Web Dev Day 4: JS & Recap

Website Development
UBC | Coding Pals

JavaScript Syntax

- SOMEWHAT similar to CSS
- Curly bracket language
- Needs semicolons after each line except after curly brackets

Variables

What is a variable?

- A variable is a way for us to store a value, which can also be changed or modified later

Three ways to define variables:

- Var (not preferred)
- Let
- Const

Data Types

A data type refers to what kind of data is being stored in a variable

Examples of data types:

- Number
- String
- Boolean (true/false)
- Null

Variables are *dynamically typed* in JavaScript, meaning that they can change from one data type to another after being defined (unless const)

JavaScript Operators

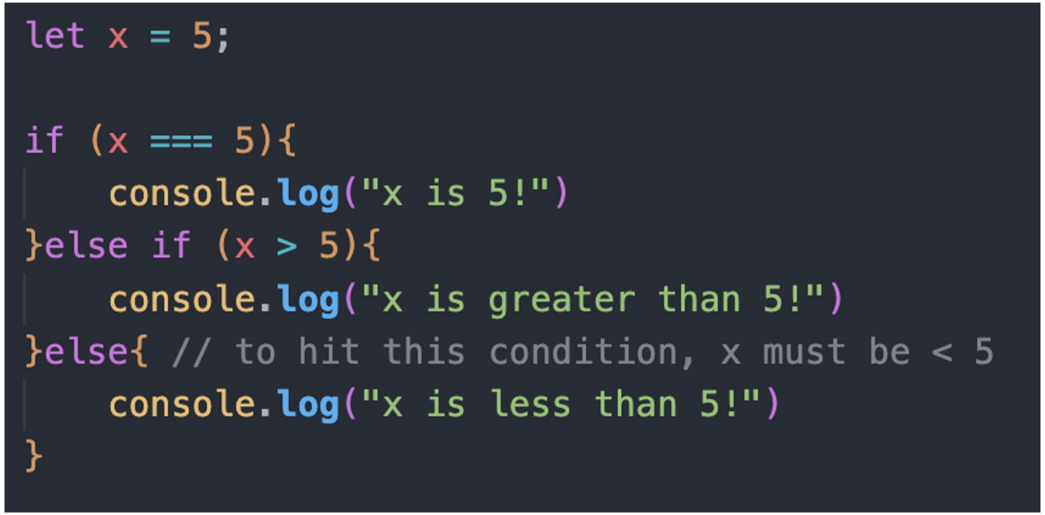
- + : Addition
- - : Subtraction
- * : Multiplication
- / : Division
- ** : Exponential
- % : Modulus
- ++ : Increment (Add by 1)
- -- : Decrement (Subtract by 1)
- = : Assign

If-Else Statements

If-else statements come in handy when you need to run conditional logic

- Checking equality (==) vs. strict equality (===)

Syntax example:



```
let x = 5;

if (x === 5){
    console.log("x is 5!")
}else if (x > 5){
    console.log("x is greater than 5!")
}else{ // to hit this condition, x must be < 5
    console.log("x is less than 5!")
}
```

Scope (Global vs Local)

If you define variables in a local scope (ex. within an if statement), you cannot call them in a wider scope

Ex:

```
if (variable === true){
        let x = 5; // defining a variable here
}
console.log(x); // will not run, because the variable ‘x’ is not defined in this scope
```

Functions

Functions are used to avoid repeating code

Defined by:

```
function functionname(){
        // code inside the function goes here
}
```

Functions can return values or print data or change some information

Calling a Function from HTML

Function defined in JS:

```
function clicked(){
    document.getElementById("divid").innerHTML = "Hello JavaScript!";
}
```

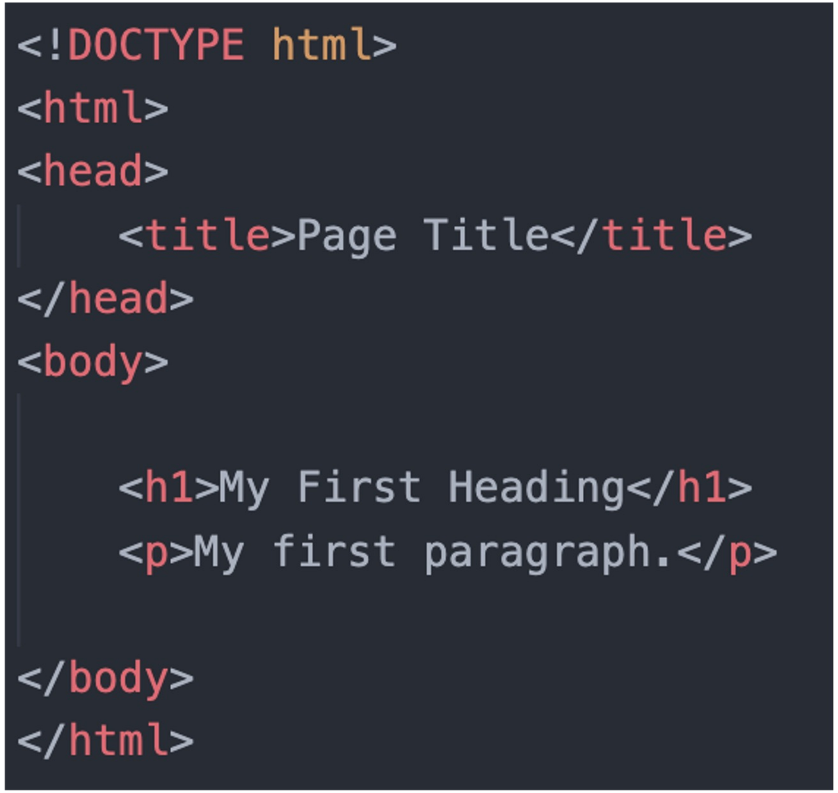
Calling the function in HTML:

```
<button onclick='clicked()'>Click Me!</button>
```

HTML Recap

HTML Syntax

- Opening & closing tags (<> </>)
- Always contains:
  - <!DOCTYPE html>
  - Opening & closing <html> tags
- Indented tags (for readability)
- Child & Sibling tags
- Order of sibling tags matters
- Attributes



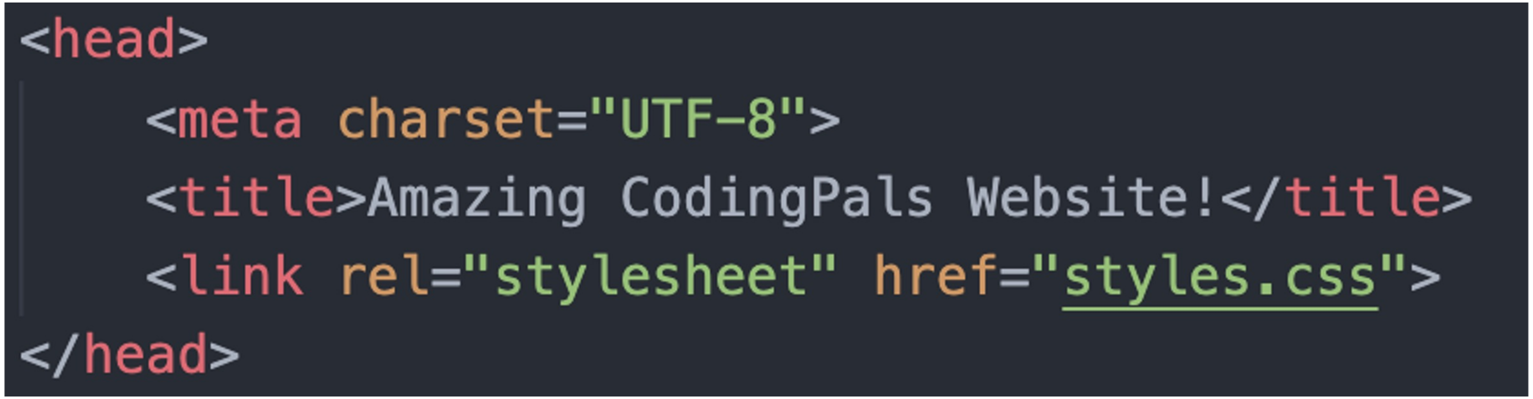
```html
<!DOCTYPE html>
<html>
<head>
    <title>Page Title</title>
</head>
<body>

    <h1>My First Heading</h1>
    <p>My first paragraph.</p>

</body>
</html>
```

HTML Head Section

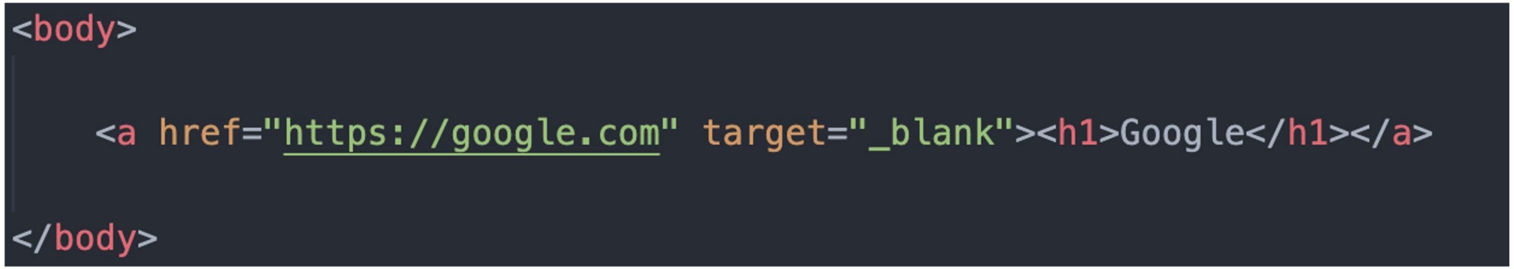
- Meta tags
- Title tags
- Link tags
- Additional attributes & properties



```html
<head>
    <meta charset="UTF-8">
    <title>Amazing CodingPals Website!</title>
    <link rel="stylesheet" href="styles.css">
</head>
```

Anchor (Hyperlink) Tags

- Denoted with the <a> tag
- Used to link to different websites (also can link to images, but not very common)
- Can nest other tags
- Href attribute
    - Href must start with ‘https://’
- Target attribute



```
<body>

    <a href="https://google.com" target="_blank"><h1>Google</h1></a>

</body>
```

Google

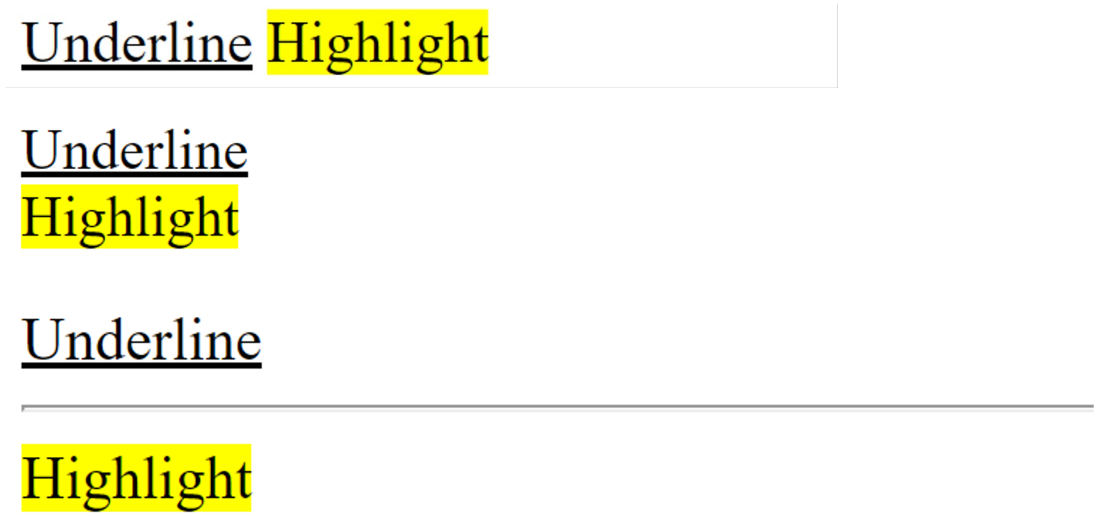
Single Tags

Breaks, horizontal rule

<br>

<hr>

No closing tag (</>)


Underline Highlight

Underline
Highlight

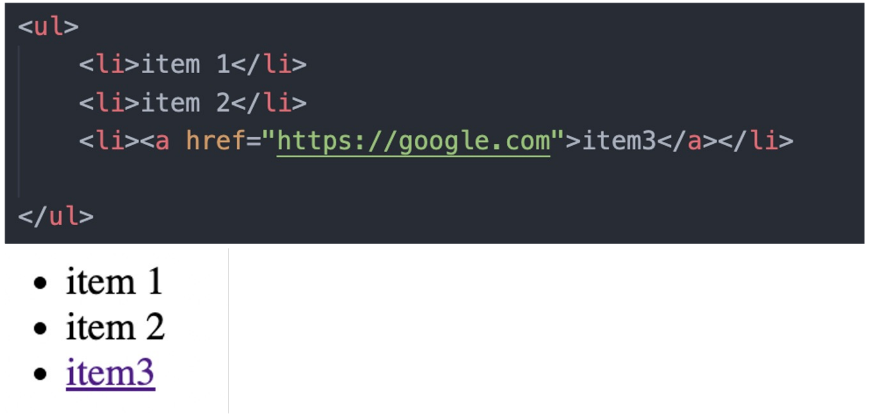
Underline

Highlight

Lists

- Unordered (<ul>) vs. Ordered (<ol>)
- Individual elements denoted by <li>
- Attribute to define type of ordering
- Notice the indents

Unordered List:

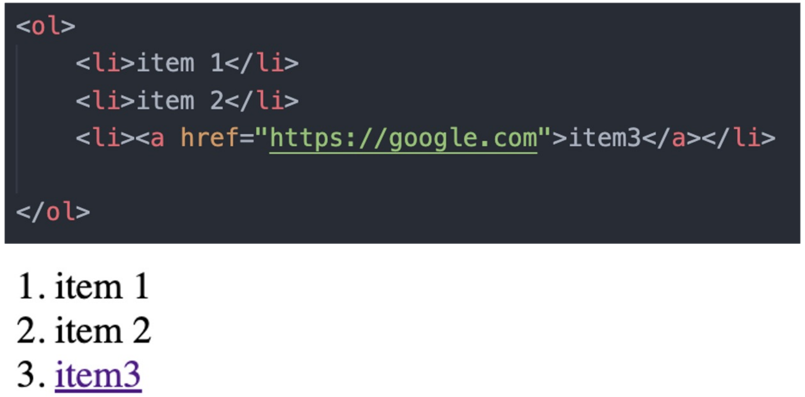


```
<ul>
    <li>item 1</li>
    <li>item 2</li>
    <li><a href="https://google.com">item3</a></li>

</ul>
```

- item 1
- item 2
- item3

Ordered List:



```
<ol>
    <li>item 1</li>
    <li>item 2</li>
    <li><a href="https://google.com">item3</a></li>

</ol>
```

1. item 1
2. item 2
3. item3

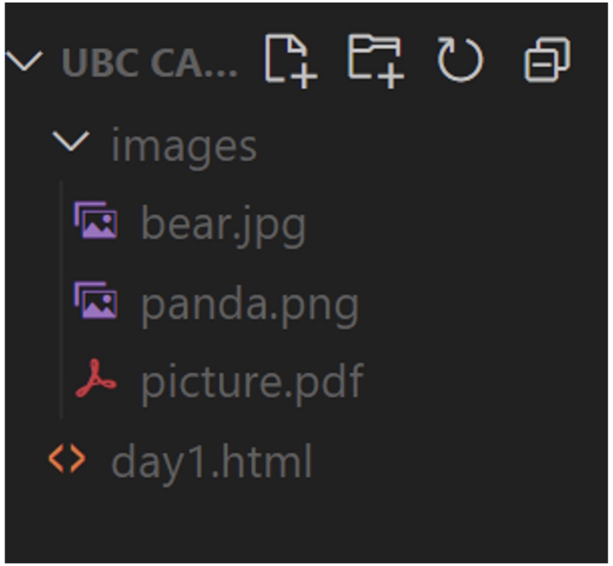
Images

- Denoted with the <img> tag (single tag, no need for closing tag)
- Attributes:
    - Src (source)
    - Alt (alternative link)
    - Width & height (html considers aspect ratios)

Images need to be referenced with a path

Create a folder named “images” under the main folder directory

<img src=”*image path*” alt=”*text*”>



```
UBC CA...
  images
    bear.jpg
    panda.png
    picture.pdf
  day1.html
```

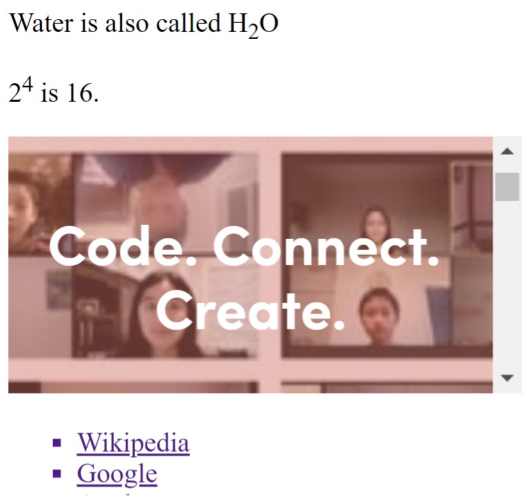
Videos

- Denoted with the <video> tag
- Attributes
  - Src (source; can define multiple and the browser will play the first one that is compatible)
  - Controls (gives the option to play, pause, etc.)
  - Width & height
  - Poster (thumbnail of a video)
  - Autoplay
  - Loop
- Like images, videos must be referenced by a path
- Optional text between opening and closing <video> tags to display a message if none of the src videos are compatible

YouTube Videos/ iFrame

iFrame is essentially an webpage embedded inside another page. And iFrame is also the easiest way to embed a Youtube video into a webpage.

<iframe src=”*source*” title=”description”></iframe>



Water is also called H_2O

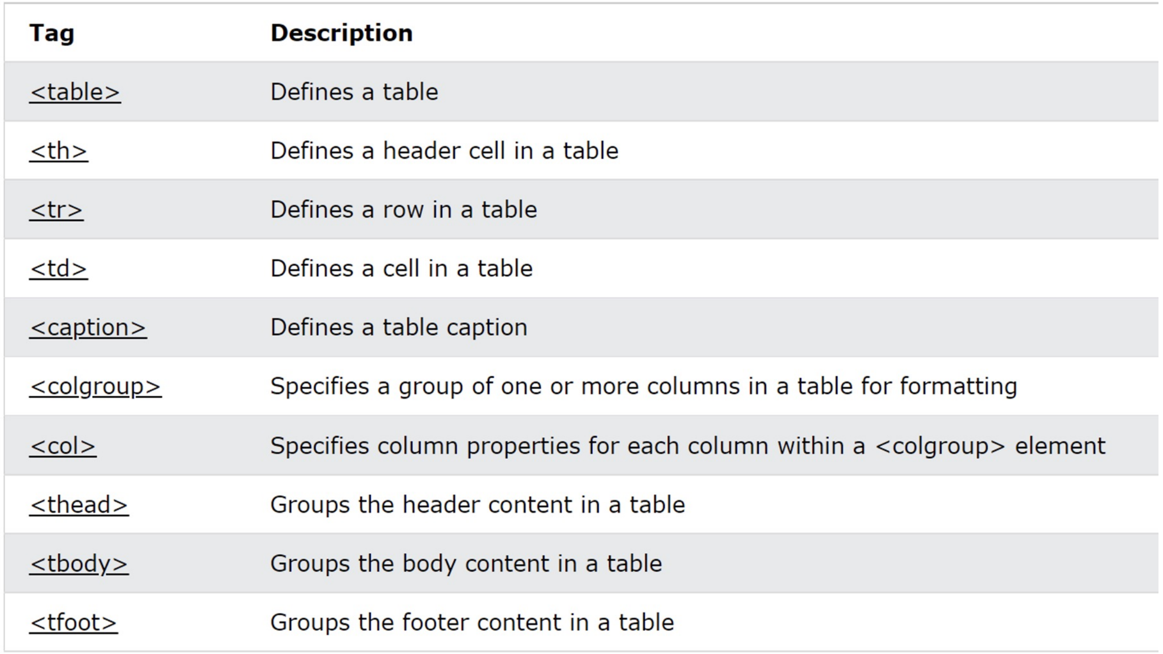
2^4 is 16.

- Wikipedia
- Google

Tables

The syntax for table is very similar to the syntax for a list, but there are a few more elements.

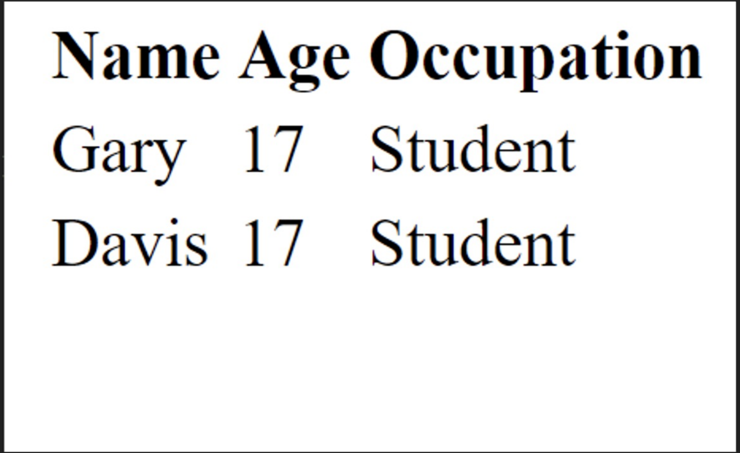
For an example code for the table element, refer to next slide.



Tag	Description
<table>	Defines a table
<th>	Defines a header cell in a table
<tr>	Defines a row in a table
<td>	Defines a cell in a table
<caption>	Defines a table caption
<colgroup>	Specifies a group of one or more columns in a table for formatting
<col>	Specifies column properties for each column within a <colgroup> element
<thead>	Groups the header content in a table
<tbody>	Groups the body content in a table
<tfoot>	Groups the footer content in a table

Table Syntax

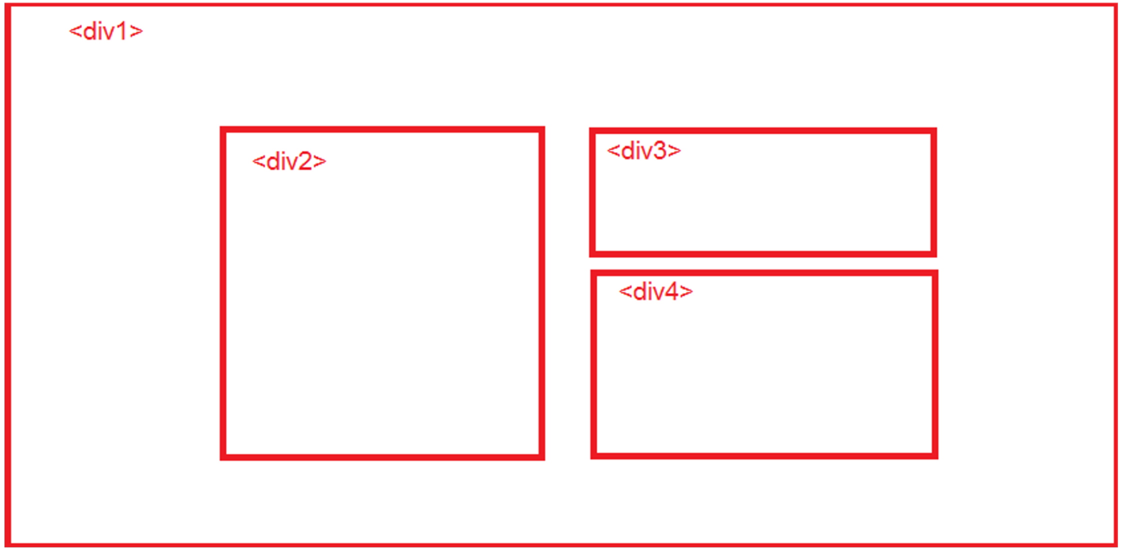
```
<table>
    <thead>
        <tr>
            <th scope="col">Name</th>
            <th scope="col">Age</th>
            <th scope="col">Occupation</th>
        </tr>
    </thead>
    <tbody>
        <tr>
            <td>Gary</td>
            <td>17</td>
            <td>Student</td>
        </tr>
        <tr>
            <td>Davis</td>
            <td>17</td>
            <td>Student</td>
        </tr>
    </tbody>
</table>
```



Name	Age	Occupation
Gary	17	Student
Davis	17	Student

Divs and Spans

- Inline vs Block elements
  - <a> is an example of inline, <p> is an example of block
- Divs are for block elements while Spans are for inline elements
- Divs and spans define “sections” of HTML to group it all under one category



IDs and Classes

IDs and Classes are used to assign an identifier to an HTML tag

Referenced when styling specific elements

- IDs can only be used once
- Classes can be used for multiple tags

Will go more into detail for CSS

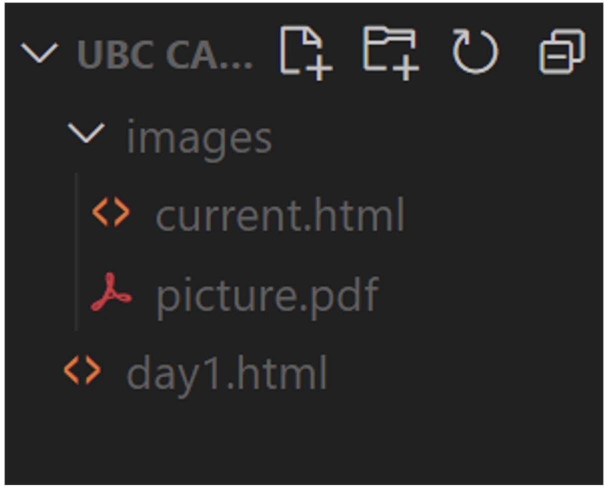
CSS Recap

Directories and Relative Paths

To access files within the same system, we can use paths instead of https links

To access the picture.pdf file from current.html, we can:

- href=”picture.pdf”
- href=”/images/picture.pdf”
- href=”../images/picture.pdf”



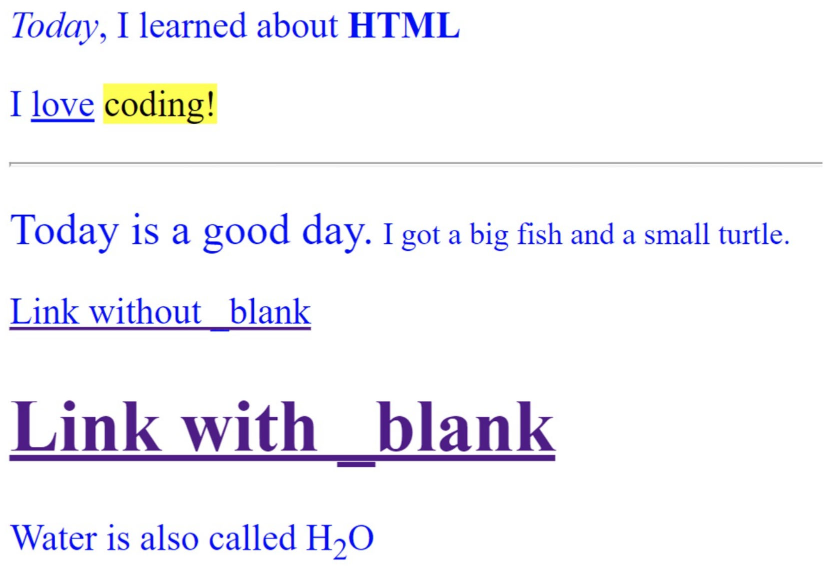
Notice how “/images/picture.pdf” is different from “images/picture.pdf”

Element Selector

In CSS, you can select an entire type of elements to make changes on. For example, you can select the <p> element and change its attributes, which will cause all <p> elements in that webpage to be altered.

```
p {
        color: blue;
}
```

Note: the "*" selector targets all elements in the HTML document



Today, I learned about **HTML**

I love coding!

Today is a good day. I got a big fish and a small turtle.

Link without _blank

Link with _blank

Water is also called H_2O

Targeting Specific IDs and Classes

We can also target IDs and Classes to style

- This is why it is important to give HTML tags IDs and Classes

. and #

- Use a “.” before the class name to target a class
- Use a “#” before the ID name to target an ID

It is also possible to assign one element to multiple classes.

Colours

```
.classname {
        /* the three following lines do the same thing */
        color: red;
        color: rgb(255,0,0);
        color: #ff0000
}
```

Can also define background colours with “background-color”

Use RGBA to define an opacity value at the end [ex: rgba(255,0,0,0.5)]

Text

Text can be customized in many different way using CSS!

Some example of properties that we can play around with are:

- color
- background-color
- text-align
- text-decoration
- text-transform
- letter-spacing

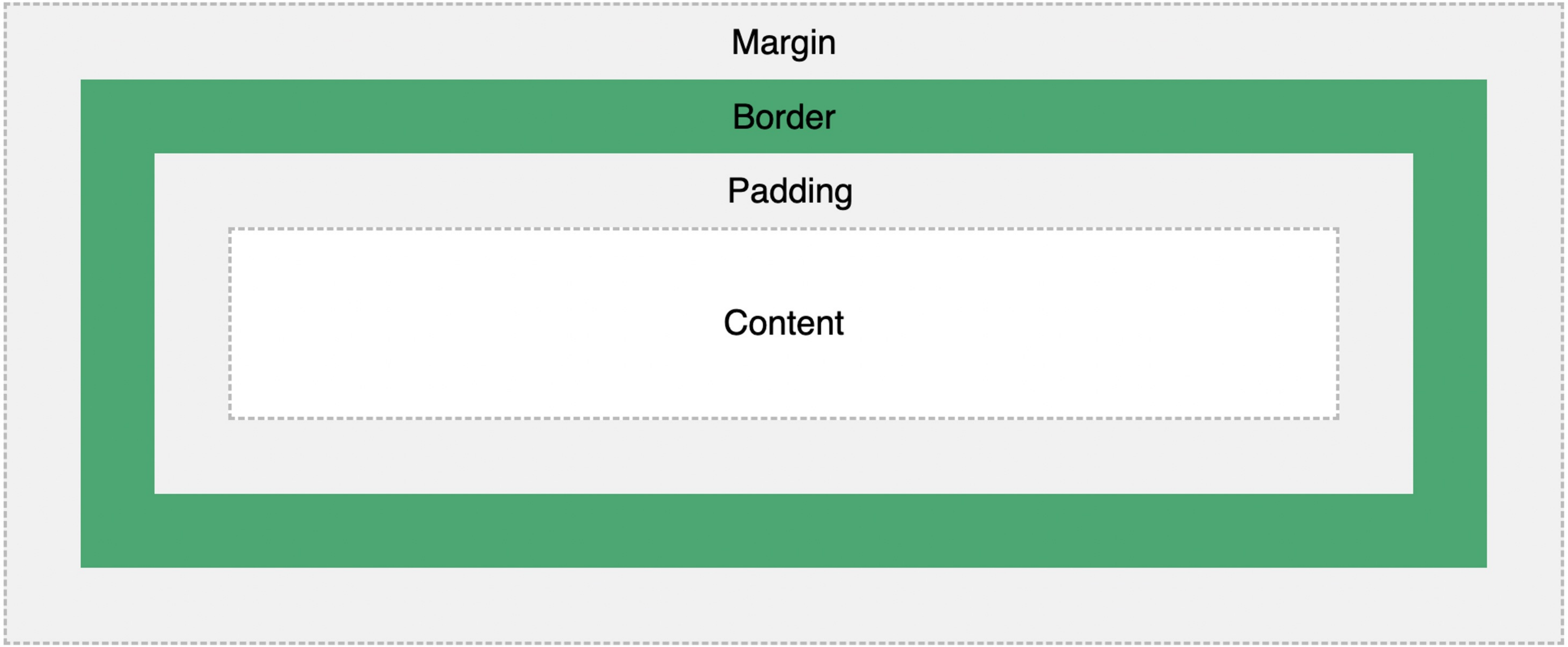
Width and Height

Certain tags need widths and heights defined

We can define width and height in two ways:

1. Absolute units: px, cm, etc. (do not add a space between the number and the unit)
2. Relative units:
   a. Rem: relative to the font size of the root element
   b. Em: relative to the font size of the element

Box Model (Padding, Border, Margin)



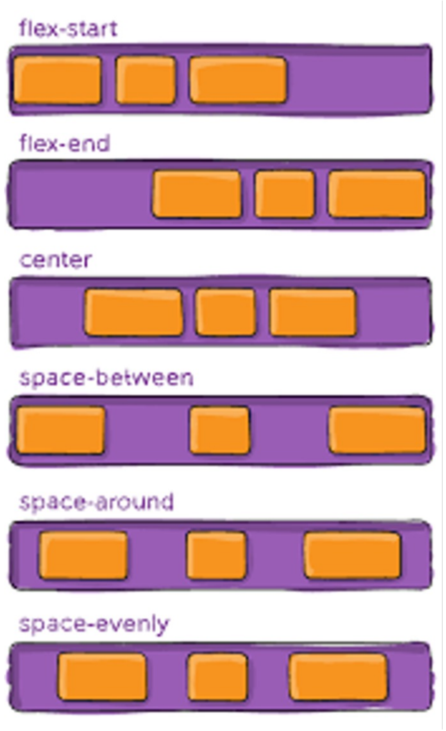
Display & Justify Content

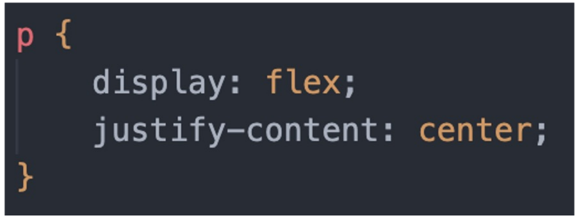
Display lets us manipulate the positioning and placement of elements

We can change inline elements to block and block elements to inline

By using "display: flex" we can change spacing with the "justify-content" property as shown below in the following ways:

I. **flex-start** = Default value. Items are positioned at the beginning of the container
II. **flex-end** = Items are positioned at the end of the container
III. **center** = Items are positioned in the center of the container
IV. **space-between** = Items will have space between them
V. **space-around** = Items will have space before, between, and after them
VI. **space-evenly** = Items will have equal space around them





```
p {
    display: flex;
    justify-content: center;
}
```

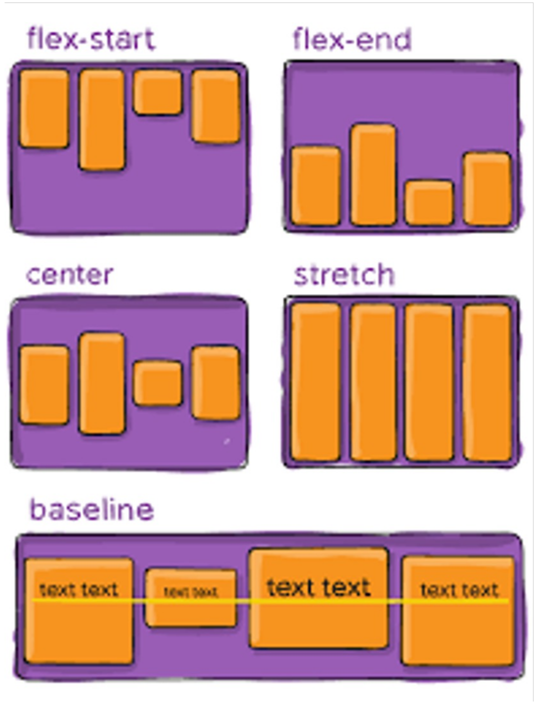
Align Items

Another useful attribute with flex displays is called align-items

For all the elements in your div they need to be aligned

Options:

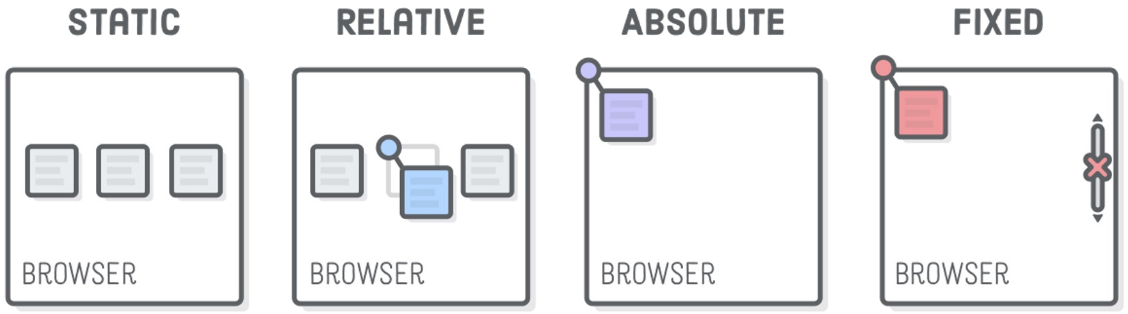
I. flex-start = Align everything to the top
II. flex-end = Align everything to the bottom
III. center = Align everything in the center
IV. stretch = Stretches everything to top & bottom
V. baseline = Align texts to be on the same level



Position

- Static
- Relative
    - top, bottom, left, right
- Absolute
- Fixed
- Sticky
- Z-index

Absolute vs. Fixed vs. Sticky



Resources

MDN Web Docs

“Dictionary” for HTML, CSS, & JS

https://developer.mozilla.org/

W3 Schools

Modules that explain all the components

https://w3schools.com/