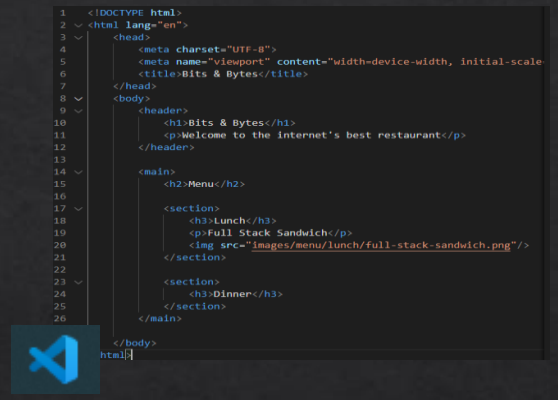
Module 3
CSS Selectors



Session Objectives:

HTML document structure
CSS Selectors and Layout Positioning
Box Model

Module 3
CSS Selectors

Elements

These are the building blocks of an HTML page, and are also referred to as nodes. A few of the elements/nodes shown in the listed code are:

html, body, head
title, div, p etc.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=de
    <title>Document</title>
</head>
<body>
    <p>My cat is very grumpy</p>
</body>
</html>
```

Tags

The tag defines HTML elements, and it is represented within angular brackets < >
example: <html>, <head>, <body>

Attributes

An attribute defines the properties of an HTML element. Here are some of the attribute examples from the listed code:
example: <meta name="viewport"... (name is an attribute for meta)

Module 3
CSS Selectors

Box Model

Every element in web design is a rectangular box.
The content, padding, border, and margin can be used to calculate the amount of space that an element takes up.

HTML Code

```
<p>We at Acme Co. guarantee that you'll
love our products. We sell:</p>

<ul>
    <li>Paper goods</li>
    <li>School supplies</li>
    <li>Laptops</li>
</ul>

<p>Visit our website to learn more.</p>
```

Rendered content

1 We at Acme Co. guarantee that you'll love our products. We sell:

2
- Paper goods
- School supplies
- Laptops

3 Visit our website to learn more.

There are three types of "boxes" with different layout behaviors: block, inline, and inline-block. It's important to understand the default layout behavior of these boxes and how they interact

Module 3
CSS Selectors

Block

A block element always starts on a new line and takes up the full width available, meaning that it stretches out to the left and right as far as it can. If it's contained inside another element, it only takes up the width of that element, not the entire page. By default, block elements use as much height as the content needs, but it's possible to change the height and width.

Examples of block level elements include:

-
- <h1> - <h6>
- <p>
- <form>
- <header>
- <footer>
- <section>

Module 3
CSS Selectors

Inline

An inline element doesn't start on a new line and only takes up as much height and width as necessary. It stays inline to the text that it's contained in. You can't set the height and width of an inline element.

Examples of inline elements include:

-
- <a>
- <img>
-
-

Inline-block

An inline-block element is very similar to inline, but you can set the height and width of it. This can be useful if you want to change the size of certain elements but don't want them to start on a new line.

Examples of inline-block elements include:

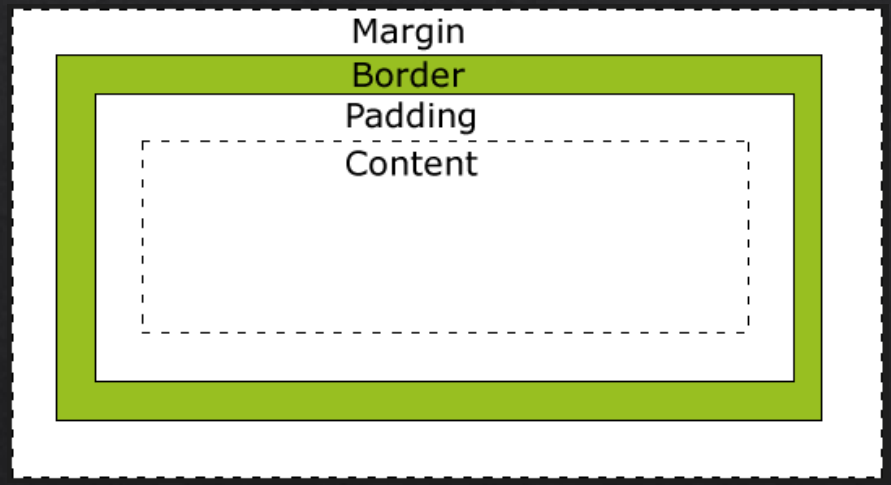
- <select>
- <button>

Module 3
CSS Selectors

Spacing

Box Model

Every element in web design is a rectangular box.
The content, padding, border, and *margin* can be used to calculate the amount of space that an element takes up.




```
div {
     width: 200px;
     border: 5px solid lightgreen;
     padding: 20px;
     margin: 15px;
 }
```

- Content - The content of the box, where text and images appear
- Padding - Clears an area around the content. The padding is transparent
- Border - A border that goes around the padding and content
- Margin - Clears an area outside the border. The margin is transparent

Module 3
CSS Selectors

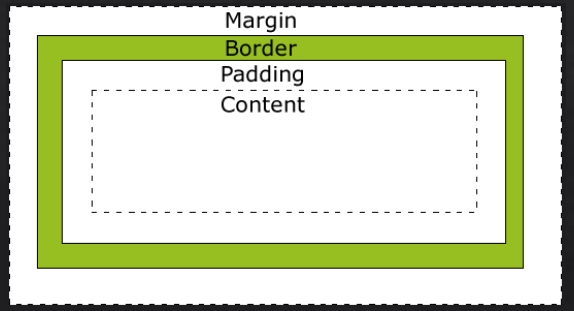
Every element in web design is a rectangular box.
The content, padding, border, and margin can be used to calculate the amount of space that an element takes up.

Calculate the total space taken up on the screen (in pixels)

CSS:

```
div {
    width: 200px;
    height:50px;
    border: 5px solid lightgreen;
    padding: 20px;
    margin: 12px 4px;
}
```

height width



Double the pixel counts for padding, margin, and border + width/height.

Total Space: Width: 258px; height: 124px

Module 3
CSS Selectors

Basic selectors

A CSS selector selects the element or elements that you want to style on the page. There are many ways to write selectors based on how broad or specific you want to be.

CSS is used to change the default styles of HTML elements in the browser. You can change element colors, borders, widths, heights, fonts, and display styles. To define these new rules, you first need to "select" the elements that you want to change.

Module 3
CSS Selectors

Element selector

The element, or type, selector selects all HTML elements of a certain type. If you want to style all elements, the CSS selector is div:

```
div {
  ...
}
```

Class selector

If you only want to style some elements but not others of the same type, you can apply a class attribute to the HTML elements, and write your CSS selector to apply to that class:

```
<div class="notice"></div>
<span class="notice"></span>

.notice {
    ...
}
```

Note how the selector starts with a .. This is how you denote a class in CSS

Module 3
CSS Selectors

ID selector

If you want to apply your style to only one element, you can apply an id attribute to the HTML element, and write your CSS selector to apply to that id:

```
<div id="info"></div>

#info {
    ...
}
```

Note how the selector starts with a #. This is how you denote an id in CSS.

Note: IDs should be unique

In your HTML document, no two elements should have the same ID. IDs are meant to be unique in HTML. There's nothing that strictly enforces this, but certain things won't work if you reuse an ID anywhere.

Module 3
CSS Selectors

Advanced selectors

Now that you've seen basic selectors, there are a few more "advanced" selectors to know. These may not be used as much as the other selector types, but you might find yourself needing one of these selectors or come across it in an existing project.

Module 3
CSS Selectors

Universal selector

The universal selector selects every element on a page. This could be useful in situations where you want to set some "default" values for the page, like font size and color. The universal selector is a single *:

```
* {
    ...
}
```

Attribute selector

The attribute selector selects elements that have the defined attribute, and optionally attribute value too. You could think of this as a longhand version of the class and id selectors, but for any attribute that you have on an element. If the attribute value isn't provided, the selector matches as long as an attribute exists. The attribute selector uses [] to denote the attribute:

```
input[type] {
    ...
}
input[type=text] {
    ...
}
<input type="button" value="This only matches the first selector" />
<input type="text" value="This matches both selectors" />
```

There are ways to write selectors that match if the attribute value starts with, ends with, or contains a value. You can read more about attribute selectors on the MDN page for Attribute Selectors

Module 3
CSS Selectors

Pseudo-class selector

A pseudo-class defines a special state of an element based on user interaction or its location. A pseudo-class begins with a colon :. This example applies the CSS rule when the user hovers the mouse over any <a> element:

```
a:hover {
    ...
}
```

Other pseudo-classes include:

:focus for when the user interacts with an input element, like typing, or is ready to type in a text box.
:active to select the element the user is actively clicking on; typically used for a elements, but can be applied to any type.
:first-child and :last-child selects every element that's the first or last child of its parent.
:enabled and :disabled matches every enabled or disabled element; mostly used on input elements in a form.

There are many more pseudo-classes. These are some of the more common ones. You can read more about them at the MDN page for Pseudo-classes

Module 3
CSS Selectors

Combinators

Another powerful feature of CSS selectors is that you can combine them to specify the elements you want from the page. There are several combinators that you can use to narrow down your selection.

Combo combinator

You can combine selectors into very specific selections by combining them:

```
button[type=submit].danger {
    ...
}
```

This code example uses an element selector, an attribute selector, and a class selector to apply the rule to any button element that has an attribute of type equal to submit that also has a class of danger. There can be no spaces between any of these parts if you want them all to apply to the same element.

Module 3
CSS Selectors

Multiple combinator

You can have a set of styles apply to different elements by separating those elements with a comma. The following style applies to all buttons and all checkboxes and all select elements:

```
button, input[type=checkbox], select {
    ...
}
```

See the following diagram for another example and how the elements are selected in the document tree:

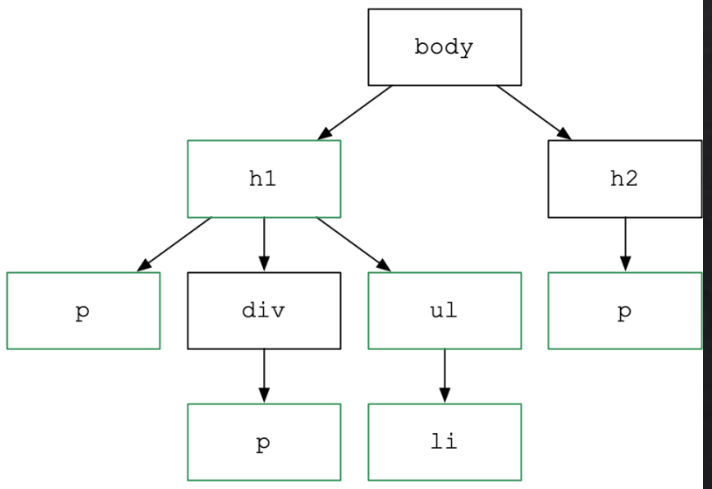
CSS Style

```
h1, .special {
    color: green;
}
```

HTML Code

```
<h1>
    <p class="special">…</p>
    <div>
        <p class="special">…</p>
    </div>
    <ul class="special">
        <li class="special">…</li>
    </ul>
</h1>
<h2>
    <p class="special">…</p>
</h2>
```

Document Tree



Module 3
CSS Selectors

Descendant combinator

Descendant combinators select all the elements that are inside another element. The following rule applies to any p element that's anywhere within a div element that has a class of aside:

```
div.aside p {
    ...
}
```

For example, the styles apply to the following p elements:

```
<div class="aside">
  <p>Styles will apply to this paragraph</p>
  <div class="section">
      <p>Styles will apply to this paragraph also because it is still in
side the div.aside</p>
  </div>
</div>
```

Module 3
CSS Selectors

Child combinator

Child combinators select all the elements that are direct children of another element. The following rule applies to any p element that's directly under a div element that has a class of aside:

```
div.aside > p {
    ...
}
```

For example, the styles apply to only the first p element:

```
<div class="aside">
  <p>Styles will apply to this paragraph</p>
  <div class="section">
      <p>Styles will *NOT* apply to this paragraph because it is not a d
irect child of div.aside</p>
  </div>
</div>
```

Module 3
CSS Selectors

CSS rules

Along with the selector, a CSS rule consists of at least one property and value. For instance, if you wanted to change the background color of all divs to red, you could write a rule like this:

```
div {
    background-color: red;
}
```

In this example, background-color is the property, and red is the value that's being set to that property. Note that the property and value are separated by a colon : and the CSS rule has a semi-colon ; at the end. The semi-colons are important when you have multiple property-value pairs.

If you wanted to also style those divs to be only 500 pixels wide, the CSS rule would look like this:

```
div {
    background-color: red;
    width: 500px;
}
```

You can find all the CSS properties that can be changed on MDN's [CSS Reference](https://developer.mozilla.org/en-US/docs/Web/CSS/Reference page)

Module 3
CSS Selectors

Element Positioning

- The **normal flow** of a page is for elements to appear left to right and top to bottom based on the order in which they appear in the HTML document and the rules of block and inline display.
- **Static** position by default means the element conforms to normal flow.
- **Relative** position means **relative to where it would otherwise be positioned in the normal flow**.
    - You can set the top, right, bottom, and left positioning attributes.

- **Absolute** position places the element relative to the parent ancestor—that is, the containing element—**exactly where you specify**.
    - These elements are removed from the flow of the page.
    - Setting both top and bottom, or both left and right allows you to "stretch" an element's dimensions.

- **Fixed** position is **relative to the browser window** and does not scroll with the page.
    - You can set the top, right, bottom, and left positioning attributes.

Module 3
CSS Selectors

Units of measurement

Every CSS declaration includes a property-value pair. Depending on the property, the value may require a unit of measurement. You saw these earlier in this chapter with values like 20px. px stands for "pixel" and represents one device pixel or dot on a screen. There are many units of measurement that are used for various CSS properties, but this section covers the most frequently used ones.

When including a length value, the unit identifier is required and is case insensitive. It must come immediately after the numeric part of the value, with no space in between. If the length is zero, the unit identifier isn't required:

Absolute versus relative length units

There are two types of lengths in CSS: absolute and relative.

Module 3
CSS Selectors

New Tools!