This is Section.

Week 8: Web Development

Gabe LeBlanc
(today's slides courtesy of Andrew Holmes)

Attendance Form: cs50.ly/section8

Gradescope: view your inline comments!

Today’s plan

01 HTML

The building blocks

03 JavaScript

La casa Madrigal

02 CSS

A little polish

04 PSET

Ball’s in your court!

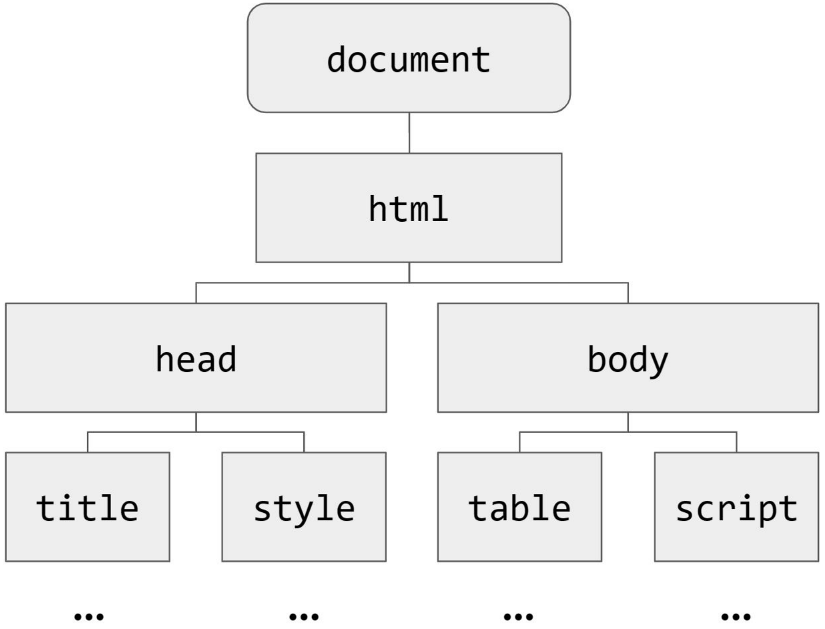
01

HTML

The building blocks

Document Object Model (DOM)

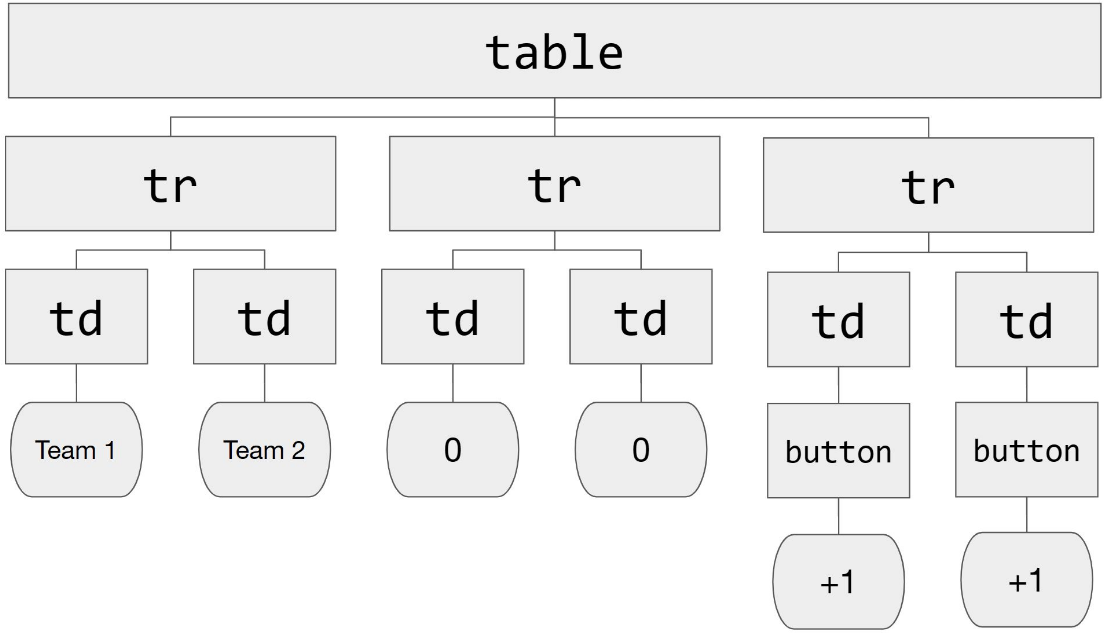
HTML



Document is a **tree** structure

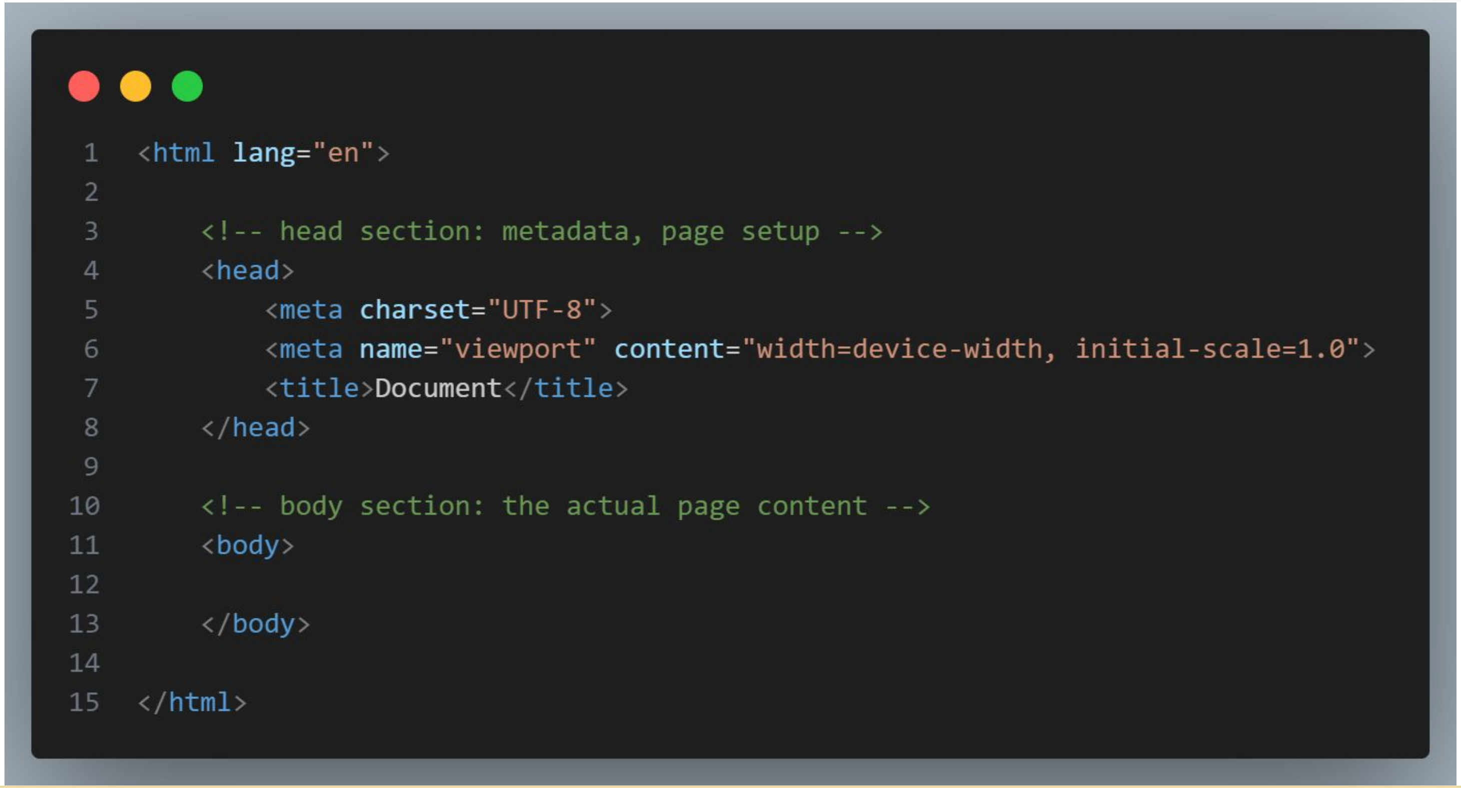
Document Object Model (DOM)

HTML



HTML

HTML



```html
<html lang="en">

    <!-- head section: metadata, page setup -->
    <head>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <title>Document</title>
    </head>

    <!-- body section: the actual page content -->
    <body>

    </body>

</html>
```

HTML tags

Tags (almost) always appear **in pairs:**

```
<opentag>
  content
</closetag>
```

Can you think of a tag that doesn't need closing?

one example:

```
<link>
```

HTML attributes

HTML

You can modify tags using **attributes**

```
<p style="color:blue">
  hi!
</p>
```

hi!

A million possibilities

HTML

There’s loads of tags, here are some of the most common:

```
<div>, <p>, <img>, <link>, <hr>,
<br>, <head>, <body>, <h1/2/3>,
<header>, <section>, <span>, <ul>,
<ol>, <li>, <a> and many more
```

And a million more

HTML

And even more attributes. Here's some key ones:

`id`	`class`	`style`
`href`	`src`	`onclick`

HTML example

scoreboard.html on github.com/gblanc25/cs50 → EXTRAPRACTICE → week8

02

CSS

Making it pretty

Cascading Style Sheets

CSS

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

Cascading Style Sheets

CSS

```
p
{
    background-color: blue;
    color: white;
}
```

Cascading Style Sheets

CSS

```
<style>
  p
  {
        background-color: blue;
        color: white;
  }
</style>
```

Cascading Style Sheets

CSS

Generally we avoid

We aim to keep these separate – tweaking form should never risk damaging function, easier to manage code.

Function
HTML

Form
CSS

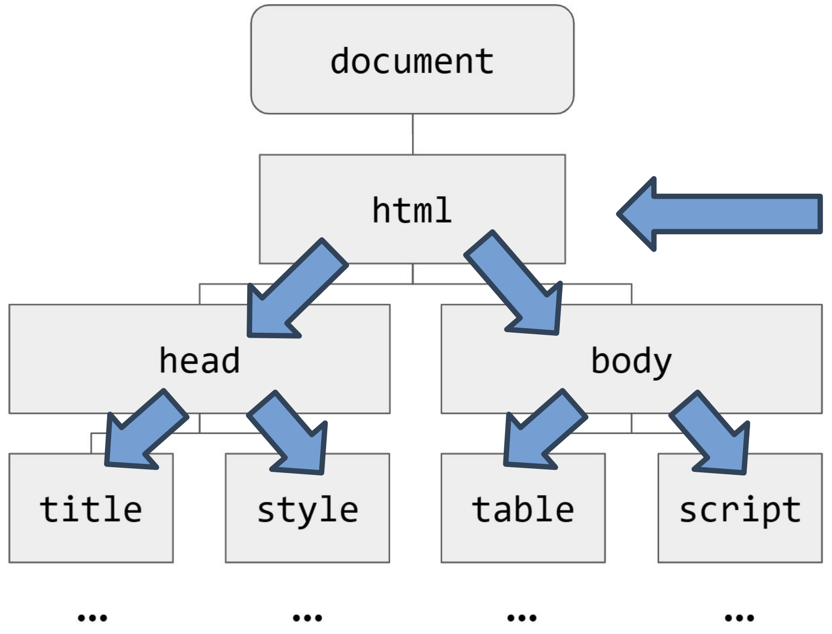
Cascading Style Sheets

CSS

To use a CSS file, we use link:

```
<link rel=“stylesheet”
      href=“styles.css”>
```

Why *cascading* style sheets?



`background-color: blue;`

Lower elements by default ‘**inherit**’ style. A change ***cascades*** down the tree.

Previous example

CSS

```
p
{
      background-color: blue;
      color: white;
}
```

Currently affects **all paragraphs!**

Let’s spice it up

CSS

In index.html :

```
<p id=“special-p”>
```

In styles.css:

```
#special-p {
   background-color: red;
}
```

Fighting the waterfall

CSS

We have this CSS, what color is the paragraph?

```
#special-p {
  background-color: red;
}
p {
  background-color: blue;
}
```

Classes vs ids

```
<div class=“home-item theme-1”
     id=“about-section”>   … </div>
```

id should be **unique**, to give a specific element unique styling.

Classes can be re-used, and elements can (and often) have multiple classes.

Classes vs ids

CSS

```
<div class="home-item theme-1"
     id="about-section">   ... </div>
```

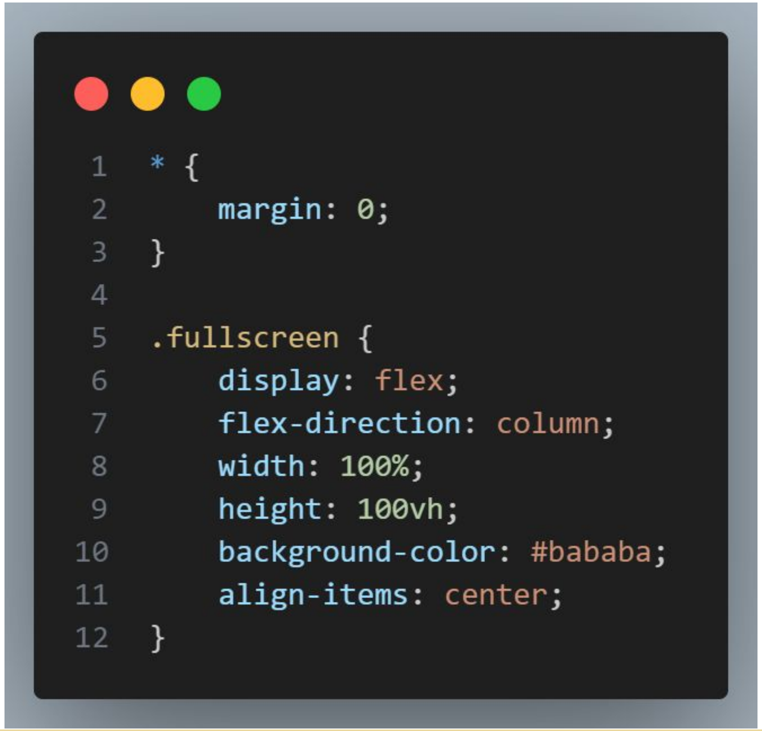
```
.home-item { ... }

#about-section { ... }
```

CSS example

CSS

Here's an example of some CSS I wrote for the intro slide. * modifies **all elements like in SQL**.

Then I have some code that only affects objects in the 'fullscreen' class.



```css
* {
    margin: 0;
}

.fullscreen {
    display: flex;
    flex-direction: column;
    width: 100%;
    height: 100vh;
    background-color: #bababa;
    align-items: center;
}
```

Common CSS selectors

CSS

```
display   color     padding   margin
height    width     border    font-weight
display   text-align    flex
position  outline

and many more!
```

CSS example

scoreboard.html on github.com/gblanc25/cs50 →
EXTRAPRACTICE → week8

03

Javascript

It’s getting lively

Javascript

Can write JavaScript within HTML script tags, or again, as preferred, in a separate file.

We write JavaScript in **.js files**.

You can include the via **<script src="…"></script>**

It is best to include these at the **end of the body** – load the content first, the animations second (unless required).

Console.log

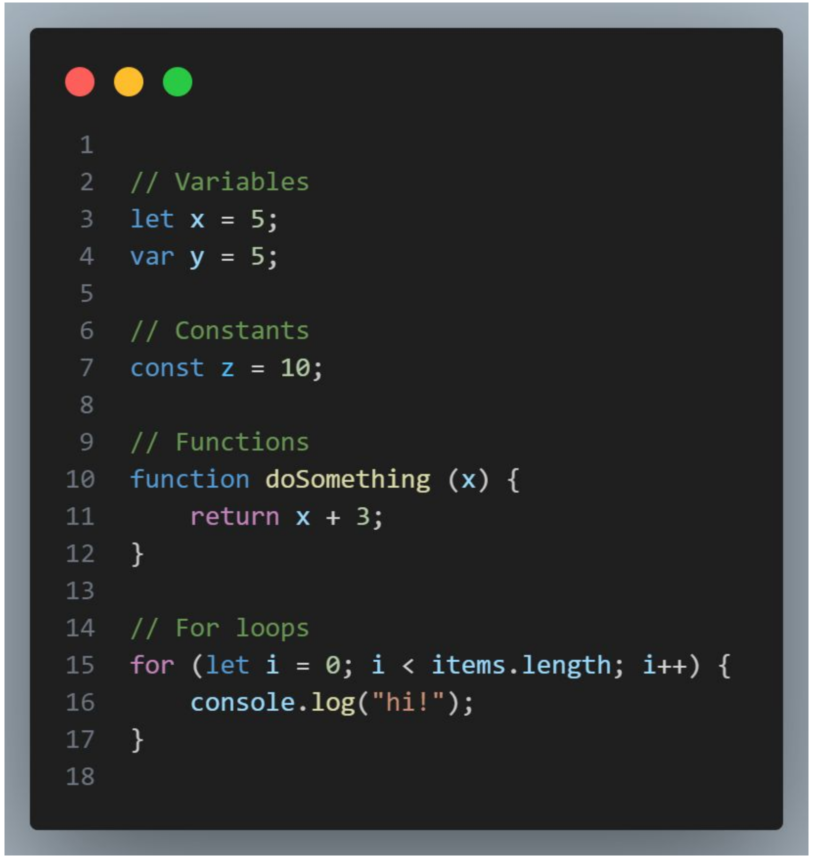
You can print like in other languages via **`console.log(...)`**

This will print output to the **console** in **your browser**.

Basic JS

JS in many ways resembles a blend of C and Python syntax.

Curly braces are back. For loop syntax is back. Types are not.



```javascript

// Variables
let x = 5;
var y = 5;

// Constants
const z = 10;

// Functions
function doSomething (x) {
    return x + 3;
}

// For loops
for (let i = 0; i < items.length; i++) {
    console.log("hi!");
}

```

JavaScript

Basic JS

JavaScript

Javascript allows us to manipulate the DOM. To do so, we need to get DOM objects:

```
.getElementById
document.getElementById(‘main-p’)
.querySelector
document.querySelector(‘.item’)
.querySelectorAll
document.querySelectorAll(‘.item’)
.parentElement
  button.parentElement
```

Basic JS

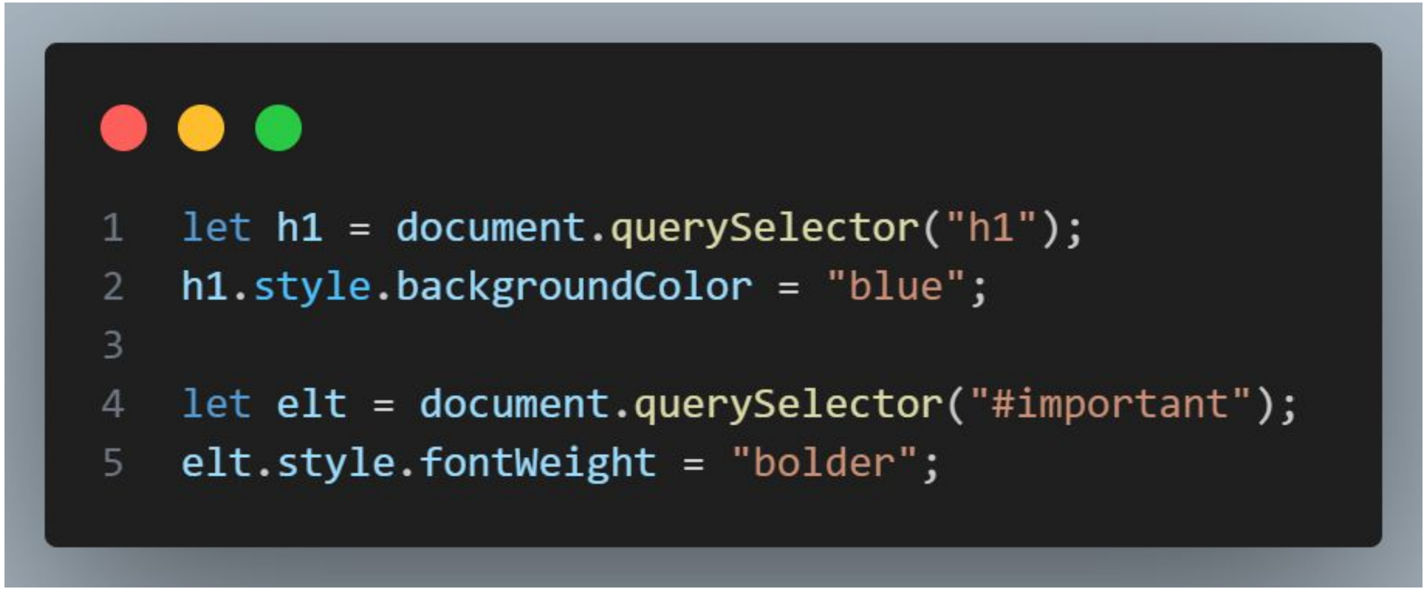
JavaScript

Javascript also allows us to **manipulate** the DOM:

```
.createElement
let p = document.createElement(‘p’);
.innerHTML
p.innerHTML(‘New paragraph!’);
.style
p.style.backgroundColor = “green”;
```

Basic JS

JavaScript



```javascript
let h1 = document.querySelector("h1");
h1.style.backgroundColor = "blue";

let elt = document.querySelector("#important");
elt.style.fontWeight = "bolder";
```

Variables can represent actual **objects on the DOM/things on the page!**

Event listeners

JavaScript

```
<DOM_ELT>. addEventListener(<event>, <fun>)
```

This creates an ‘event listener’ attached to this DOM element, every time <event> is detected at this element, it executes <fun>.

```
button.addEventListener(‘click’,
            function() {alert(‘clicked!’)})
```

Anonymous functions

JavaScript

```
button.addEventListener('click',
      function() {alert('clicked!')})

function handleAlert() {
    alert("Loaded!");
}

document.addEventListener('DOMContentLoaded',
                          handleAlert);
```

Javascript example

scoreboard.html on github.com/gblanc25/cs50 → EXTRAPRACTICE → week8

04

PSET

Trivia time