FEWD WEEK 2 • CLASS 3:

CSS Core Concepts

https://slides.com/jennifermeade/fewd-2-3/live

A QUIZ TOO?!?

SHOW OF HANDS: About Your Assignment

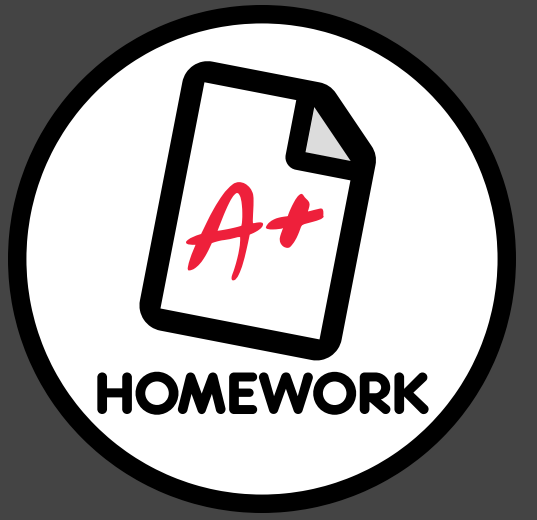
Were you able to successfully complete the homework following the instructions?

- ☐ Nailed It!
- ☐ Ummm... you gave us the answers... not that I'm saying I looked at them...
- ☐ Wait, there was homework?

YOUR HOMEWORK

- You did an amazing job no matter how far you got.
- Last week we covered massive amounts of HTML; about **60%** of all of the CSS2.1 spec feature; and a little CSS3!
- After we review the assignment, we need to learn how to submit them via GitHub.

HOMEWORK REVIEW



OBJECTIVES

- Working with GitHub to submit assignments
- CSS Selectors in depth
- Demystifying cascade and inheritance

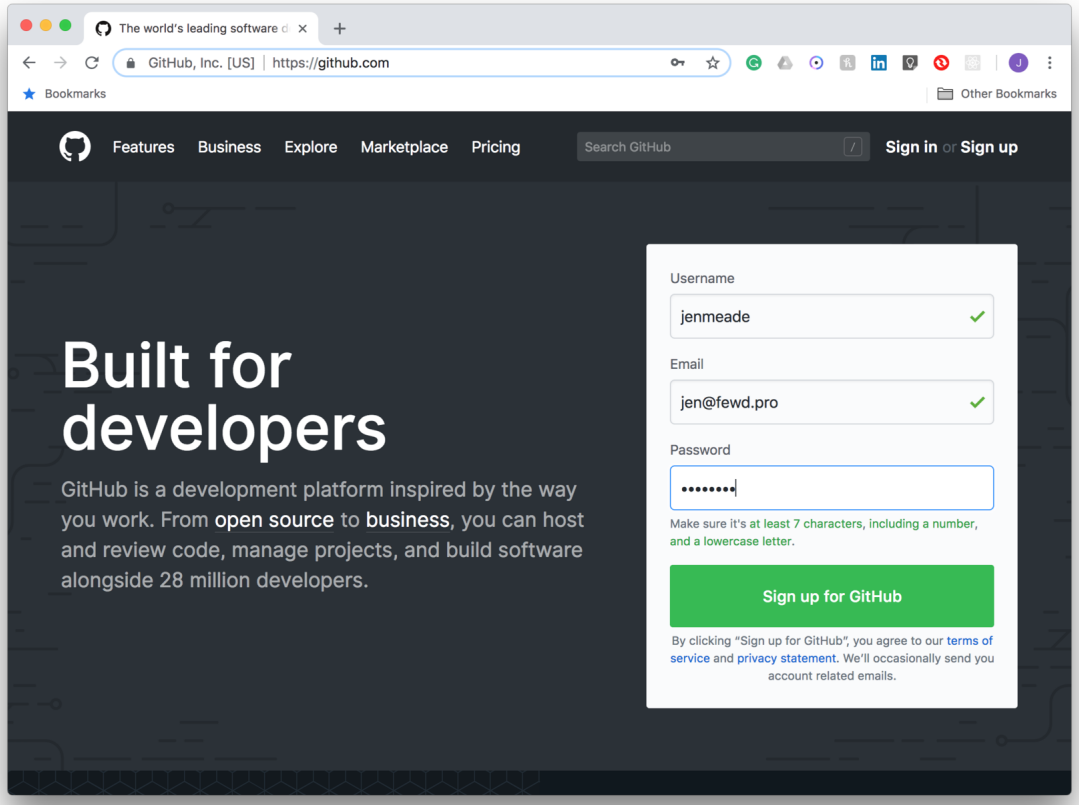
SUBMITTING YOUR ASSIGNMENTS

Or... Let's Learn GitHub!!!

WHAT IS GITHUB PAGES?

GITHUB ACCOUNT

If you don't already have a GitHub account, create one now.

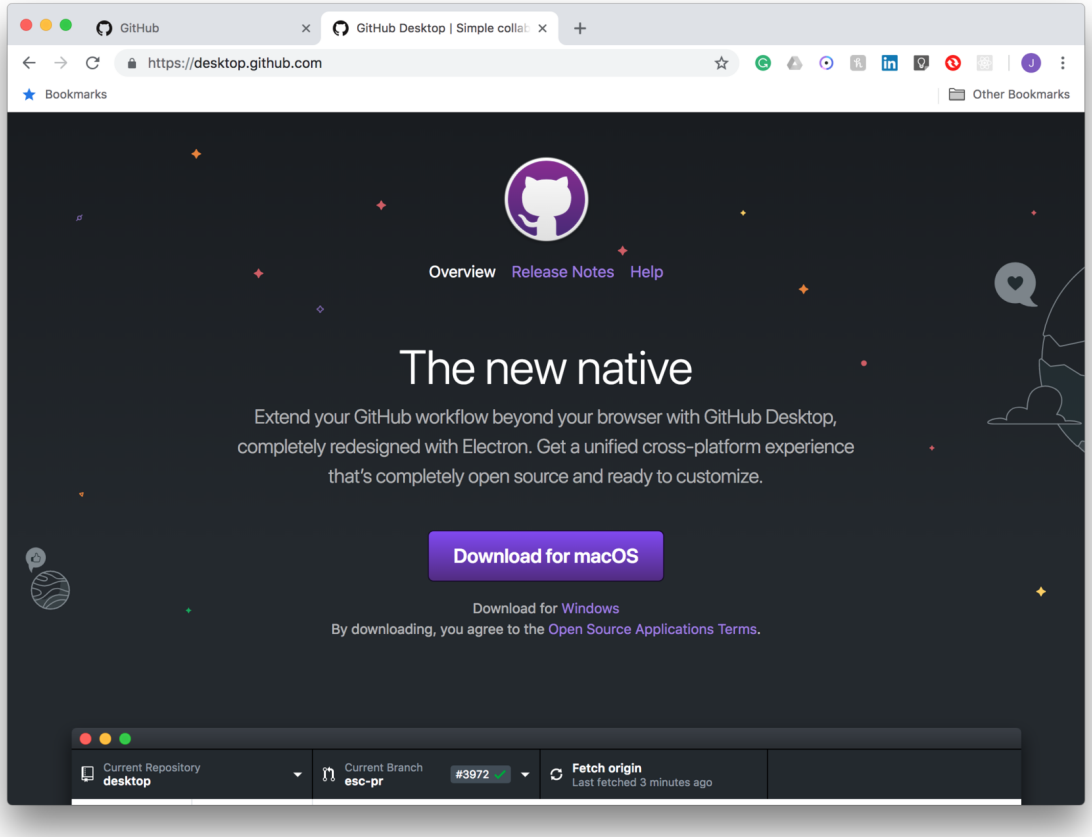


GitHub

1. Sign up at github.com.
2. Select the free plan.
3. Verify your email.

DOWNLOAD GITHUB DESKTOP

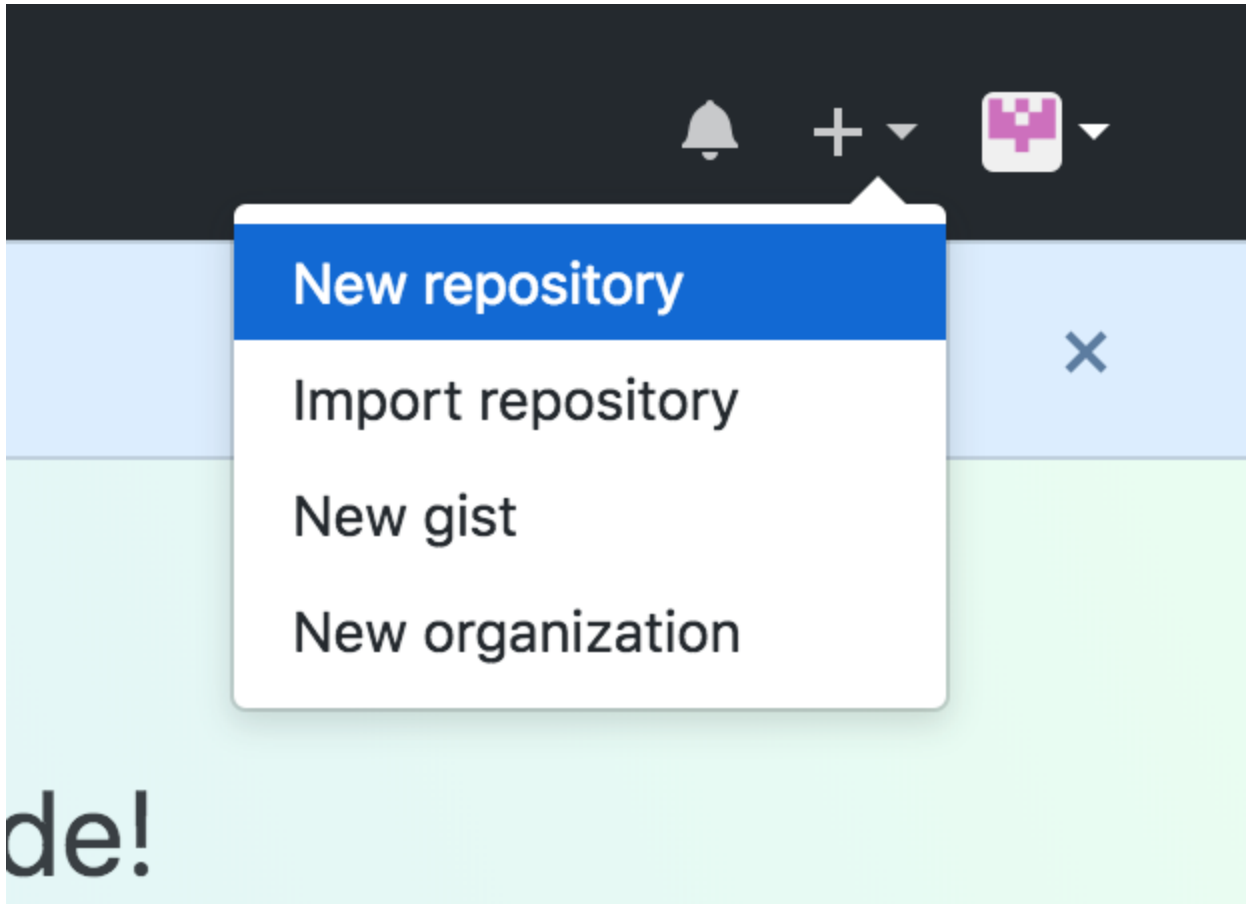
Download and install Github Desktop.



1. Download the installer: **desktop.github.com**
2. Launch the installer and follow the onscreen prompts.

CREATE A REPOSITORY

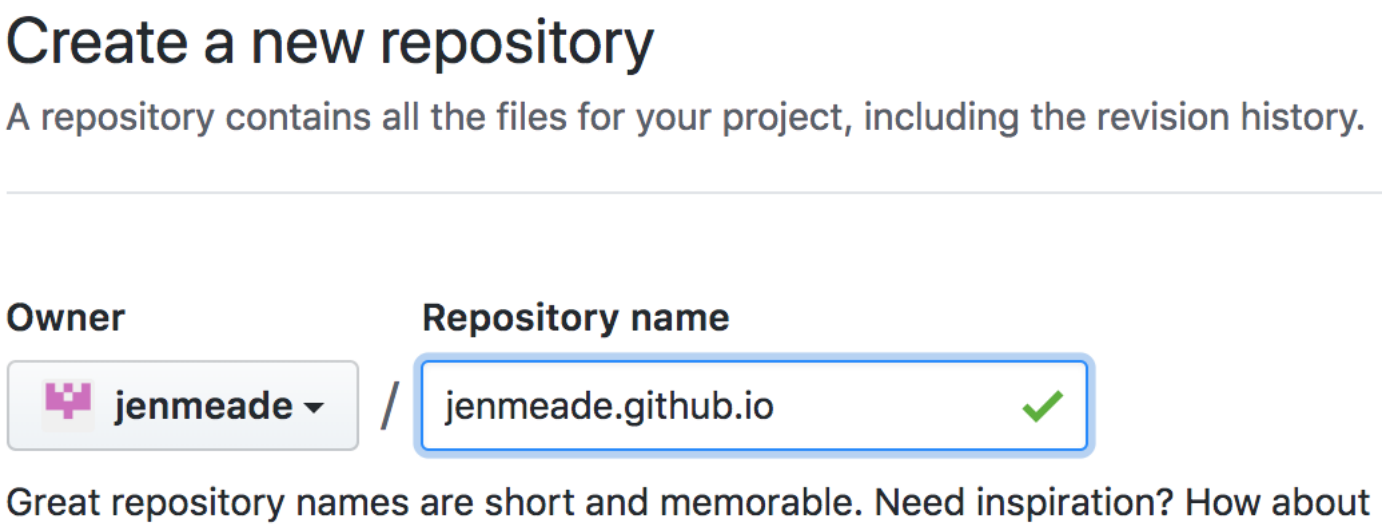
Back in GitHub, click the **plus** symbol in the upper right corner and choose **New repository**.



GitHub

NAMING YOUR REPO

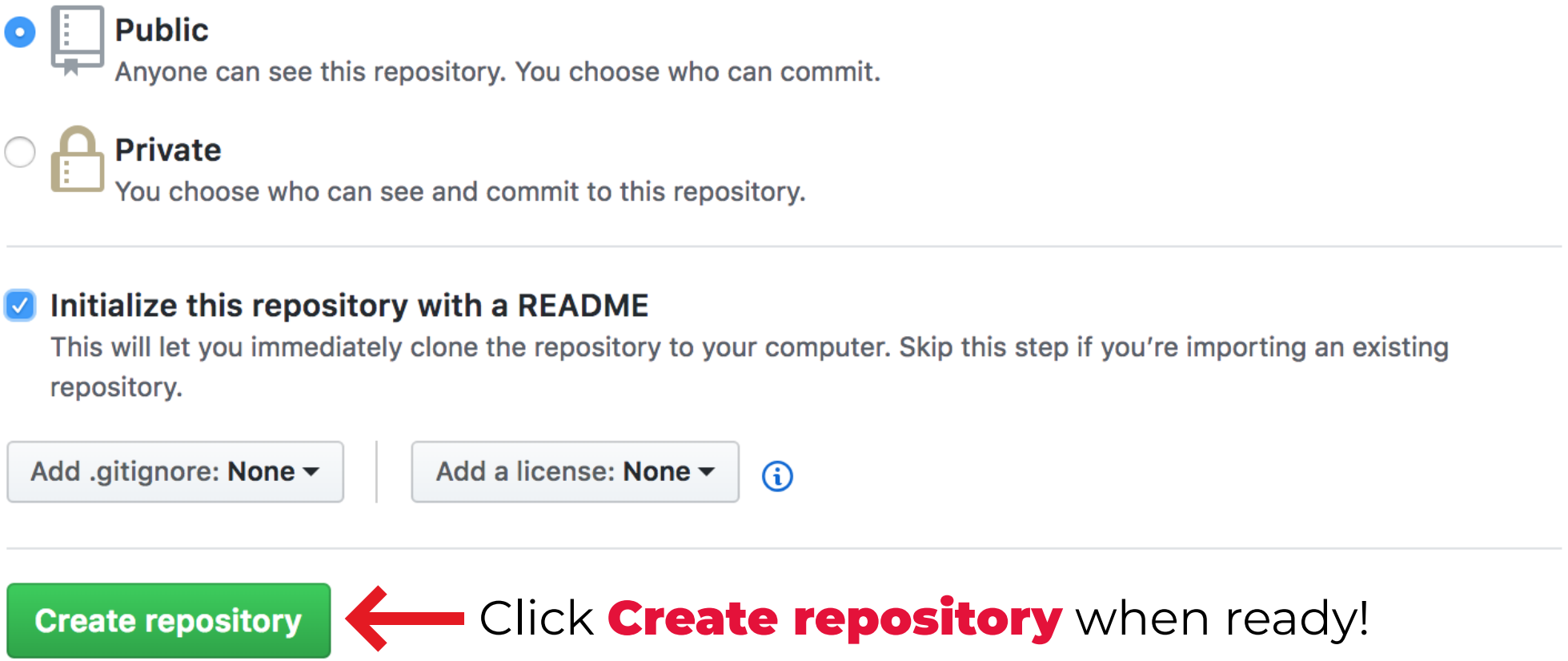
Enter the name of the repository as:
***your-username*.github.io**



Make sure your username and repository name match exactly!

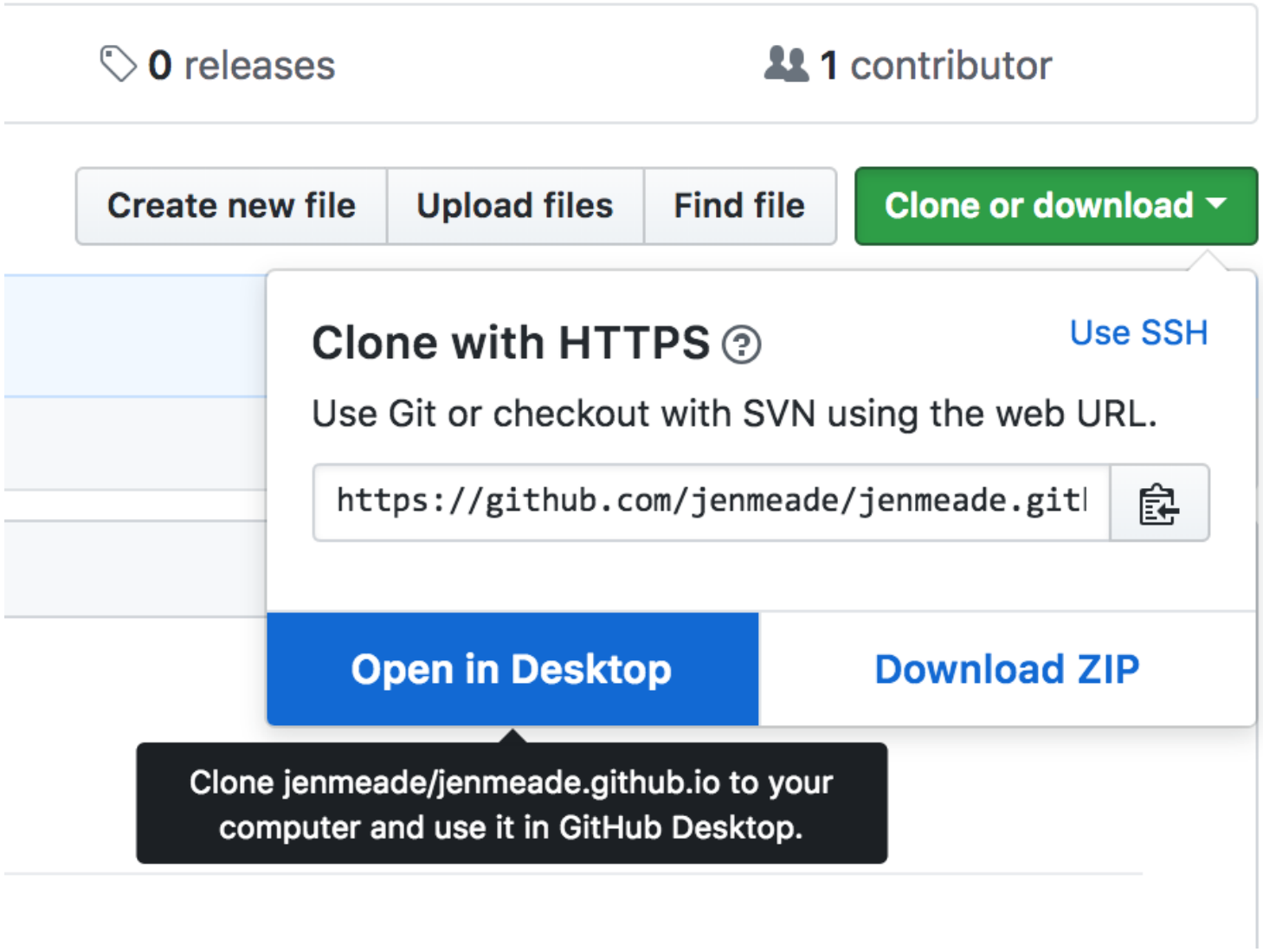
INITIALIZING YOUR REPO

Set the repo to **Public** and select **initialize this repository with a README option**.



Click **Create repository** when ready!

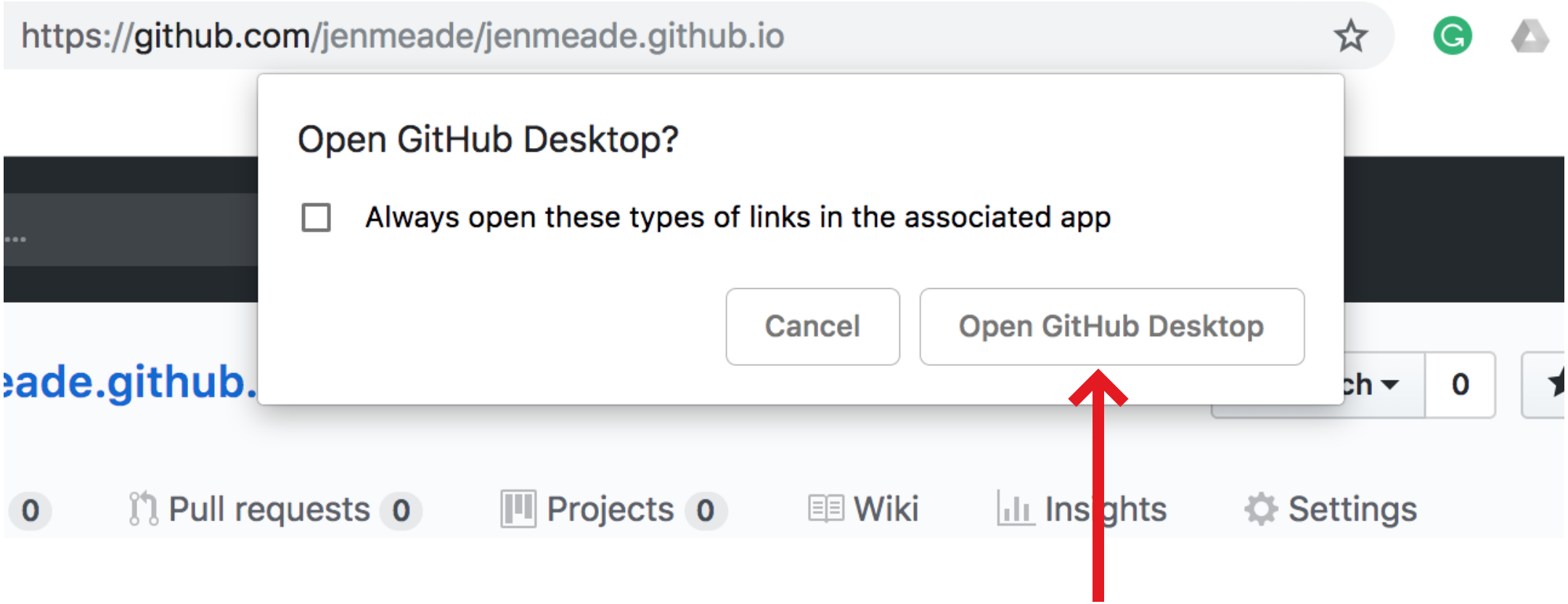
SETTING UP YOUR WEBSITE



← Click **Clone or download**

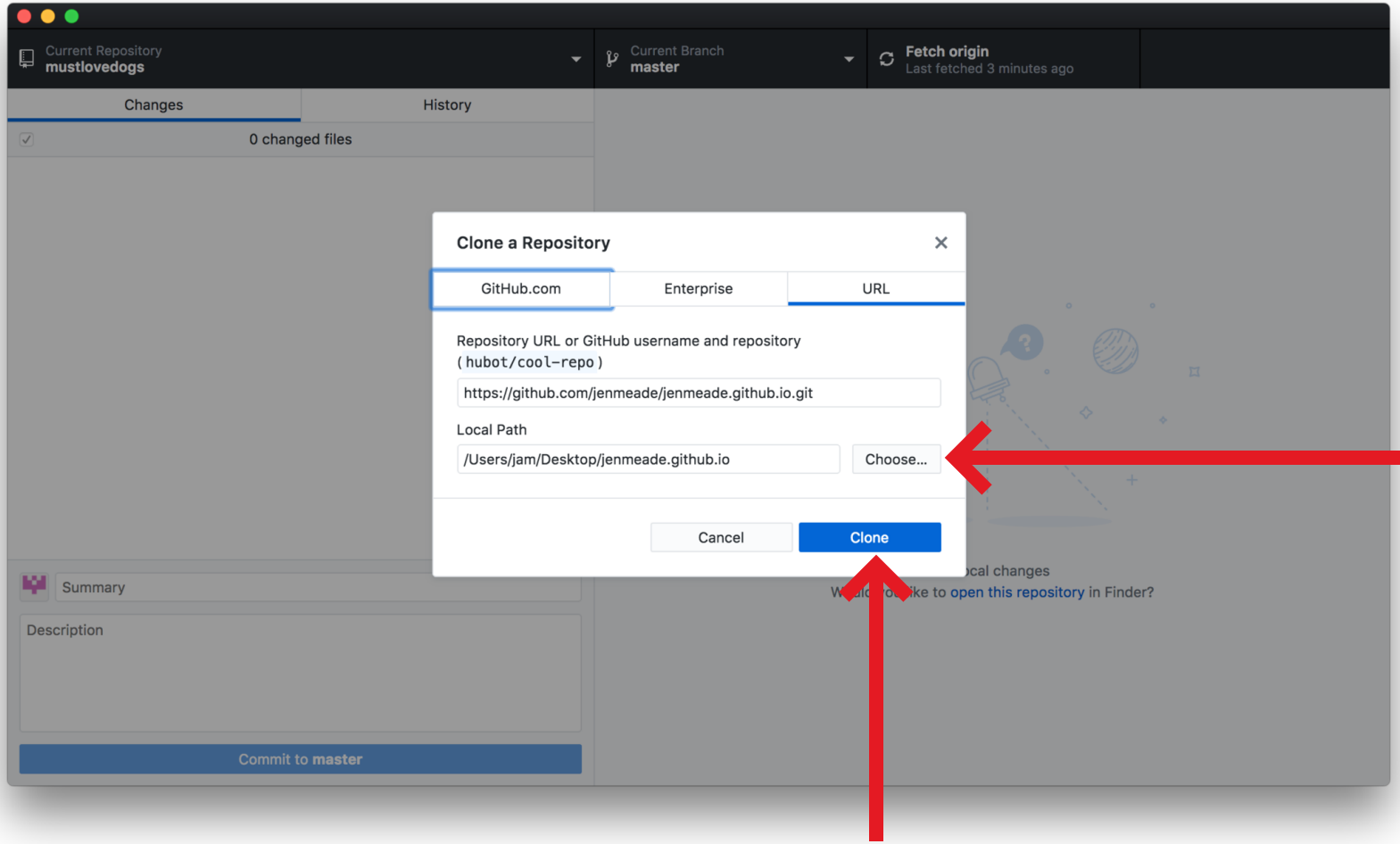
← Click **Open in Desktop**

ACCEPT ANY WARNINGS



Click **Open GitHub Desktop**

CHOOSE A LOCATION FOR YOUR LOCAL REPO



Click **Choose...**

GitHub Desktop creates the folder, so just choose where you'd like to store the folder for your repo.

Click **Clone** when ready!

LET'S ADD A FILE

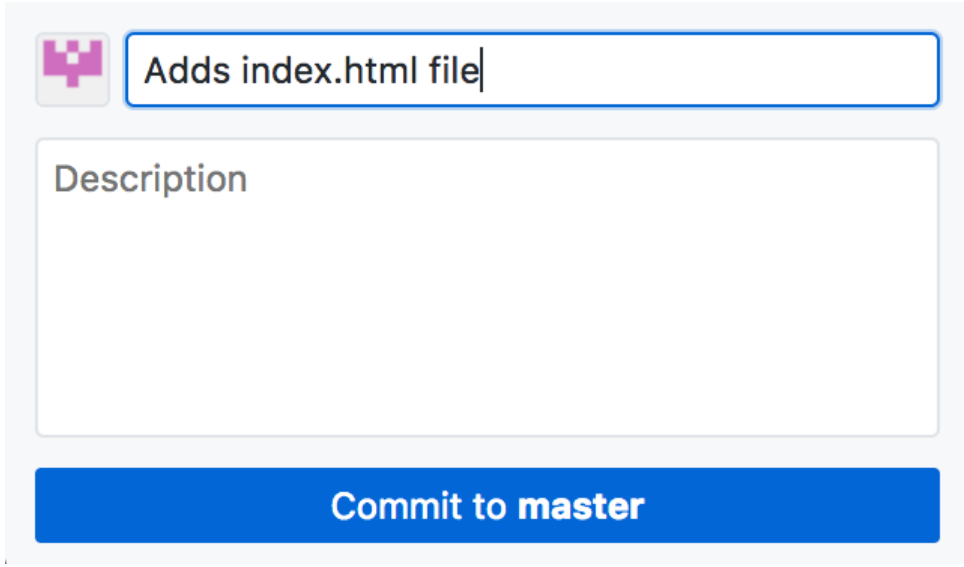
1. Drag the newly created folder on to Atom to launch it as a new project.
2. Right-click the folder in Atom and choose **New File...**
3. Name the file **index.html**.
4. Add your page boilerplate.
5. Add an h1 element and give the page a heading.
6. Save the file.

SYNC'ING UP WITH GITHUB

- Back in Github Desktop, you'll see that your new file addition is listed in the changes.
- Git will track all of the files you add, modify and delete from this folder making it easy to rollback changes if needed!
- To sync your changes with GitHub.com, you have to **commit** and then **push** the changes to the server.

GIT COMMIT

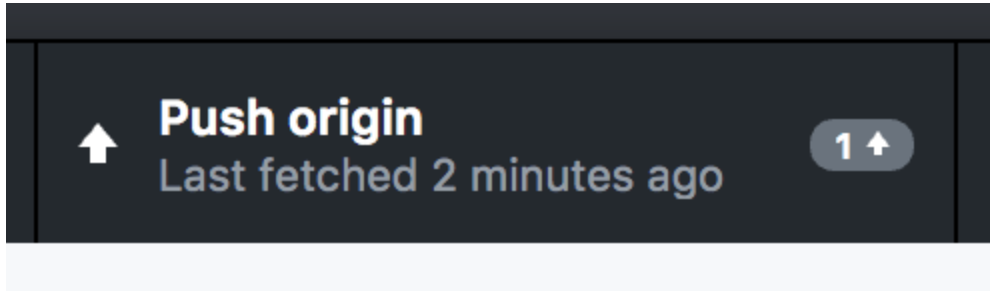
- In GitHub Desktop, add a commit message in the field at the bottom left
- After you add a message, click the **Commit to master** button



It's customary, to write commit messages in the present tense.

GIT PUSH

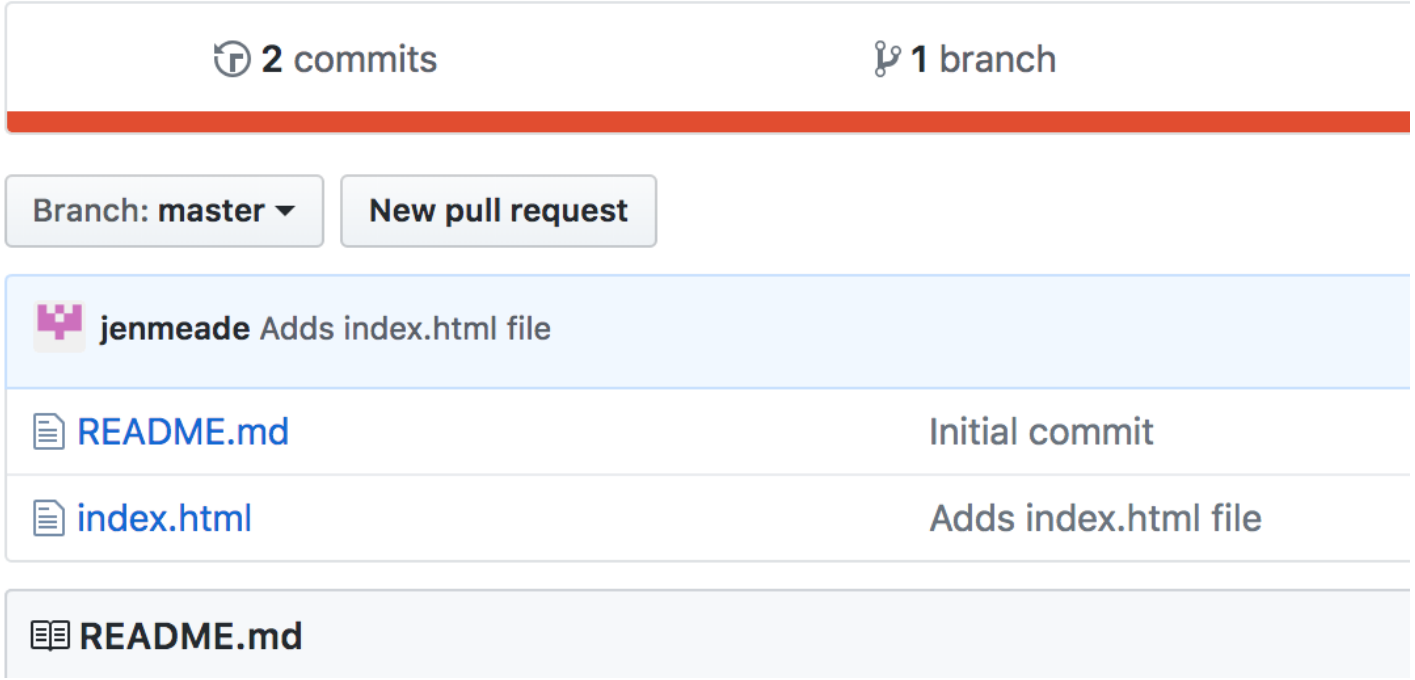
- To push the file to the server, you'll click the **Push origin** button on the right in the menubar



If the files have not been committed, you will not see the Push origin option.

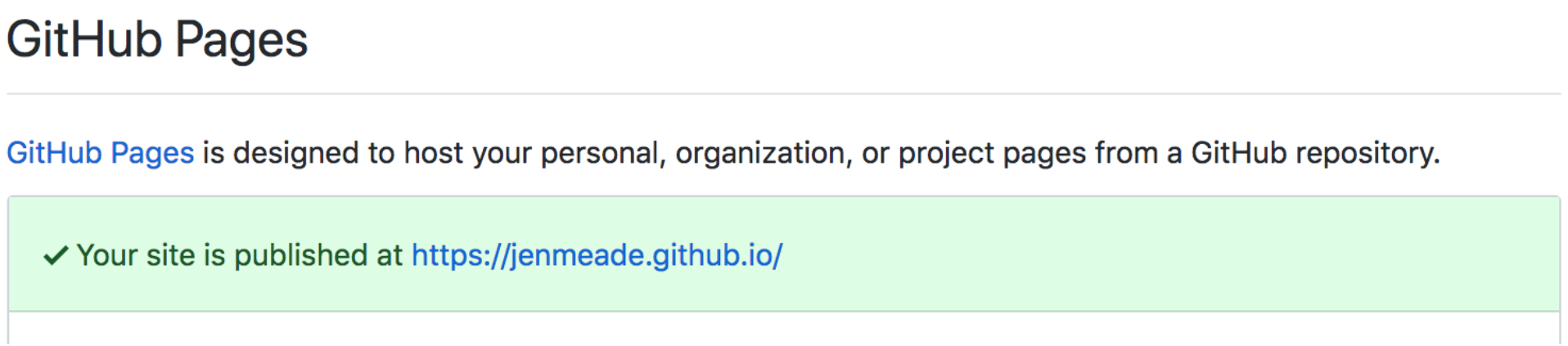
CHECK YOUR FILES ON GITHUB

- Refresh the repository page in your browser and you should now see your file!



CHECK OUT YOUR NEW WEBPAGE!

- Click the **Settings** tab on the top right.
- About halfway down the page under the heading **GitHub Pages**, is a link to your live site. Go ahead and click it...



LINKING TO YOUR HOMEWORK

- Back in Atom, add an **unordered list** to the index.html.
- Inside the first **list item** add an **anchor tag**.
- Inside the anchor, type: **Assignment #1**.
- Right-click on the project folder in Atom's navigator pane and choose **Add Folder**.
- Name the folder: **homework1**
- Link the **index.html** file that we'll put inside your homework folder using a relative path inside the anchor tag **href**.

ADDING YOUR HOMEWORK

- Go to the folder where you created your homework assignment.
- Copy the **index.html and the two folders** containing your css file and images.
- Navigate to your github.io folder and paste the files and folders directly into the homework1 folder you created.

TEST YOUR LINK

- Before syncing your files to GitHub on the web, let's test the homework link.
- Double-click on the main index.html file that is in your local github.io folder.
- Click the link to your homework to make sure it opens.

If the link doesn't work, check that the anchor tag is written as: `<a href="homework1/index.html">Assignment #1</a>` and that the folder is named correctly with no caps or spaces!

COMMIT AND PUSH

- Back in GitHub Desktop, the changes you made to the index.html and the new homework folder with its contents should display in the changes.
- Add a **commit message**, such as: Updates index.html and adds homework assignment #1.
- Click the **Commit to master** button.
- Click the **Push origin** button.

HOMEWORK IS SUBMITTED!

Next week, you get to do it yourself!

CSS SELECTORS IN DEPTH

CSS SELECTORS

1. **Element Tags** ✔
2. **Classes & IDs** ✔
3. Combinators
4. Attributes
5. Pseudo Classes

CLASSES & ID QUICK REVIEW

IDS

- An ID name is unique. It may only be used once on a page.
- An element may only have one ID

```
<div id="extra-special">
```

```
#extra-special {
...
}
```

CLASSES

- Classes are reusable as many times as you want
- An element can have as many classes as you want

```
<div class="big primary">
```

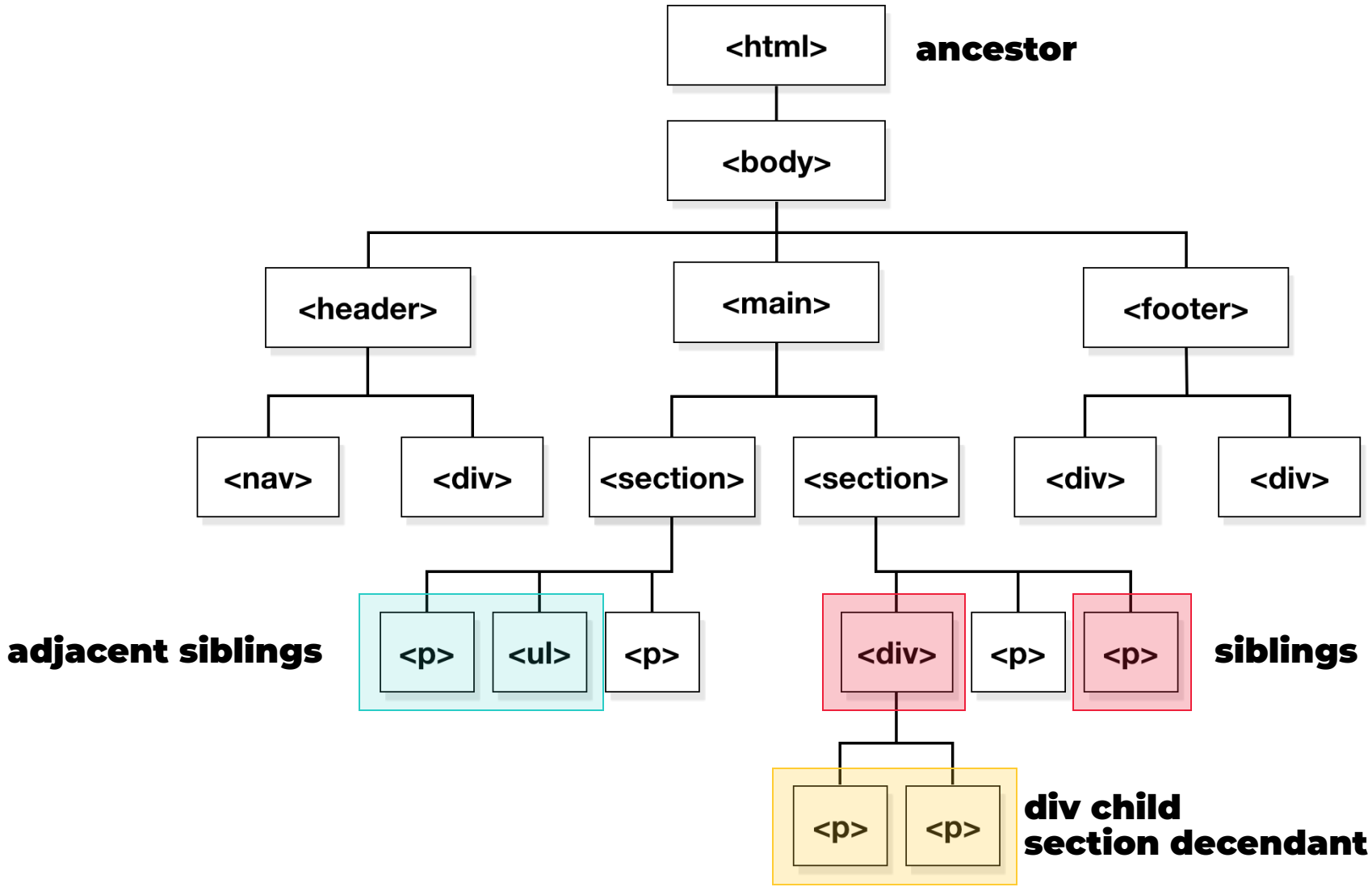
```
.primary {
  ...
}
.big {
  ...
}
```

COMBINATORS

UNDERSTANDING THE DOM

- The Document Object Model represents the page so that programs can change the document structure, style, and content.
- We can visualize our HTML as a DOM tree, sort of like a family tree visualizes the relationships between family members.

DOM: A WEBPAGE FAMILY TREE



CSS COMBINATORS

- CSS Combinators allow us to target elements on the page based on their relationship to one another.
- Combinators always follow the page source order.
- The target of the combinator selector is always the last element in the selector.

USING CSS COMBINATORS

- **Descendant** *space* : Applies to any matching descendants (**any number of levels below**) of the first selector.
- **Child** `>`: Applies only to matching children (only **one level below** the first selector) of the first selector.
- **General Sibling** `~`: Applies to any element that is a sibling of the first element, as long as it appears **after** the first element in the source order.
- **Adjacent Sibling** `+`: Must be the **very next sibling** in the source order following the first selector.

COMBINATORS IN ACTION

```
/* targets paragraphs that are descendants of divs */

div p {
  color: red;
}

/* targets paragraphs that are direct children of sections */

section > p {
  color: blue;
}

/* targets paragraphs are adjacent siblings of another paragraph */

p + p {
  background-color: yellow;
}

/* targets paragraphs are siblings of divs */

div ~ p {
  background-color: orange;
}
```

HEADS UP: It's always the **last** element that we're targeting.

USING COMBINATORS

ATTRIBUTES

ATTRIBUTE SELECTORS

- Attributes let us target elements by their attributes
- The main syntax for attributes is:

```
[attribute="value"] {
  /* styles here */
}
```

TARGETING ATTRIBUTES

```
/* Target the attribute that contains a value */

[title~="flower"] {
  /* styles for elements with title="flower" or title="Summer flower" */
}

/* Target the attributes that start with a value */

[href^="http://"] {
  /* styles for anchors that link to external addresses */
}

/* Target the attribute that ends with a value */

[class$="-box"] {
  /* matches classes like large-box and small-box */
}
```

ATTRIBUTE SELECTORS

PSEUDO CLASSES

PSEUDO CLASSES

- A CSS pseudo-class is a keyword added to a selector that specifies a special state of the selected element(s)
- For example, when an element is being hovered over, or when it is the first or last element in its parent

FIRST-CHILD

```
/* Selects any <p> that is the first element
   among its siblings */

p:first-child {
  color: lime;
}
```

- The element matching the selector **must** be the first element inside its parent
- There's a corresponding last-child

FIRST-OF-TYPE

```
/* Selects any <p> that is the first element
   of its type among its siblings */

p:first-of-type {
  color: red;
}
```

- Unlike the first-child, this element can be anywhere in the source order among its siblings so long as it is the first instance of that type of element
- There are last-of-type and only-of-type as well

NOT

```
/* Selects any element that has a class of blue
   and is NOT a paragraph */

.blue:not(p) {
  color: blue;
}
```

- The not pseudo class can be used on its own or in combination with other selectors

HOVER

```
/* Selects any <img> element when "hovered" */

img:hover {
  border: 5px solid blue;
}
```

- The hover pseudo class allows for powerful interactions without Javascript!

NTH-CHILD & NTH-OF-TYPE

```
/* Selects the 1st, 4th, 7th, 10th... div
   among any group of siblings */

div:nth-child(3n+1) {
  color: lime;
}
```

- The value in the paren can be **even**, **odd**, or a formula in the form of **An+b**
- The formula is calculated with every value of n starting with zero. So, (3*0)+1 = **1**, (3*1)+1= **4**, (3*2)+1= **7**, etc.
- To select just the 3rd element use `:nth-child(2)`

PSEUDO CLASSES

CASCADING & INHERITANCE

CASCADING & INHERITANCE

- Many CSS properties inherit their values from their ancestors and most accept a value of **inherit**
- Properties can be overridden when you provide a rule that has **more specificity**
- Any rules that are **not specifically overridden** continue to be inherited

SPECIFICITY

Specificity is a calculation based on the selectors used. Each type of selector is weighted:

1. Inline Styles (highest)
2. IDs
3. Classes & Attributes
4. Element Tags (lowest)

CALCULATING SPECIFICITY

```
#main-nav > a.nav-item {
  background-color: red;
}
```

```
header > nav .nav-item {
  background-color: blue;
}
```

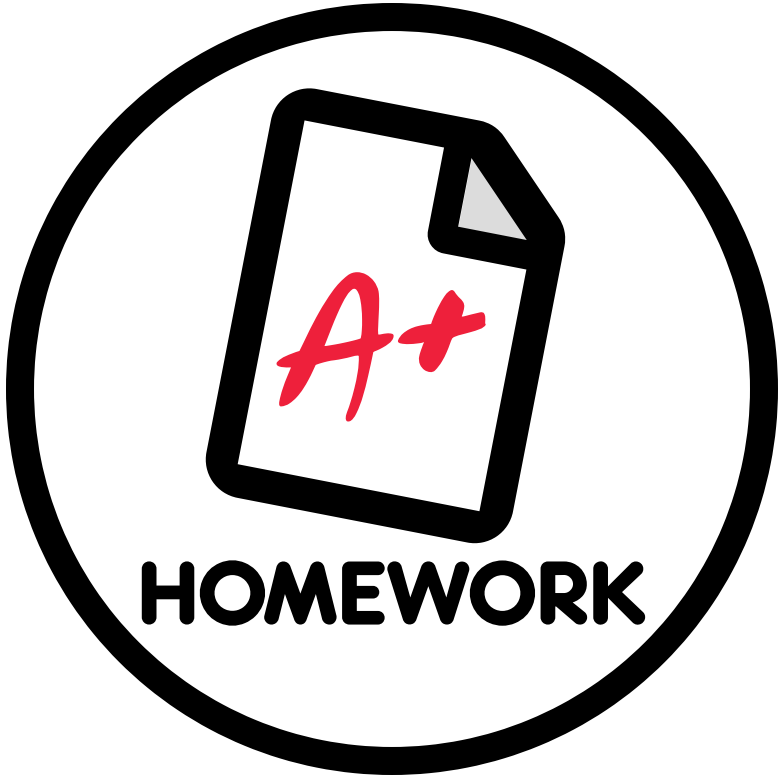
```
header nav#main-nav > a.nav-item {
  background-color: green;
}
```

inline	id	class	element
0	1	1	1
0	0	1	2
0	1	1	3

WHEN SPECIFICITY MATTERS

- When there are multiple rules that contradict one another, the specificity is calculated to determine which is applied.
- For identically weighted rules, the last declaration for wins!
- Inline styles trump **almost** all others.
- One exception is the special `!important` attribute.

GUESS WHICH?



WEEK 1 HOMEWORK

https://github.com/jmeade11/FEWD/Class3/homework

HOMEWORK FOR NEXT CLASS

- Read this article on **Margin Collapse** (https://www.sitepoint.com/collapsing-margins/)
- Add some spice to your homework homepage... Be creative and make it your own!

EXIT SURVEY

https://goo.gl/EB4XFw

GO BUILD AWESOME THINGS!