Sign in up front

- If your name is not on the lsit, write it down and indicate whether you are enrolled or waitlisted
- Grab a blank sheet of paper and make a name tag

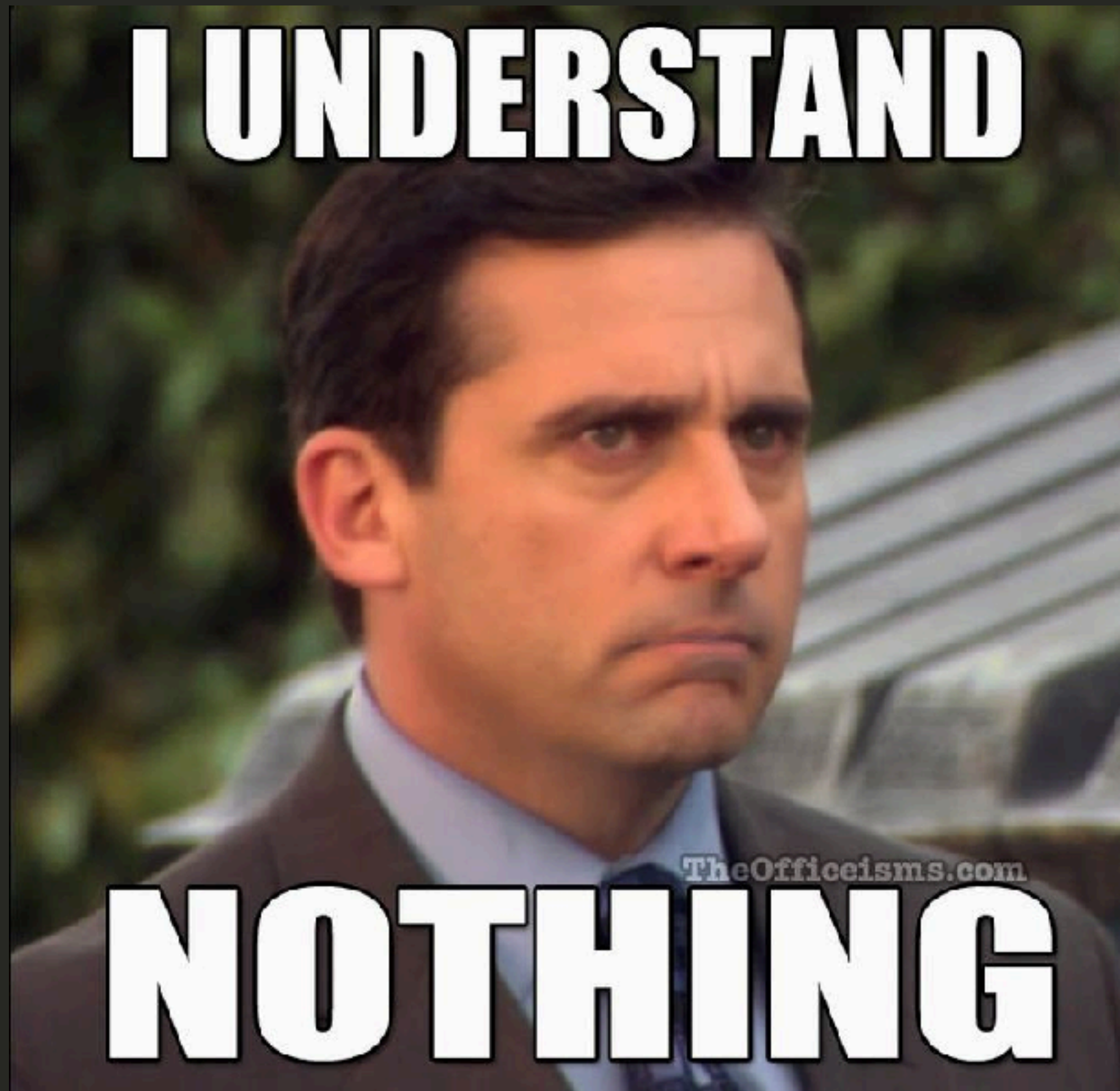# Lab 1: Version Control & HTML

Tricia Ngoon

slides by Michael Bernstein, Scott Klemmer, & Philip Guo

Course Objectives

- Learn some basic web programming and code literacy skills to build a one-page website or portfolio
- Learn some practical tips for future coding projects
- Prepare for more rigorous upper division design courses

Inspiration for this course

Course website

- Website will have all course info, including the syllabus, lab slides, and submission links for assignments
- tngoon.github.io/design90/

Grading

- P/NP course
- Requirements to pass:
  - Regular attendance (1 excused absence/tardy allowed)
  - 3 out of 4 assignments graded w/ a check or higher (on check -, check, and check + scale)
  - Regular participation in class
- Assignments due the following Wednesday at 11:59pm after class (but we'll have class time to complete them)

Understanding the command line

- On Mac, the command line is the Terminal
- On Windows, use Git Bash (make sure this is installed along with git and Github)
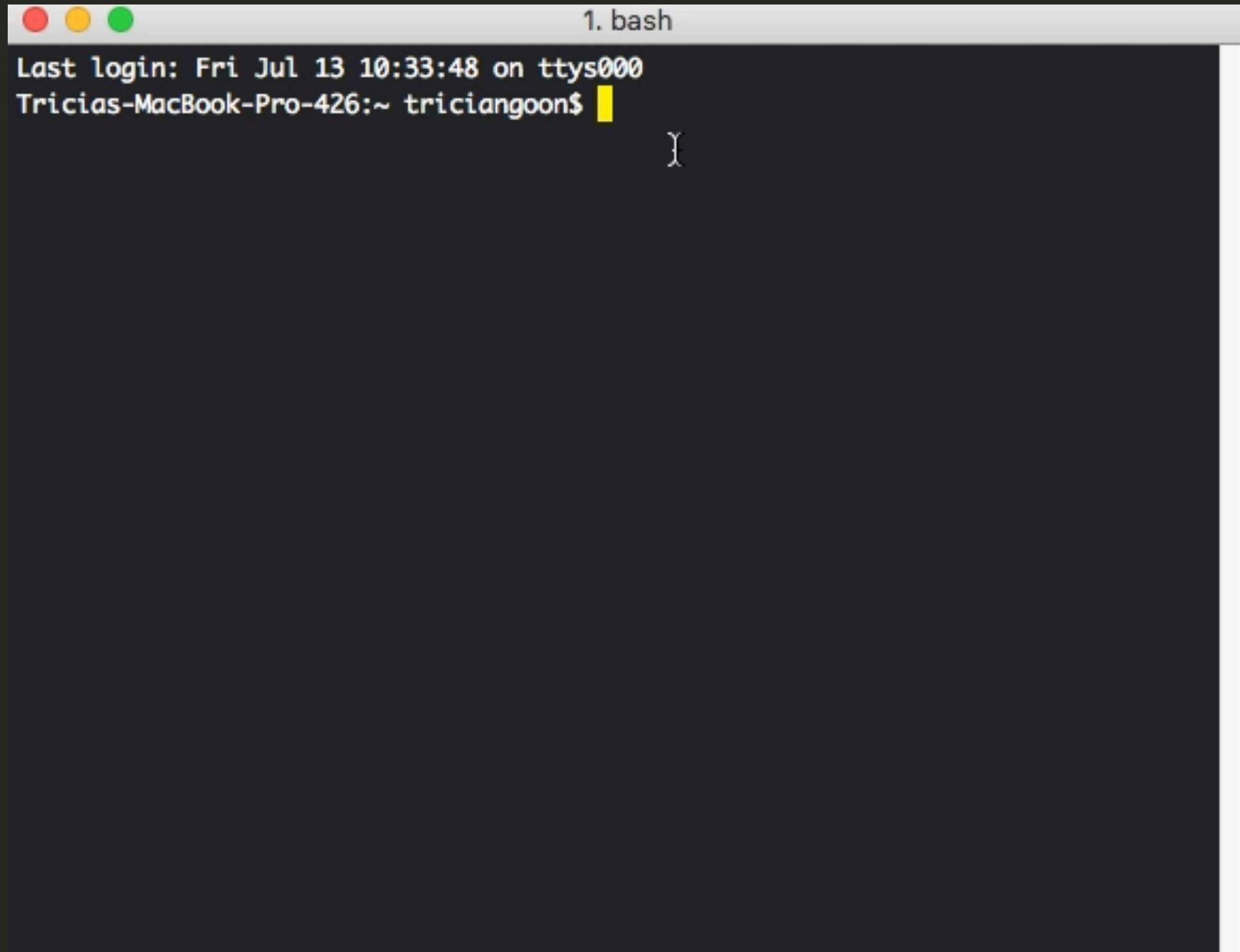
Directories are basically "folders"
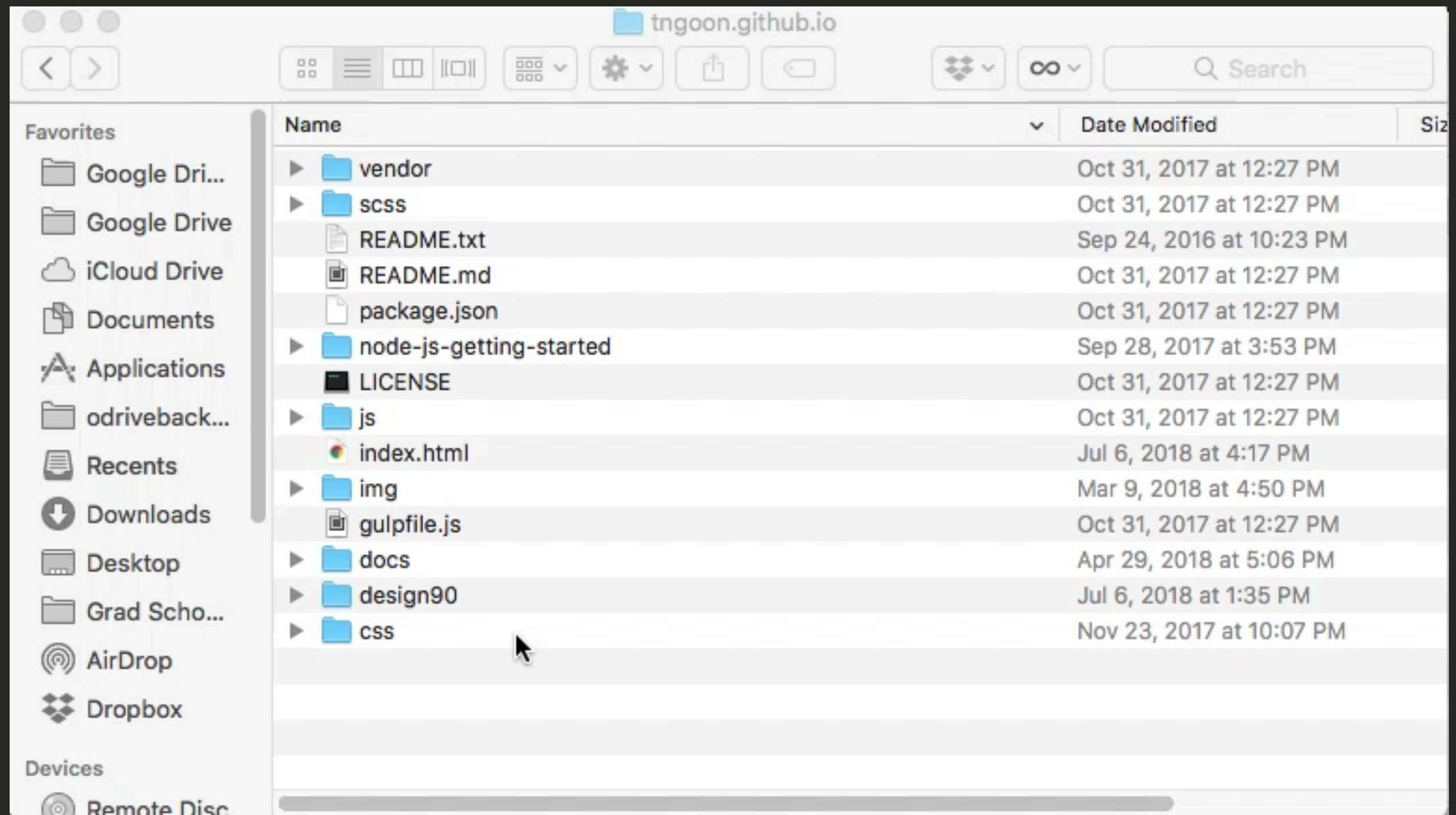
Common command line prompts

- cd = change directory

- pwd = present working directory

- cd ../ = move back one directory

- ls = list all files in current directory

- rm = remove

- Other tips:
  - hitting tab acts as an auto-complete in your terminal
  - Cmd+E (Ctrl+E) takes you to the end of the command, Cmd+A (Ctrl+A) takes you to the beginning
  - Hitting the up arrow button ^ will automatically paste the last command you typed

Navigating directories on the command line

Source Control

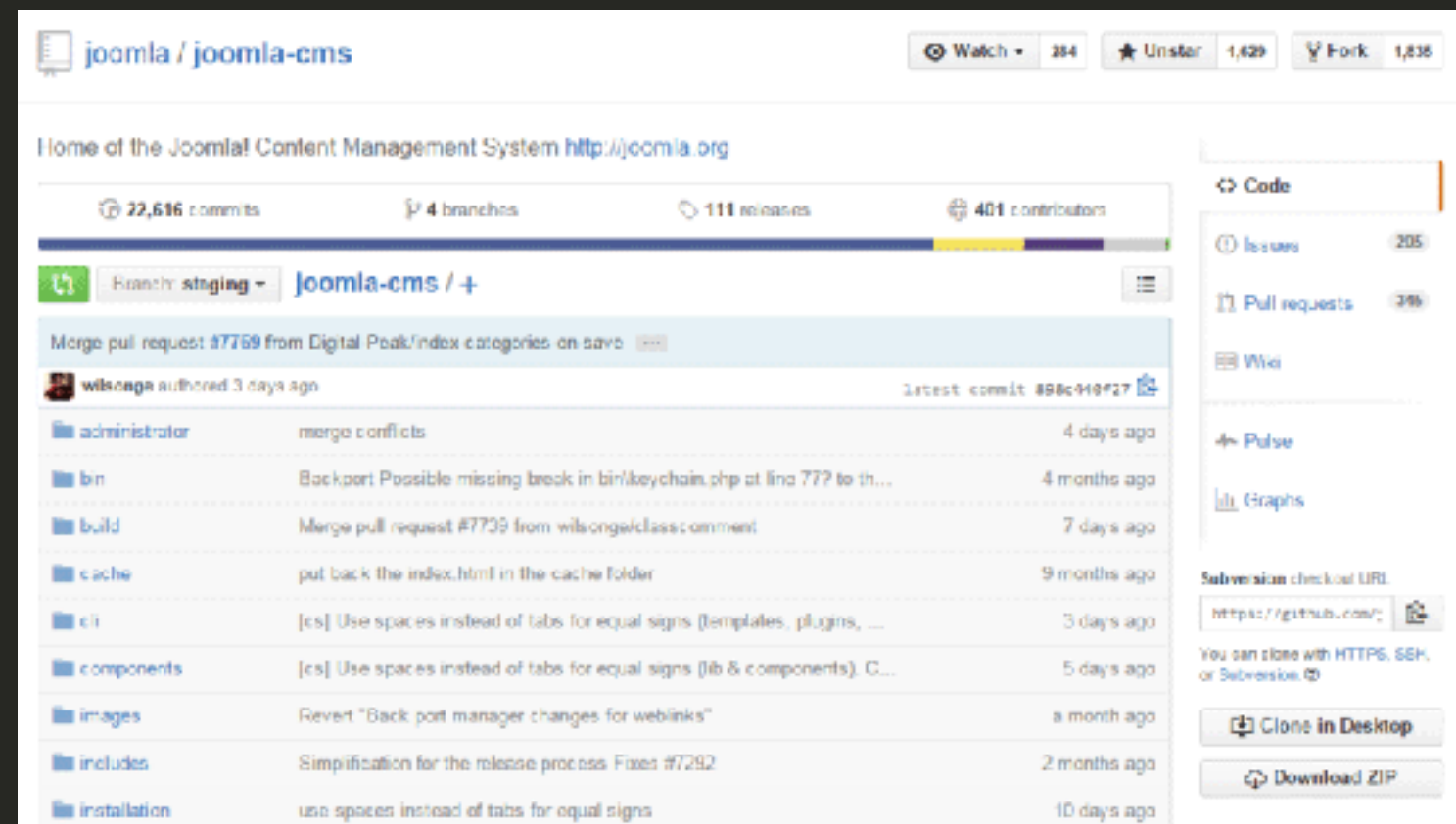# Why should you use version control?

- "My hard drive crashed!"
- "My friend changed everything, and I can't get back the original!"
- You: "I edited complicatedstuff.js." Friend: "I did too." Both: "…F&@*!"

Why should you use version control?

- Back up your code safely in the cloud: GitHub
- Make checkpoints and push to the cloud: git
- Deal with team edits: git conflicts
- Make a web page: static HTML

Why Learn Git?

- Git is a distributed version control system (VCS) that allows collaboration by groups of people on the same documents.

- GitHub is a website that is free to host your Git code repositories and keeps a timeline of your changes.

Understanding Version Control

git pull

GitHub website

git push

Local computer

Make changes

Stage changes

git add

Save changes
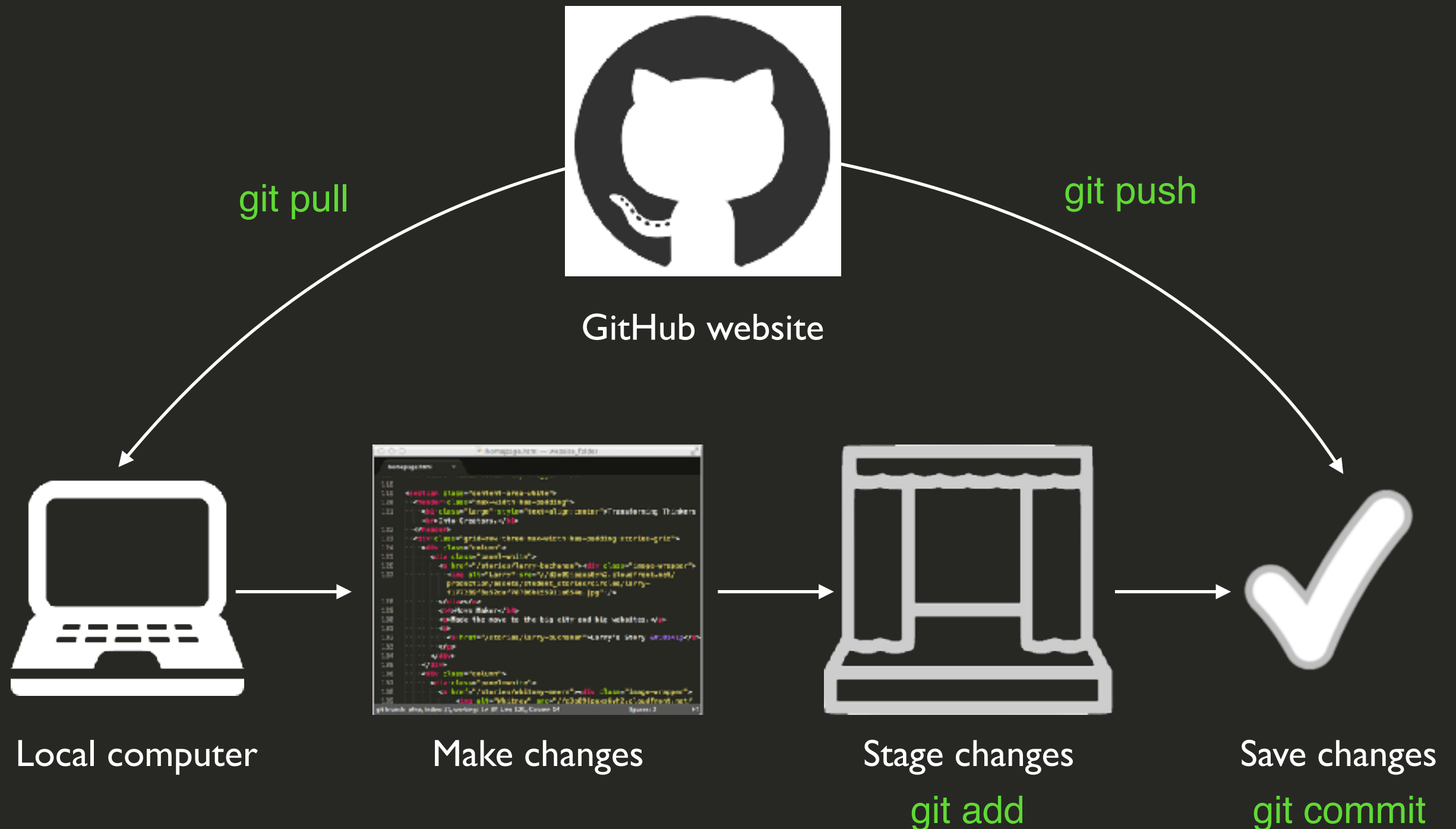
git commit

Important git commands

- git init - initializes git within your repository
- git status - git tells you whether and which files have uncommitted changes
- git add <filename> - Tells git that a file should be included in the commit
- git commit -m "Commit message" - Performs a local commit. It is not pushed to the server.
  Make as many commits as you like.
- git pull - Merges in any new commits from the server
- git push - Shares all local commits back with the git server

Order of steps

- git add→ git commit→ git pull→ git push
- git status can be regularly used to check if you've added/committed all your desired changes
- git pull can be performed at any stage before git push

```
Last login: Tue Aug  7 10:37:19 on ttys000
Tricias-MacBook-Pro-426:~ triciangoon$
```

Other basic git commands

- git commit -a -m "Commit message" performs a local commit of all files that have changed (instead of git add each of them)

- git add . adds all your changed files at once so you don't have to add by each filename

- git log shows your history of commits

- git diff shows what changed but has not yet been committed

Let's get started!

Assignment 1 goals

- Turn the portfolio page scaffold into a full web site and push it to GitHub pages
- Your webpage should contain:
  - A header with your name in it
  - An "About" paragraph with information about you
  - An image of yourself
  - A description of your projects
  - A hyperlink to your Resume/CV in your About paragraph
- Stretch Goal: add more sections and styling

Forking the repository

- Go to github.com/tngoon/dsgn90

- Fork the repository. and copy the URL. IMPORTANT:
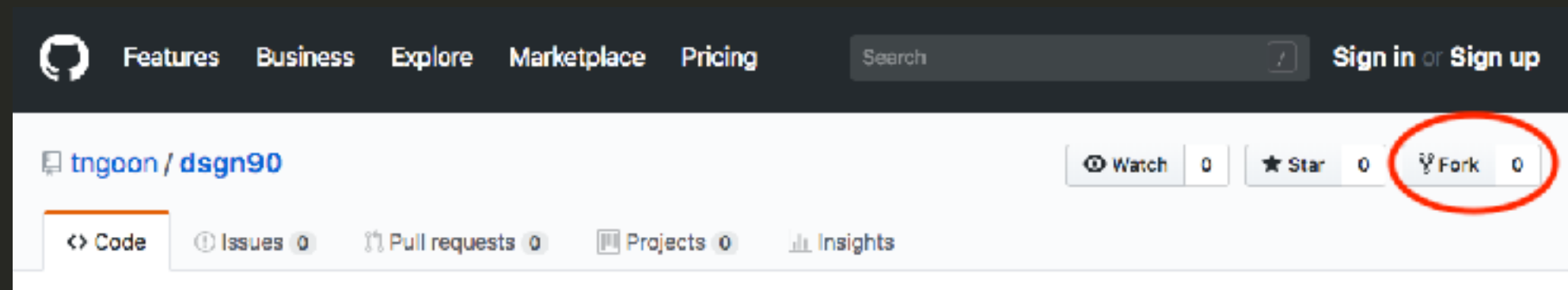

- Now if you go back to your Github homepage, you'll see a repository called <YourUsername>/dsgn90

Making a Github Pages site

- You can host your own static website from Github pages
- In your forked repository (should now be <YourUsername>/dsgn90labs), go to "Settings" in the toolbar.
- Rename the repository to <YourUsername>.github.io — This will allow Github Pages to draw from this repository to make the site.

Clone the repository to your local computer
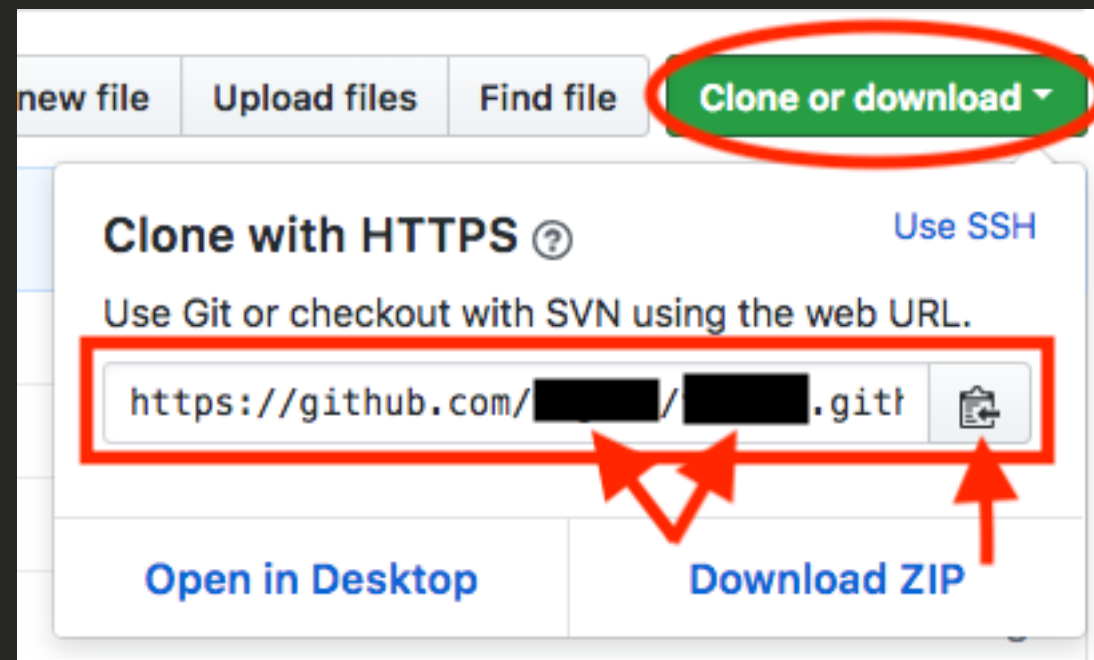
IMPORTANT: Make sure YOUR username is where those black bars are. If you try to clone my repo (tngoon/dsgn90 or tngoon/tngoon.github.io), you will be unable to make edits and will need to start this process over.

Clone the repository to your local computer

- In your Terminal or Git Bash, type git clone <git URL you copied> (Clone this repo in a place you're able to easily access or navigate to -like your Desktop)
- ** IMPORTANT: Make sure the URL you are cloning is your Github repo (<YourUsername>.github.io)
  - If you do not ensure it is your Github repo, you will be unable to make changes, and you'll have to start over.

Let's look at file structure

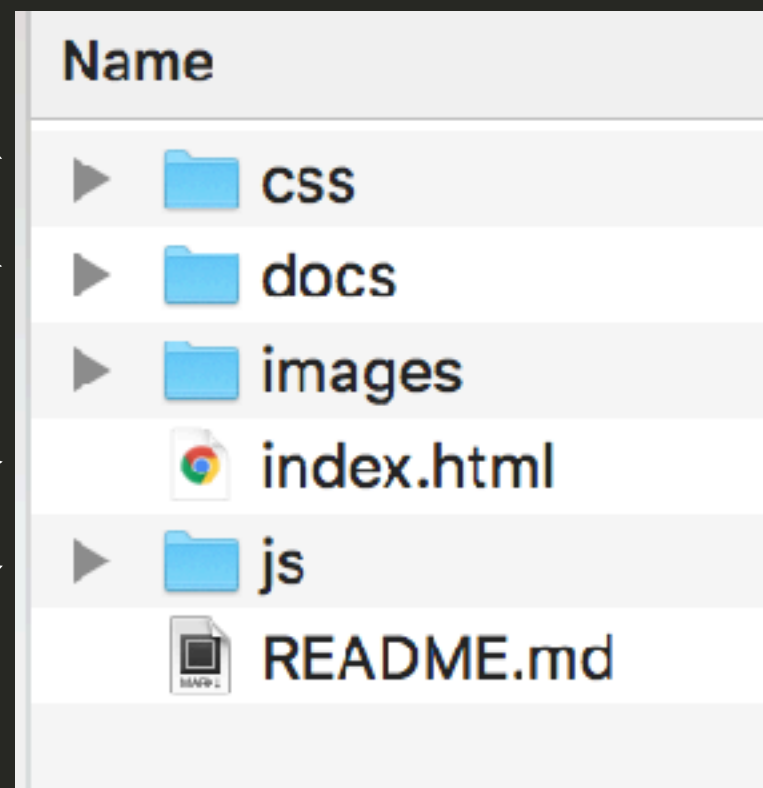- Open your repo in your Finder (Mac) or Explorer (Windows) window

Contains your stylesheets →

Any documents you link in your website →

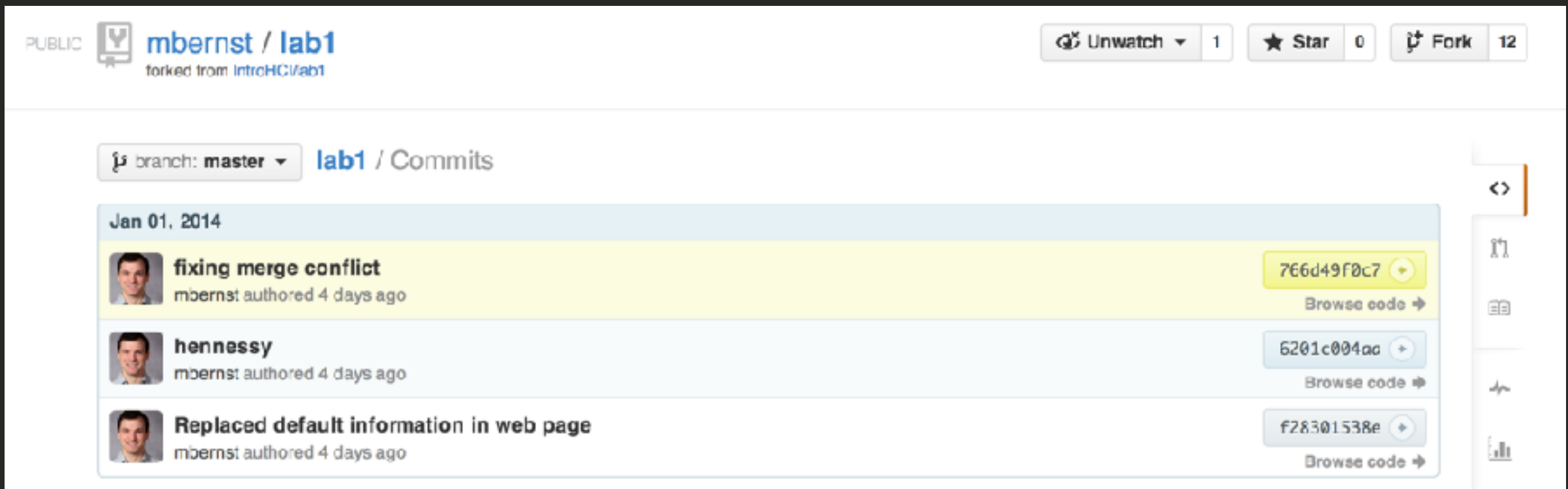Contains any images you use in your website →

Your HTML file that structures your webpage →

Contains JavaScript files for interactivity →

Name

- ▶ 📁 css
- ▶ 📁 docs
- ▶ 📁 images
- 🌐 index.html
- ▶ 📁 js
- 📄 README.md

GitHub website reflects your commit

Browse to index.html using the file on your hard drive, open it from browser

Tricia Ngoon

I am a fourth-year Ph.D student in Cognitive Science at UCSD. Etiam vitae tortor. Vestibulum suscipit nulla quis orci. Donec elit libero, sodales nec, volutpat a, suscipit non, turpis.Donec mi odio, faucibus at, scelerisque quis, convallis in, nisi. Cras dapibus. Pellentesque dapibus hendrerit tortor.

Projects

Activity: Inventing code

- Step 1: Decompose the parts of the webpage (text, images, etc.)

- Step 2: make a syntax for each of these elements

- Step 3: Organize those parts into your "pseudo-code"

Tricia Ngoon

I am a fourth-year Ph.D student in Cognitive Science at UCSD. Etiam vitae tortor. Vestibulum suscipit nulla quis orci. Donec elit libero, sodales nec, volutpat a, suscipit non, turpis.Donec mi odio, faucibus at, scelerisque quis, convallis in, nisi. Cras dapibus. Pellentesque dapibus hendrerit tortor.

Projects

Activity: Inventing code — Reflection

- Does your "code" make sense?

- How might one make an error within your code?

Tricia Ngoon

I am a fourth-year Ph.D student in Cognitive Science at UCSD. Etiam vitae tortor. Vestibulum suscipit nulla quis orci. Donec elit libero, sodales nec, volutpat a, suscipit non, turpis.Donec mi odio, faucibus at, scelerisque quis, convallis in, nisi. Cras dapibus. Pellentesque dapibus hendrerit tortor.

Projects

Site Structure Basics

- HTML = hypertext markup language. It gives a site its structure and content.

- CSS = cascading styles sheet. It gives your content in HTML styling and formatting.

HTML Element

`<p>Hello World!</p>`

↑ HTML Element type

↑ Closing tag

`<a href="www.google.com">Click here</a>`

↑ attribute

↑ attribute link

↑ link text

```html
<body>
    <div>
        <!-- h1 through h6 are headers. The higher the number, the s
        <!-- p means paragraph -->
        <div>
            <h1>Michael Bernstein</h1>
            <p>human-computer interaction &middot; social computing
        </div>

        <h4>Projects</h4>
        <!-- divs are invisib          that stack vert
        <!-- img tags are imag         iver random im
        <!-- a are anchors, a          yperlinks  Use   href a
        <div>
            <a href="project.html">
                <img src="images/lorempixel.people.1.jpeg" alt="Lore
                <p>Waiting in Line</p>
            </a>
        </div>
        <div>
            <a href="project.html">
                <img src="images/lorempixel.ci          lt="Lorem
                <p>Needfinding</p>
            </a>
        </div>
        <div>
            <a href="project.html">
                <img src="images/lorempixel.technics.1.jpeg" alt="Lo
                <p>Prototyping</p>
            </a>
        </div>
        <div>
            <a href="project.html">
                <img src="images/lorempixel.abstract.1.jpeg" alt="Lo
```
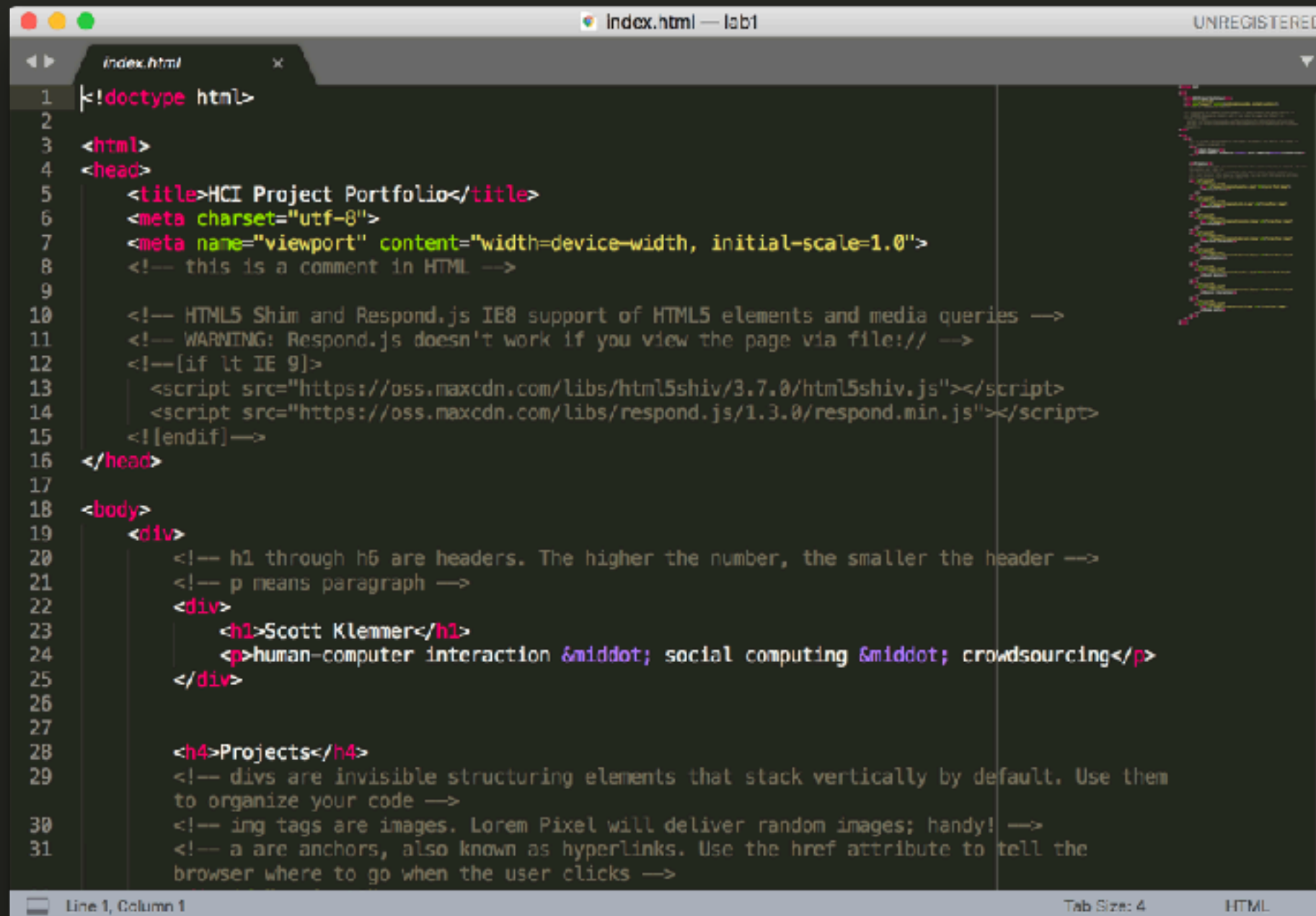
Heading 1

paragraph

div box (invisible)

image

hyperlink

HCI Project Portfolio

msb-lab2.herokuapp.com

Michael Bernstein

human-computer interaction · social computing · crowdsourcing

Projects

Waiting in line

33

HTML syntax & data types

- HTML elements are contained within carrot brackets (`<p>`).
- Most HTML elements require an opening and a closing tag (`<p></p>`).
- The `<head>` tag indicates things that will be read first by the server. Content is not displayed in the webpage.
- The `<body>` tag indicates the actual main content of the webpage.

Practical tips

- Check your browser console for error messages (in Chrome hit the 3 dots in the right corner > More Tools > Developer Tools. Go to Console tab)
- Write a small amount of code at a time and check it incrementally
- Check all variable names, brackets, strings, etc.

Open index.html in your text editor

- This HTML will be rendered by the server

Divs & Text

- <div> = invisible structuring elements that stack vertically by default.

<div>[some other code here]</div>

- <h1>-<h6> = headers for section titles, subtitles, etc. H1 is the largest header.

<h1>Tricia Ngoon</h1>

- <p> = paragraph. Used for non-header text like bodies of articles, "About Me" text, etc.

<p>This is a sentence.</p>

Divs & Text

Tasks:

- Add your name to the <h1> in index.html
- Add an "About Me" header (should be smaller than h1)
- Add a paragraph with text about yourself beneath the "About Me" header
- Describe your projects in the <p> tags under the "Project" header

Images & Videos

- <img> = image. Images have a source tag (src) to display the intended image. Images can be sourced from a directory or from the internet. Videos work the same way.

<img src="images/cats.png">

<video src="https://youtube.com/video">

Images & Videos

Tasks:

- Replace the lorem image under "About Me" with an image of yourself
  - Do this by placing a photo in the "images" folder and sourcing to the right target image
- Add images for each of your projects

Hyperlinks & lists

- <a> = anchor. Used for linking text to URLs or to other HTML pages.

Google

- and = unordered and ordered lists. Creates the list ("parent" element).

- = list element. Makes your specific list items ("children" elements).

  Finish Design 90 Assignment

Hyperlinks & Lists

Tasks:

- Add a hyperlink to your Resume/CV
  - Add a PDF of your Resume/CV to the "docs" directory and direct your hyperlink to this document.
- (Optional) Add a list to your webpage

Other HTML tips

- It's important to put attributes in quotes (<p class="info">). No quotes will be read as variables, whereas quotes indicate strings.

- You can comment out code using this: <!-- -->

<!-- This is a comment. It will not show up on your webpage or affect its functionality -->

Add Styling

Good webpages don't look like this

Cascading Style Sheet (CSS)

- CSS syntax includes a selector and properties in curly brackets

```
img .thumbnail {
    width: 50px;
    height: 100px;
    border: 1px solid #434343;
}

.contact-info {
    font-size: 10pt;
    color: #cccccc;
}

.project {
    background-color: gray;
    margin-left: 10px;
```

CSS Rules

Selector ⟶ h1 {

Property ⟶ color: red;

font-family: Arial;

}

Classes vs. ids

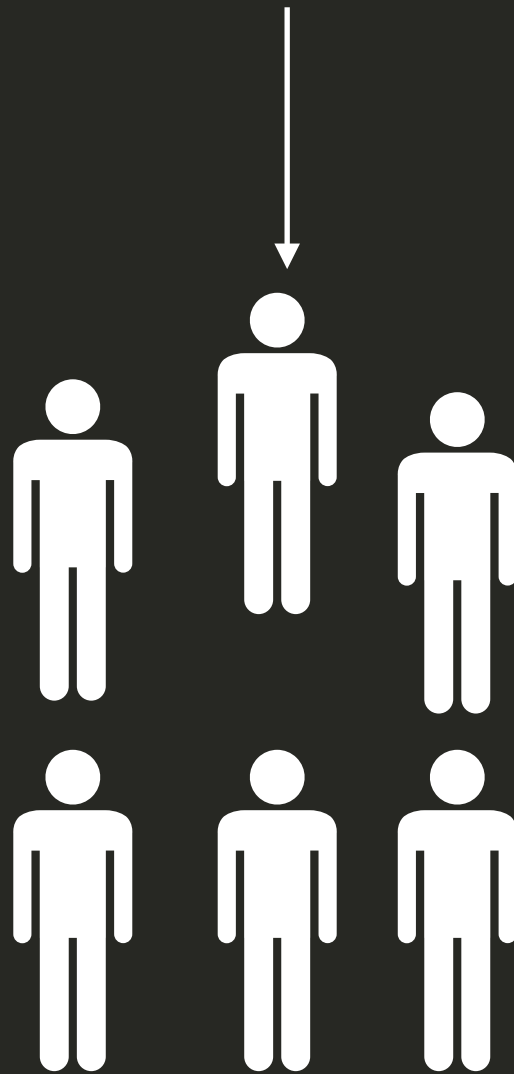- Class - a global attribute that can apply to multiple HTML elements `<h1 class="important"><p class="important">`

- id - a unique attribute that can only apply to a single HTML element `<h1 class="important" id="name">`

Question: Why do you think classes vs. ids are important?

Classes vs. ids

Make the entire class red

class="dsgn90"

Classes vs. Ids
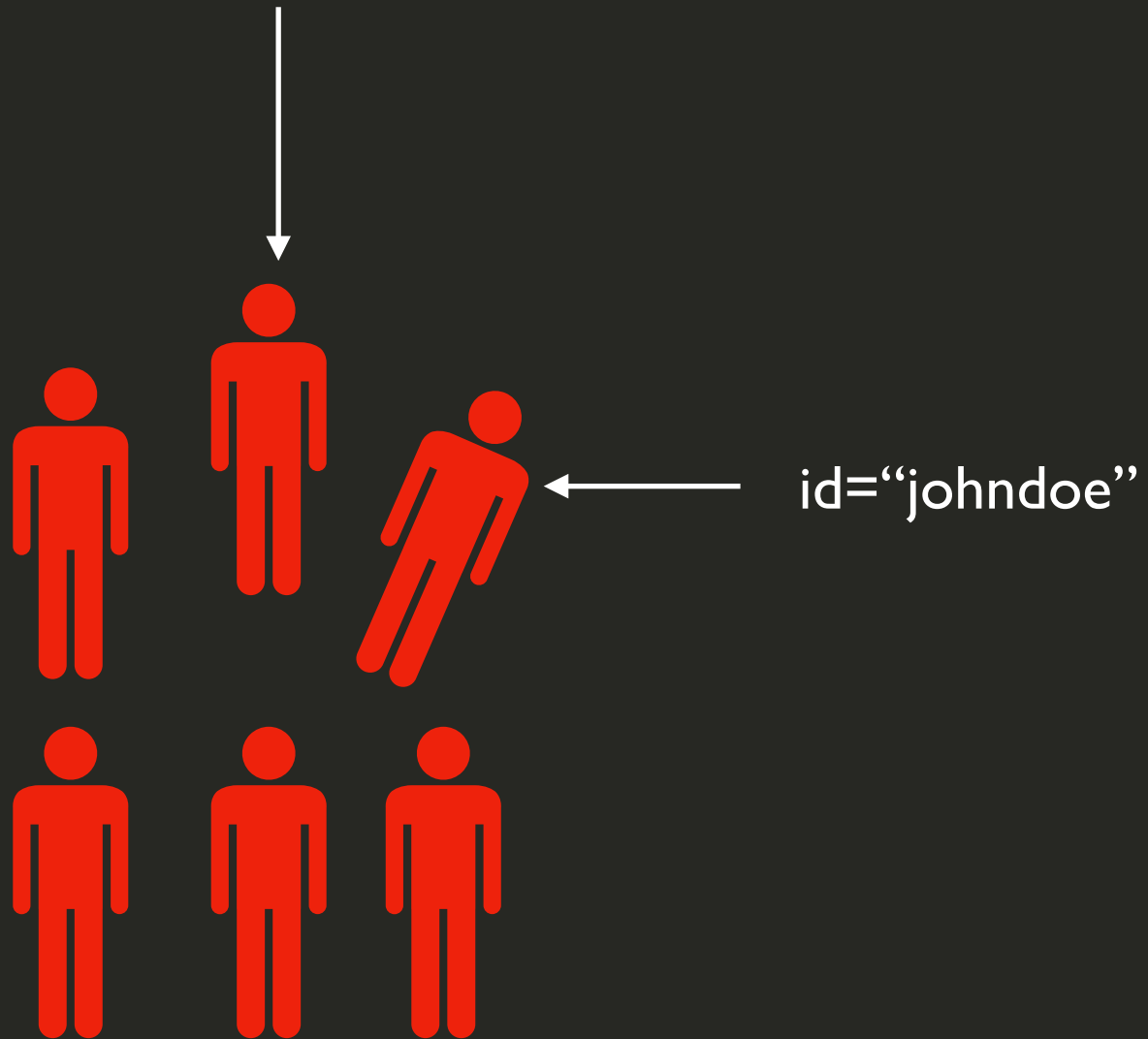

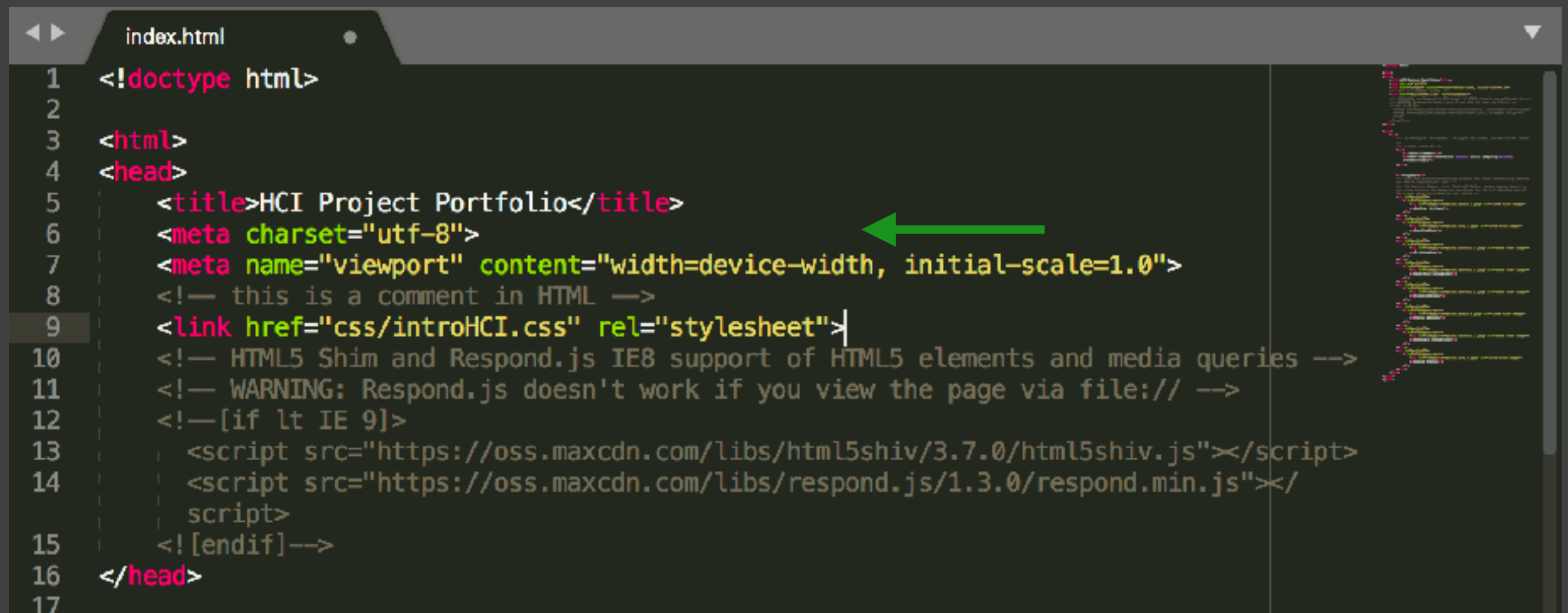Make the entire class red

class="dsgn90"

Classes vs. Ids

Classes vs. Ids

- In CSS, classes and ids are specified in different ways.
- Classes are specified by a . in front of the class name (i.e. .project)
- Ids are specified by a # in front of the id name (i.e. #project2)
- This is an important distinction when styling your webpage.

Open lab1/static/index.html in

- In order for your styling to apply to your HTML, link your CSS file in your HTML file:
  `<link href="css/main.css" rel="stylesheet">`
- CSS imports go inside the `<head>` element

```html
<!doctype html>

<html>
<head>
    <title>HCI Project Portfolio</title>
    <meta charset="utf-8">          ⟵
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <!-- this is a comment in HTML -->
    <link href="css/introHCI.css" rel="stylesheet">
    <!-- HTML5 Shim and Respond.js IE8 support of HTML5 elements and media queries -->
    <!-- WARNING: Respond.js doesn't work if you view the page via file:// -->
    <!--[if lt IE 9]>
      <script src="https://oss.maxcdn.com/libs/html5shiv/3.7.0/html5shiv.js"></script>
      <script src="https://oss.maxcdn.com/libs/respond.js/1.3.0/respond.min.js"></script>
    <![endif]-->
</head>
```

Did it work?

· Refresh index.html in your browser. Do you see the dotted border?

Tricia Ngoon

I am a fourth-year Ph.D student in Cognitive Science at UCSD. Etiam vitae tortor. Vestibulum suscipit nulla quis orci. Donec elit libero, sodales nec, volutpat a, suscipit non, turpis.Donec mi odio, faucibus at, scelerisque quis, convallis in, nisi. Cras dapibus. Pellentesque dapibus hendrerit tortor.

Add class="project" to the projects

- For every <div id="project1">, <div id="project2">, numbered 1 through 3, add class="project"
- Example: <div class="project" id="project1">

```
26  </div>
27
28
29  <h4>Projects</h4>
30  <!-- divs are invisible structuring elements that stack vertically by default. Use them to
    organize your code -->
31  <!-- img tags are images. Lorem Pixel will deliver random images; handy! -->
32  <!-- a are anchors, also known as hyperlinks. Use the href attribute to tell the browser where
    to go when the user clicks -->
33  <div id="project1">
34      <a href="project.html">
35          <img src="images/lorempixel.people.1.jpeg" alt="Lorem Pixel image">
36          <p>Waiting in Line</p>
37      </a>
38  </div>
39  <div id="project2">
40      <a href="project.html">
41          <img src="images/lorempixel.city.1.jpeg" alt="Lorem Pixel image">
```

56

Trying out CSS

- Task:
  - Add the projects class selector to css/main.css
  - Give it the property: margin-left: 20px;
  - Reload your index.html. What change did you notice?

Pushing your changes to Github

- In your terminal add, commit, and push your changes.
- After a couple minutes, visit your github.io address and see whether it worked

Assignment 1 goals

- Turn the portfolio page scaffold into a full web site and push it to GitHub pages
- Your webpage should contain:
  - A header with your name in it
  - An "About" paragraph with information about you
  - An image of yourself
  - A short description of your projects
  - A hyperlink to your Resume/CV in your About paragraph
- Stretch Goal: add more sections and styling

Once your site is up, submit Assignment 1 at

https://tinyurl.com/dsgn90-a1

Due by Aug 15 by 11:59pm

Today we learned…

- Using Github and git for source control
- Site structure through HTML and basic CSS
- HTML elements: divs, headers, paragraphs, images, hyperlinks
- CSS properties