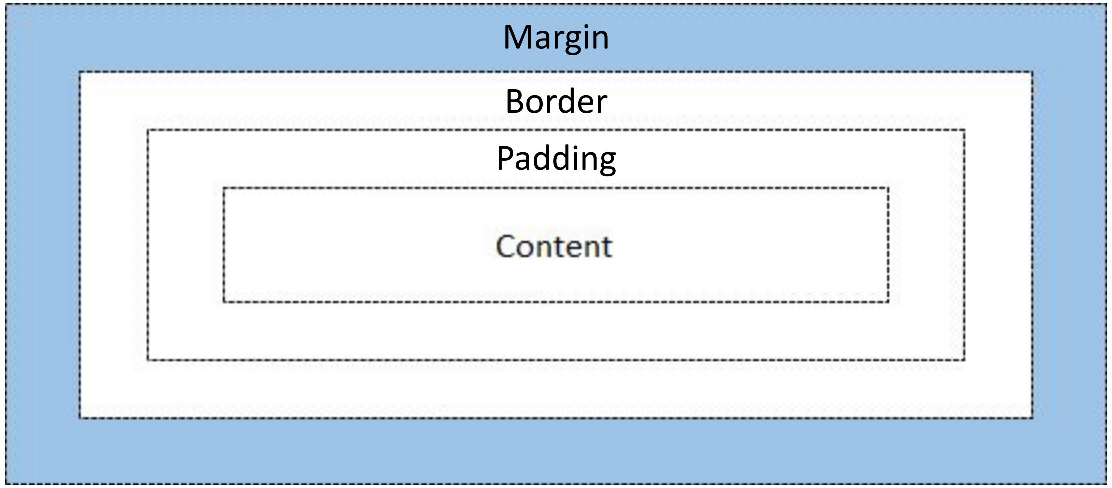
CSS box model

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



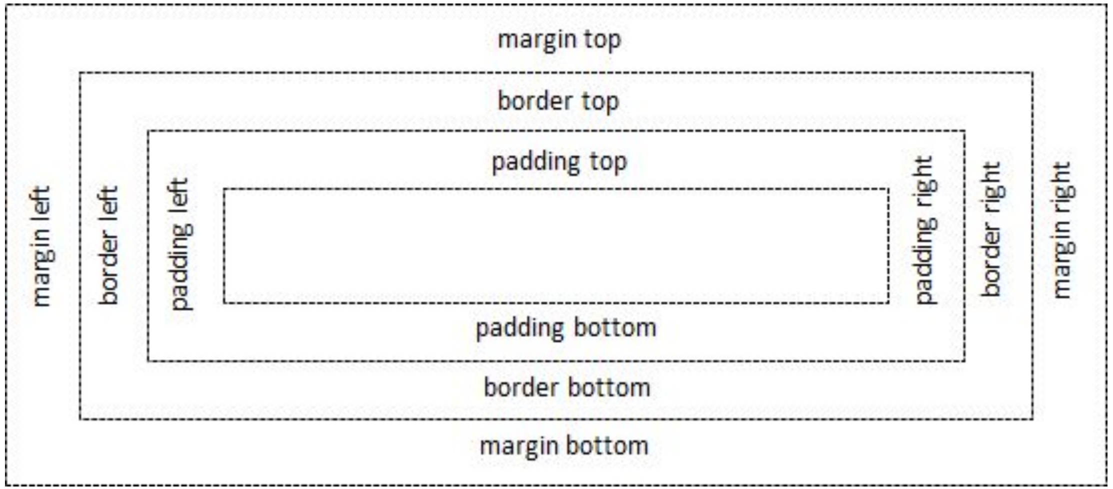
CSS box model

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



CSS box model

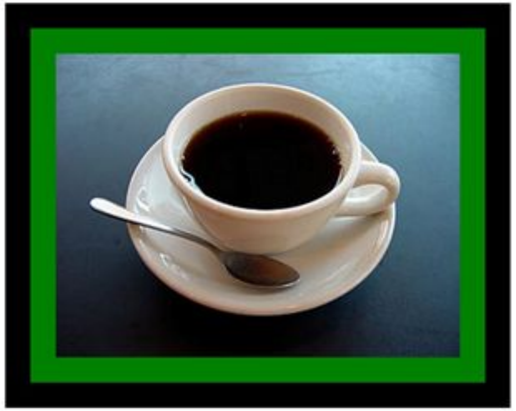
Just like with the border properties, the width of the margins and padding can be set either for all sides at once, or once side at a time.



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;
}
```

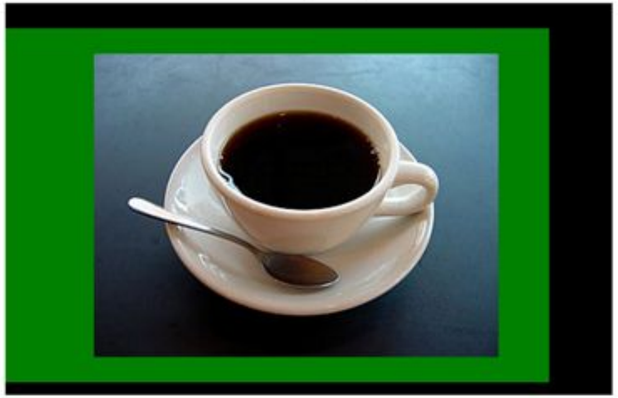


auctor, ac tempus sapien dignissim. Curabitur tincidunt metus in augue accumsan ullamcorper. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Curabitur vehicula tincidunt euismod. Cras feugiat mi at nisi semper, eu mollis quam porttitor. Aliquam erat volutpat. 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.