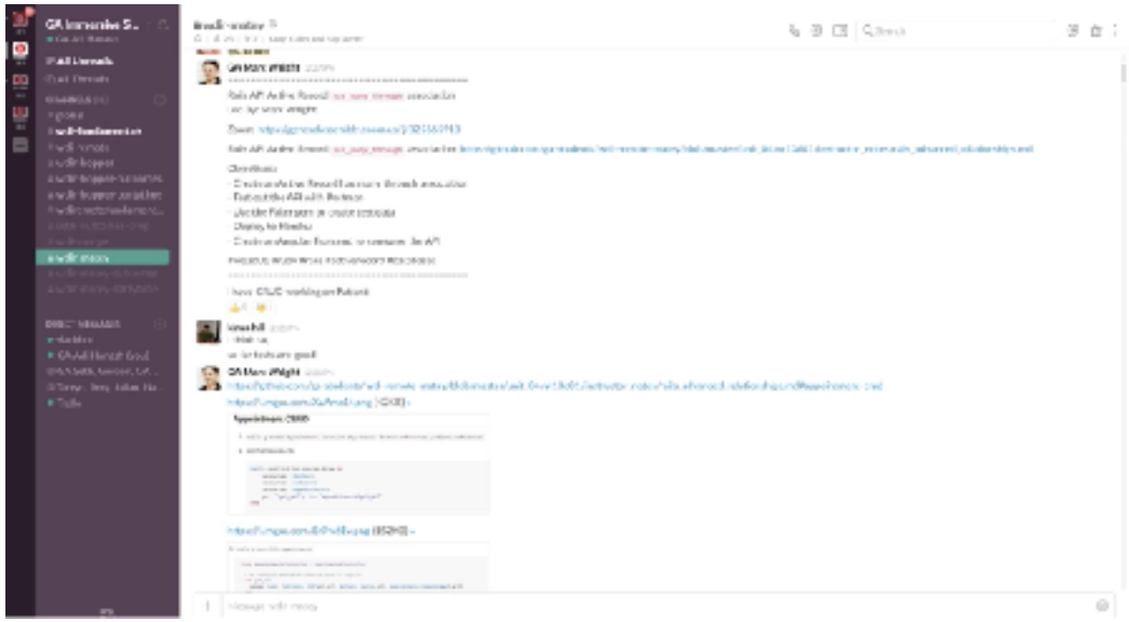
LET'S GET EVERYTHING SET UP!

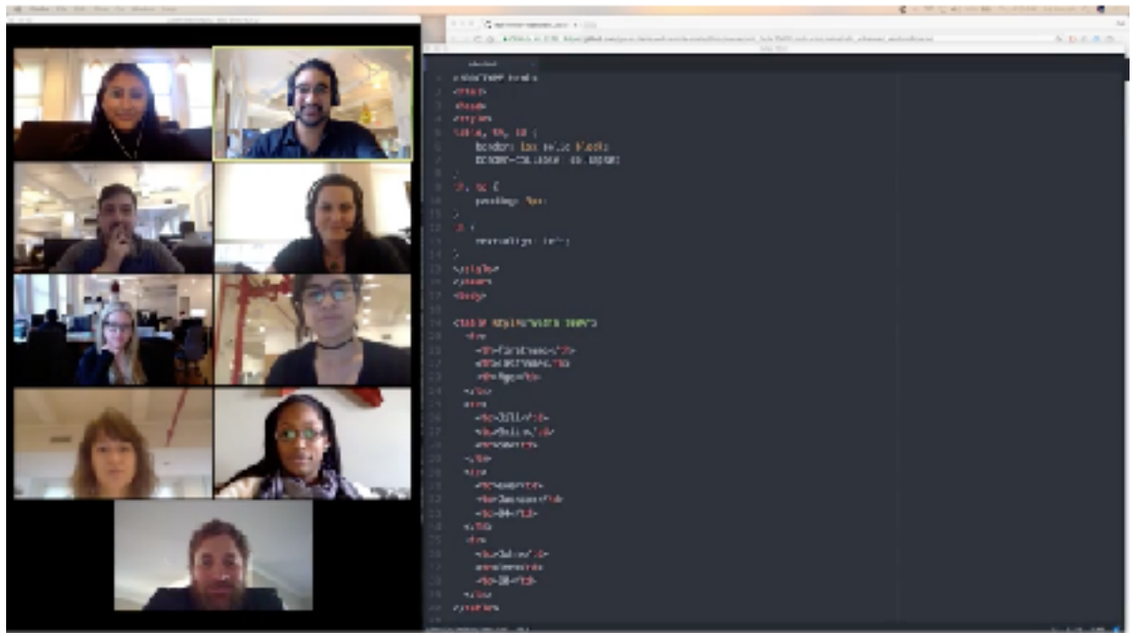
1. Navigate to the FEWD Dashboard and download the Lesson 3 starter code and slides. You'll want to keep the dashboard open for other links and resources we'll be referencing in class.
2. Move the starter code and slides from your Downloads folder to the **fewd** folder we added to your Desktop last class.
3. Double-click on starter code folder to unzip it
4. After you've unzipped, be sure to delete the original .zip file to avoid confusion and make sure you don't unzip it again later!
5. Open the entire **fewd** folder with Atom (either drag and drop the folder on the Atom icon in the dock on Mac, or open Atom and go to file > open… and select the starter code folder.
6. Log into Slack and join the class 3 channel.

SETUP — WITH EXTERNAL MONITOR

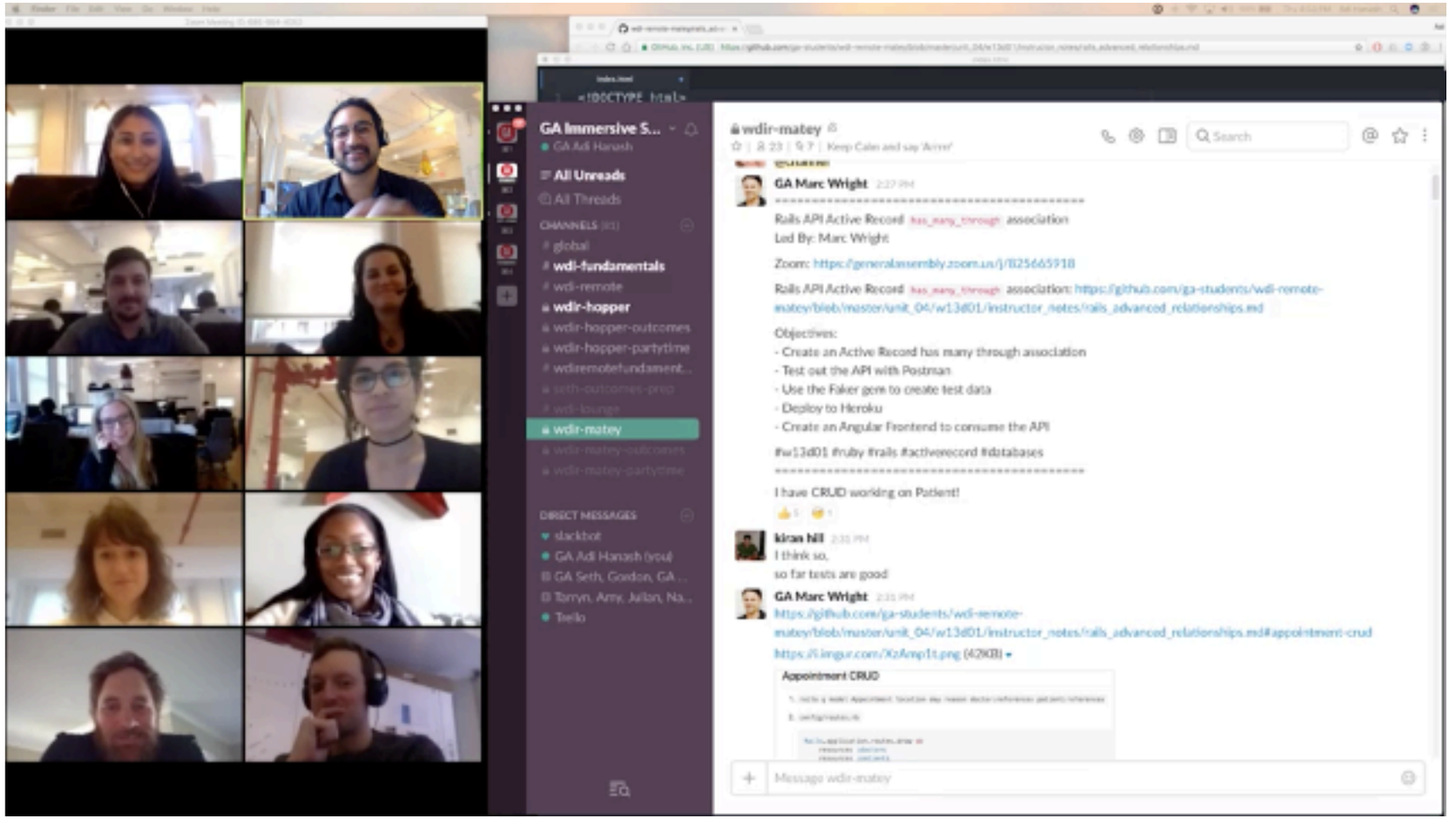
PRIMARY MONITOR:



EXTERNAL MONITOR:



SETUP — NO EXTERNAL MONITOR



WEEKLY OVERVIEW

WEEK 2	CSS Selectors / Layout
WEEK 3	Layout Lab & Wireframing / Responsive Design

AGENDA

- Review
- Relative URLs — Part 2
- CSS Selectors
- Lab Time

LEARNING OBJECTIVES

- Use padding and margin to add spacing to a page.
- Describe inheritance in CSS.
- Add classes and IDs to HTML elements and apply CSS styles to elements based on class and ID.
- Apply CSS to elements based on their relationships.

TODAY'S LAB:

CSS SELECTORS

REVIEW

ACTIVITY

KEY OBJECTIVE

- Continue practicing HTML & CSS

TYPE OF EXERCISE

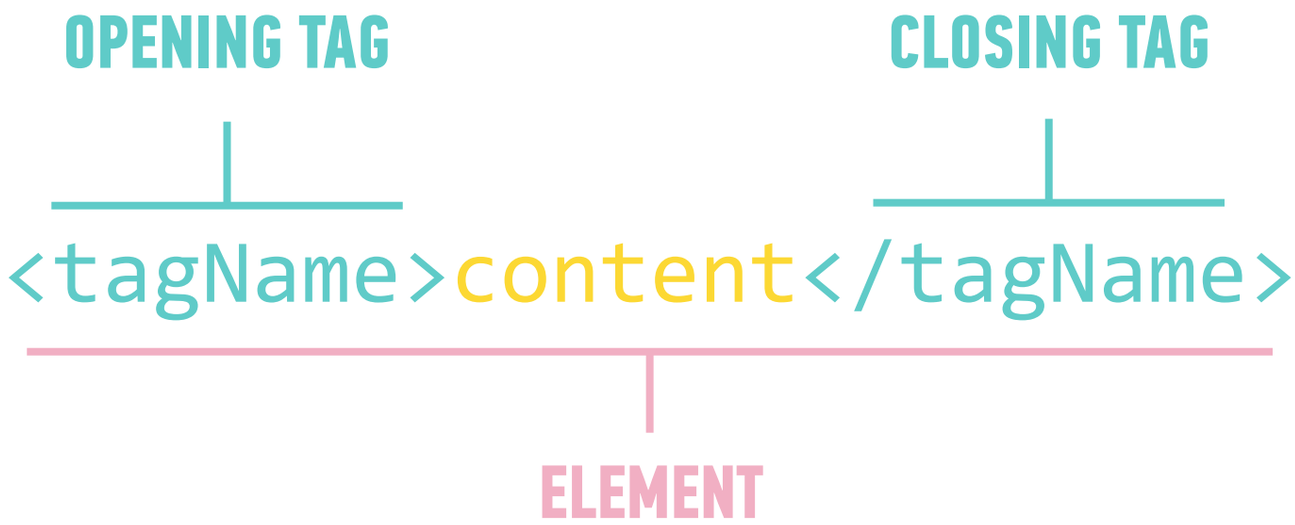
- Individual

TIMING

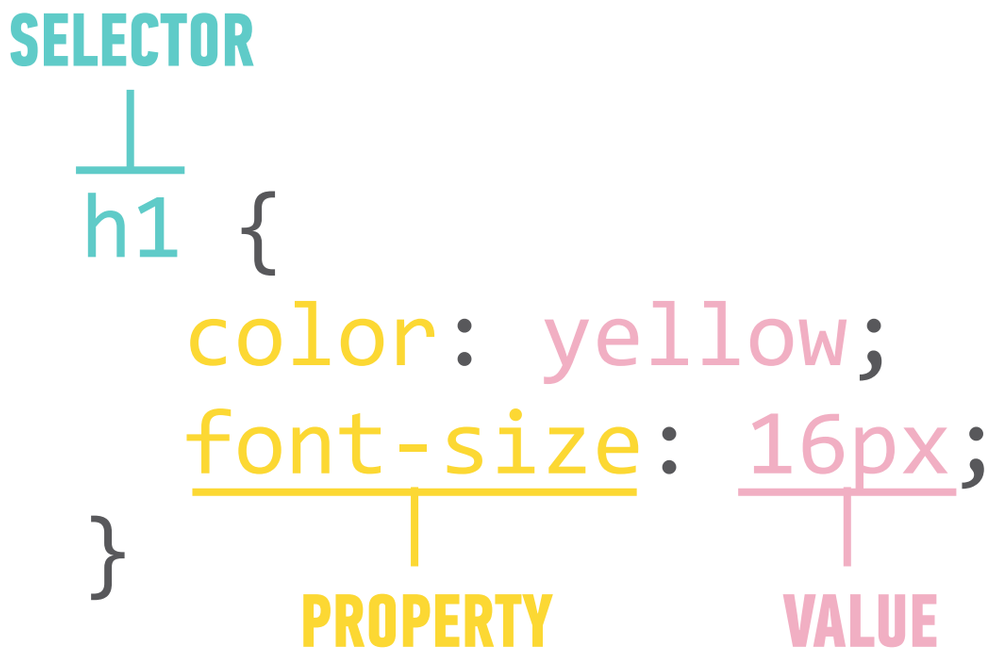
10 min

1. Screen share your Lazy Days site with your groups.
2. What were you excited to be able to accomplish? What would you like to be able to add in the future?
3. Were you able to complete any of the bonus tasks? Adding padding and margin? Hover effect? A favicon? Show your group how you did it!

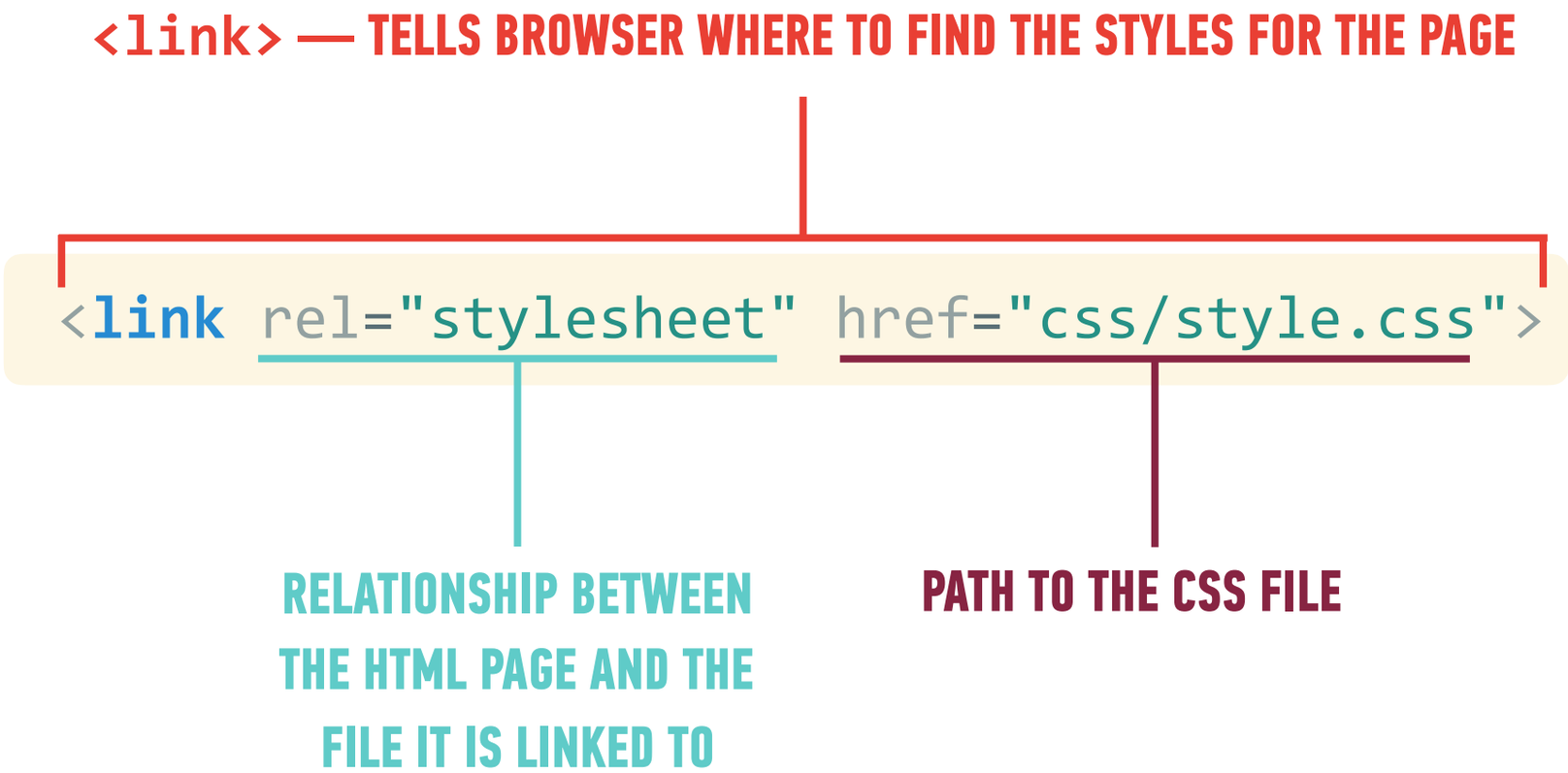
HOW DO I WRITE HTML?



HOW DO I WRITE CSS?



ADDING A CSS FILE — THE LINK ELEMENT



ATOM SHORTCUTS

`link + tab` then put the path to the CSS file for the `href`

```
link + tab   <link rel="stylesheet" href="">
```

ADDING A CSS FILE

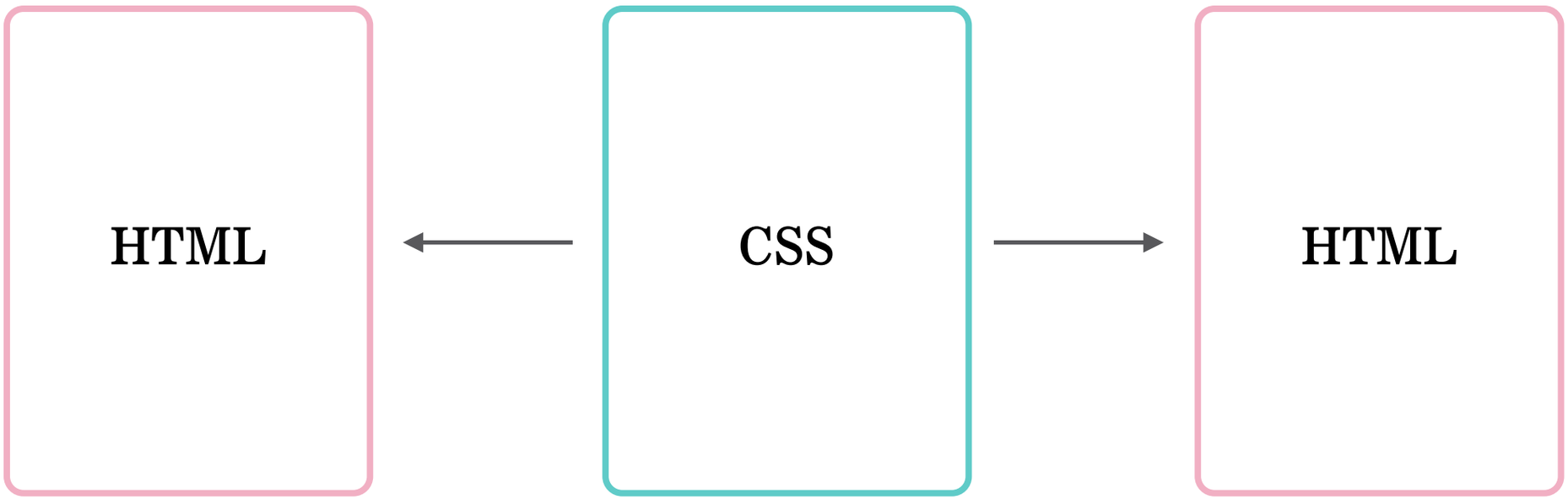
MUST GO IN THE <HEAD> OF THE DOCUMENT!!!

```
<head>
    <meta charset="UTF-8">
    <title>My Website</title>
    <link rel="stylesheet" href="css/style.css">
</head>
```

INTERNAL VS. EXTERNAL CSS

BENEFITS OF USING AN EXTERNAL STYLESHEET:

- Multiple pages can use same stylesheet (Don't repeat yourself!)
- Only have to make changes in one file
- Keep content separate from presentation



HTML5 STRUCTURAL ELEMENTS

nav – Used to group together major navigational blocks on a page

```
<nav>
    <a href="home.html">Home</a>
    <a href="about.html">About</a>
    <a href="contact.html">Contact</a>
</nav>
```

To add styles to the **nav**:

```
nav {
  background-color: #dfdfdf;
}
```

HTML5 STRUCTURAL ELEMENTS

`header` – used to group elements in the opening section of a page.

Examples of content that may be inside of the header:

- any main headings on your site
- the navigation
- a search form for the site
- the site logo
- any other introductory content.

```
<header>
    <h1>My Portfolio Site</h1>
    <nav>
        <a href="home.html">Home</a>
        <a href="about.html">About</a>
        <a href="contact.html">Contact</a>
    </nav>
</header>
```

To add styles to the **`header`**:

```
header {
  background-color: #dfdfdf;
}
```

HTML5 STRUCTURAL ELEMENTS

`footer` – Used for the concluding section of a page.

Examples of content a footer might include:

- copyright info
- social media links

```
<footer>
    <p>Copyright 2017.</p>
</footer>
```

To add styles to the **`footer`**:

```
footer {
  background-color: #dfdfdf;
}
```

PSEUDO CLASSES

Test hover effects here

```
/* unvisited link */
a:link {
    color: #FF0000;
}

/* visited link */
a:visited {
    color: #00FF00;
}

/* mouse over link */
a:hover {
    color: #FF00FF;
}

/* selected link */
a:active {
    color: #0000FF;
}
```

FAVICONS

```
<head>
    <meta charset="UTF-8">
    <title>Lazy Days</title>
    <link rel="shortcut icon" href="favicon.ico" type="image/x-icon" >
    <link rel="stylesheet" href="css/style.css">
</head>
```

Article on adding a favicon

GITHUB

WHY WOULD WE USE GITHUB?

- Think back to the last time you wrote a report or paper.
- Did you happen to save different versions, like `myawesomepaper_v1`, `myawesomepaper_v2`, etc?
- That's called **version control**– and managing it on your own can be tricky.
- Numbering file names can help you understand which file is the most recent, but how can you tell the difference between each version?

WHY WOULD WE USE GITHUB?

- This is where GitHub can help! GitHub is a free, open-source tool where developers store code online.
- GitHub automatically tracks any changes you make to your code. It allows you to compare versions of your code, add comments and notes to different versions, share your code, and more!

WHY USE GITHUB?

HISTORY

- Since GitHub stores a history of the code, it allows developers to go back in time if something breaks.

COLLABORATION

- Allows multiple developers to work on the same project. Much like Google Drive lets multiple people collaborate on the same document, GitHub allows this for code.
- You can see who worked on what.

FEEDBACK

- GitHub allows for feedback to be given on the code which, hopefully, increases code quality.

SETTING THINGS UP

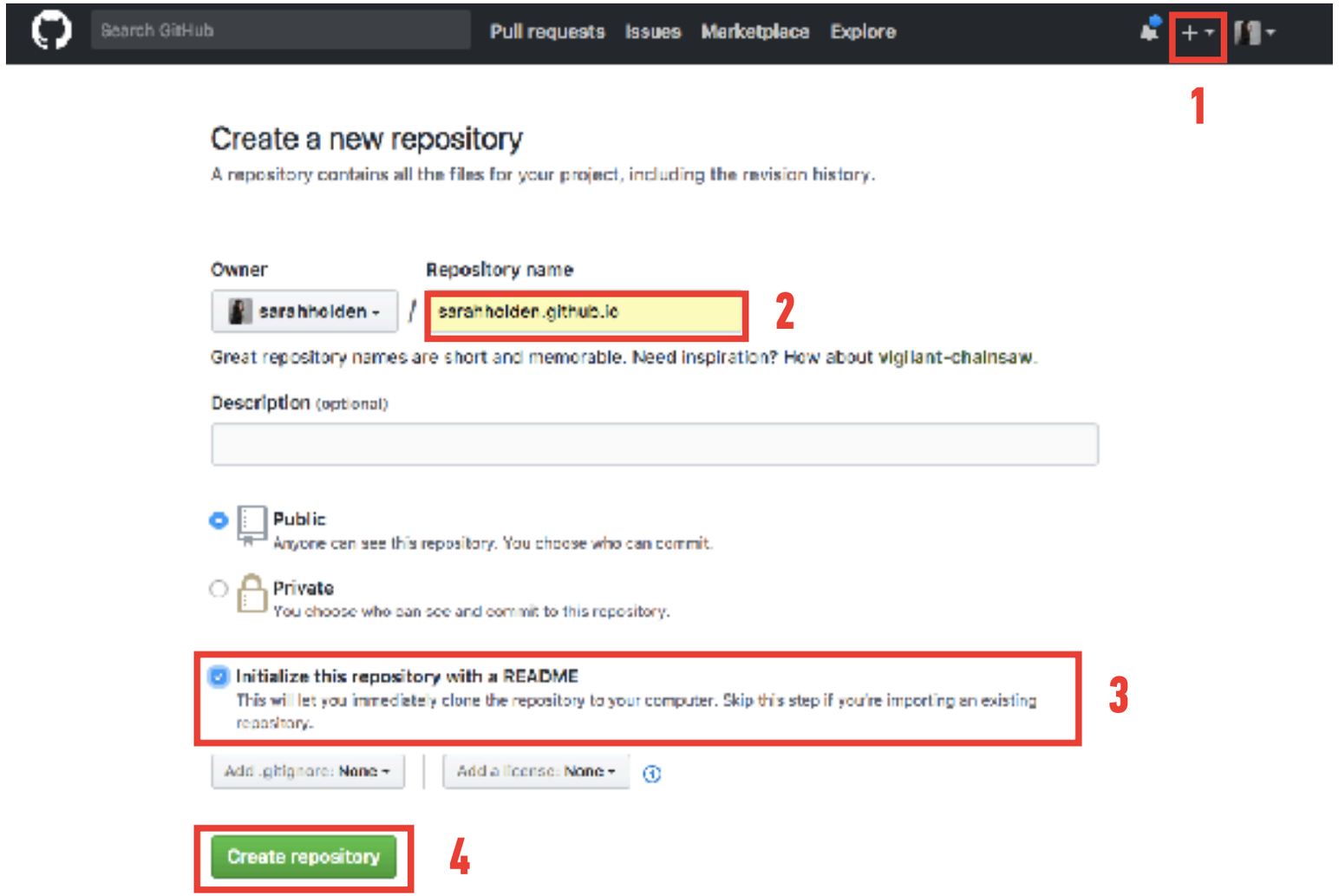
GITHUB SETUP

In GitHub, the folder in the cloud where you can store, update, and share all the files that make up your website is called a **repository**.

GITHUB SETUP

Do these steps in **Chrome**. Other browsers may not have the drag and drop feature enabled.

GITHUB SETUP



1. Click the + button in the upper right corner

2. Enter a repository name. You'll want to name it using a specific format:

   `yourusername.github.io`

   for example:

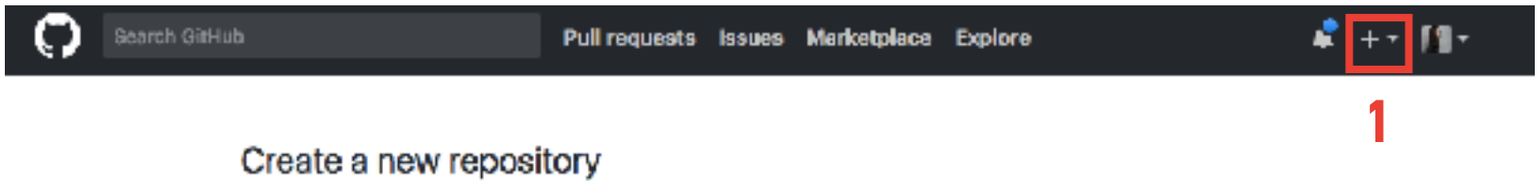
   sarahholden.github.io

3. Check the initialize this repository with a README checkbox

4. Click the "Create Repository" button.

GITHUB SETUP

1. Click the + button in the upper right corner



GITHUB SETUP

2. Enter a repository name. You'll want to name it using a specific format:

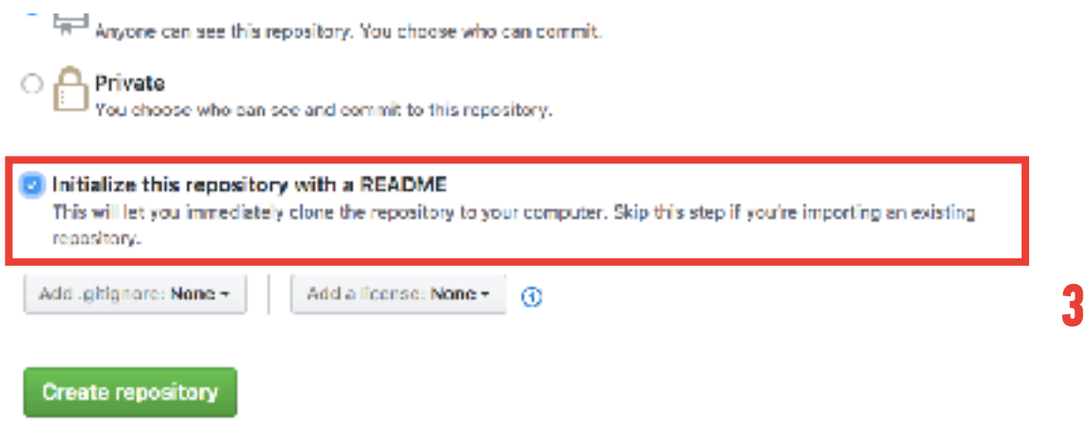
   `yourusername.github.io`

   for example:

   sarahholden.github.io

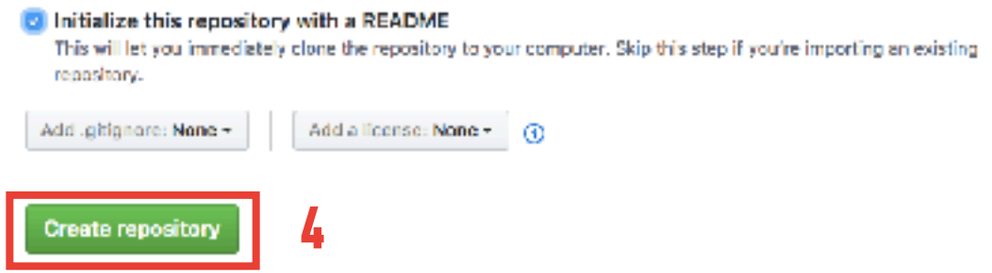
GITHUB SETUP

3. Check the "Initialize this repository with a README" checkbox



GITHUB SETUP

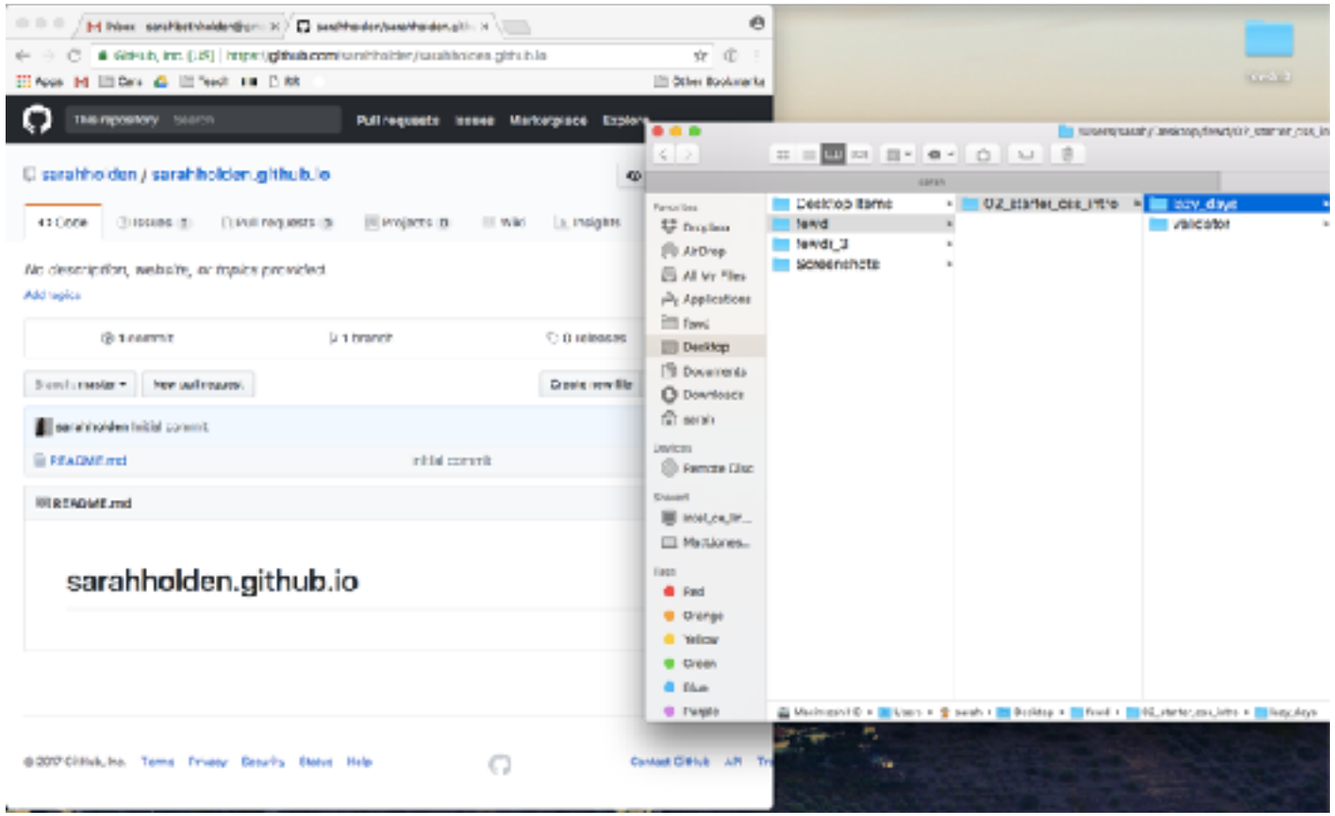
4. Click the "Create Repository" button.



GITHUB SETUP

5. Drag and drop the lazy_days folder (your homework) from the fewd folder to this screen.

Make sure to drag and drop the entire folder, not just the files inside the folder!!

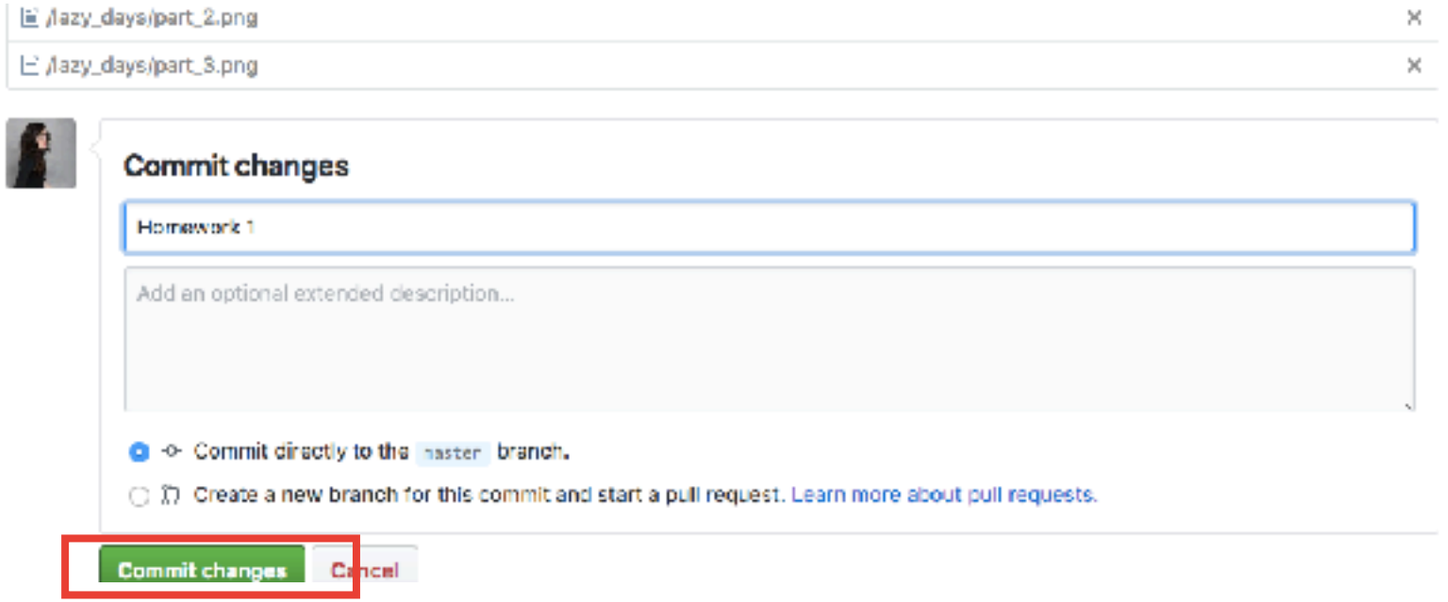


GITHUB SETUP

6. Scroll down and enter a commit message (required step)

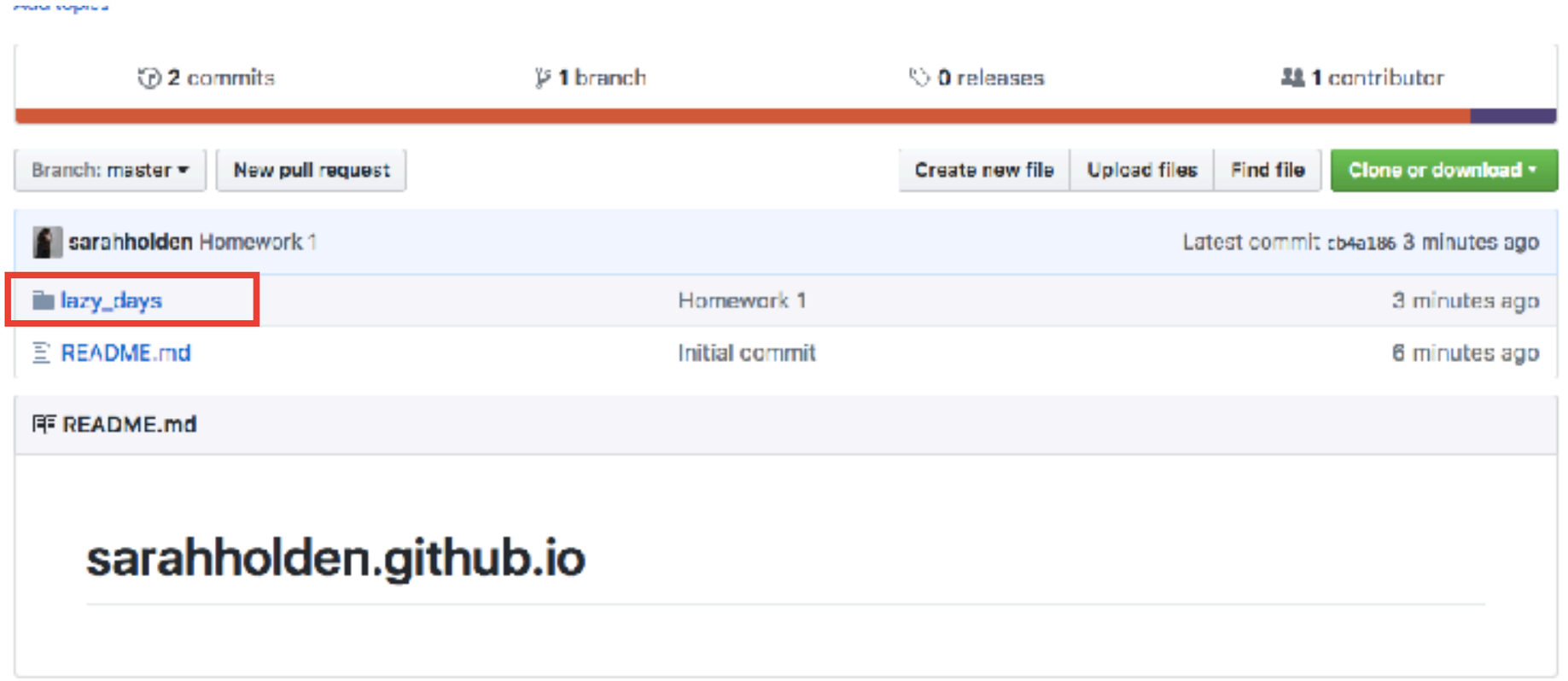
GITHUB SETUP

7. Enter a commit message (required step)



GITHUB SETUP

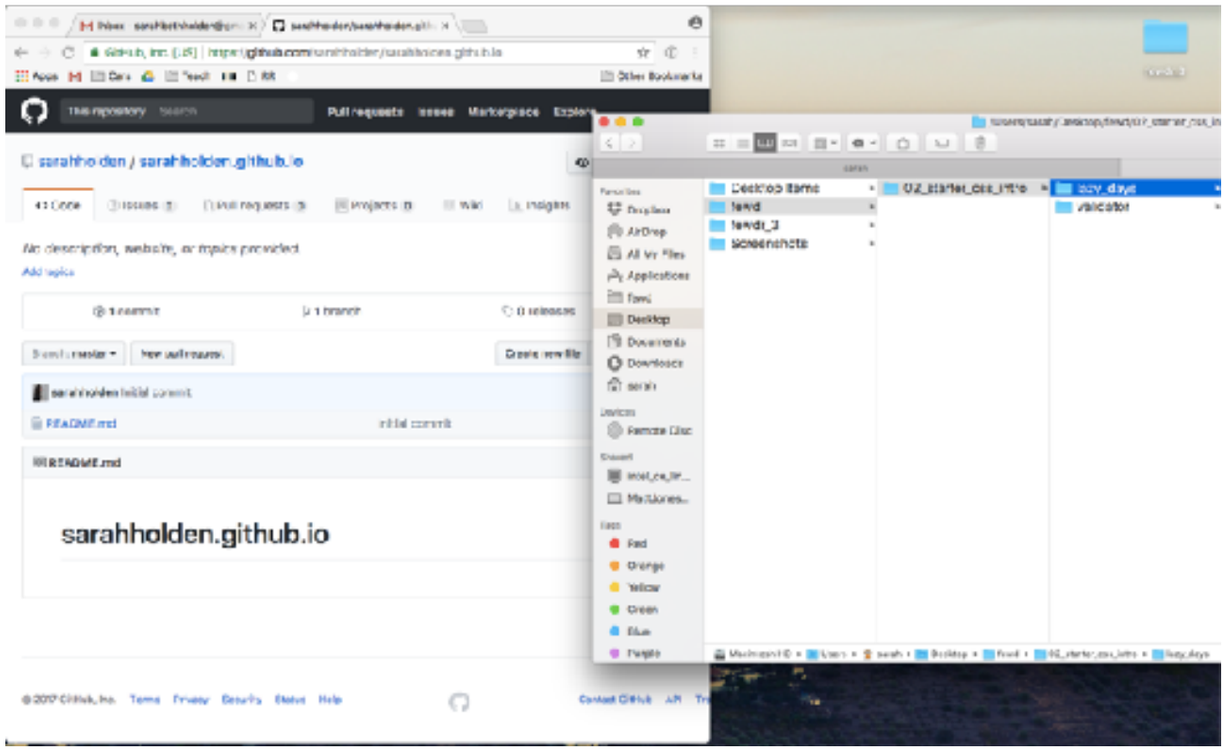
8. You should now see the lazy_days folder in your main homework repository.



GITHUB SETUP

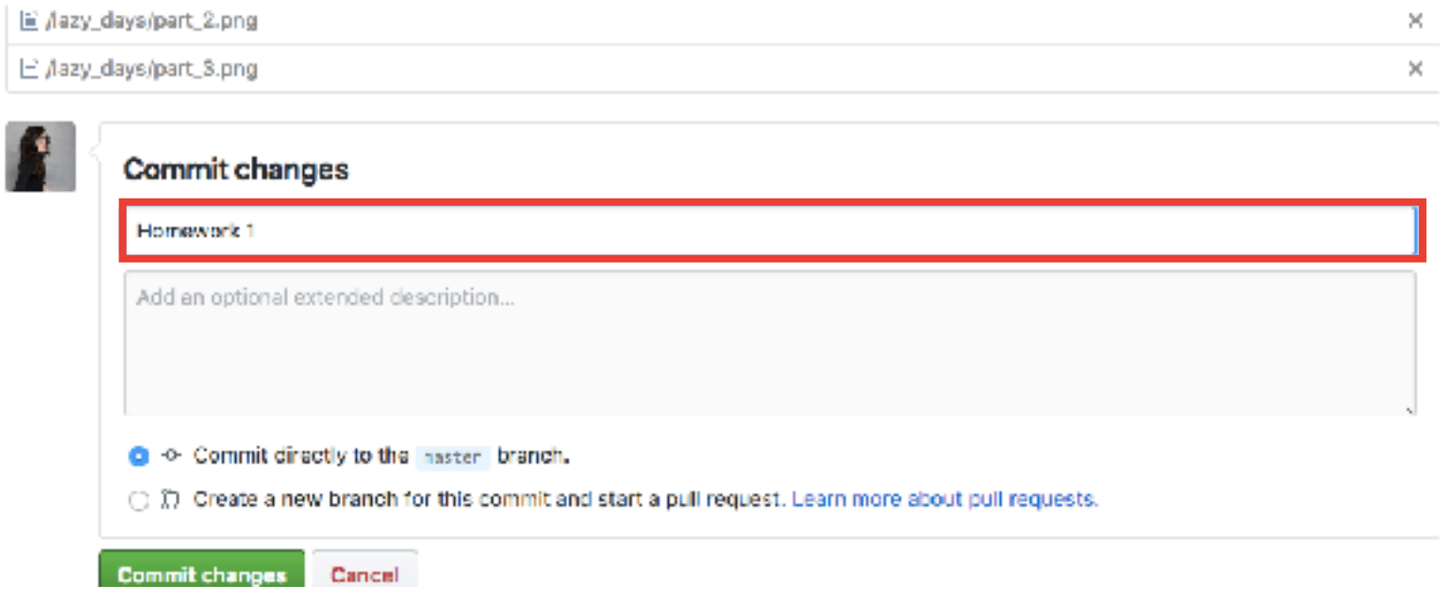
Initial setup is complete! No need to create any new repositories in coming weeks.

Every time homework is due, you'll come back to this repository in GitHub in Chrome (`yourgithubusername.github.io`) and drag and drop the folder for the homework assignment for that week to this repository page.



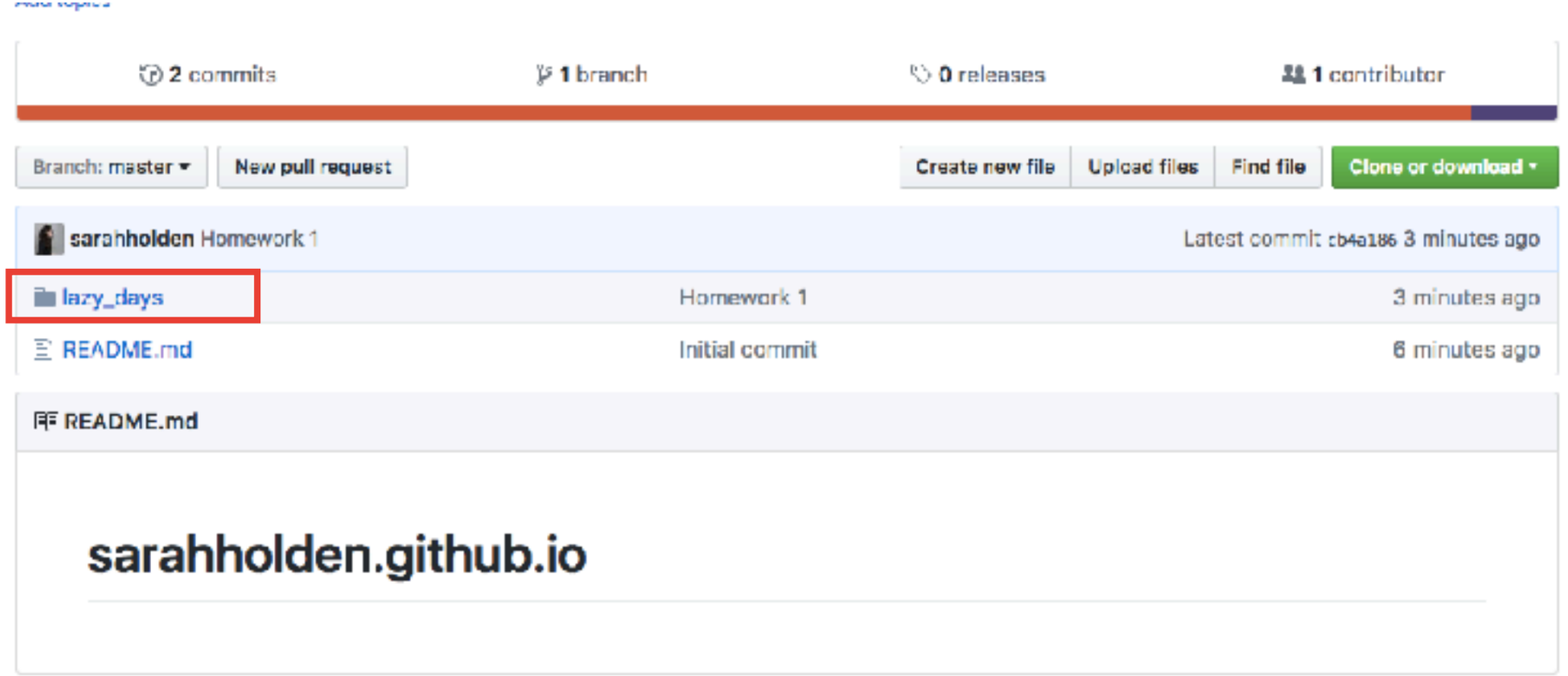
GITHUB SETUP

After dragging and dropping the current assignment folder, you'll always need to enter a commit message and click "Commit changes".



GITHUB SETUP

Make sure to double-check each time you submit an assigment that you see the folder listed after dragging and dropping.



IMPORTANT NOTES

1. You'll always drag and drop all assigments to this same repository on GitHub each week. You won't need to create any other repositories for the class (although you're welcome to do so for personal use and to store personal coding projects)
2. Make sure to drag and drop the **entire** folder for each assignment each week (not just the html and css files in the folder).
3. No need to slack Madeline when your homework is uploaded each week. As long as you see that homework folder listed on your repository page in GitHub, she'll have a link for that repository and can go there to find your assignment each week.

BOX MODEL

CODEALONG

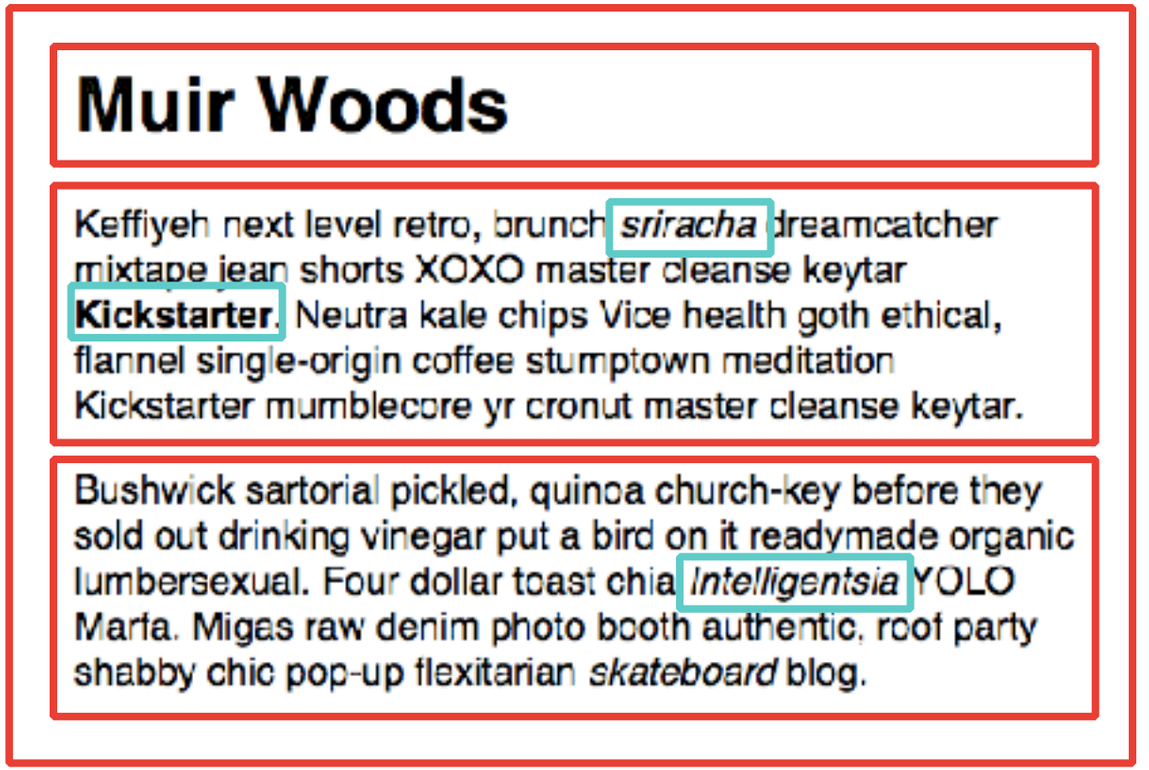
Follow along in this CodePen

REMEMBER THAT THING WE MENTIONED?

Muir Woods

Keffiyeh next level retro, brunch *sriracha* dreamcatcher mixtape jean shorts XOXO master cleanse keytar **Kickstarter**. Neutra kale chips Vice health goth ethical, flannel single-origin coffee stumptown meditation Kickstarter mumblecore yr cronut master cleanse keytar.

Bushwick sartorial pickled, quinoa church-key before they sold out drinking vinegar put a bird on it readymade organic lumbersexual. Four dollar toast chia *Intelligentsia* YOLO Marfa. Migas raw denim photo booth authentic, roof party shabby chic pop-up flexitarian *skateboard* blog.



CONTENT

CONTENT

```
<h1>My Site Title</h1>
```

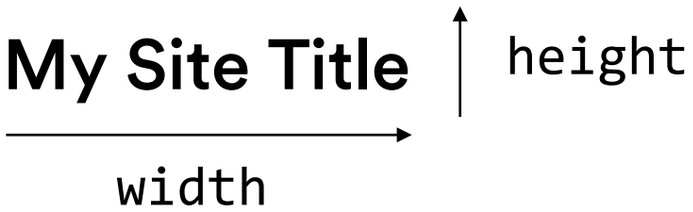
BOX MODEL ANALOGY

Read more about the suburban neighborhood example here.

CONTENT

Content ⟶ **My Site Title**

CONTENT



CONTENT

```
h1 {
  width: 300px;
  height: 40px;
}
```

BORDERS

BORDERS

BORDER SHORTHAND

For a border on all sides:

```
border: width style color;          border: 1px solid red;
```

For a border on one side:

```
border-top: width style color;      border-top: 1px solid red;
border-right: width style color;    border-right: 1px solid red;
border-bottom: width style color;   border-bottom: 1px solid red;
border-left: width style color;     border-left: 1px solid red;
```

BORDER SHORTHAND

Why isn't my border showing up?

```
border-color: red;
border-size: 1px;
```

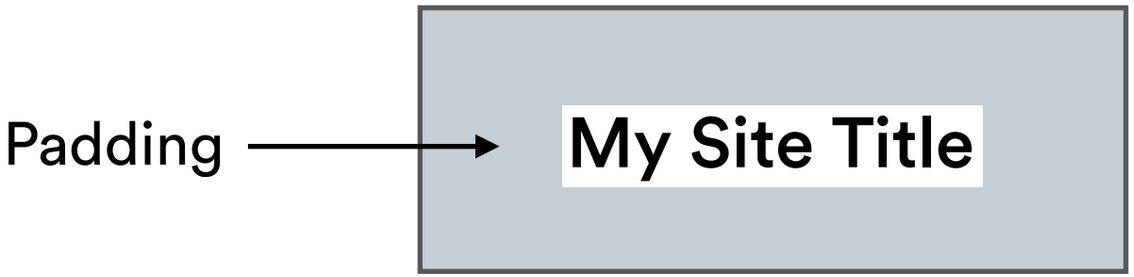
The default border-style is none! So why not always specify all three properties?

```
border: 1px solid red;
```

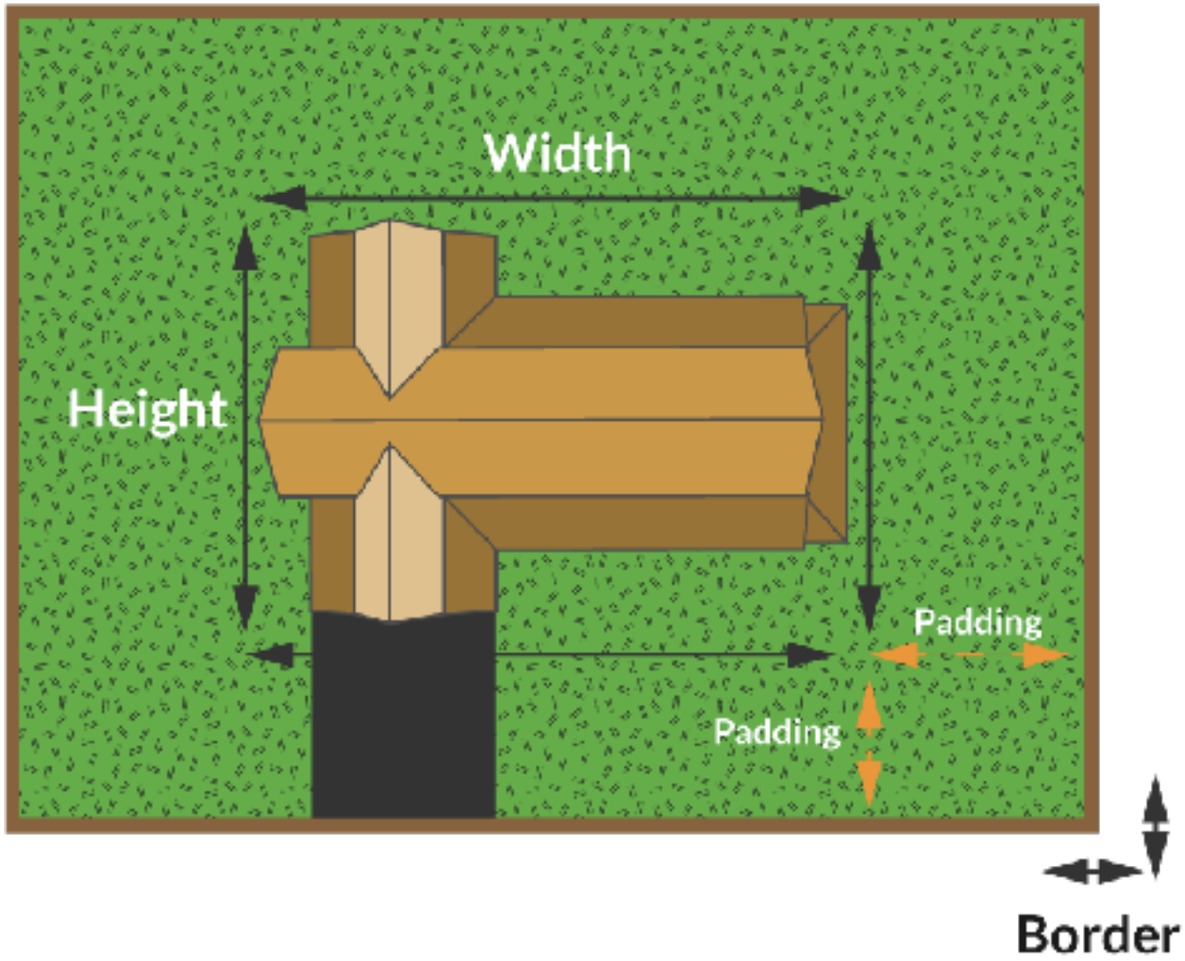
PADDING

PADDING

We can add `padding` to get space between our content and the border



PADDING



BOX MODEL — PADDING SYNTAX

For even padding on all sides:

```
padding: ##px;    padding: 20px;
```

For padding on one specific side:

```
padding-top: ##px;    padding-top: 20px;
```

For different size padding for each side:

```
padding: top right bottom left;    padding: 20px 10px 5px 10px;
```

Memory device: Trouble!

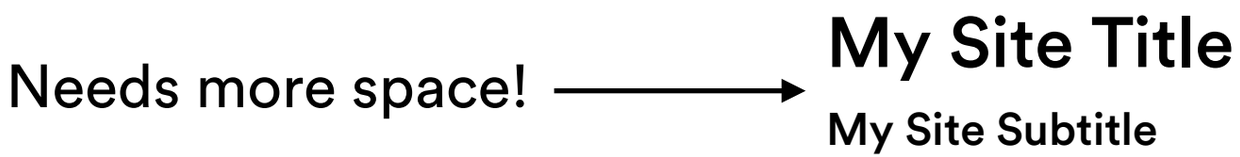
Shorthand if top/bottom are the same and left/right are the same:

```
padding: top/bottom right/left;    padding: 20px 10px;
```

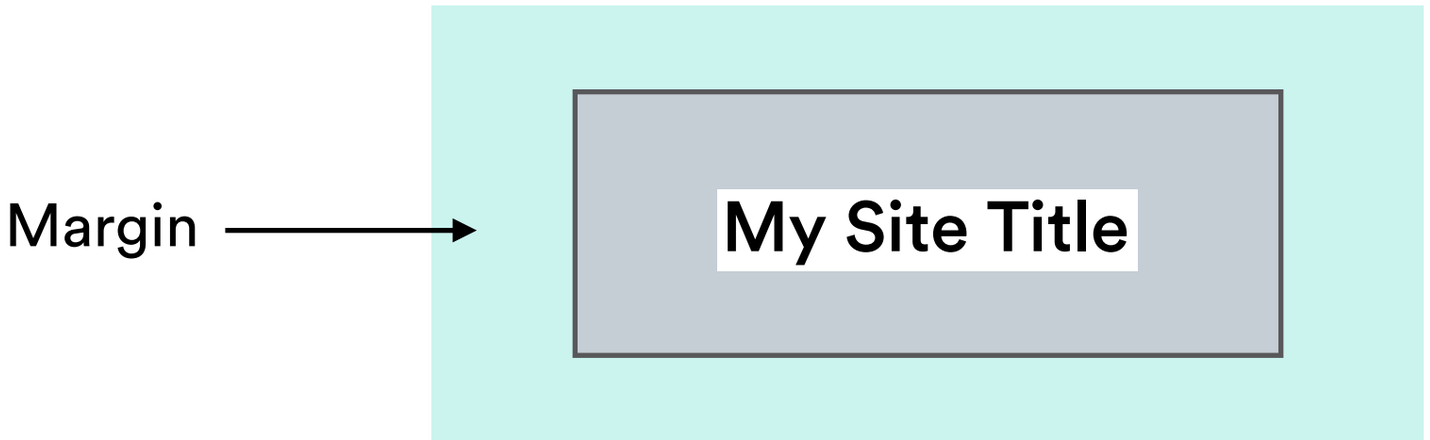
MARGIN

MARGIN

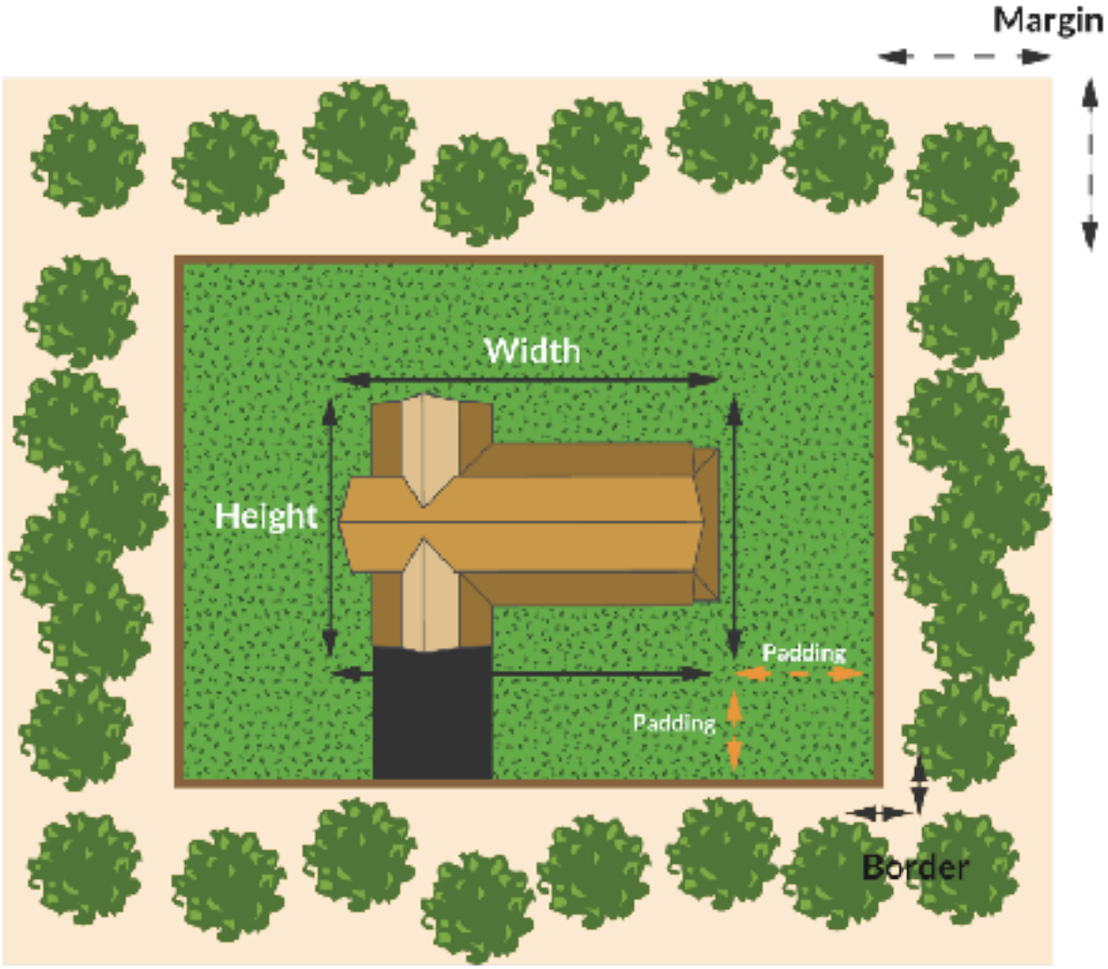
We can add `margin` to add space between elements on our pages.



MARGIN



MARGIN



BOX MODEL — PADDING SYNTAX

For even margin on all sides:

```
margin: ##px;    margin: 20px;
```

For margin on one specific side:

```
margin-top: ##px;    margin-top: 20px;
```

For different size margin for each side:

```
margin: top right bottom left;    margin: 20px 10px 5px 10px;
```

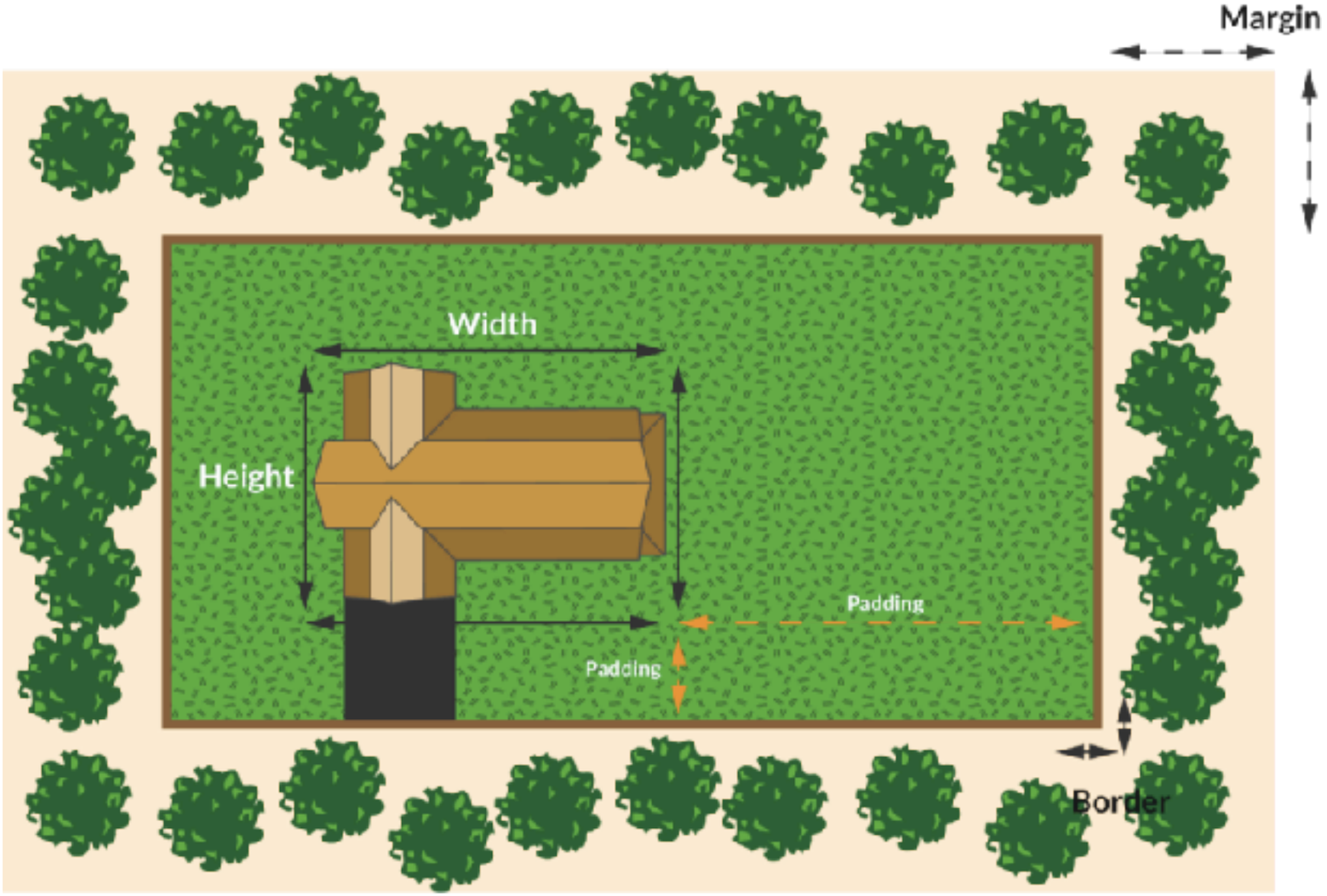
Memory device: Trouble!

Shorthand if top/bottom are the same and left/right are the same:

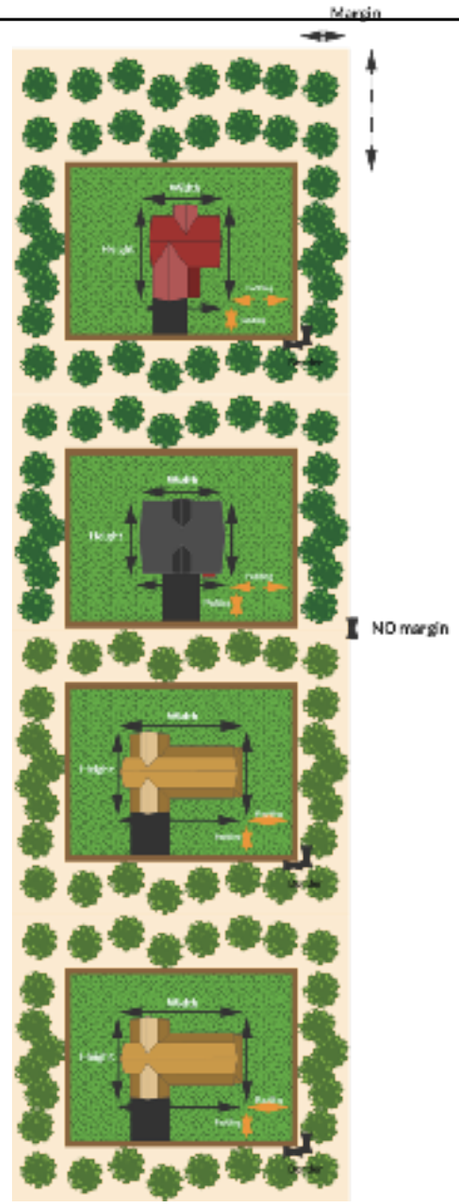
```
margin: top/bottom right/left;    margin: 20px 10px;
```

PADDING VS. MARGIN

PADDING



MARGIN



REVIEW

- What would we use padding for on a site?
- What would we use margin for on a site?
- Take a look at the Lazy Days design. How could we get more space inside the header to have the background color take up more room?
- Take a look at the Lazy Days design. How could we add space between things — between the header and the articles, between the h2 and the h3?

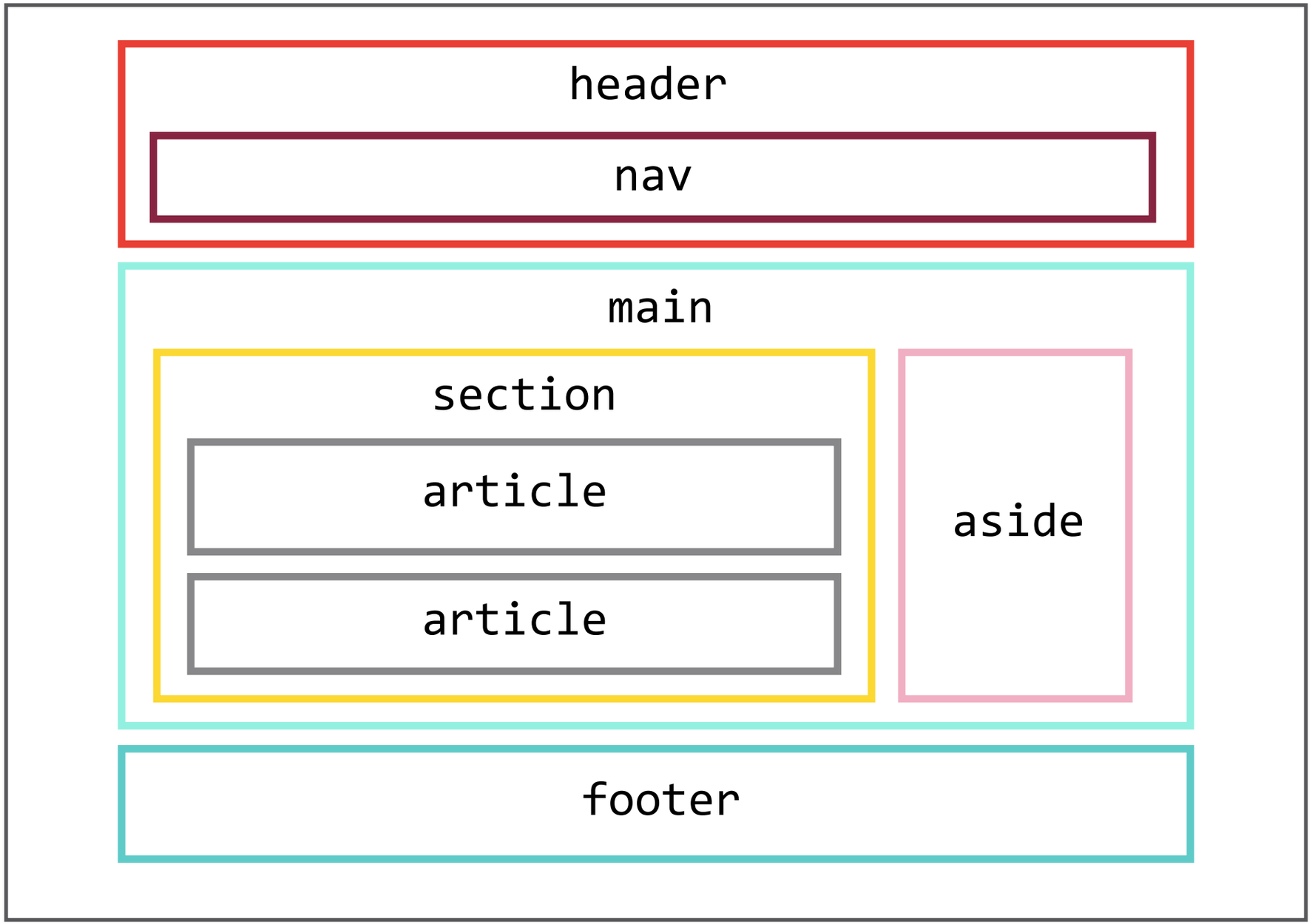
MARGIN

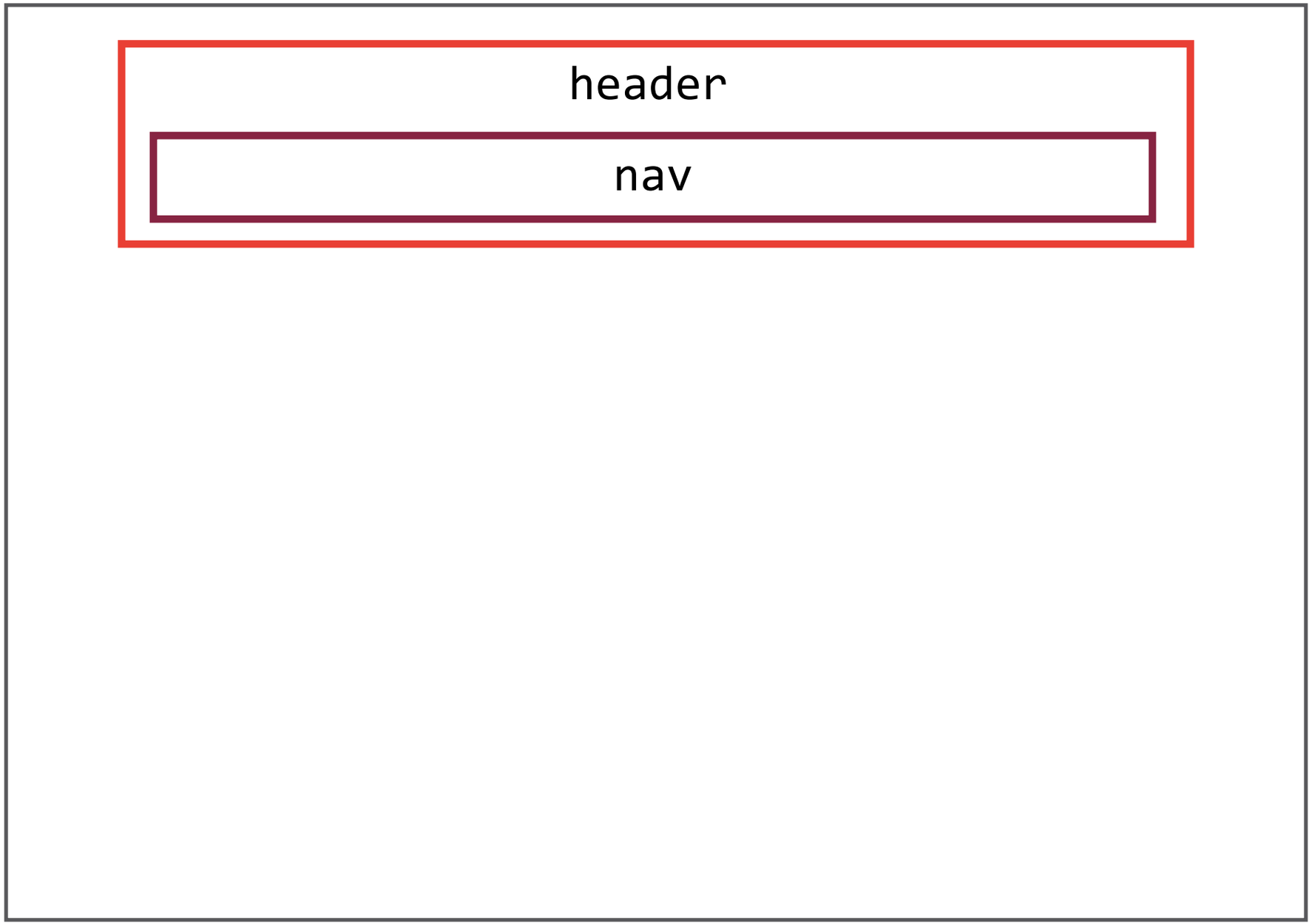
CONTENT

Code for centered layout:

```
main {
    max-width: 800px;
    margin: 0 auto;
}
```

HTML5 STRUCTURAL ELEMENTS


header
nav
main
section
article
article
aside
footer


header
nav

HTML5 STRUCTURAL ELEMENTS

nav – Used to group together major navigational blocks on a page

```
<nav>
    <a href="home.html">Home</a>
    <a href="about.html">About</a>
    <a href="contact.html">Contact</a>
</nav>
```

To add styles to the **nav**:

```
nav {
  background-color: #dfdfdf;
}
```

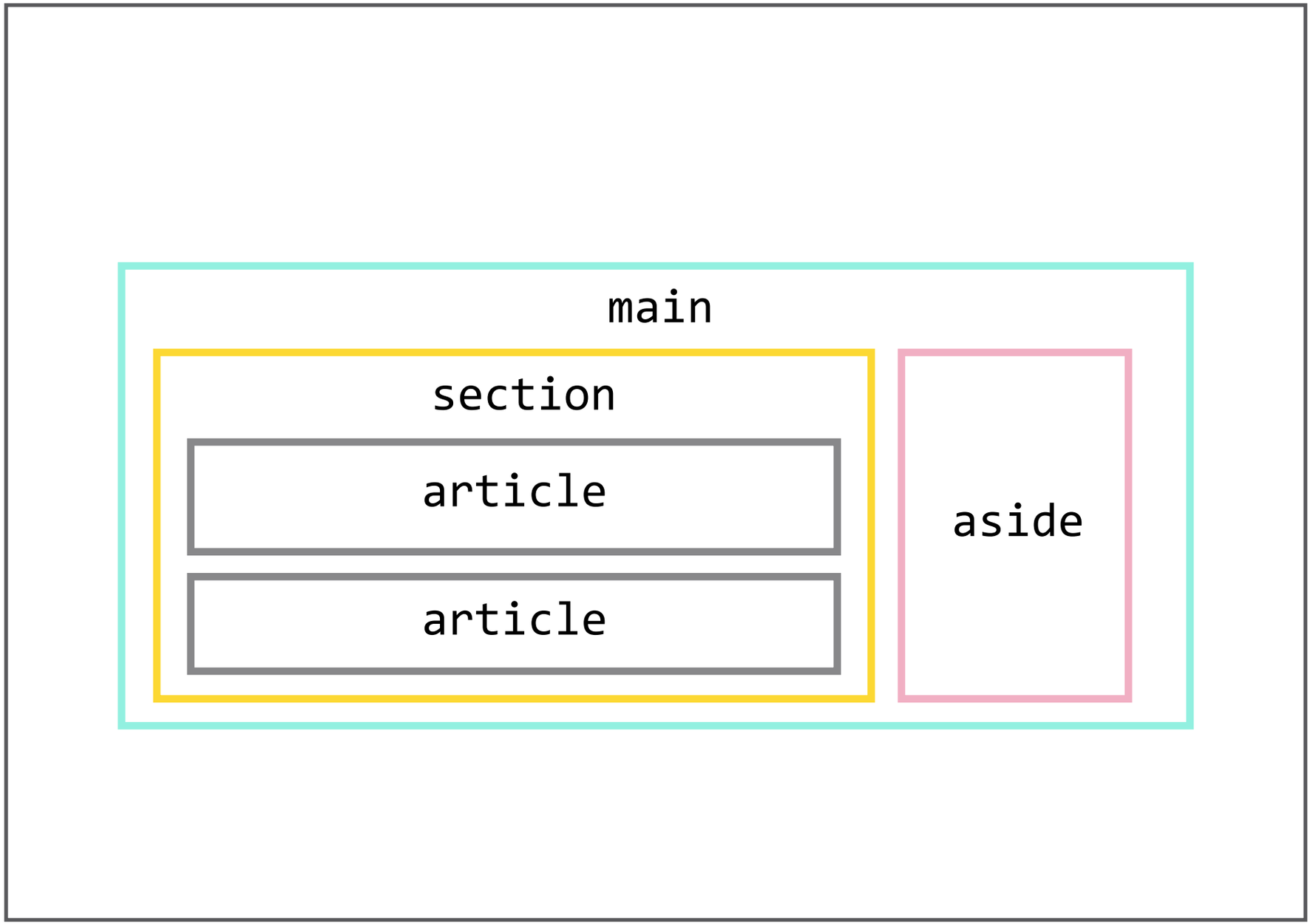
HTML5 STRUCTURAL ELEMENTS

`header` – used to group elements in the opening section of a page.

Examples of content that may be inside of the header:

- any main headings on your site
- the navigation
- a search form for the site
- the site logo
- any other introductory content.

```
<header>
    <h1>My Portfolio Site</h1>
    <nav>
        <a href="home.html">Home</a>
        <a href="about.html">About</a>
        <a href="contact.html">Contact</a>
    </nav>
</header>
```


main
section
article
article
aside

HTML5 STRUCTURAL ELEMENTS

`main` – Groups content that is unique to a particular page. This would exclude any content on a site that might be repeated on multiple pages, such as the navigation and footer.

```
<main>
    <h2>Down the Rabbit Hole</h2>
    <p>Alice was beginning to get very tired of sitting by her sister on the bank, and
    of having nothing to do...</p>

    ...

    <h2>About Alice</h2>
    <p>Alice is a fictional character and protagonist of Lewis Carroll's children's
    novel Alice's Adventures in Wonderland (1865) and its sequel ... she steps through
    a mirror into the Looking-Glass Land.</p>
</main>
```

HTML5 STRUCTURAL ELEMENTS

`article` – Represents a self-contained, standalone section of an HTML document.

This element could be used just once on a page - for example, a blog post may be a single article - or multiple times - for example, an online newspaper page may have numerous articles.

```
<article>
    <h2>My Very Interesting blog article</h2>
    <p>Alice was beginning to get very tired of sitting by her sister on the bank, and of
    having nothing to do: once or twice she had peeped into the book her sister was reading,
    but it had no pictures or conversations in it, "and what is the use of a book," thought
    Alice "without pictures or conversation?"</p>
    <p>So she was considering in her own mind (as well as she could, for the hot day made
    her feel very sleepy and stupid), whether the pleasure of making a daisy-chain would be
    worth the trouble of getting up and picking the daisies, when suddenly a White Rabbit
    with pink eyes ran close by her.</p>
</article>
```

HTML5 STRUCTURAL ELEMENTS

`section` – Represents a general section of an HTML document, typically with a heading, which doesn't have a more specific semantic element to represent it

For example, a portfolio page could be split into an about section, a portfolio section, and a contact section.

```
<section>
    <h2>About Alice</h2>
    <p>Alice is a fictional character and protagonist of Lewis Carroll's children's novel
    Alice's Adventures in Wonderland (1865) and its sequel, Through the Looking-Glass
    (1871). A mid-Victorian era child, Alice unintentionally goes on an underground
    adventure after accidentally falling down a rabbit hole into Wonderland; in the
    sequel, she steps through a mirror into the Looking-Glass Land.</p>
</section>
```

HTML5 STRUCTURAL ELEMENTS

`aside` – Represents a section of a document that contains content that is tangentially connected to the main content of the page.

Examples include:

- additional notes
- reference links
- glossaries

```
<aside>
  <p>The manuscript was illustrated by Dodgson himself who added 37 illustrations.</p>
</aside>
```

footer

HTML5 STRUCTURAL ELEMENTS

`footer` – Used for the concluding section of a page.

Examples of content a footer might include:

- copyright info
- social media links

```
<footer>
    <p>Copyright 2017.</p>
</footer>
```

	DESCRIPTION
HEADER	Does it contain introductory content, such as the site title and navigation?
NAV	Does it contain a major navigational block?
FOOTER	Does it contain footer content?
MAIN	Does it contain content that is unique to that particular page?
ARTICLE	Is it self-contained, standalone content? Would it make sense on its own?
ASIDE	Is the content **not** required to understand the rest of the page?
SECTION	Is it a general section of the page that could have a heading?
DIV	Do you need a wrapper for styling and none of the above apply?

ACTIVITY

EXERCISE

KEY OBJECTIVE

- Identify content sections

TYPE OF EXERCISE

- Partner

TIMING

8 min

1. Go through each slide and discuss which structural element you would use for each.

01/23/2016

A VISUAL GUIDE TO THE SOUTHWEST

Lorem ipsum dolor sit amet, consectetur adipisicing elit. Eius soluta incidunt, eum quos consequuntur aliquam nobis velit mollitia iste, natus ex repellendus delectus quisquam et quae, in atque a. Perspiciatis excepturi ipsum temporibus, eius odit deleniti quasi voluptates eaque aliquam, iste consequatur repudiandae illo a. Voluptates eligendi reiciendis corporis itaque soluta fugit atque quaerat ipsum reprehenderit sapiente, voluptatem repellat eveniet fuga natus tempore modi vero distinctio nulla accusantium recusandae, omnis. Esse aliquam ea, similique repudiandae nihil numquam, neque enim perspiciatis. Facilis dolor praesentium impedit quidem, non unde accusantium, commodi error, soluta dolores modi blanditiis similique amet his earum, reiciendis consectetur animi in. Exercitationem distinctio necessitatibus molestias quas laboriosam nam modi porro accusantium blanditiis, beatae incidunt aliquid nostrum laborum, magni cum recusandae autem doloremque qui perferendis iste non quaerat fugit. Asperiatur voluptatem impedit, enim!

Lorem ipsum dolor sit amet, consectetur adipisicing elit. Eius soluta incidunt, eum quos consequuntur aliquam nobis velit mollitia iste, natus ex repellendus delectus quisquam et quae, in atque a. Perspiciatis excepturi ipsum temporibus, eius odit deleniti quasi voluptates eaque aliquam, iste consequatur repudiandae illo a. Voluptates eligendi reiciendis corporis itaque soluta fugit atque quaerat ipsum reprehenderit sapiente, voluptatem repellat eveniet fuga natus tempore modi vero distinctio nulla accusantium recusandae, omnis. Esse aliquam ea, similique repudiandae nihil numquam, neque enim perspiciatis. Facilis dolor praesentium impedit quidem, non unde accusantium, commodi error, soluta dolores modi blanditiis similique amet his earum, reiciendis consectetur animi in. Exercitationem distinctio necessitatibus molestias quas laboriosam nam modi porro accusantium blanditiis, beatae incidunt aliquid nostrum laborum, magni cum recusandae autem doloremque qui perferendis iste non quaerat fugit. Asperiatur voluptatem impedit, enim!

ABOUT US

Lorem ipsum dolor sit amet, consectetur adipisicing elit. Rerum fugiat illo sunt quasi incidunt veniam cumque minus totam non ea vel at eius, accusamus, doloribus dolore at ipsum quaerat molestias.

Explicabo consectetur tempora perferendis voluptas itaque, corporis at ipsum quisquam eveniet, facilis perspiciatis. Eum ad adipisci, earum fugit perspiciatis deserunt veniam eos? Impedit expedita assumenda, saepe excepturi ad voluptatibus. Eveniet.

RECENT ISSUES

- Issue Nineteen — Camping
- Issue Eighteen — Food
- Issue Seventeen — Signs
- Issue Sixteen — Friends
- Issue Fifteen — Carnivals
- Issue Fourteen — Gatherings

EMAIL ADDRESS

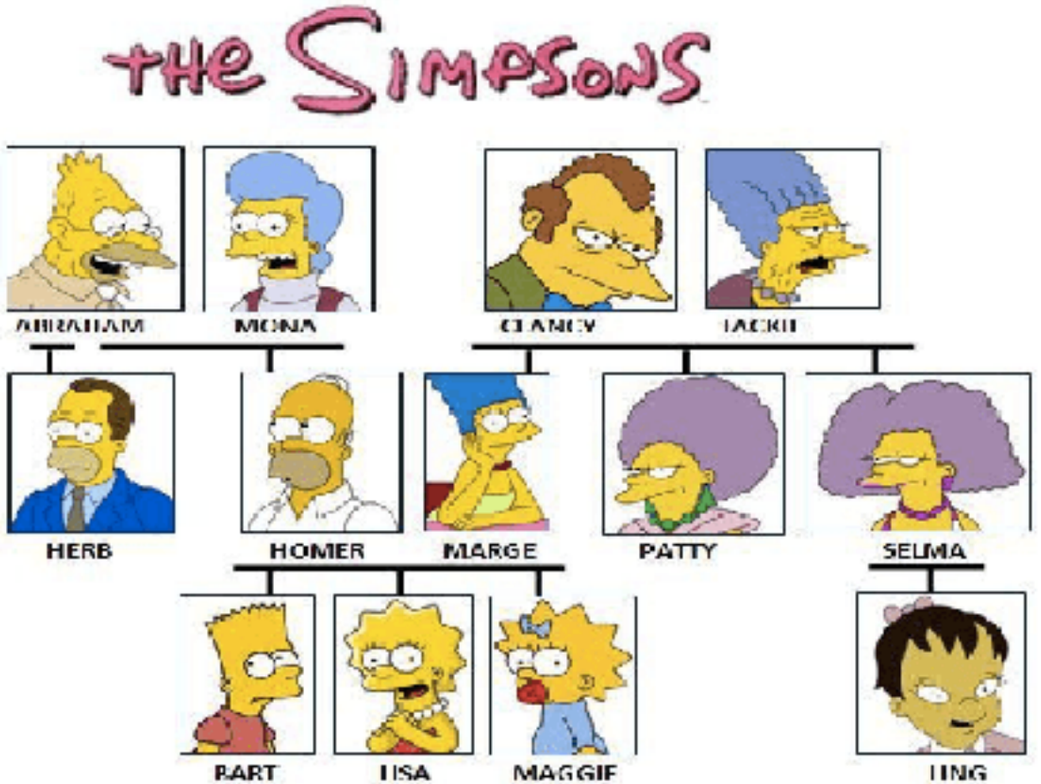
KEEP IN TOUCH

©2016 Sarah Holden

HTML STRUCTURE

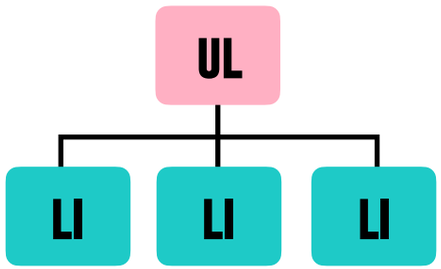
PAGE STRUCTURE — RELATIONSHIPS

- We often use the same terms we would use to describe relationships between family members to describe the relationships between HTML elements.



PAGE STRUCTURE — NESTING

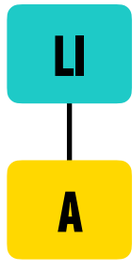
- Here we can say that our <ul> is the **parent** of our <li>s
- We can also say that our <li>s are **children** of the <ul>




```
<!DOCTYPE html>
<html lang="en">
<head>
     <meta charset="UTF-8">
     <title>The Evolution of Denim</title>
</head>
<body>
    <h1>The Evolution of Denim</h1>
    <ul>
        <li>Dark Wash <a href="linkgoeshere">Jeans</a></li>
        <li>Stone Wash</li>
        <li>Chambray</li>
    </ul>
</body>
</html>
```

PAGE STRUCTURE — NESTING

- Similarly, we can have an <a> tag that is nested inside, or wrapped by, our <li> element.



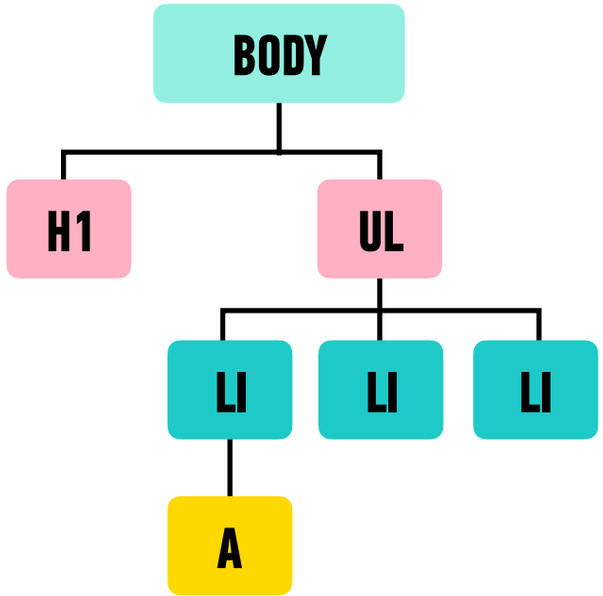

```
<!DOCTYPE html>
<html lang="en">
<head>
     <meta charset="UTF-8">
     <title>The Evolution of Denim</title>
</head>
<body>
    <h1>The Evolution of Denim</h1>
    <ul>
        <li>Dark Wash <a href="linkgoeshere">Jeans</a></li>
        <li>Stone Wash</li>
        <li>Chambray</li>
    </ul>
</body>
</html>
```

<a>

<li>

PAGE STRUCTURE — NESTING

- Similarly, we can say that all of our HTML content, our h1, our ul, our li elements, are "wrapped" by the body, or "nested" inside the body since they are within the opening and closing body tags

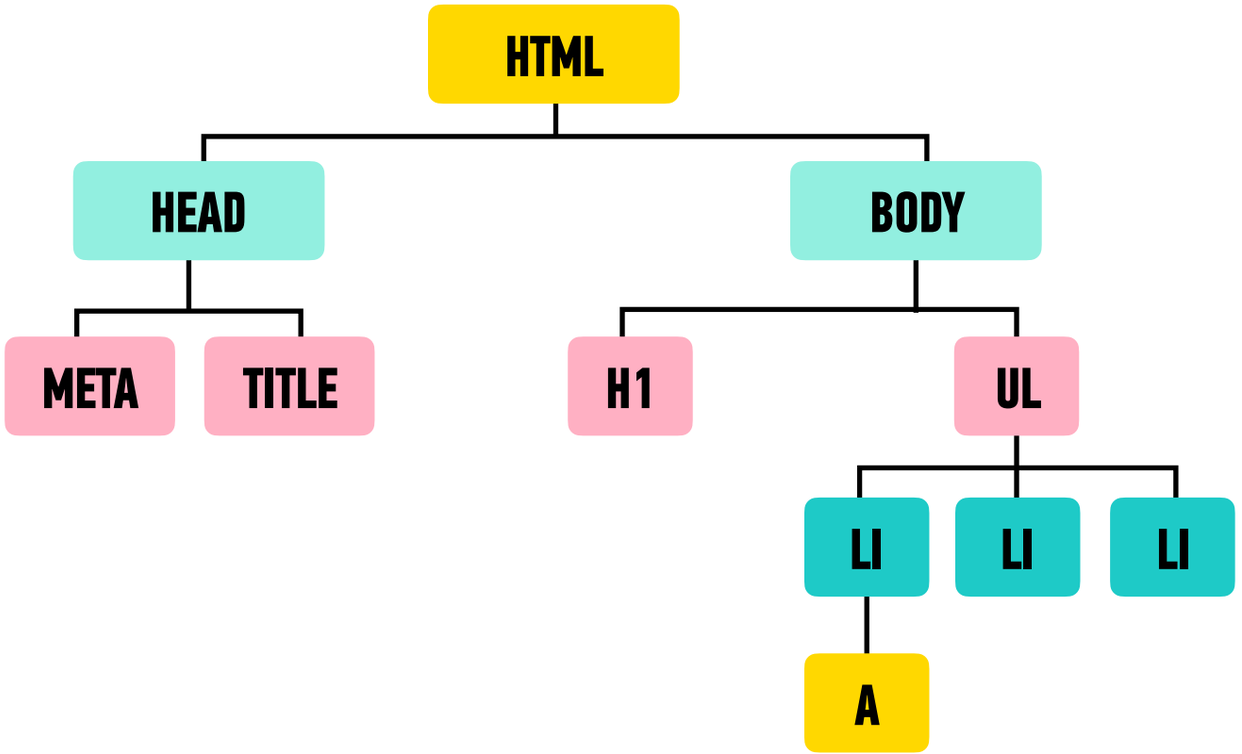


body

```
<!DOCTYPE html>
<html lang="en">
<head>
     <meta charset="UTF-8">
     <title>The Evolution of Denim</title>
</head>
<body>
    <h1>The Evolution of Denim</h1>
    <ul>
        <li>Dark Wash <a href="linkgoeshere">Jeans</a></li>
        <li>Stone Wash</li>
        <li>Chambray</li>
    </ul>
</body>
</html>
```

HTML

DOM TREE




```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>The Evolution of Denim</title>
</head>
<body>
    <h1>The Evolution of Denim</h1>
    <ul>
        <li>Dark Wash <a href="linkgoeshere">Jeans</a></li>
        <li>Stone Wash</li>
        <li>Chambray</li>
    </ul>
</body>
</html>
```

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>About Us</title>


</head>
<body>

    <h1>About Us</h1>

    <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit.</p>

    <ul>
        <li><a href="http://www.bigsurcamp.com/">Big Sur Campground &amp;amp; Cabins</a></li>
        <li><a href="http://www.parks.ca.gov/?page_id=570">Pfeiffer Big Sur State Park</a></li>
        <li><a href="http://www.riversidecampground.com/">Riverside Campground &amp;amp; Cabins</a></li>
        <li><a href="http://www.fernwoodbigsur.com/">Fernwood Resort</a></li>
    </ul>

    <h2>Another Section</h2>

    <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit.</p>

</body>
</html>
```

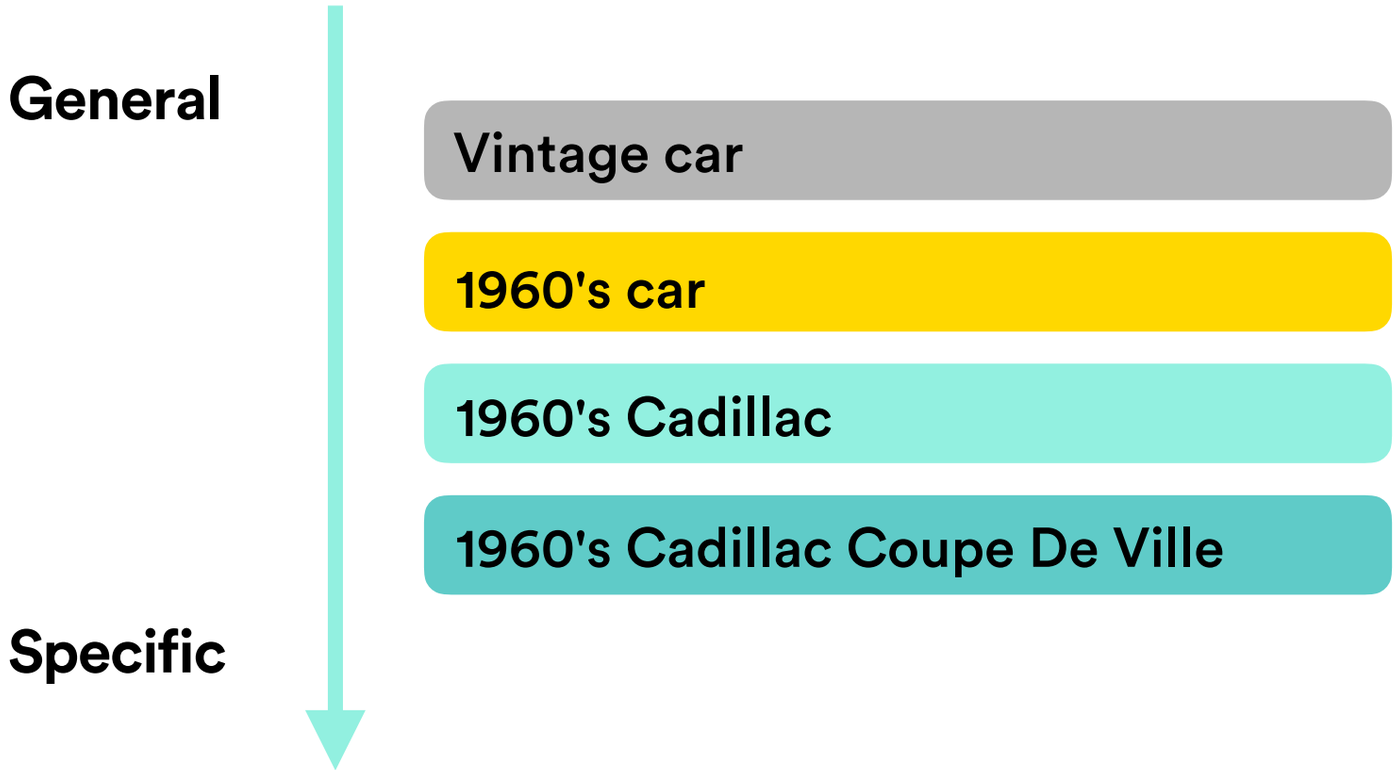
PAGE STRUCTURE — INDENT!

- It is best practice to indent any tags that are nested inside other tags.
- This makes our code easier to read and will help us see if we forget a closing tag.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>The Evolution of Denim</title>
</head>
<body>
    <h1>The Evolution of Denim</h1>
    <ul>
        <li>Dark Wash <a href="linkgoeshere">Jeans</a></li>
        <li>Stone Wash</li>
        <li>Chambray</li>
    </ul>
</body>
</html>
```

CSS SELECTORS

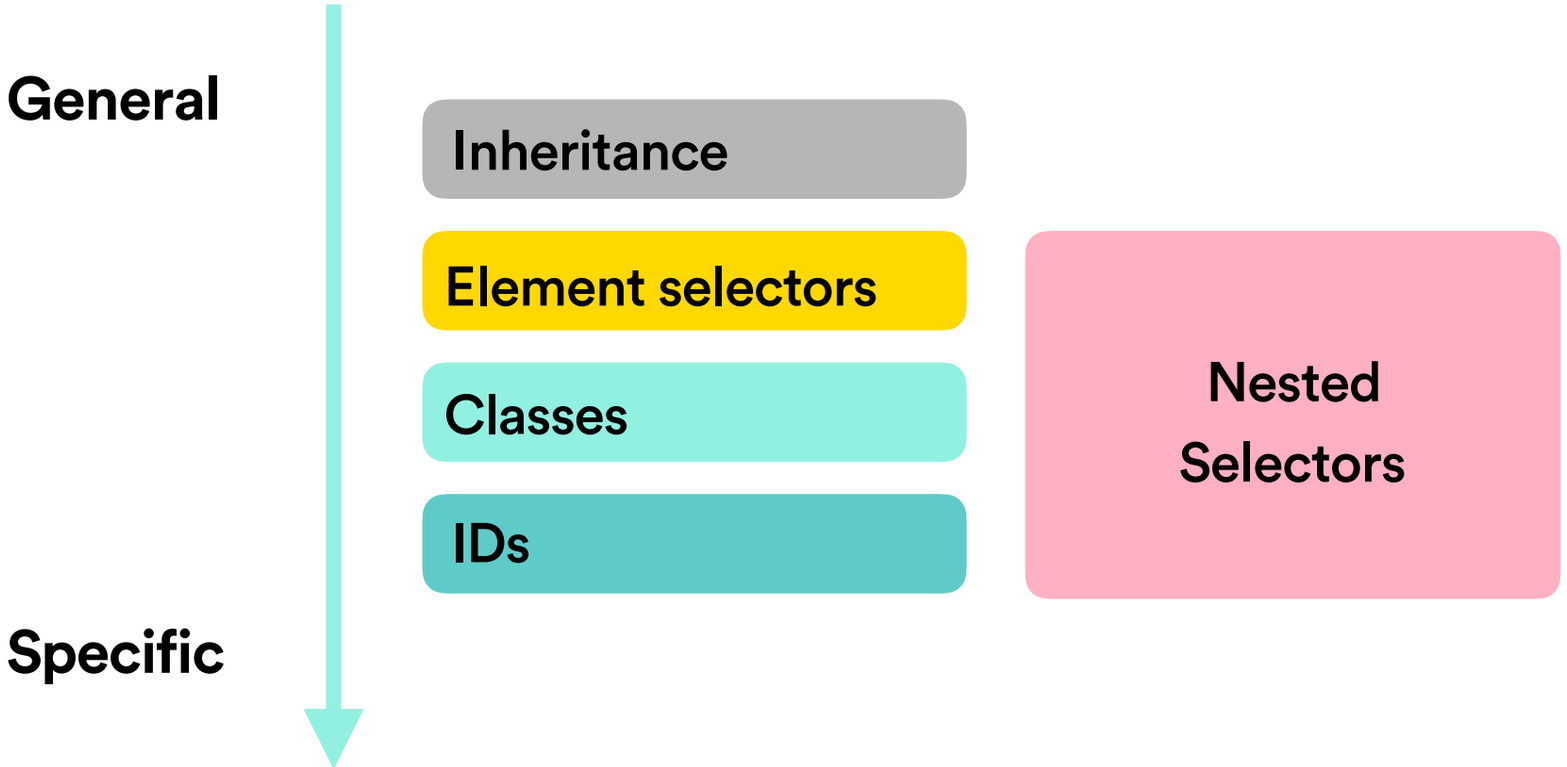
MORE ABOUT CASCADING



MORE ABOUT CASCADING — GENERAL TO MORE SPECIFIC

SPECIFICITY:

The *more specific rule* will take precedence over the more general rule



INHERITANCE

INHERITANCE — SETTING BASE STYLES

- Certain properties are passed on from a parent element down to its children
- If you specify the *font-family* or *color* properties on the body element, they will be inherited, or applied, to most child elements **unless there is a more specific rule that applies**.

Inherited properties you'll use in this course:

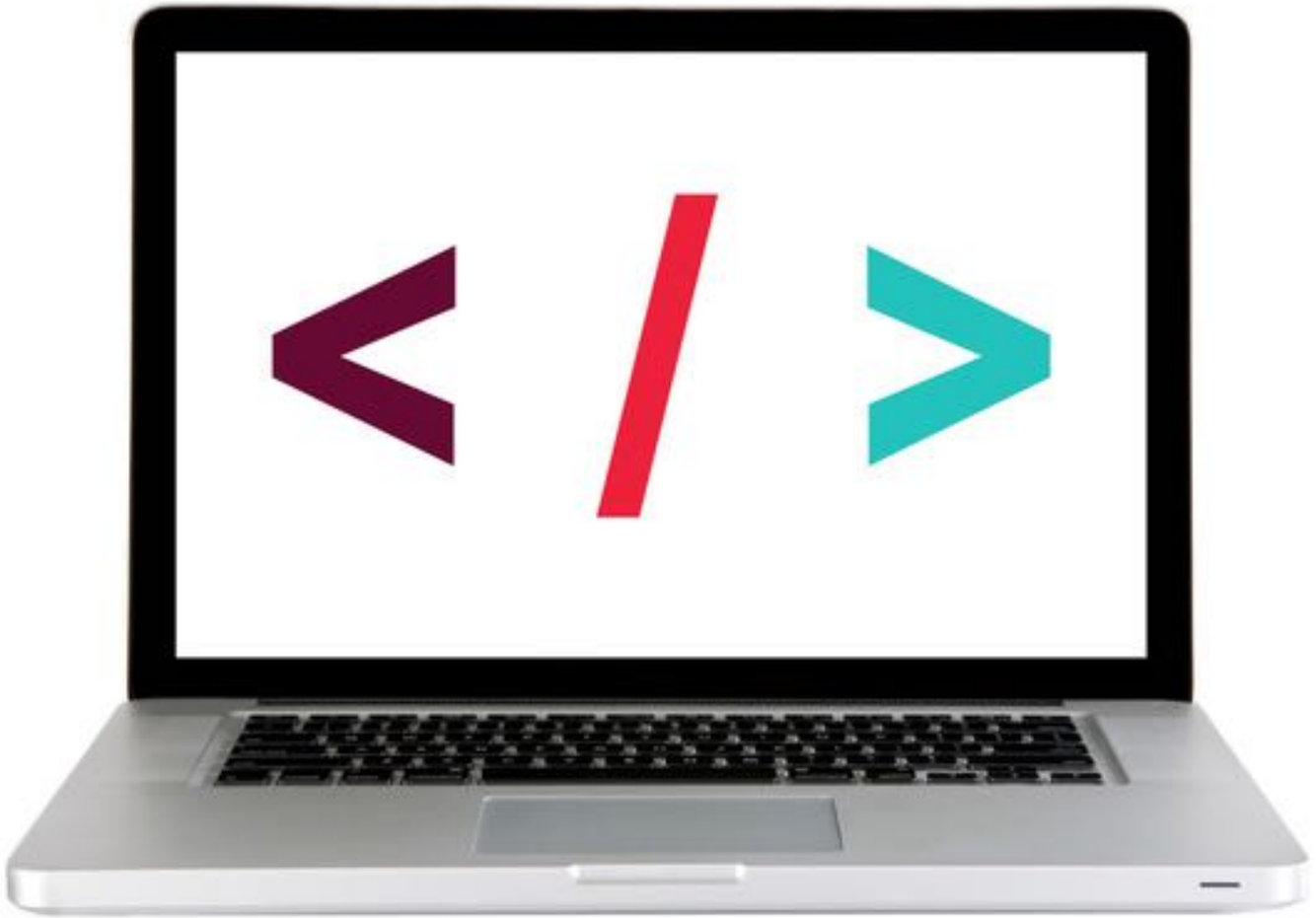
- color
- font-family
- font-size
- font-style
- font-weight
- letter-spacing
- line-height
- list-style
- text-align
- text-indent
- text-transform
- word-spacing

INHERITANCE — SETTING BASE STYLES

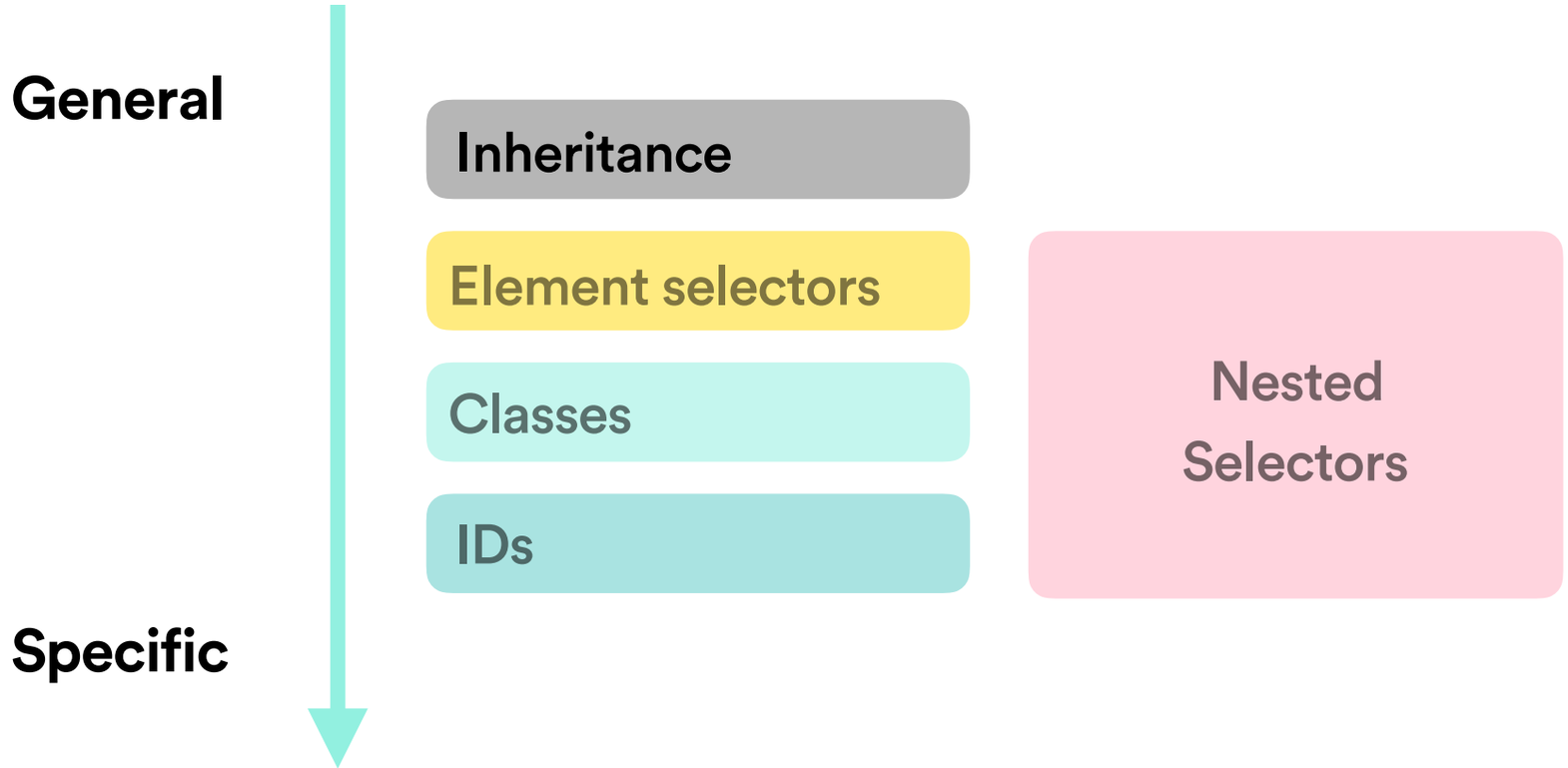
- Our browser adds default styles for some elements, so they will not inherit some properties.

	PROPERTIES NOT INHERITED	HOW TO OVERRIDE
`a`	`color`, `text-decoration`	`a { text-decoration: none; color: pink; }`
`h1 - h6`	`font-weight`	`h1 { font-weight: normal; }`

LET'S TAKE A CLOSER LOOK



MORE ABOUT CASCADING — GENERAL TO MORE SPECIFIC



ACTIVITY

KEY OBJECTIVE

- Practice using inheritance to set up default styles for a webpage

TYPE OF ACTIVITY

- Starter Code > Specificity

TASKS

1 min

1. Follow instructions under step 1

ACTIVITY

KEY OBJECTIVE

- Explain inheritance in CSS. How can you use inheritance in your stylesheets to write less code in the long run?

TYPE OF ACTIVITY

- Turn and Talk

TASKS

1 min	1. Discuss the question with your groups
2 min	2. Pick one person to jot down your thoughts and share them via Slack.

ACTIVITY

EXERCISE

KEY OBJECTIVE

- Plan out the Monument Magazine site.

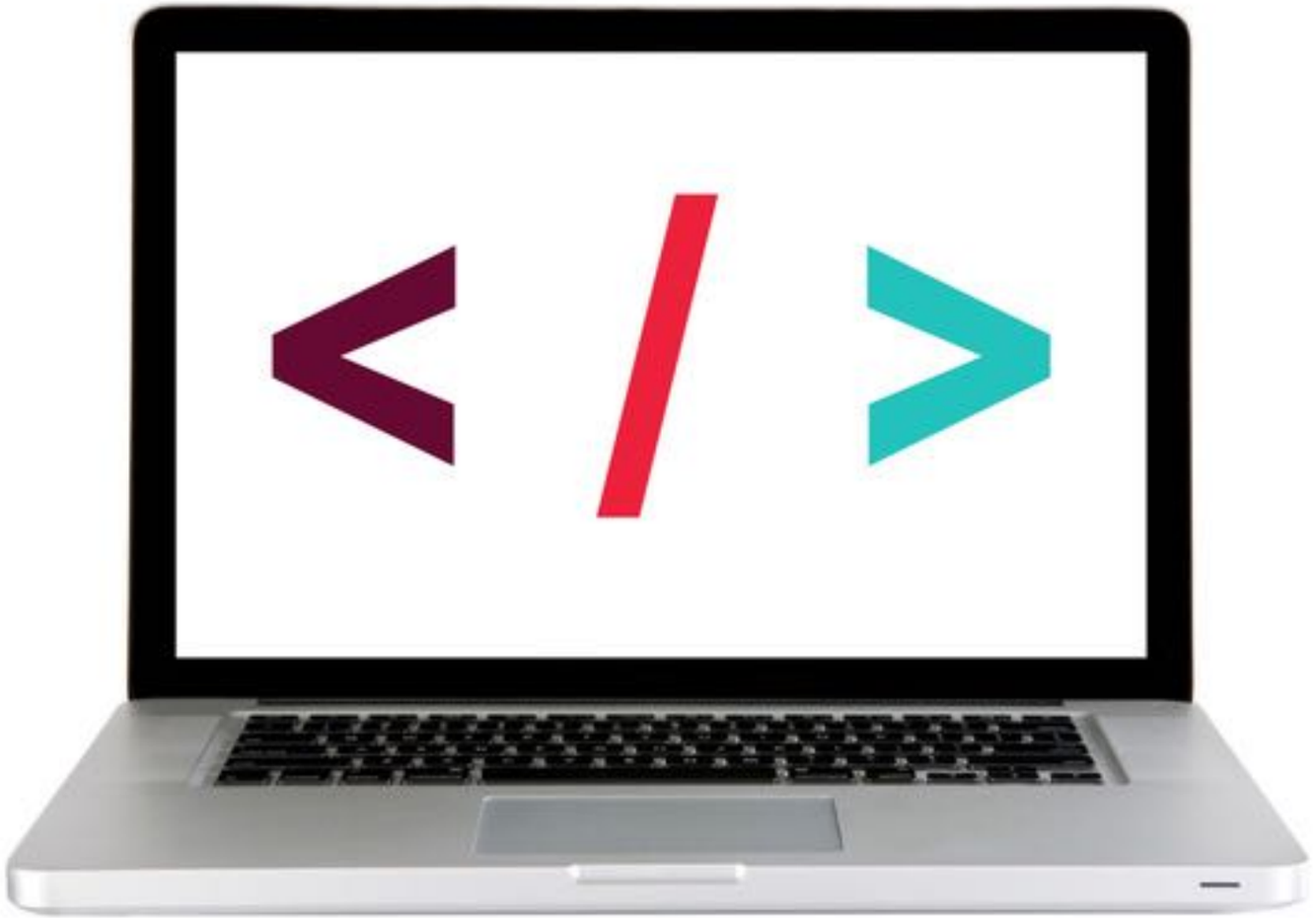
TASKS

Consider the following questions with your groups:

1. What is the most common font-family?
2. Is there a common color for the text?
3. Is most of the text uppercase? Lowercase?
4. How about the anchors? Are they underlined?
5. Is there a common color for the anchors?

NESTED SELECTORS

LET'S TAKE A CLOSER LOOK



SELECTORS — MOST COMMON

SELECTOR:	MEANING:	EXAMPLE:
TYPE	Selects an element	`a {}`
DESCENDANT	Selects an element that is a **descendent** of another element	`p a {}`
UNIVERSAL	Selects **all** elements in a document	`* {}`
MULTIPLE	Select multiple elements	`h1, h2 {}`

SELECTORS

Which element here will be italic?

```
p a {
    font-style: italic;
}
```

SELECTORS

- The **last** element in the selector string is *always* the one that we are styling.
- Try reading right-to-left!

```
p a {
    font-style: italic;
}
```

```
<p>Want to get in touch? Send us an <a href="#">email!</a></p>
```

ANCHORS THAT ARE
DESCENDENTS OF A PARAGRAPH

SELECTORS

```
<header>
    <h1><a href="">Nested Selectors</a></h1>
    <nav>
        <a href="">Home</a>
        <a href="">About</a>
        <a href="">Contact</a>
        <a href="">Blog</a>
    </nav>
</header>
```

SELECTORS

```
<header>
    <h1><a href="#">Nested Selectors</a></h1>
    <nav>
        <a href="#">Home</a>
        <a href="#">About</a>
        <a href="#">Contact</a>
        <a href="#">Blog</a>
    </nav>
</header>
```

SELECTORS

```
<footer>
    <nav>
        <a href="#">About</a>
        <a href="#">Contact</a>
        <a href="#">Blog</a>
    </nav>
</footer>
```

ACTIVITY

EXERCISE

LOCATION

- starter_code folder > nested_selectors folder

KEY OBJECTIVE

- Practice using nested selectors

TIMING

5 min

1. Follow the instructions in the style.css file.

CLASSES AND IDS

TARGETING SPECIFIC ELEMENTS

- Classes & IDs allow us to add ‘labels’ to elements so we can target them in our CSS.

TARGETING SPECIFIC ELEMENTS

CLASSES

- Classes are used to group elements together

```
<div class="alert">Content</div>
```

```
.alert {
    color: red;
    font-size: 20px;
}
```

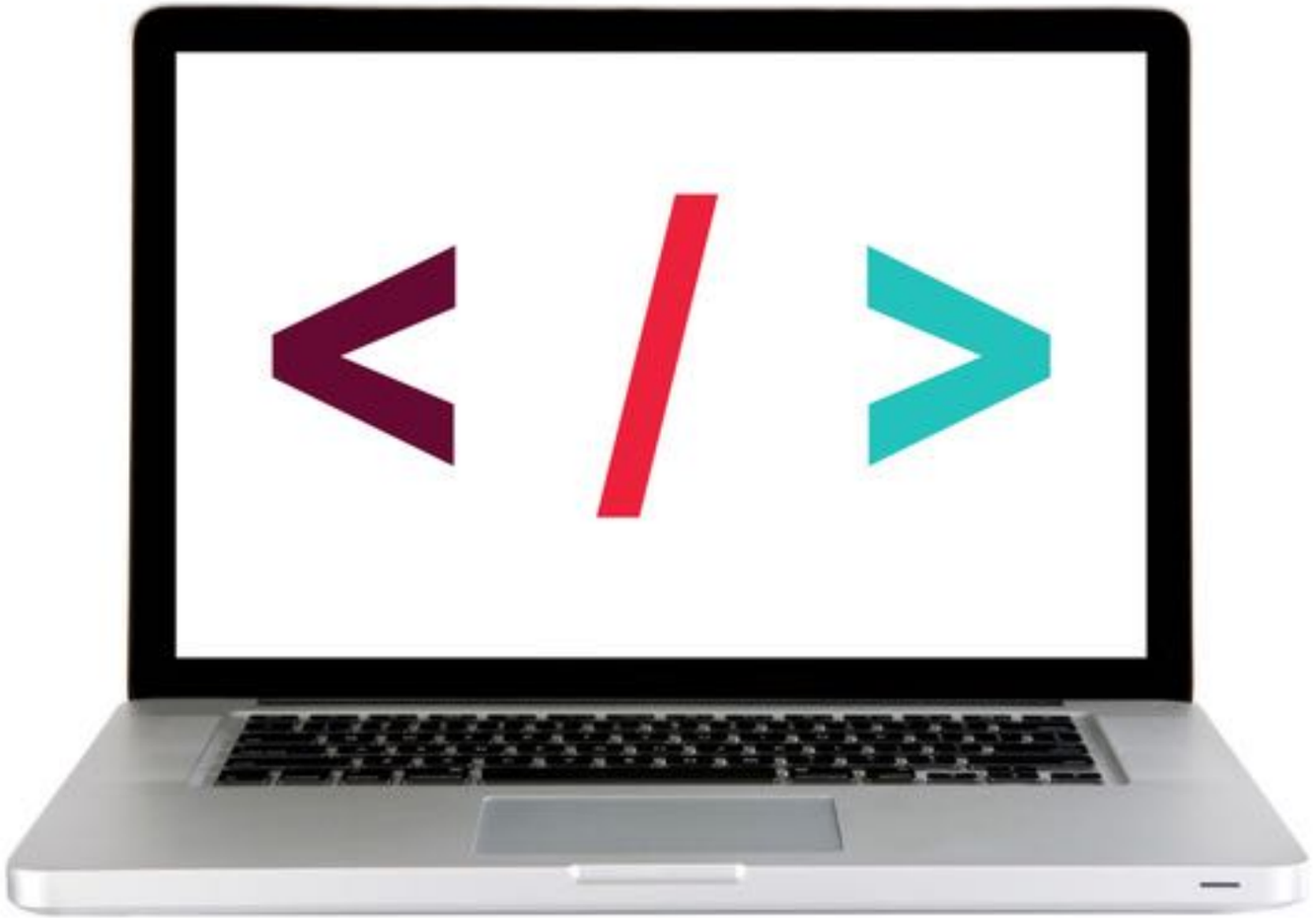
IDS

- Ids are used to target *one specific element*
- Each element can only have one id
- **Important**: two elements on the same page cannot have the same id

```
<nav id="main-nav">Content</nav>
```

```
#main-nav {
   text-align: center;
}
```

LET'S TAKE A CLOSER LOOK



ACTIVITY

EXERCISE

LOCATION

- starter_code folder > classes_and_ids folder

KEY OBJECTIVE

- Use classes and IDs to target elements

TIMING

1 min	1. Look at the image provided. Which ice cream items would you use a class to style? How about an ID?
3 min	2. Follow steps 1 - 3 in your CSS file (towards the end)

BEST PRACTICES

SELECTORS

- Avoid class and ID names that are similar to names already used by HTML and CSS
- Avoid classitus! Only use classes when you need to
- Add a class or ID to the parent and use nested selectors when possible, instead of adding a class or id to each individual element in a section.

NESTED SELECTORS

```
.contact p a {
    color: red;
}
```

NESTED SELECTORS

```
.contact p a {
    color: red;
}
```

```
<div class="contact">
    <p>Want to get in touch? Send us an <a href="#">email!</a></p>
</div>
```

ANCHORS *THAT ARE DESCENDENTS OF*
A PARAGRAPH *THAT IS A DESCENDENT OF*
AN ELEMENT WITH THE CLASS CONTACT

SELECTORS

```
<header>
    <h1><a href="">Nested Selectors</a></h1>
    <nav class="main-nav">
        <a href="">Home</a>
        <a href="">About</a>
        <a href="">Contact</a>
        <a href="">Blog</a>
    </nav>
</header>
```

SELECTORS

```
<section>
    <h2>About Me</h2>
    <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit.</p>
    <button>Read More</button>
</section>
```

https://codepen.io/sarahholden/pen/mwmMrz

SELECTORS

```
<section id="about">
    <h2 id="about-heading">About Me</h2>
    <p id="about-paragraph">Lorem ipsum dolor sit amet, consectetur adipisicing elit.</p>
    <button id="about-button">Read More</button>
</section>
```

SELECTORS

```
<section id="about">
    <h2>About Me</h2>
    <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit.</p>
    <button>Read More</button>
</section>
```

LAST RULE

CSS CASCADING — THE SPECIFICITY GAME

Question: *Will the anchor have an underline or no underline? Why?*

HTML:

```
<li>
    <p>Visit this <a href="#">cool</a> site.</p>
</li>
```

CSS:

```
li a { /* Score: 2 */
    text-decoration: underline;
}

p a { /* Score: 2 */
    text-decoration: none;
}
```

CSS CASCADING — THE SPECIFICITY GAME

Question: *Will the anchor have an underline or no underline? Why?*

HTML:

```html
<li>
    <p>Visit this <a href="#">cool</a> site.</p>
</li>
```

CSS:

```css
li a { /* Score: 2 */
    text-decoration: underline;
}

p a { /* Score: 2 */
    text-decoration: none;
}
```

TIEBREAKER? LAST RULE WINS!

CSS CASCADING — THE SPECIFICITY GAME

Question: *Will the anchor have an underline or no underline? Why?*

HTML:

```
<li>
    <p>Visit this <a href="#">cool</a> site.</p>
</li>
```

CSS:

```
p a { /* Score: 2 */
    text-decoration: none;
}

li a { /* Score: 2 */
    text-decoration: underline;
}
```

TIEBREAKER? LAST RULE WINS!

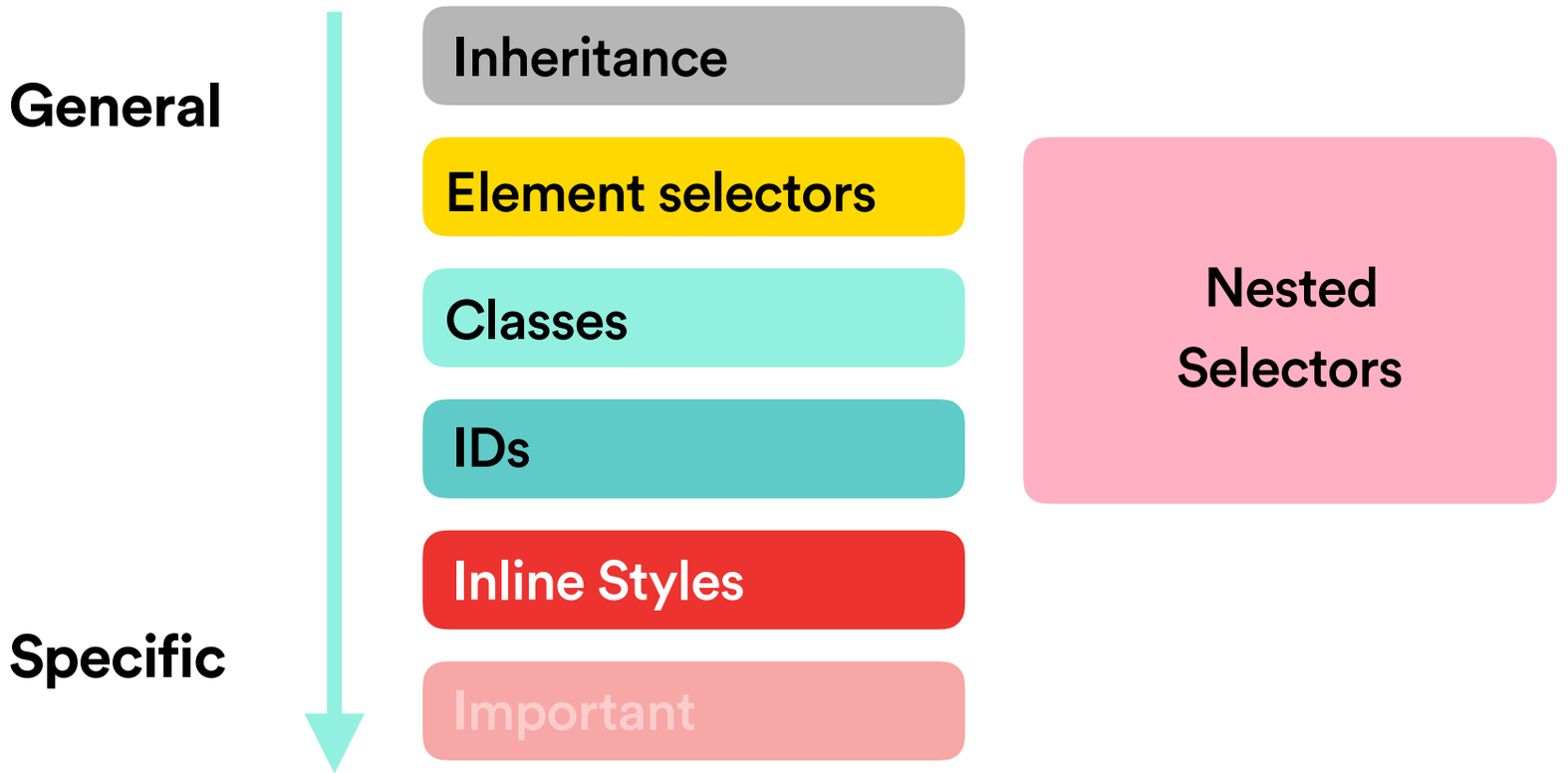
THINGS YOU SHOULD USE IF YOU WANT TO BE

INLINE STYLES

```
<li style="color: red;">Content</li>
```

BAD!!!!

MORE ABOUT CASCADING



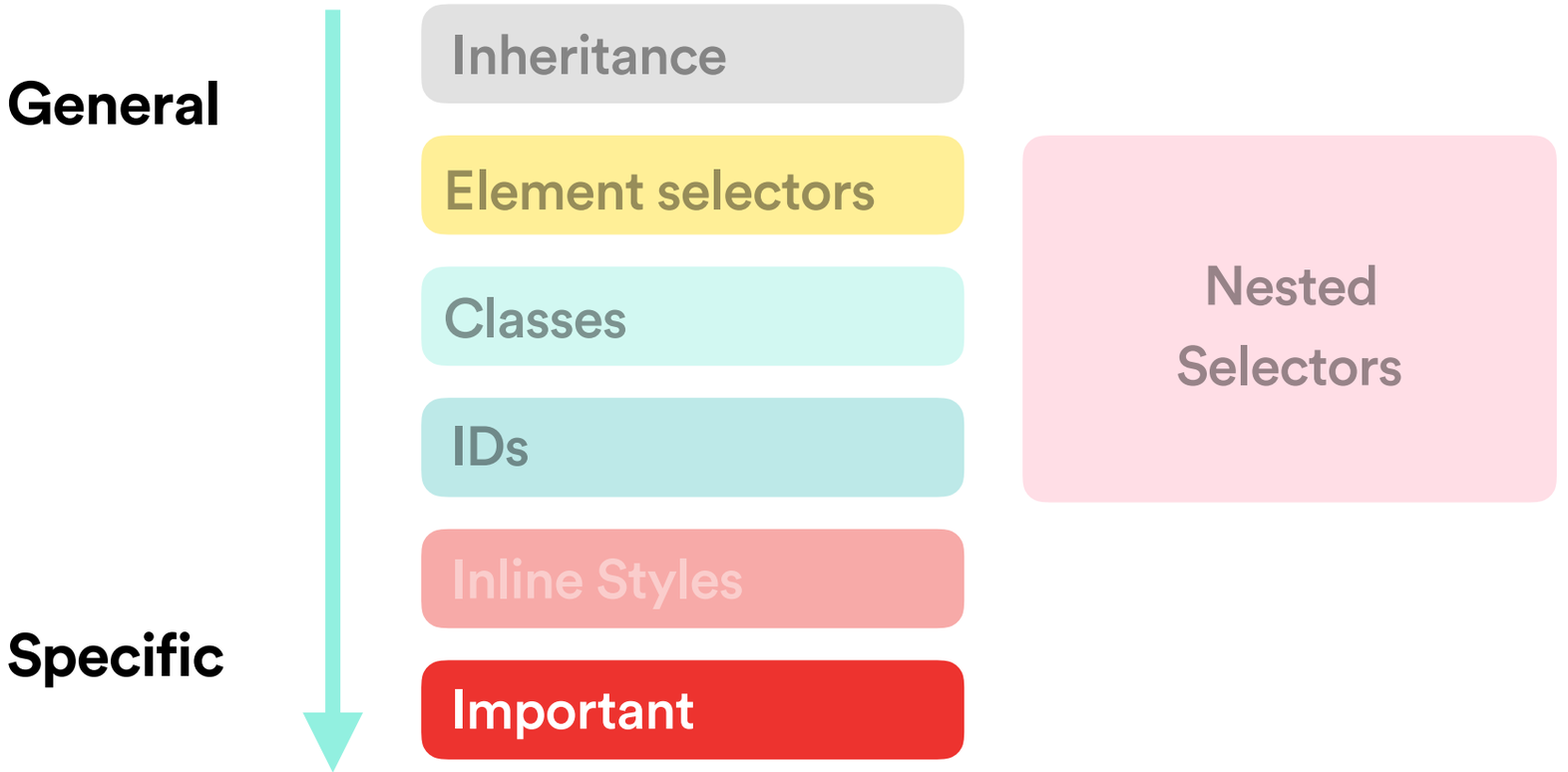
CSS IMPORTANCE

Adding **!important** after any property value indicates that it should be considered *more important than other rules that apply to the same element.*

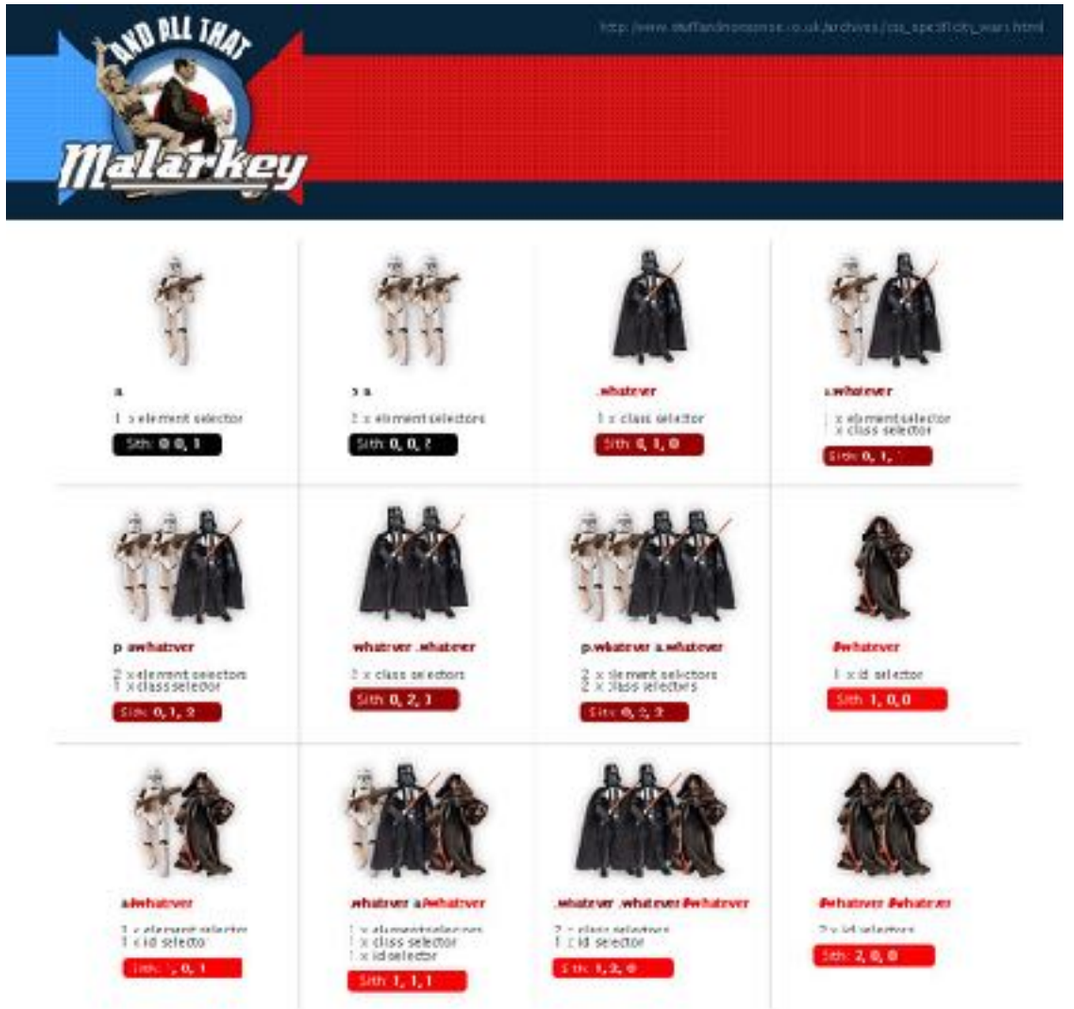
```
li {
   font-size: 100px !important;
}
```

ONLY WHEN NECESSARY!!!
(Which is almost never)

MORE ABOUT CASCADING

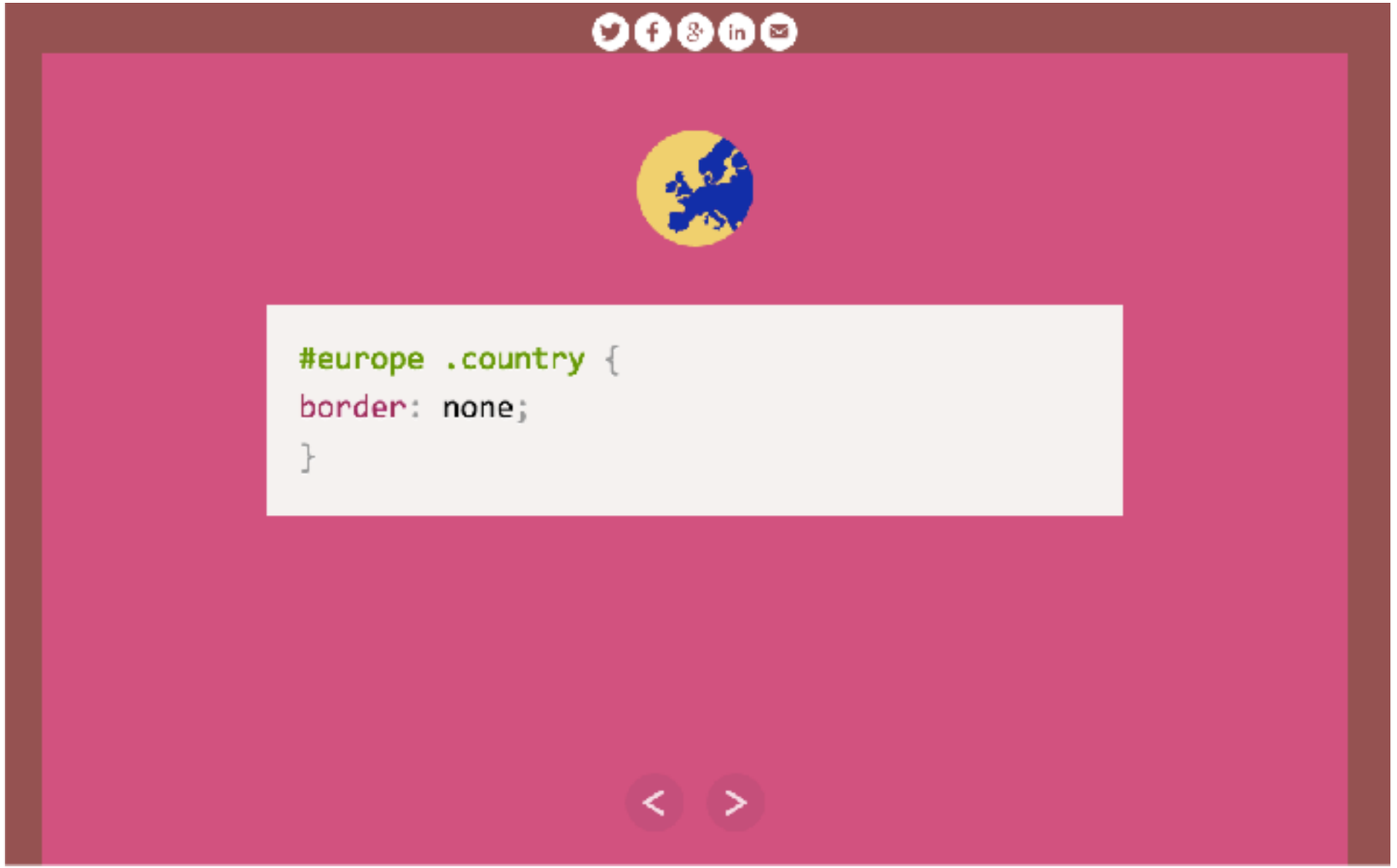


CSS CASCADING — THE SPECIFICITY GAME



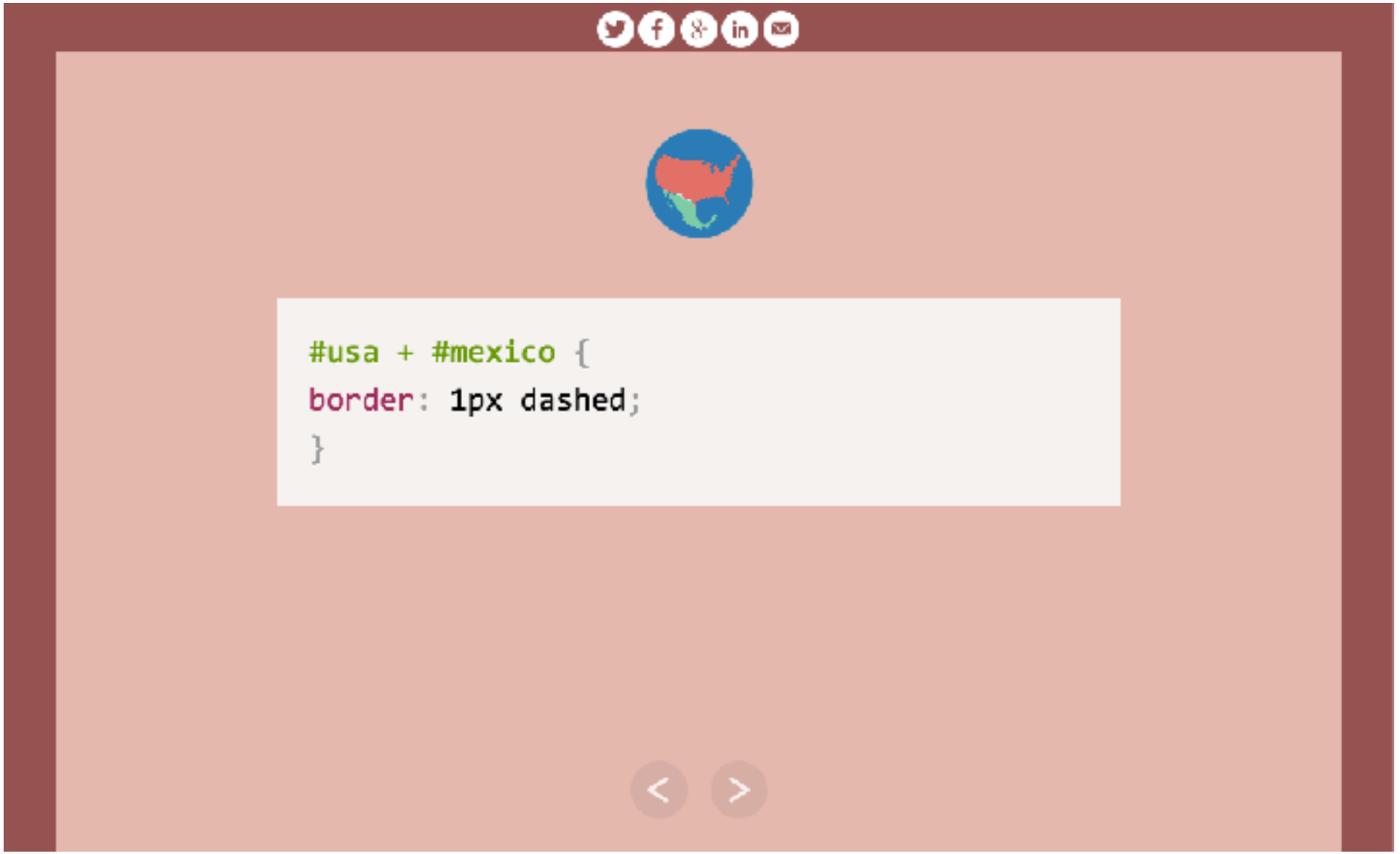
CSS JOKES

CSS PUNS



CSS PUNS

CSS JOKES



MONUMENT MAGAZINE

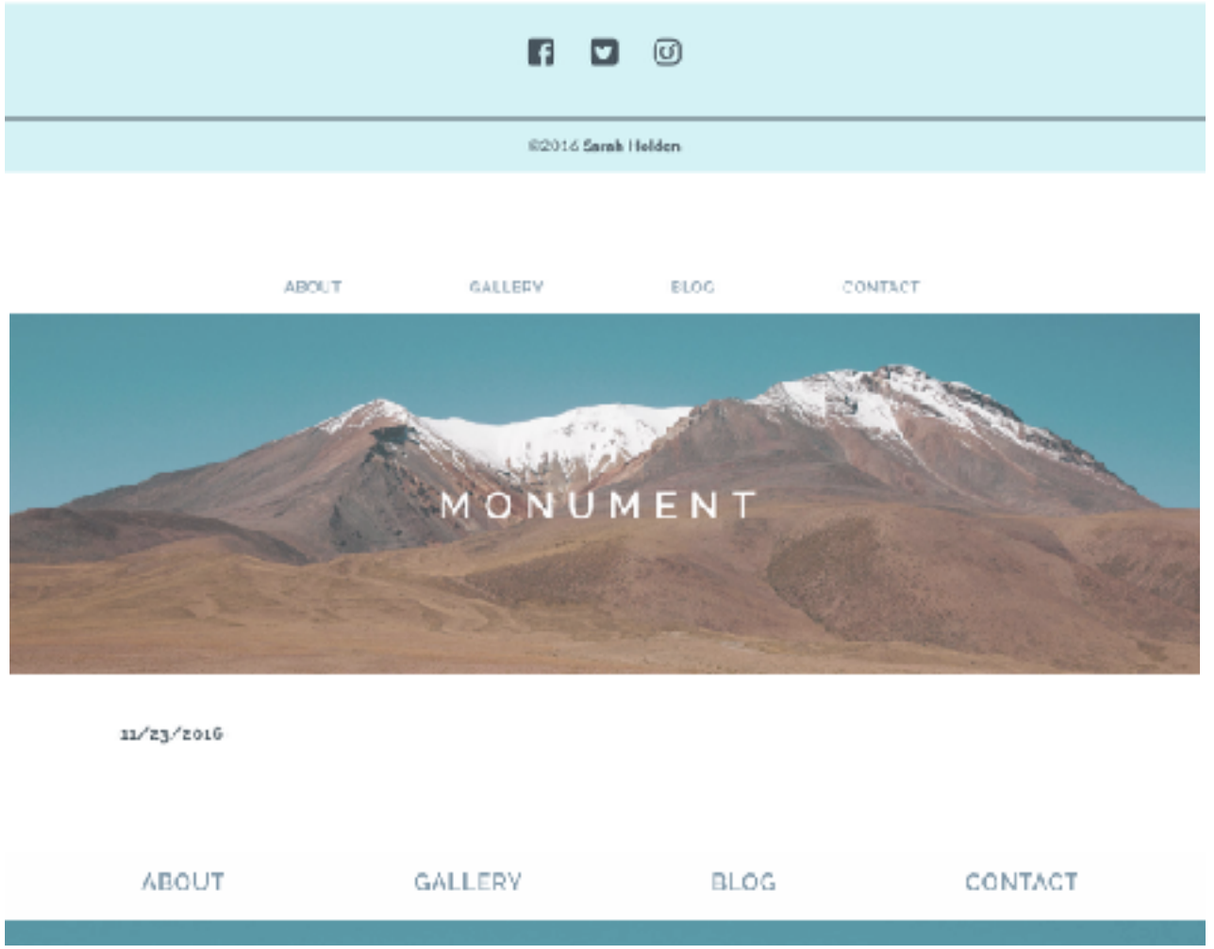
OVERVIEW

Congratulations! You have been hired as the developer for Monument Lifestyle Magazine. You are working closely with a designer and have been provided design mockups for the website. Your goal is to replicate these mockups as closely as possible.

You will be adding components to the blog page in the labs/ homework this week and will be working on building the landing page for the site during the lab sessions next week.

MONUMENT

BONUS TASKS:


©2016 Sarah Holden
ABOUT
GALLERY
BLOG
CONTACT
MONUMENT
11/23/2016
ABOUT
GALLERY
BLOG
CONTACT

LAB — MONUMENT MAGAZINE

ACTIVITY - DON'T FORGET YOUR HANDOUT!

GA ADVANCED CSS

SELECTORS:

TYPE	
DESCENDANT	
UNIVERSAL	
MULTIPLE	

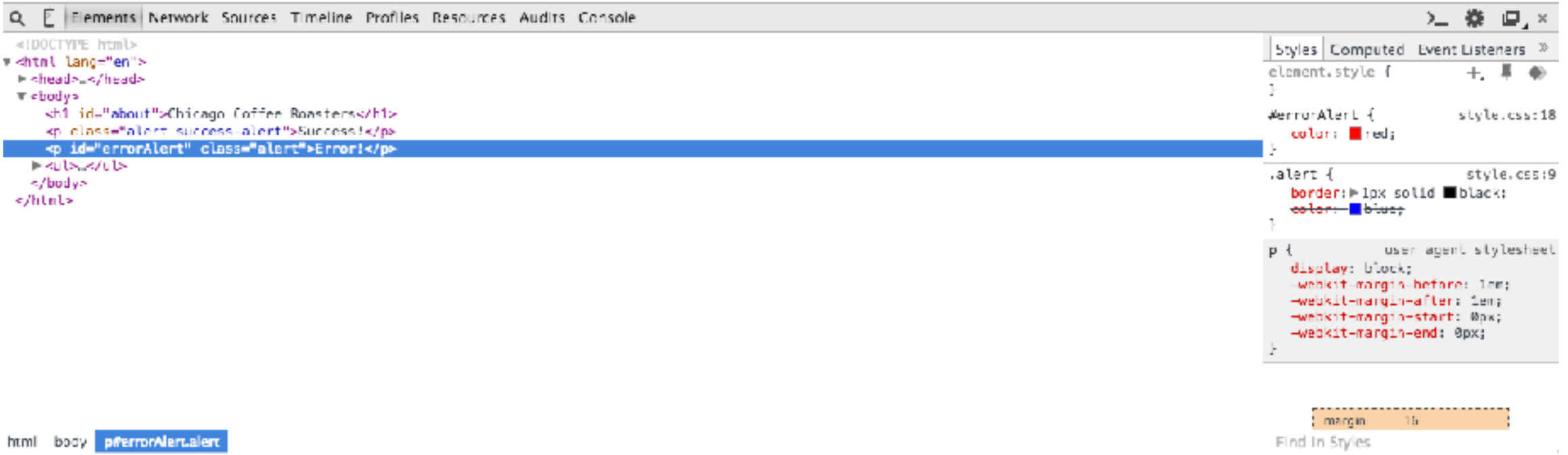
BOX MODEL:

CLASSES AND IDS:

CSS:

margin	
margin-top margin-right margin-bottom margin-left	
padding (shorthand)	
padding-top padding-right padding-bottom padding-left	
border	
border-top border-right border-bottom border-left	
height	
width	

ACTIVITY - KEEP THE INSPECT ELEMENT PANEL OPEN AT ALL TIMES!



To access the elements panel (in Chrome): Right click > Inspect Element

ACTIVITY

KEY OBJECTIVE

- Recreate the Monument Magazine site, using monument_blog.png as a reference

TASKS

1. Look through the provided HTML.
2. Start by adding styles that are the most common to the body
3. Add styles! Start general and only get more specific when you need to.
4. **Bonus 1:** Work from styleguide_medium
5. **Bonus 2:** Work from styleguide_advanced (You'll need to do some research to implement a background-image in the header and FontAwesome icons in the footer)
6. **Bonus 3:** Add a hover effect to links

LEARNING OBJECTIVES

- Use padding and margin to add spacing to a page.
- Describe inheritance in CSS.
- Add classes and IDs to HTML elements and apply CSS styles to elements based on class and ID.
- Apply CSS to elements based on their relationships.

WEEKLY OVERVIEW

WEEK 2	CSS Selectors / Layout
WEEK 3	Layout Lab & Wireframing / Responsive Design

EXIT TICKETS