



} Heroes of CSS

Web Development Boot Camp
Lesson 1.3



Admin Items

Homework #1

HTML, CSS, & Git: Code Refactor

One of the most common tasks for front-end and junior developers is to take existing code and refactor it to either meet a certain set of standards or implement a new technology. Web accessibility is an increasingly important consideration for businesses, ensuring that people with disabilities or socio-economic restrictions have access to their website, and helping them avoid litigation.

Story:

AS A marketing agency

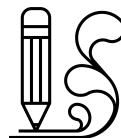
I WANT a codebase that follows accessibility standards

SO THAT our own site is optimized for search engines

Requirements:

You are required to submit the following for review:

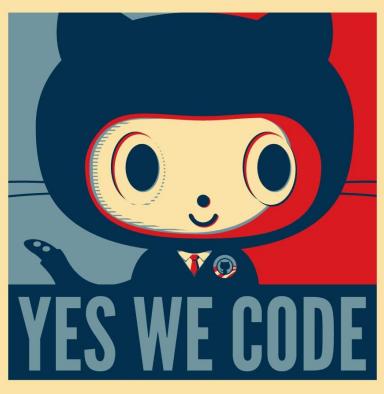
- The URL of the deployed application.
- The URL of the GitHub repository. Give the repo a unique name & include a README describing the project





Class Github Repo

<https://github.com/the-Coding-Boot-Camp-at-UT/UTA-AUS-FSF-PT-07-2020-U-C-MW>



Homework Assignment Tips

01

Really work hard on this assignment! This assignment introduces you to fundamental concepts that we'll build upon during the rest of the course.

02

Review in-class material, especially activities.

03

Work with your peers! It's much better than screaming at your computer alone.

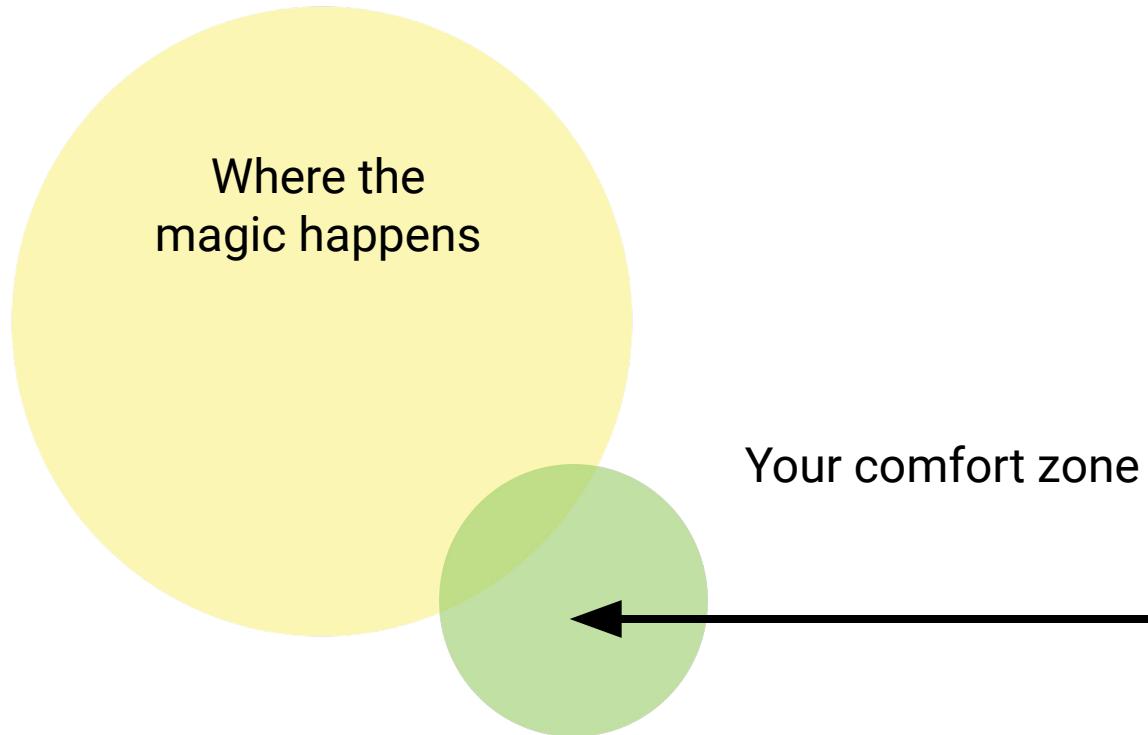
04

Ask questions on Slack! Your peers, TAs, and instructors are all here to help.



Most Important of All

Just submit *something* (even if it seems pretty crummy)!





Warning!

Brace Yourselves

Today is going to be a bit tough. But trust us—it will all look easy a few weeks from now!





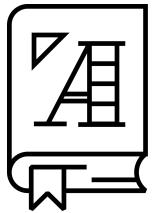
Don't expect to understand everything at once. Today is all about getting immersed.

CSS Recap

Critical Question:

What is CSS?

HTML and CSS Definitions



HTML: Hypertext Markup Language (Content)

CSS: Cascading Style Sheets (Appearance)

HTML/CSS are the “languages of the web.” Together they define both the content and the aesthetics of a webpage, handling everything from the layouts, colors, fonts, and content placement. (JavaScript is the language that deals with logic, animation, etc.)

HTML/CSS Analogy

HTML Alone	HTML and CSS
Like writing papers in Notepad.	Like writing papers in Microsoft Word.
Used to write unformatted text (i.e, content only).	Used both to write the content <i>and</i> format it (color, font, alignment, layout, etc.).
	

Basic HTML Page (No CSS)

Awesome Header

Smaller Awesome Header

Even Smaller Awesome Header

 Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.
 Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.



Menu Links

- Google
- Facebook
- Twitter

Basic HTML Page (No CSS)

Awesome Header

Smaller Awesome Header

Even Smaller Awesome Header

 Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.
 Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.



Boring

Menu Links

- Google
- Facebook
- Twitter

Critical Question: How Do We Style HTML?

Elements?

Classes?

IDs?

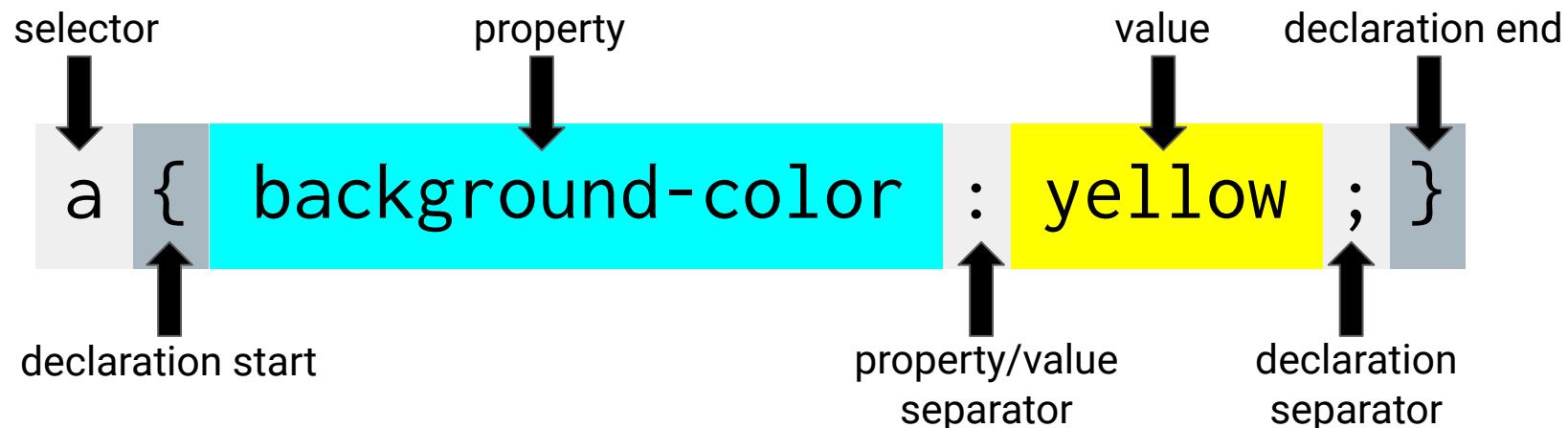


CSS Syntax

CSS works by hooking onto **selectors** added into HTML using **classes** and **identifiers**.

Classes use **.classname**, IDs use **#idname**, and elements use just their name.

Once hooked, we apply **styles** to those HTML elements using CSS.



Selectors

Element selector	Element name (p, a, div, span, etc.)	Applies to all <p> elements <pre>p { background-color: blue; }</pre>
Class selector	Period (.) + variable name (.myDiv, .phoneNumber, etc.)	Applies to all elements with class="classItem" <pre>.classItem { background-color: orange; }</pre>
ID selector	Hash (#) + variable name (#myDiv, #phoneNumber)	Applies to all elements with id="idItem" <pre>#idItem { background-color: green; }</pre>

CSS Selectors

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

```
.classItem {  
    background-color: orange;  
}
```

```
#idItem {  
    background-color: green;  
}
```



```
<p>  
    A paragraph with a blue background.  
</p>
```

```
<div class="classItem">  
    A div with an orange background.  
</div>
```

```
<div id="idItem">  
    A div with a green background.  
</div>
```

A paragraph with a blue background.

A div with an orange background.

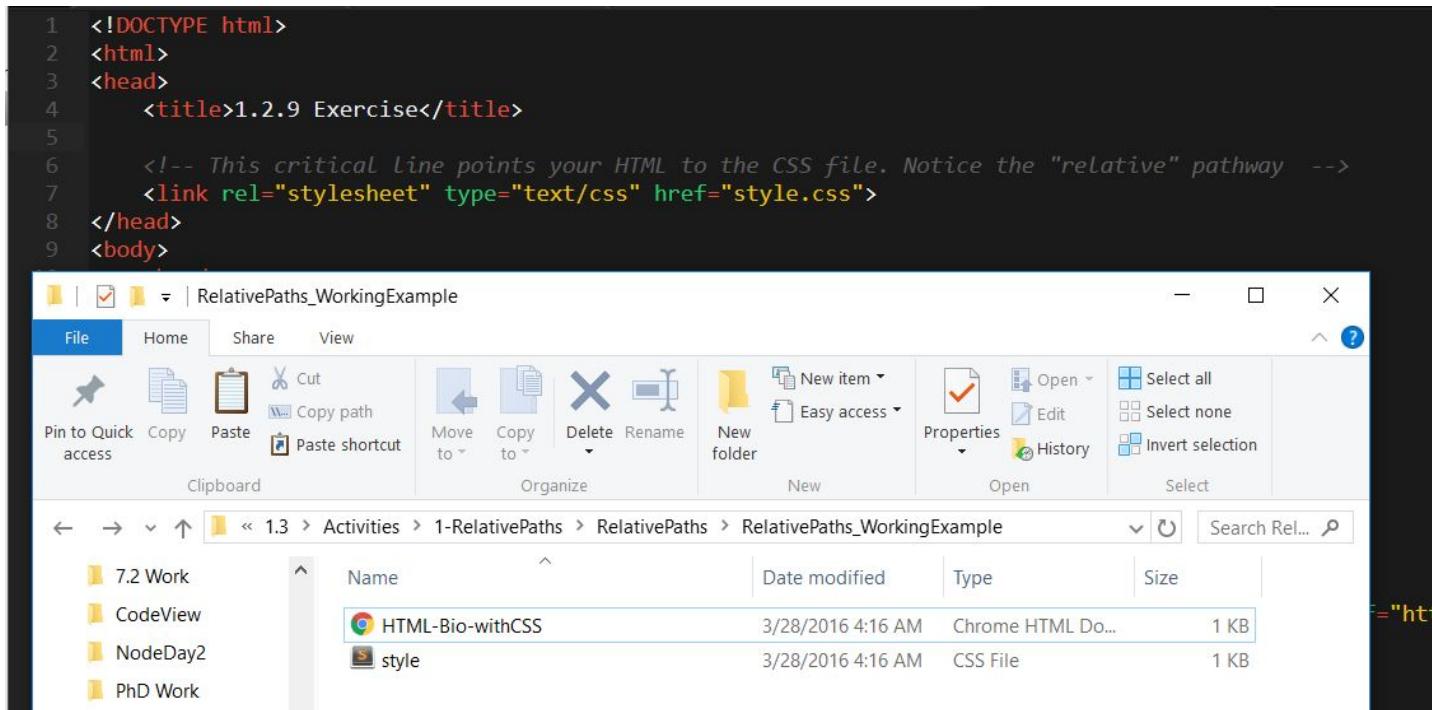
A div with a green background.

Questions?

Relative File Paths

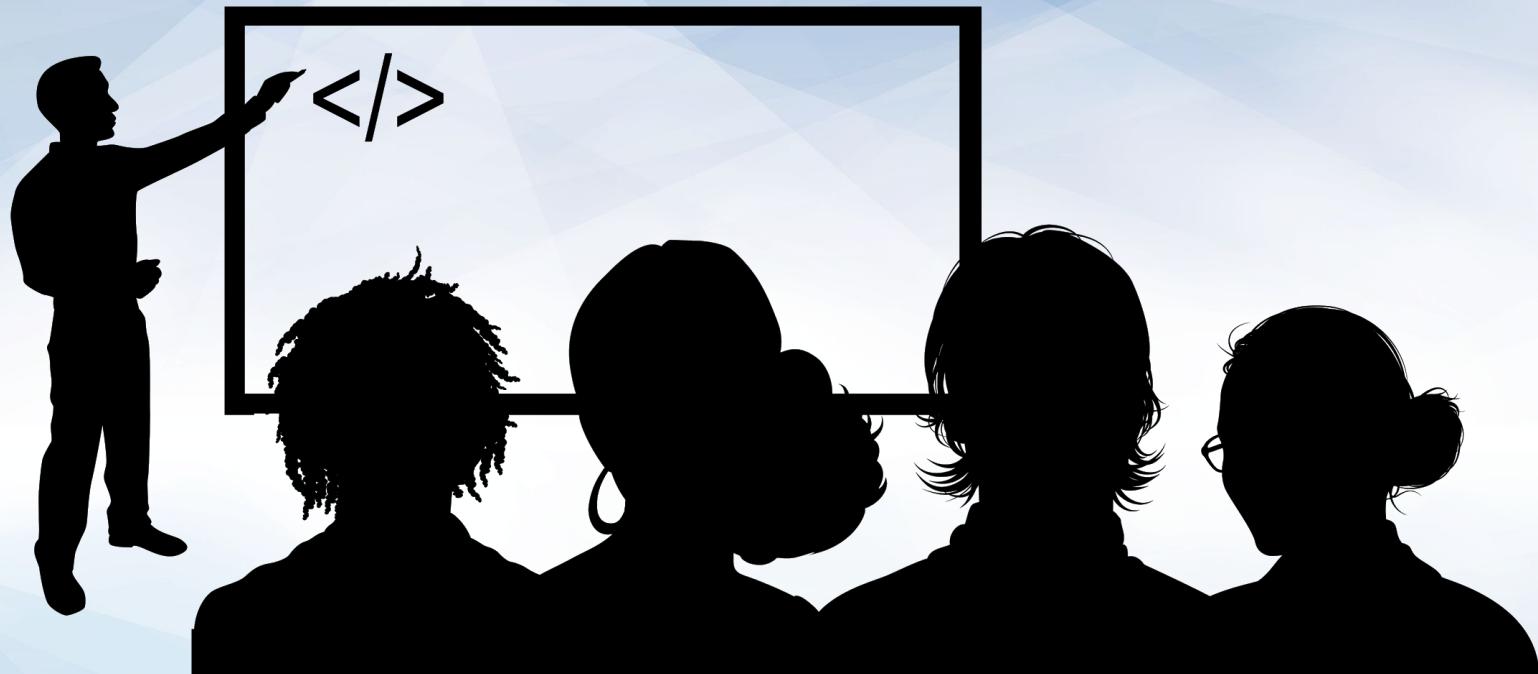
Relative File Paths

Relative file paths connect us with other files in our working directory.
In this case, style.css is in the same folder as our HTML document.



The screenshot shows a Windows File Explorer window titled "RelativePaths_WorkingExample". The address bar indicates the path: "1.3 > Activities > 1-RelativePaths > RelativePaths > RelativePaths_WorkingExample". The left sidebar shows several folders: "7.2 Work", "CodeView", "NodeDay2", and "PhD Work". The main pane displays two files: "HTML-Bio-withCSS" (a Chrome HTML Document, 1 KB) and "style" (a CSS File, 1 KB). The "style" file is highlighted with a blue selection bar. The status bar at the bottom right shows the URL "http://127.0.0.1:5000/1.3/Activities/1-RelativePaths/RelativePaths/RelativePaths_WorkingExample/HTML-Bio-withCSS".

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4     <title>1.2.9 Exercise</title>
5
6     <!-- This critical line points your HTML to the CSS file. Notice the "relative" pathway --&gt;
7     &lt;link rel="stylesheet" type="text/css" href="style.css"&gt;
8 &lt;/head&gt;
9 &lt;body&gt;</pre>
```



Instructor Demonstration Relative File Paths

Absolutely No Absolute Paths

Always use relative file paths!



If you deploy websites without them, **all of your links will fail.**



The same will happen if you move your project from one folder to another.

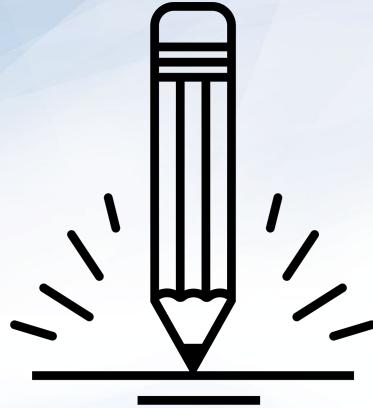


Remember, there is no such thing as a C: drive on the internet.

VERY, VERY BAD



```
<!-- BAD!!!! -->
<link rel="stylesheet" href="D:/trilogy/FullStack-Lesson-Plans/02-lesson-plans/01-
html-css-three-days/1-Class-Content/1.3/Activities/1-RelativePaths/RelativePaths/
RelativePaths_WorkingExample/style.css">
```



Activity:

Relative File Paths

Suggested Time:
10 minutes



Activity: Relative File Paths

01

Unzip the folder sent to you via Slack.

02

Edit the HTML files in all of the `RelativePaths` folders. Write relative paths that link the HTML documents with CSS stylesheets.

HINT: Check out the `RelativePaths_WorkingExample` folder.

Suggested Time: 10 Minutes



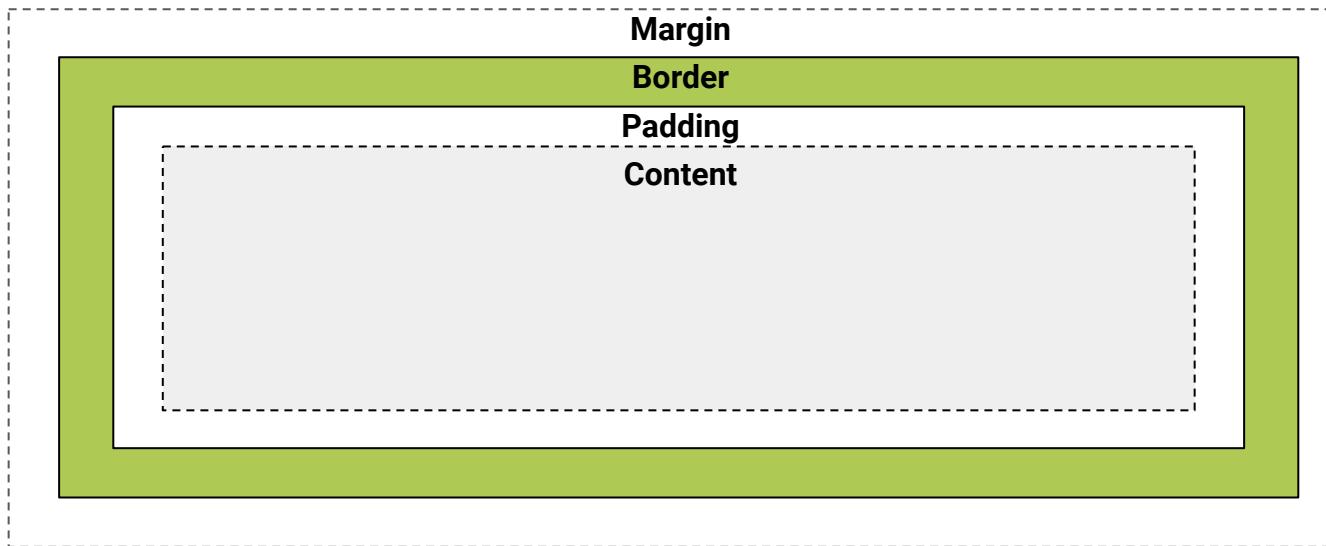
Box Model

Boxes Upon Boxes

In CSS, every element rests within a series of boxes.

Each box has customizable space properties: margin, border, and padding

Typical spacing value: 20px 10px 10px 20px (top, right, bottom, left)



Activity: Box Model

```
#box {  
  
background-color: #1E5792;  
width: 400px;  
height: 440px;  
margin: 10px 30px 20px 50px;  
color: #fff;  
padding: 25px 10px 30px 20px;  
border-style: solid;  
border-width: 22px;  
border-color: #113152;  
  
}
```

How wide is the blue #box?

How tall is the blue #box?

Total element width = content width + left padding + right padding + left border + right border + left margin + right margin

Total element height = content height + top padding + bottom padding + top border + bottom border + top margin + bottom margin

Suggested Time: 10 Minutes



Activity: Boxes Upon Boxes

Answer:

Width: 474 px (no margin), 554 px (with margin)

Height: 539 px (no margin), 569 px (with margin)

```
#box {  
  
background-color: #1E5792;  
width: 400px;  
height: 440px;  
margin: 10px 30px 20px 50px;  
color: #fff;  
padding: 25px 10px 30px 20px;  
border-style: solid;  
border-width: 22px;  
border-color: #113152;  
  
}
```

How wide is the blue #box?

How tall is the blue #box?

Total element width = content width + left padding + right padding + left border + right border + left margin + right margin

Total element height = content height + top padding + bottom padding + top border + bottom border + top margin + bottom margin

Suggested Time: 10 Minutes

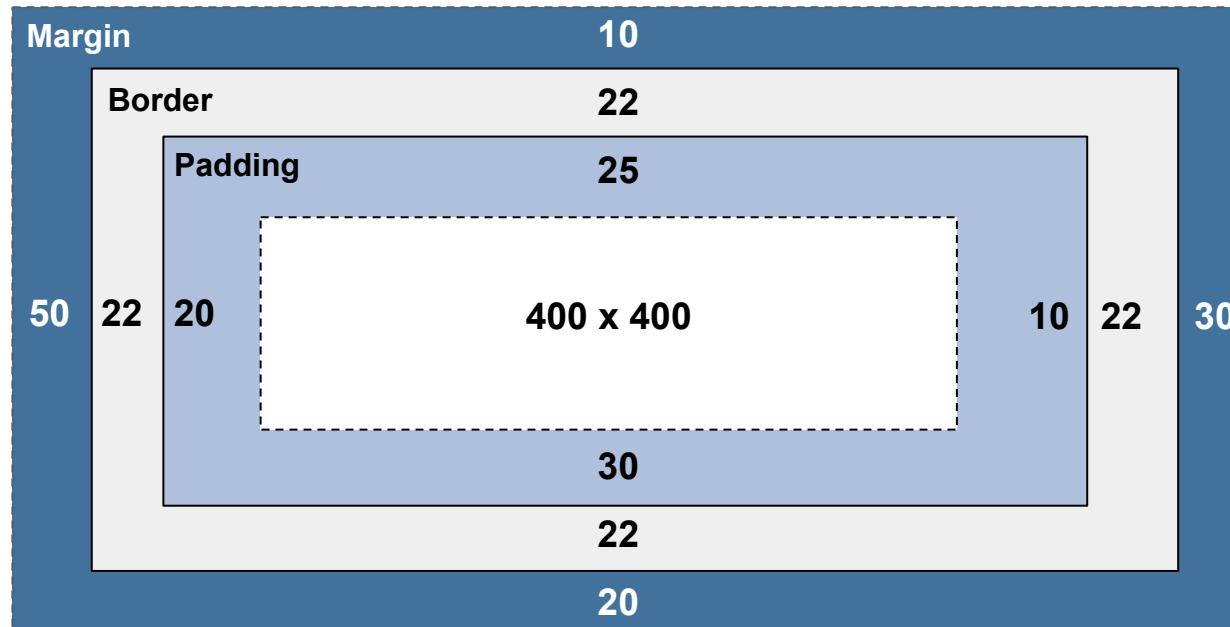


Activity: Box Model

Answer:

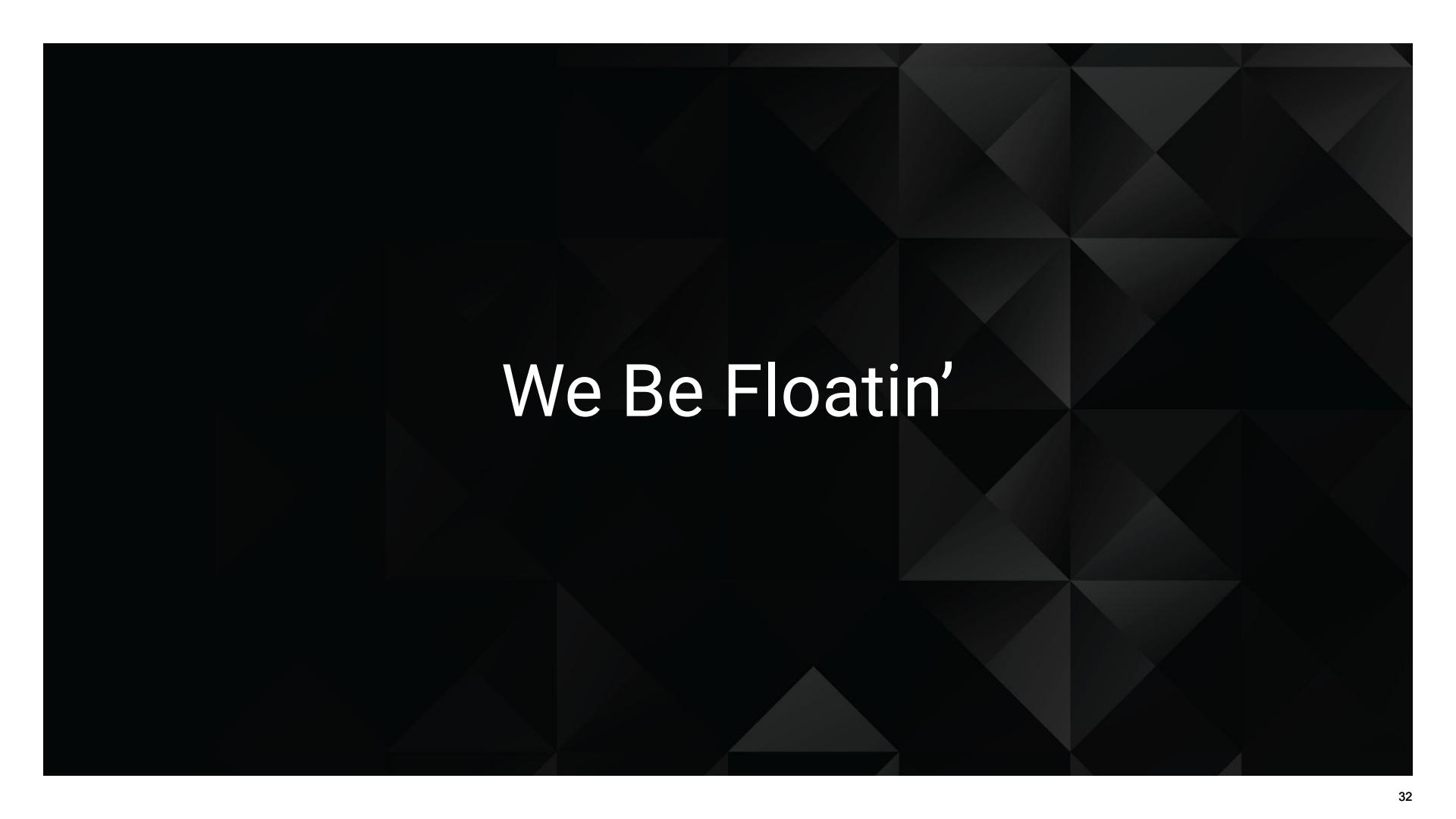
Width: 474 px (no margin), 554 px (with margin)

Height: 539 px (no margin), 569 px (with margin)



Suggested Time: 10 Minutes





We Be Floatin'



Warning

The next few topics are fairly tricky but also *very important*. Time to channel that inner genius!

The Concept of Flow

By default, every HTML element displayed in the browser is governed by a concept called **flow**.

This means that HTML elements force adjacent elements to flow around them.



Analogy: Flow and MS Word



The concept of flow is similar to wrap-text in Microsoft Word.



Just as in MS Word, with CSS you can position images to be in-line with text, on top of text, and so on.



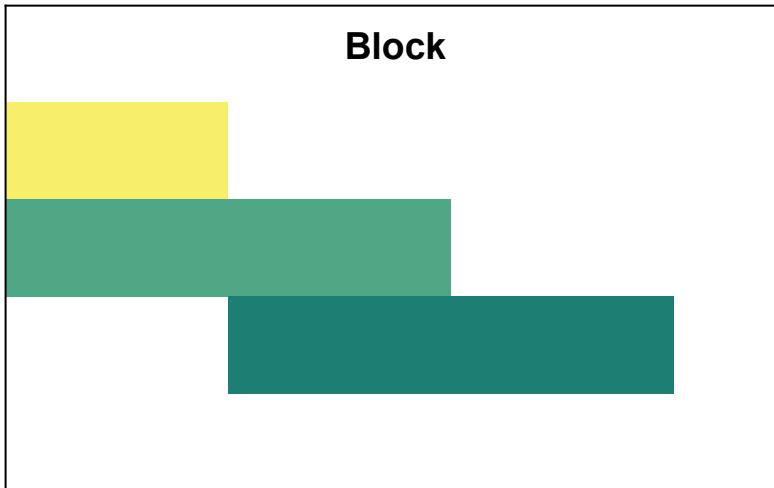
Block Elements



By default, web clients render many HTML elements as **block elements**. Paragraphs, headers, divs, and more receive this treatment.



A block element will take up an entire line of space—unless you intervene with CSS properties.



Block Elements vs. In-line Elements

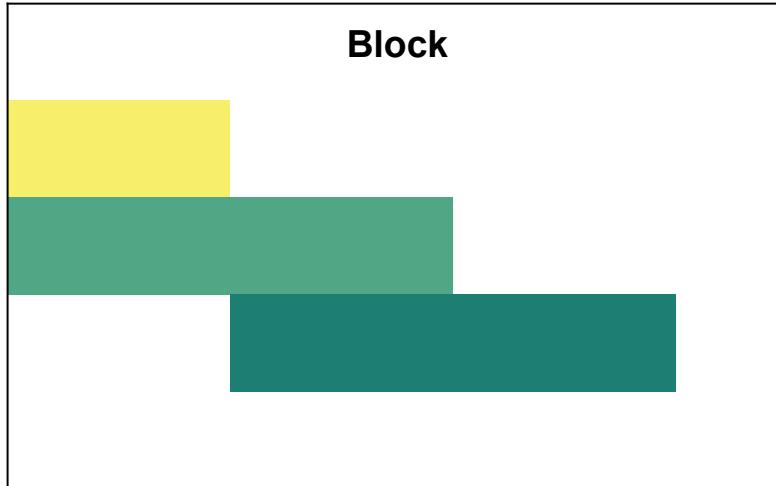


Now, contrast block elements with **in-line elements**.

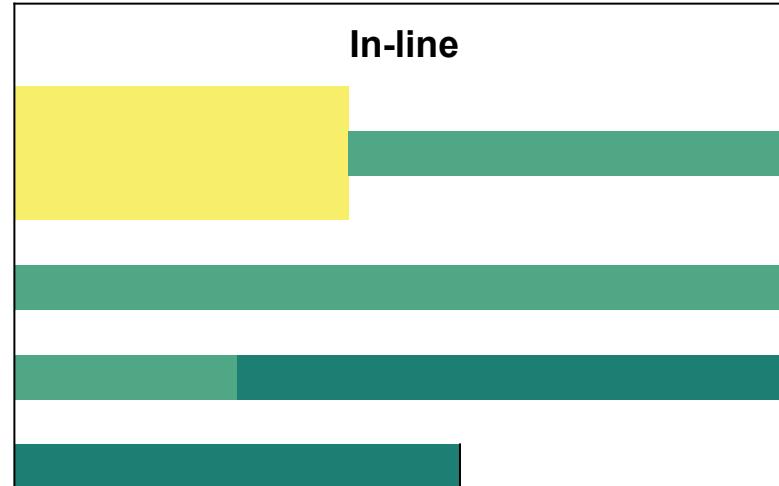


By using **float CSS properties**, we can command our website to display multiple HTML elements adjacently.

Block

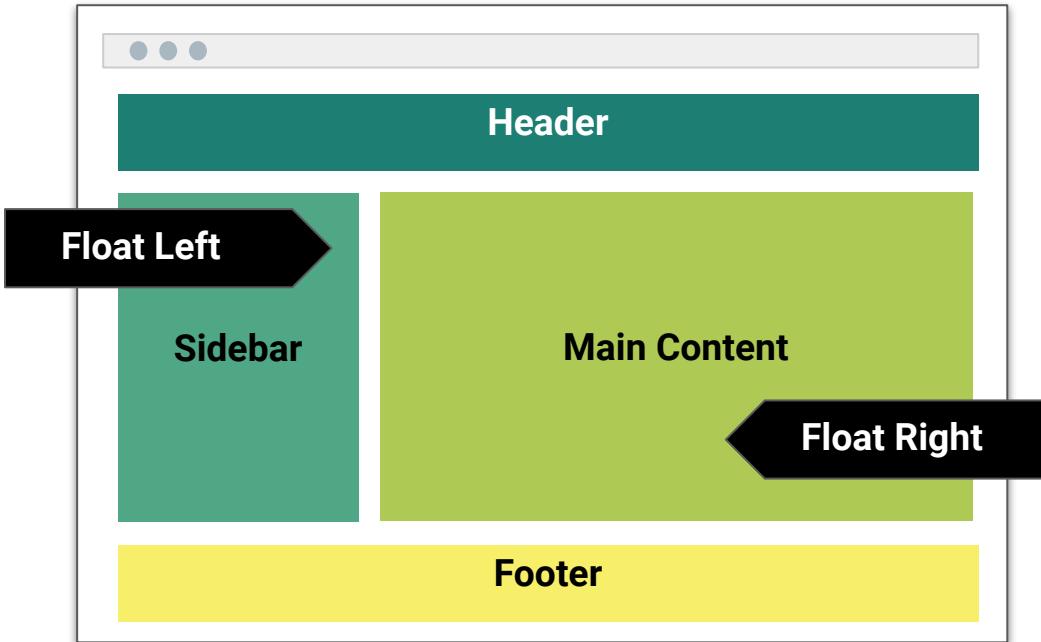


In-line



Floats

To transform these block elements into in-line elements, we use a CSS property called **float**. Floats are necessary when building web layouts.

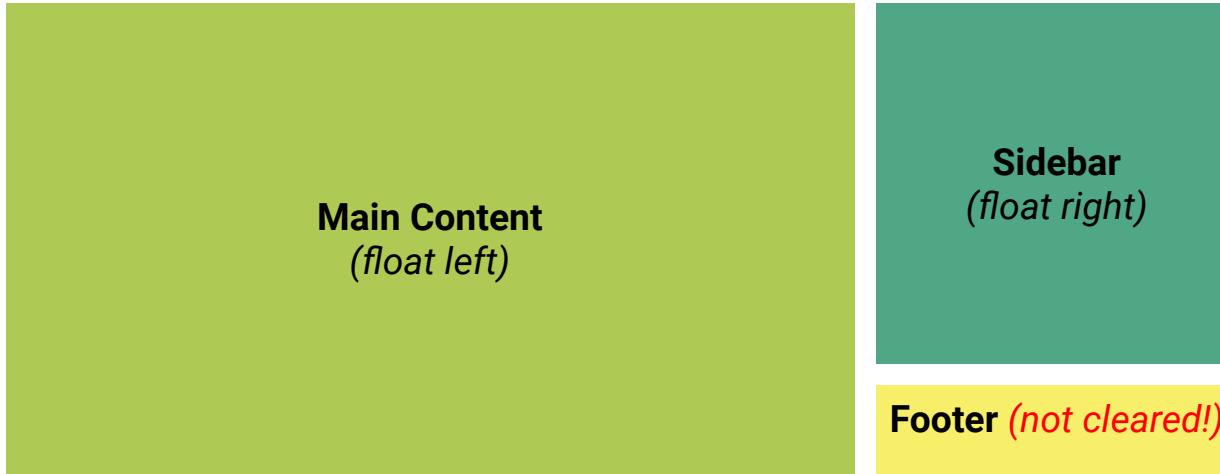


CSS

```
#sidebar {  
    float: left;  
}  
  
#main-content {  
    float: right;  
}
```

Clearing the Float

However, floats often get in the way of layouts. Sometimes we don't want to give each element the “in-line” treatment.

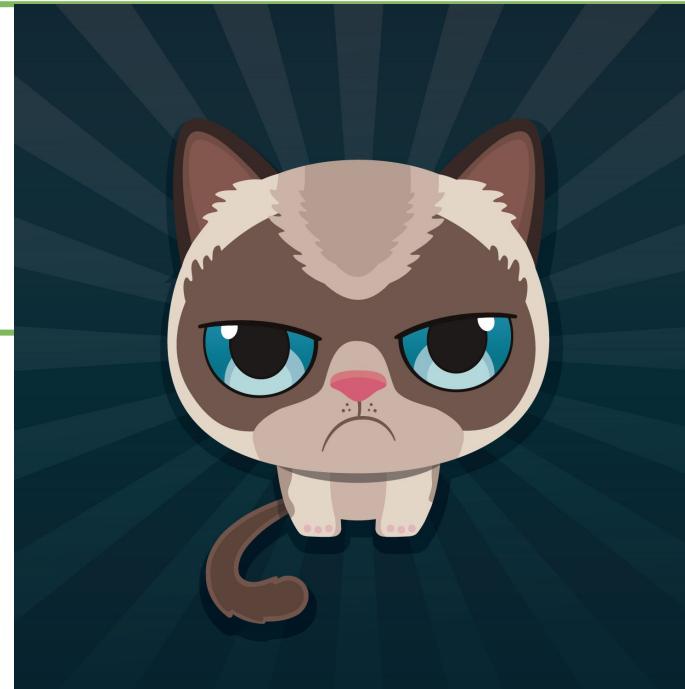


clearfix Hack

Sometimes when elements don't match up in size, we get situations like this:

<div>

Uh-oh! The image is taller than the element containing it, and it's floated, so it's overflowing outside of its container!

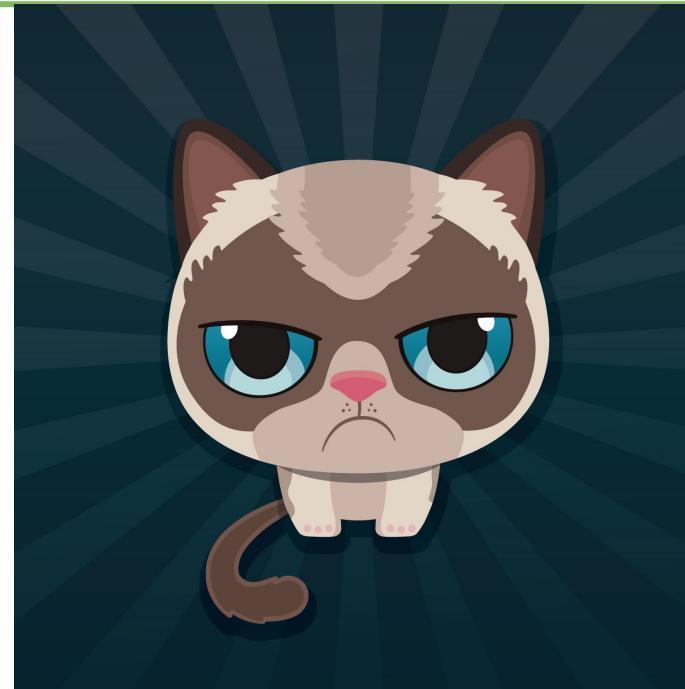


clearfix Hack

We can get around this by using the **clearfix hack**.

```
<div class="clearfix">
```

Much better!



clearfix Hack



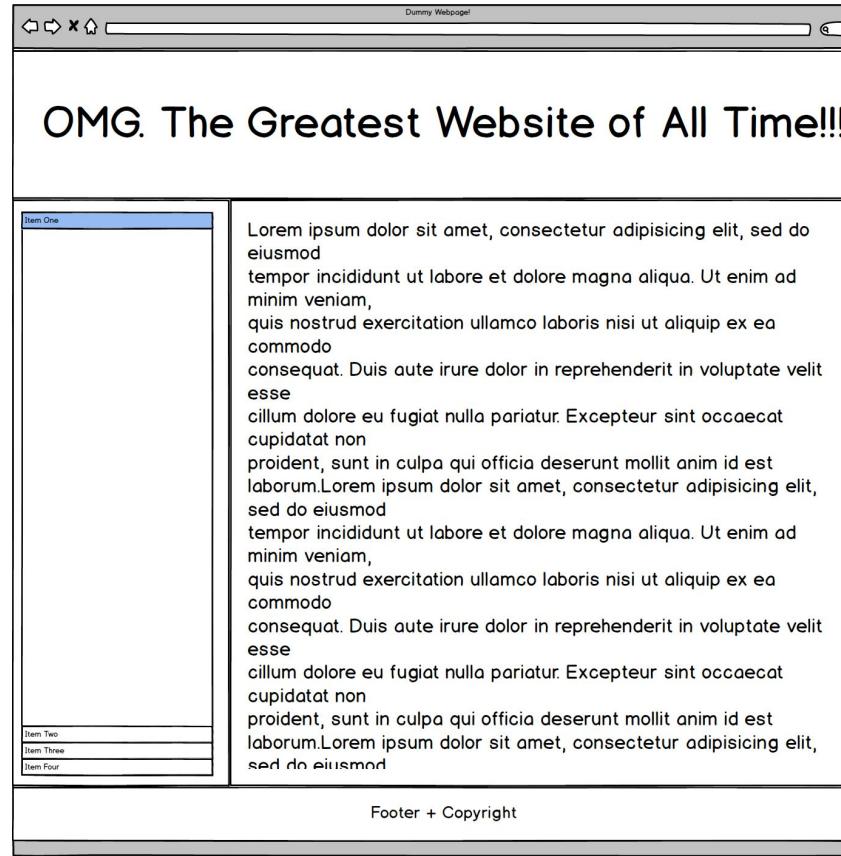
`::after` is what we call a pseudo-element. We use it to style specific parts of an element.



This will add an HTML element, hidden from view, after the content of the `.clearfix` element.
This clears the float.

```
.clearfix::after {  
    content: "";  
    display: block;  
    clear: both;  
}
```

Quick Demo



Quick Demo



2000 x 200



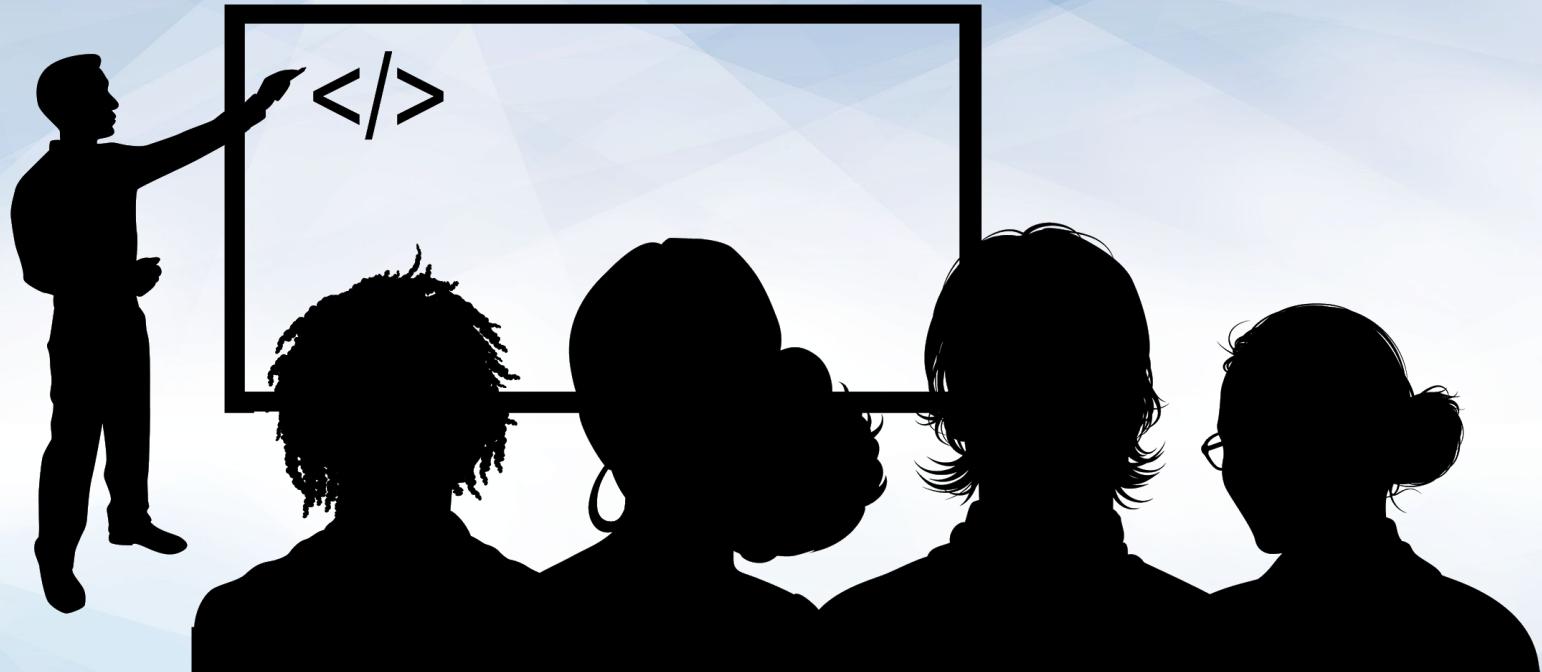
300 x 400



900 x 400



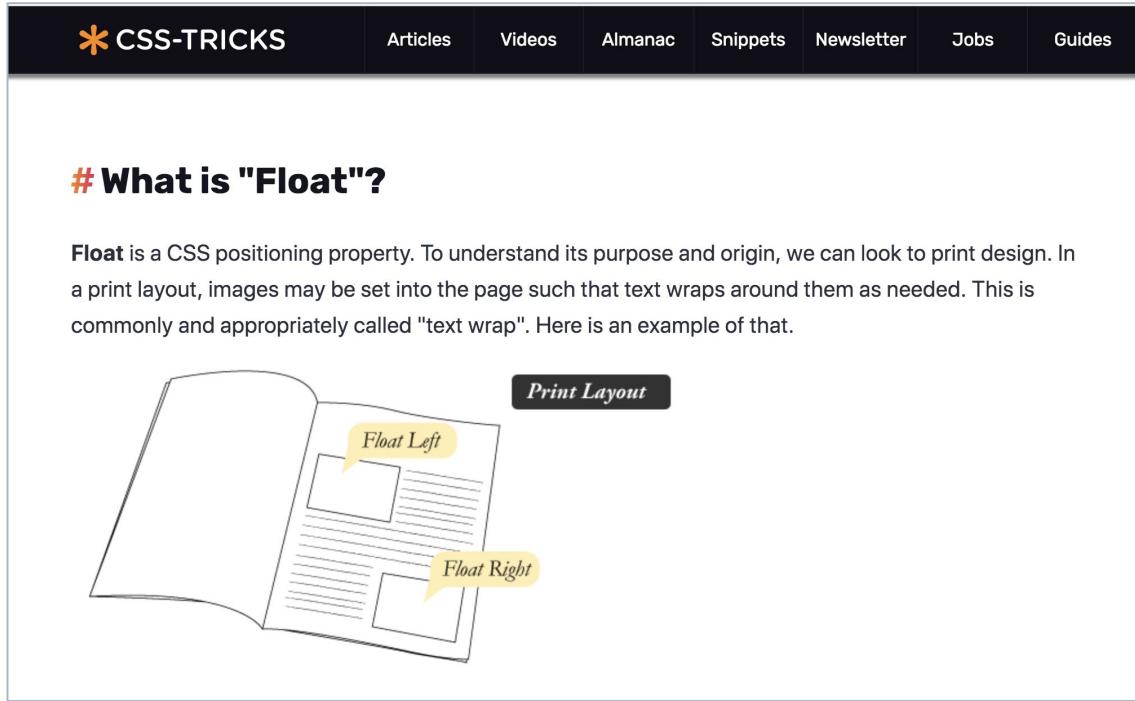
500 x 100



Instructor Demonstration
Floats

A Fantastic Guide to Floats

To all serious frontend developers, this is a necessary read:
css-tricks.com

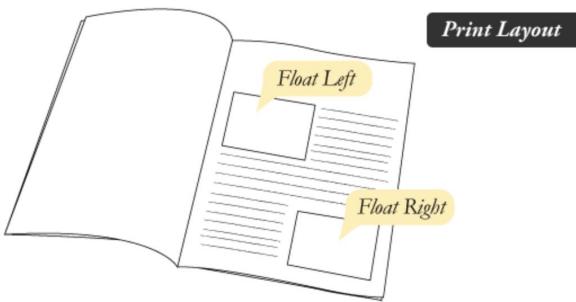


The screenshot shows the top navigation bar of the CSS-Tricks website, which includes links for Articles, Videos, Almanac, Snippets, Newsletter, Jobs, and Guides. Below the navigation, the main content area features a section titled "# What is "Float"?" with a sub-section titled "Print Layout". An illustration of an open book is shown, with text blocks labeled "Float Left" and "Float Right" indicating how content wraps around floated elements.

What is "Float"?

Float is a CSS positioning property. To understand its purpose and origin, we can look to print design. In a print layout, images may be set into the page such that text wraps around them as needed. This is commonly and appropriately called "text wrap". Here is an example of that.

Print Layout





Activity:

Float Layout

Suggested Time:
30 minutes



Activity: Float Layout



In this activity, you'll flex your newfound floating skills by creating a conceptual layout. Eyeball the design to your best ability.

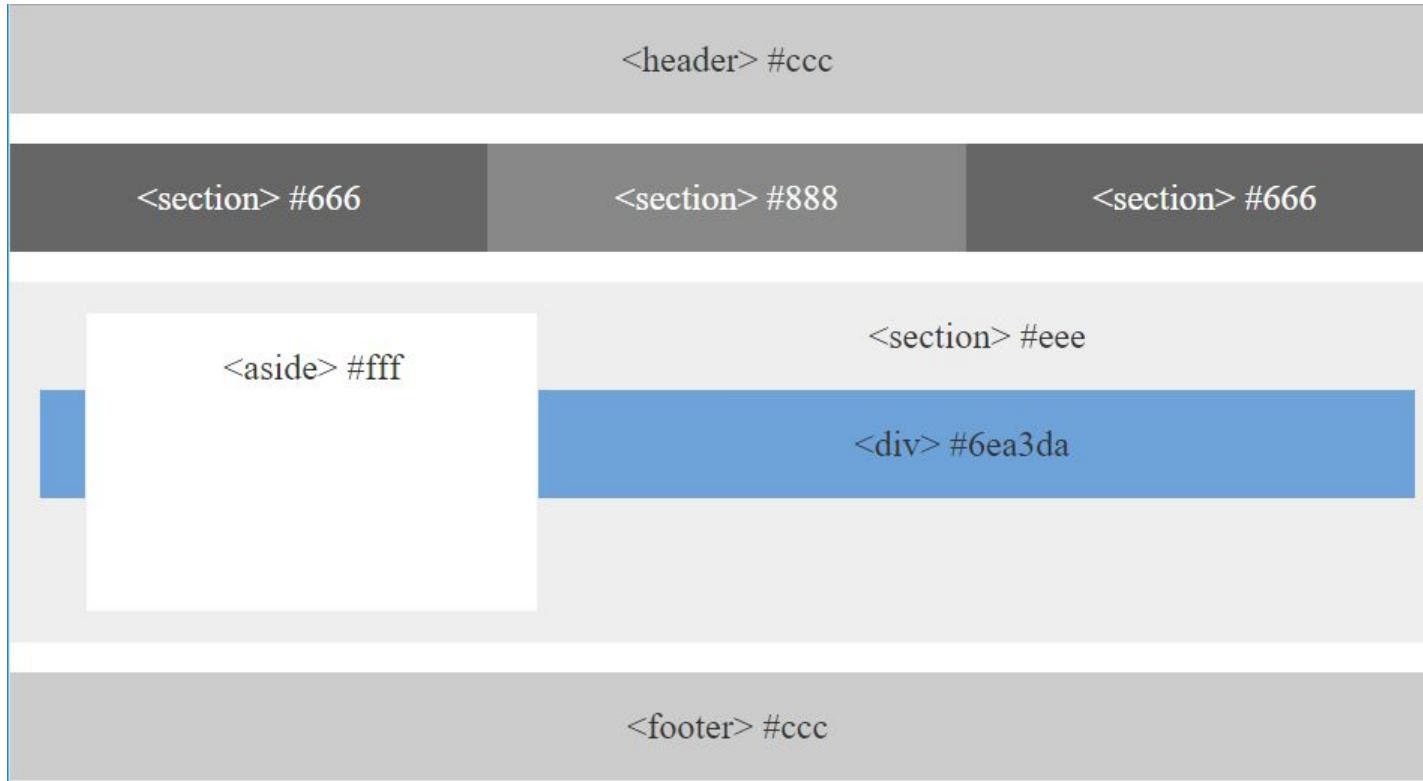


Check Slack for additional instructions.

Suggested Time: 30 Minutes



Activity: Float Layout



Suggested Time: 30 Minutes



Good work!

Your brain may rest now.





Video Walkthrough (Highly Recommended)

[Floats in CSS](#)

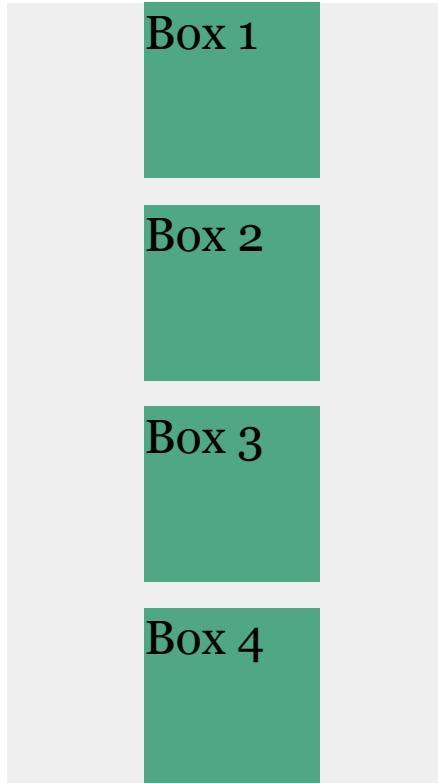
Take a Break!



CSS Positioning

Position: Static (Default)

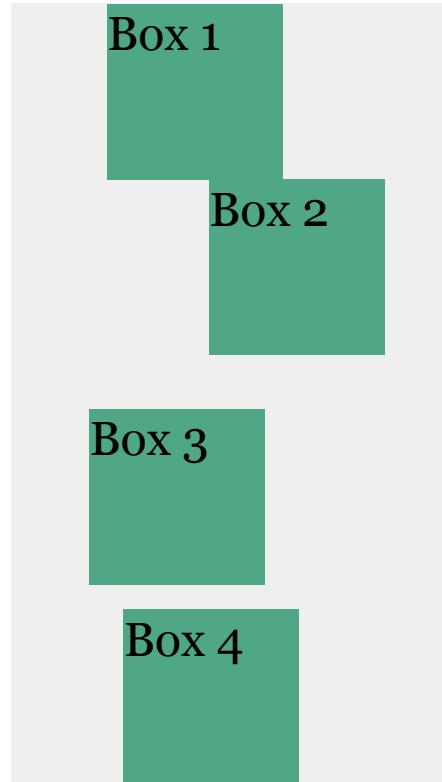
Four boxes placed statically (default):



Position: Relative

Switching the boxes to relative will nudge the boxes in relation to their “original” location.

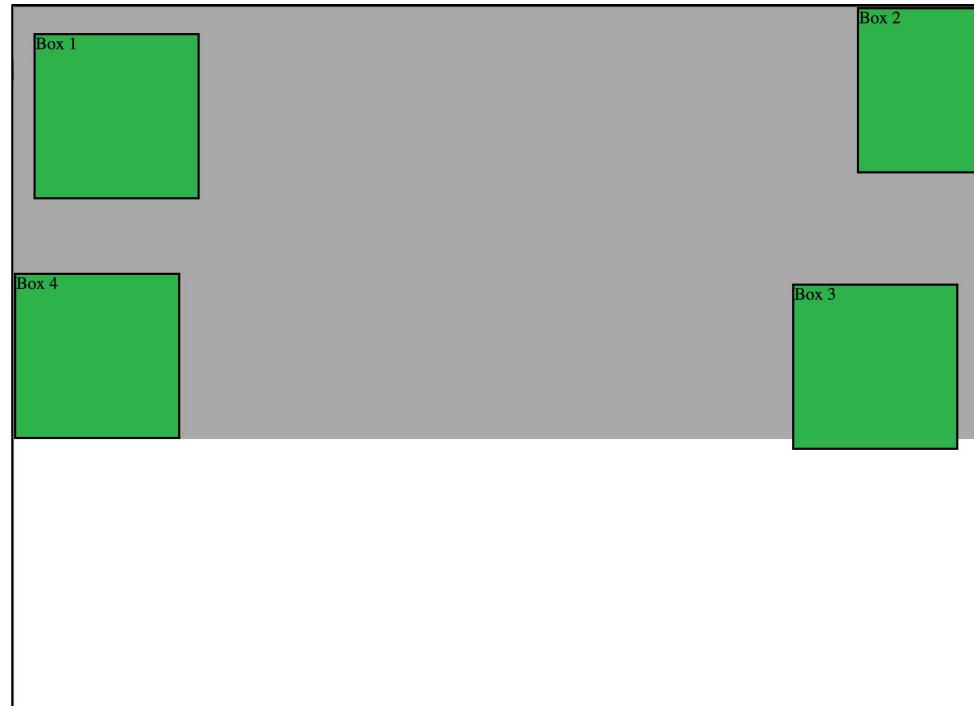
```
.box {  
  background: #2db34a;  
  height: 80px;  
  position: relative;  
  width: 80px;  
}  
.box-1 {  
  top: 20px;  
}  
.box-2 {  
  left: 40px;  
}  
.box-3 {  
  bottom: -10px;  
  right: 20px;  
}
```



Position: Absolute

Positioned relative to nearest positioned ancestor

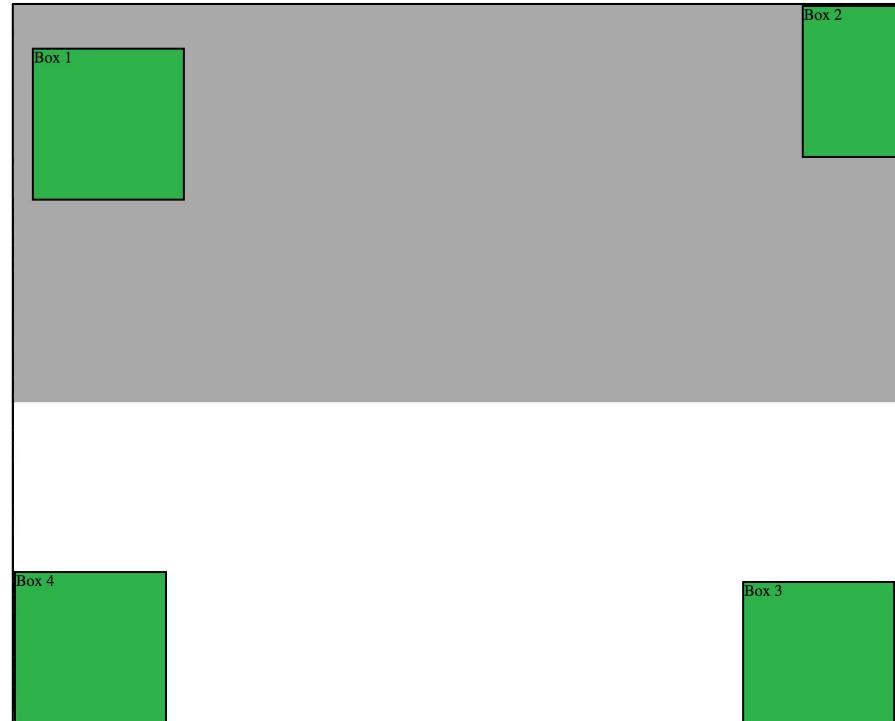
```
.box-set {  
  height: 400px;  
  background: darkgray;  
  position: relative;  
}  
.box {  
  position: absolute;  
  height: 150px;  
  width: 150px;  
  background: #2db34a;  
  border: 2px solid black;  
}  
.box-1 {  
  top: 6%;  
  left: 2%;  
}  
.box-2 {  
  top: 0;  
  right: -40px;  
}  
.box-3 {  
  bottom: -10px;  
  right: 20px;  
}  
.box-4 {  
  bottom: 0;  
}
```



Position: Fixed

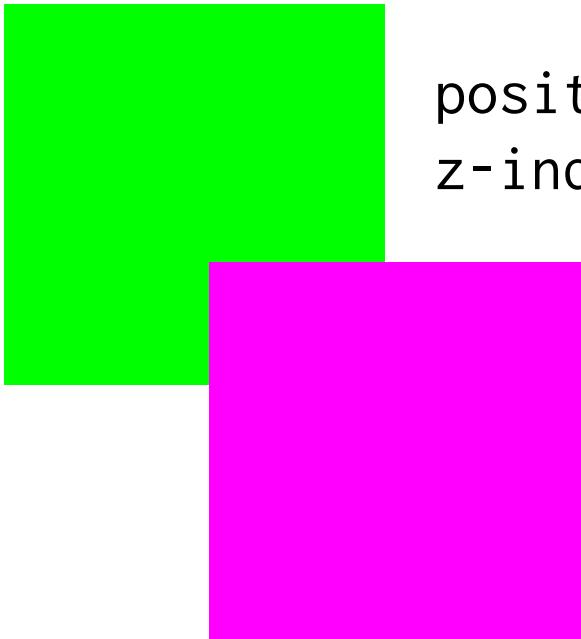
Position with exact coordinates in the browser window

```
.box-set {  
  height: 400px;  
  background: darkgray;  
}  
.box {  
  position: fixed;  
  height: 150px;  
  width: 150px;  
  background: #2db34a;  
  border: 2px solid black;  
}  
.box-1 {  
  top: 6%;  
  left: 2%;  
}  
.box-2 {  
  top: 0;  
  right: -40px;  
}  
.box-3 {  
  bottom: -10px;  
  right: 20px;  
}  
.box-4 {  
  bottom: 0;  
}
```



Layering with z-index

The z-index property allows you to layer elements on top of each other.



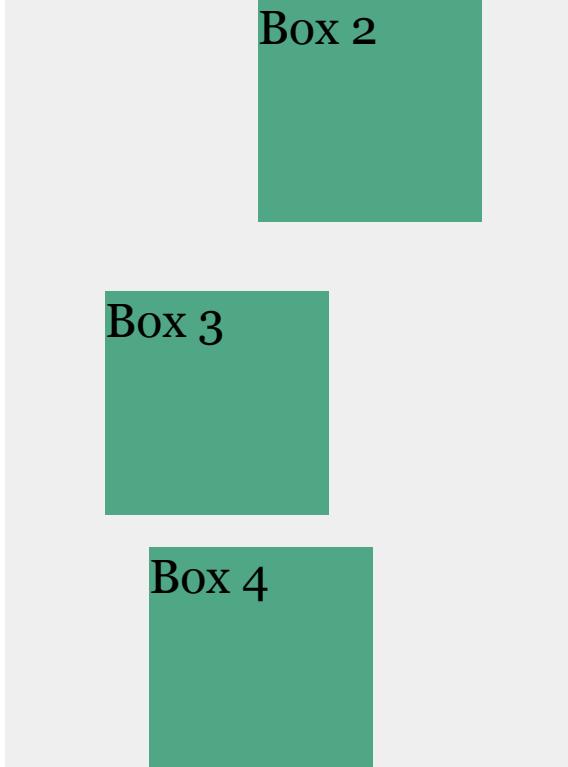
position: absolute;
z-index:1;

position: absolute;
z-index:2;

Hiding Things

Display: none allows you to hide elements from view.

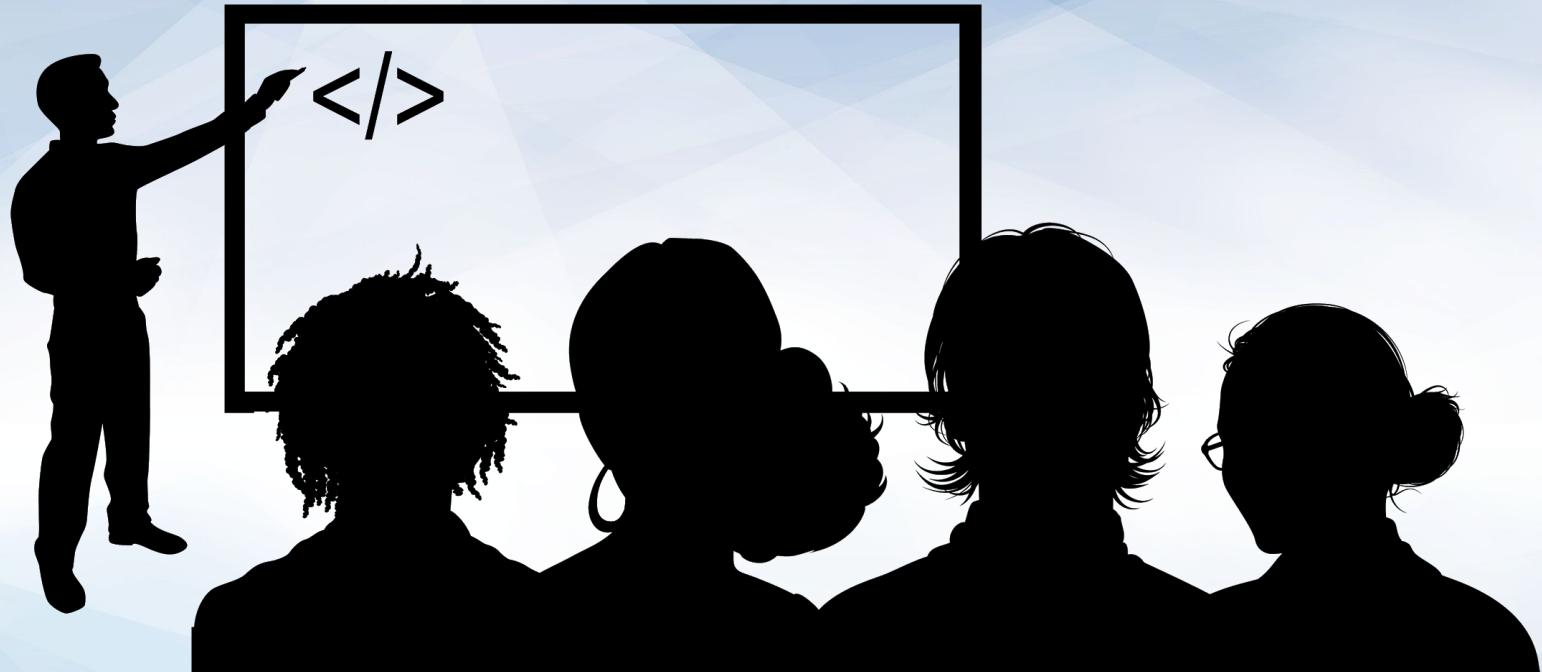
This will become useful in later sections, when we'll hide and reveal specific HTML elements of our choosing.



Box 2

Box 3

Box 4



Instructor Demonstration
CSS Positioning



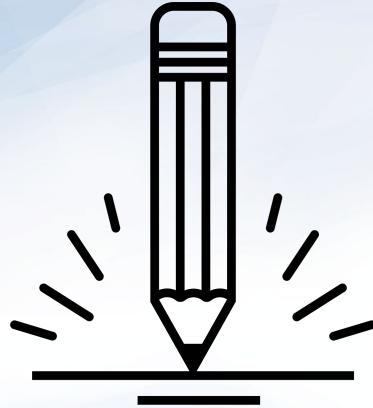
Time For a Quick Video

[Positioning in CSS](#)

Great Resource

Another great read for frontend developers:
learn.shayhowe.com





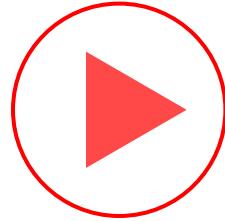
Activity: CSS Positioning

In this activity, you'll flex your newfound positioning skills by creating another conceptual layout. Eyeball the design to your best ability.

(Check Slack for additional instructions)

Suggested Time:
30 minutes





Video Walkthrough (Highly Recommended)

[CSS Positioning Layout](#)

Advice

Tips to Keep Moving Forward

01

Redo this at home.

We designed the class activities to firm up your HTML/CSS skills. Try them again at home.

02

REMEMBER!

The best way to learn web development is to practice, practice, practice!

Chrome Inspector

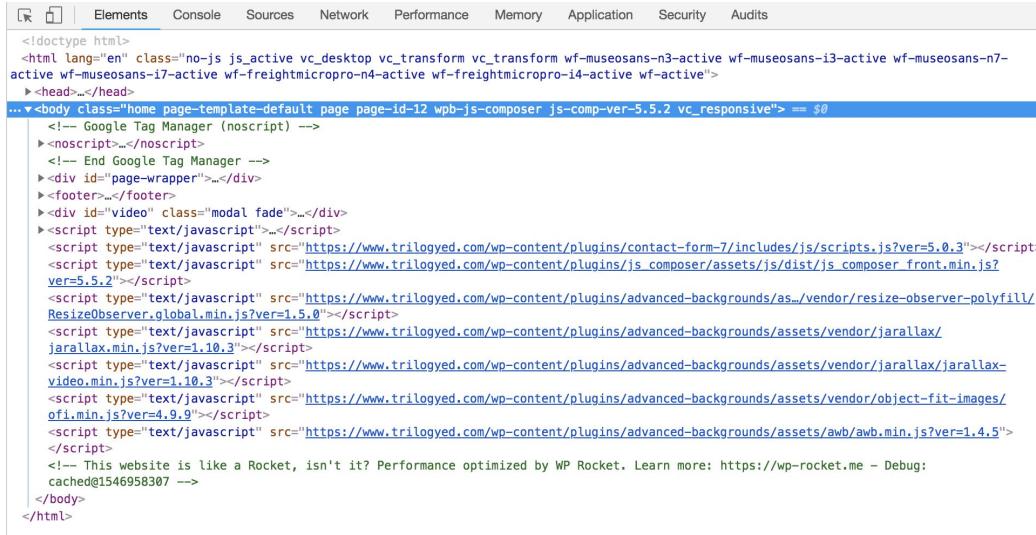
Chrome Inspector Is Your Friend



To access the Chrome inspector, right-click on a page and select **Inspect**.



It lets you inspect the HTML, CSS, and more.



A screenshot of the Chrome DevTools interface, specifically the Elements tab. The tab bar at the top includes Elements, Console, Sources, Network, Performance, Memory, Application, Security, and Audits. The main area displays the HTML source code of a webpage. A blue selection bar highlights a portion of the code within the tag, specifically the section starting with `<!-- Google Tag Manager (noscript) -->` and ending with `</body>`. The highlighted code includes various script tags and comments related to Google Tag Manager and other site scripts.

```
<!doctype html>
<html lang="en" class="no-js js_active vc_desktop vc_transform vc_transform wf-museosans-n3-active wf-museosans-i3-active wf-museosans-n7-active wf-museosans-i7-active wf-freightmicropn4-active wf-freightmicropn4-active wf-active">
  <head>...
    <!-- Google Tag Manager (noscript) -->
  </head>
  <body class="home page-template-default page-id-12 wpb_js_composer js-comp-ver-5.5.2 vc_responsive"> == $0
    <!-- Google Tag Manager (noscript) -->
  </body>
</html>
```

Chrome Inspector Is Your Friend



You can even edit the HTML/CSS in a webpage and instantly view your changes in the browser!



This works on any website, whether it's yours or not.

The screenshot shows the Chrome DevTools Elements tab with the following details:

- HTML View:** Displays the full HTML code of the page, including scripts and styles. A portion of the code is highlighted in blue, indicating an active selection for modification.
- Styles Tab:** Shows the current CSS rules applied to the selected element. It includes a "Filter" bar set to ":hov .cls +". The rules are organized into sections:
 - `element.style { }`: Contains vendor-specific font-smoothing rules.
 - `* { }`: Contains vendor-specific box-sizing rules.
 - `script { }`: Contains a user agent stylesheet rule for `display: none;`.
 - Inherited from** section: Shows styles inherited from the `body.home.page-template-default...` selector, including a `base.scss:2` rule for `position: relative;`, `min-height: 100%;`, and `font-family: freight-micro-pro;`.
 - `body { }`: Contains a `scaffolding.scss:28` rule for `font-family: "Helvetica Neue", Helvetica, Arial, sans-serif;`, `font-size: 14px;`, `line-height: 1.42857;`, `color: #333;`, and `background-color: #fff;`.



We'll come back to
this in our next class.

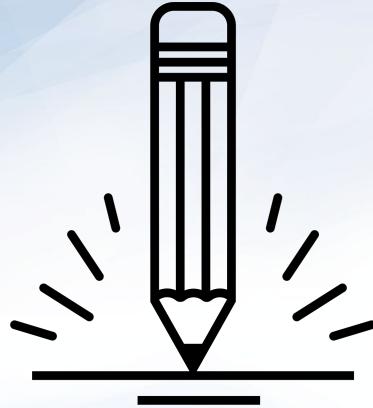
Recap

You Got This!



Questions?

Extra Material



Activity: Chrome Inspector

Choose a website you commonly visit (Amazon, Google, Huff Po, etc.) and heavily modify it using Chrome DevTools.

Suggested Time:
15 minutes



Activity: Chrome Inspector

Using Chrome Inspector, try to modify the following:



Content (change text)



Colors



Spacing



Any other CSS style rules

When you're done, send a screenshot to your class's Slack channel.



Suggested Time: 15 minutes

CSS Resets

Loading Multiple CSS Files

We can link our documents to more than one stylesheet at a time—one of the most powerful features of CSS/HTML.

By tapping into different stylesheets simultaneously, we can create complex layouts with plenty of design rules.

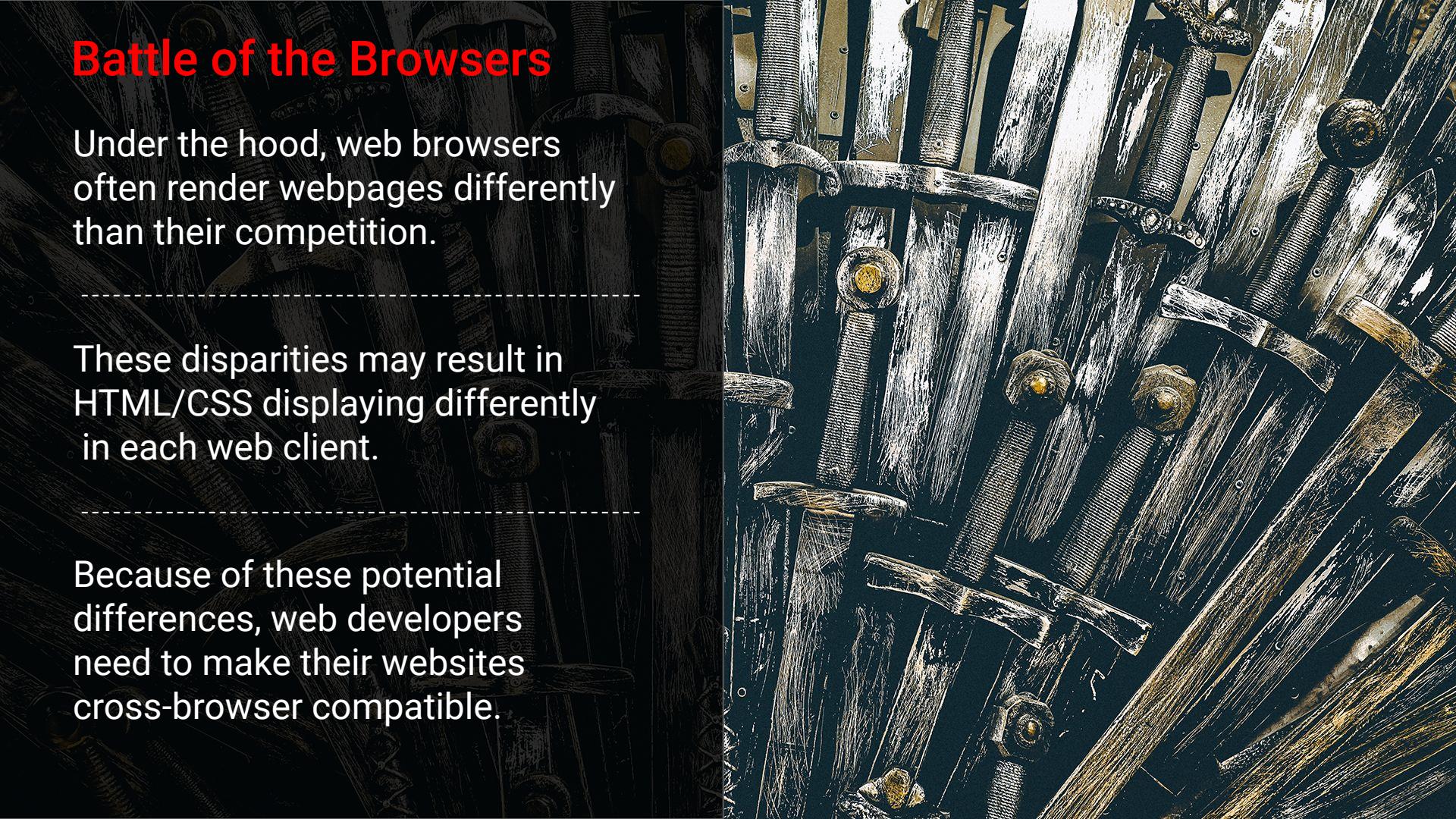


Just remember:
the loading
order matters!

```
<!DOCTYPE html>
<html>
<head>
    <title>Multiple CSS Files!!</title>
    <link rel="stylesheet" href="assets/style1.css">
    <link rel="stylesheet" href="assets/style2.css">
    <link rel="stylesheet" href="assets/style3.css">
</head>
<body>
    <header>
```

By a show of hands...
Which browser do you use?

Battle of the Browsers



Under the hood, web browsers often render webpages differently than their competition.

These disparities may result in HTML/CSS displaying differently in each web client.

Because of these potential differences, web developers need to make their websites cross-browser compatible.

Reset.css (or Normalize.css)

Reset.css will “reset” all browser-specific CSS. This means your site will appear the same in all browsers.

However, you will have to restyle everything yourself.

Header 1
Header 2
Header 3

Lore ipsum dolor sit amet, consectetur adipisicing elit. Molestiae at consectetur iste dignissimos maiores placeat deleniti eum dolore, velit ab similique eligendi commodi perspicatis excepturi labore facere. Ad, excepturi, distinctio?

Lore ipsum dolor sit amet, consectetur adipisicing elit. Soluta, reiciendis molestias inventore blanditiis ratione amet, dolore id doloribus minus iure esse, accusantium qui ex, nesciunt? Officiis, animi saepe libero quae!



List Item 1
List Item 2
List Item 3
List Item 4
[Google.com](#)
[Facebook.com](#)
[Ebay](#)

Jill Eve	Smith Jackson	50 94
-------------	------------------	----------

Why CSS Resets Matter

01

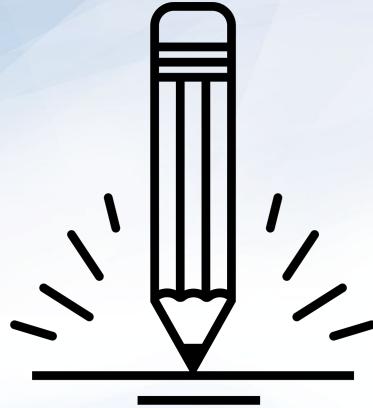
They help create browser-compatible websites.

02

They are an example of using someone else's CSS in *your* website!

03

They are a common topic in front-end developer interviews.



Activity: Reset.css

Incorporate a reset.css file in a basic HTML file. (Instructions sent out)

Note how the reset file impacts the styling of your HTML file.

Suggested Time:
10 minutes

