GENERAL ASSEMBLY

JAVASCRIPT DEVELOPMENT

Sasha Vodnik, Instructor

THE COMMAND LINE

WEEKLY OVERVIEW

WEEK 1	Installfest / The Command Line
WEEK 2	Data Types & Loops / Conditionals & Functions
WEEK 3	(holiday) / Scope & Objects

LEARNING OBJECTIVES

At the end of this class, you will be able to

- Use the most common commands to navigate and modify files / directories via the terminal.
- Initialize a local Git repository and push/pull changes to a remote Git repository.
- Run basic JavaScript code on the command line using Node.

AGENDA

- JS and web technology
- The terminal
- Git and GitHub
- Command line JS

EXIT TICKET QUESTIONS

1. Once I complete this course, will it be relatively easy to learn backend JavaScript?
2. Should Homebrew actually be an application I can view in my Mac folder?
3. I just want to be sure that my computer environment is properly prepared with all the software before next class.
4. How to set up my personal and work github accounts separately on my laptop.
5. The GitHub setup was a little confusing in setting up the RSA key, can we go over?

Think about last class:

- We installed software from the command line by typing commands
- We also installed software by downloading an installer, double-clicking it, and following the prompts

ACTIVITY

EXERCISE

KEY OBJECTIVE

- Use the most common commands to navigate and modify files / directories via the terminal window.

TYPE OF EXERCISE

- Turn and Talk

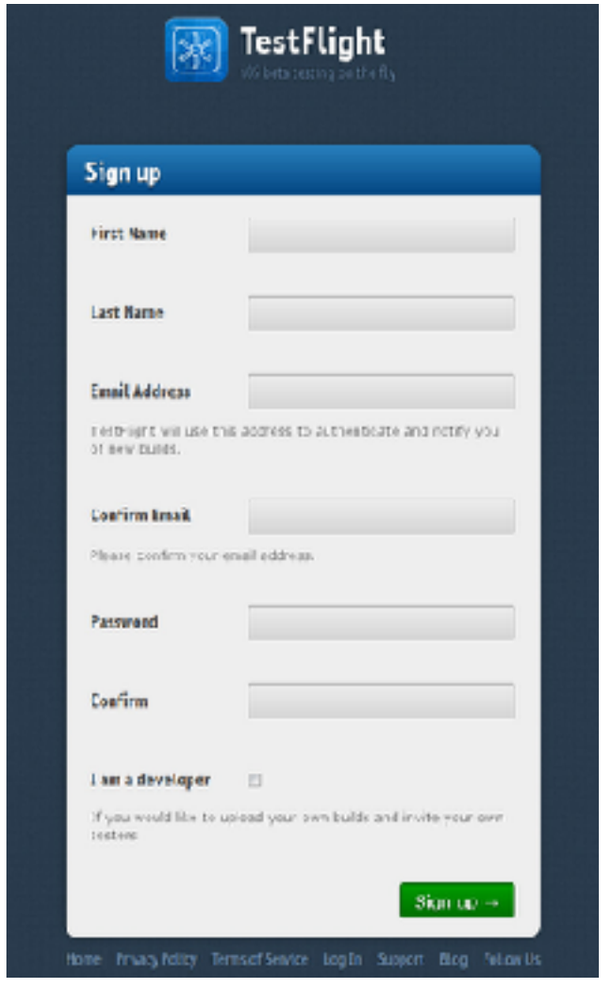
TIMING

2 min

1. List at least 2 advantages to using the command line.
2. List at least 2 disadvantages to using the command line.

JavaScript & Web Technology

WHAT CAN JAVASCRIPT DO?

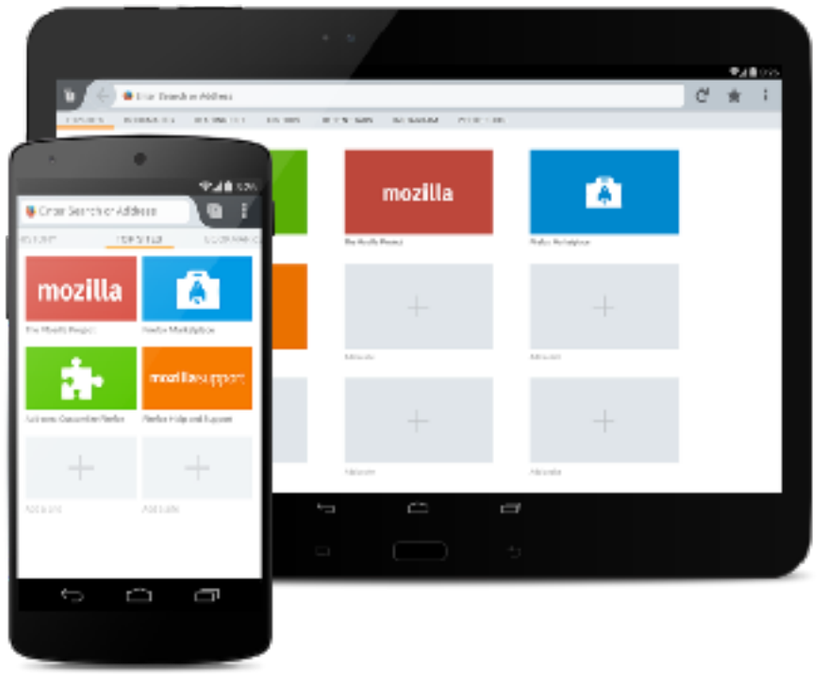


front end tasks (animations, buttons, forms)

JS

APIs, databases, back end tasks

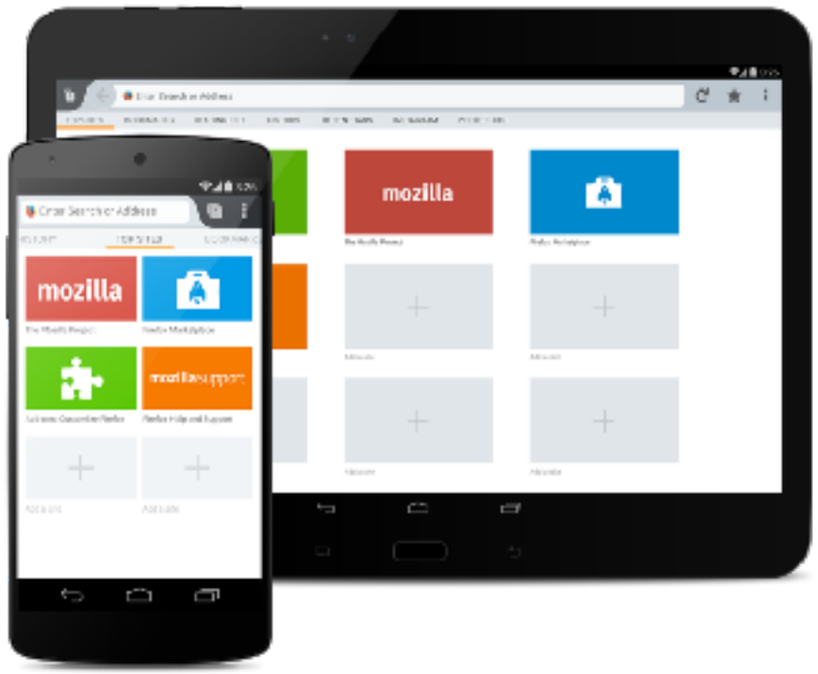
VERY FEW STEPS TO RUN



runs directly in browser within an HTML file

also runs in node.js

AND WORKS EVEN WHEN COMPUTERS ARE OFFLINE



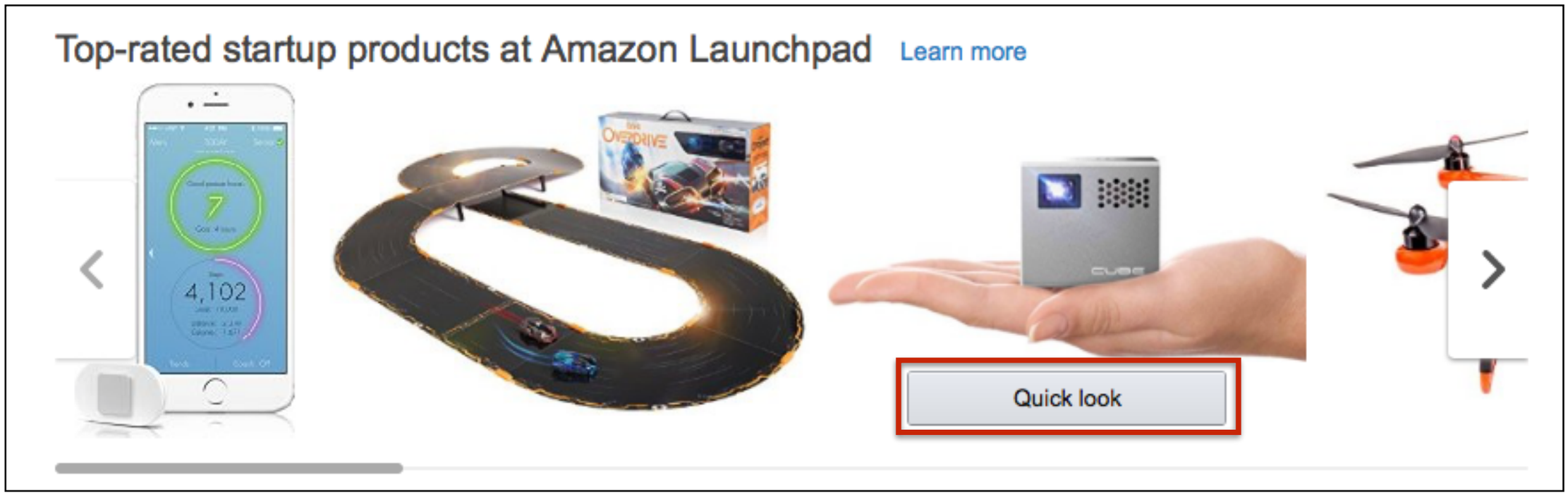
JS

runs directly in browser within an HTML file

also runs in node.js

node JS

HIGHLY RESPONSIVE INTERFACES

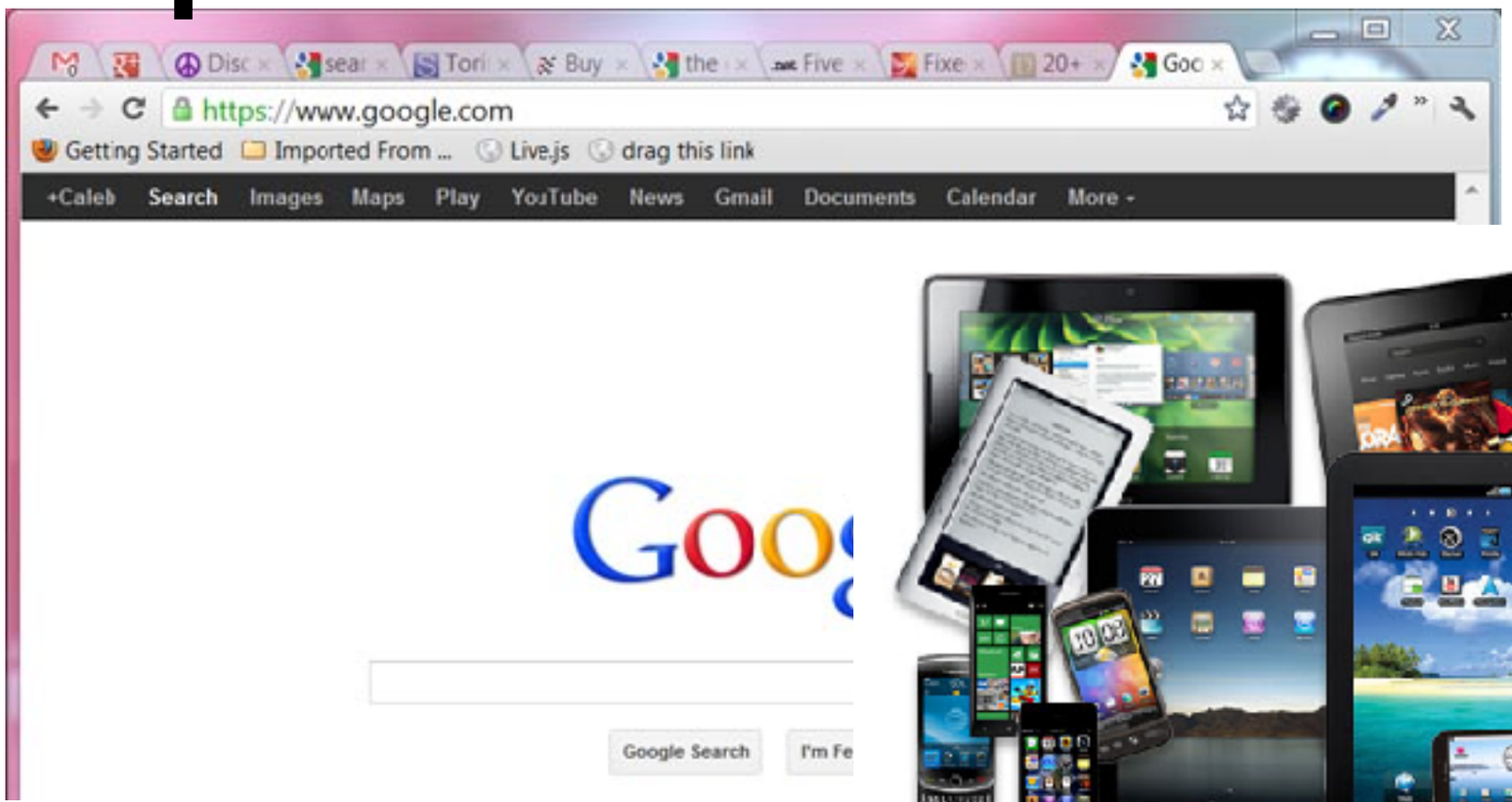


LOAD ADDITIONAL CONTENT WHEN USER NEEDS IT (AJAX)

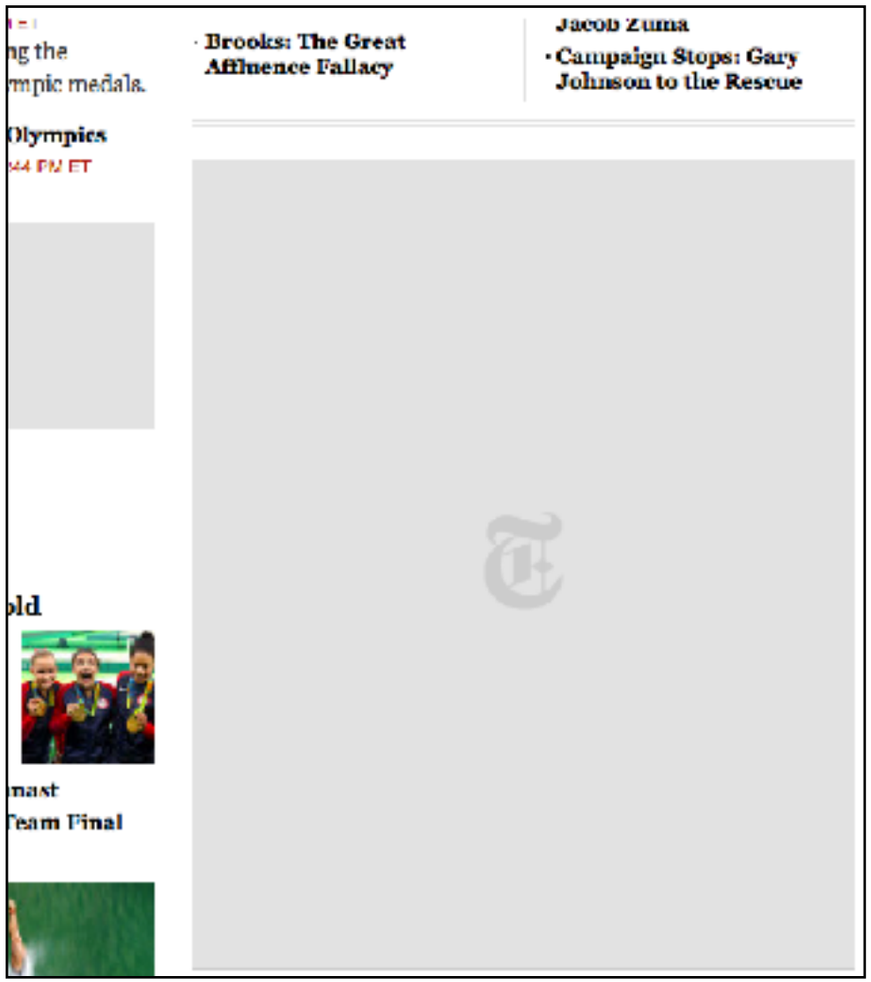
WHAT ELSE CAN JAVASCRIPT DO?

- Determine your browser functional limitations and react accordingly (progressive enhancement)
- Power website backends and physical devices (node.js)

DRAWBACK: The environment in which JavaScript operates is unknown



DRAWBACK: JavaScript can be disabled



Node.js

Node.js

- A definition (from Wikipedia):
  - In software development, Node.js is an open-source, cross-platform runtime environment for developing server-side Web applications.
- Enables JavaScript on the server (the backend)
- Written in C, C++, and JS (so, not a JS framework)
- Interprets JS using Chrome's V8 engine
- Module driven; see Node Package Manager (npm)
- All about non-blocking, asychronous input/output

Node.js

- We will not be using Node.js as a web server (backend) - see Firebase
- We will be taking advantage of Node's command line interface
- Allows us to run JavaScript from our terminal applications
- More at the end of class...

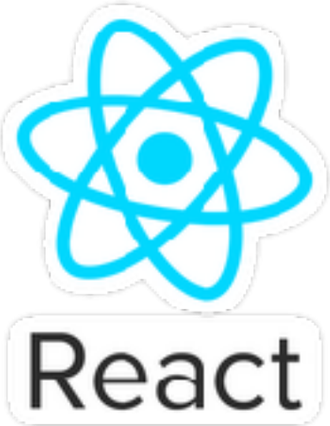
JavaScript Frameworks & Libraries

A Library

- Set of predefined functions that your code calls
- Each call performs work and returns a result (and control) to your code
- Specific, well-defined operations
- Example: jQuery

A Framework

- Opinionated architecture for building software
- Control-flow exists, you fill in with your code
- Calls your code; is always in control
- Examples: React, Angular, Vue, Ember



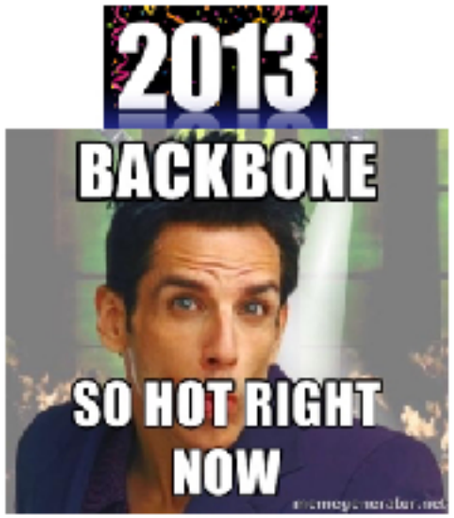
Libraries vs Frameworks

- The primary difference (source):
  - You call library
  - Framework calls you
- Please Note:
  - JSD focuses on the foundations of JavaScript as a programming language
  - We will be using the jQuery library
  - Opportunity towards class end for a framework intro


2007
GWT
SO HOT RIGHT NOW


2009
EXTJS
SO HOT RIGHT NOW


2012
KNOCKOUT
SO HOT RIGHT NOW

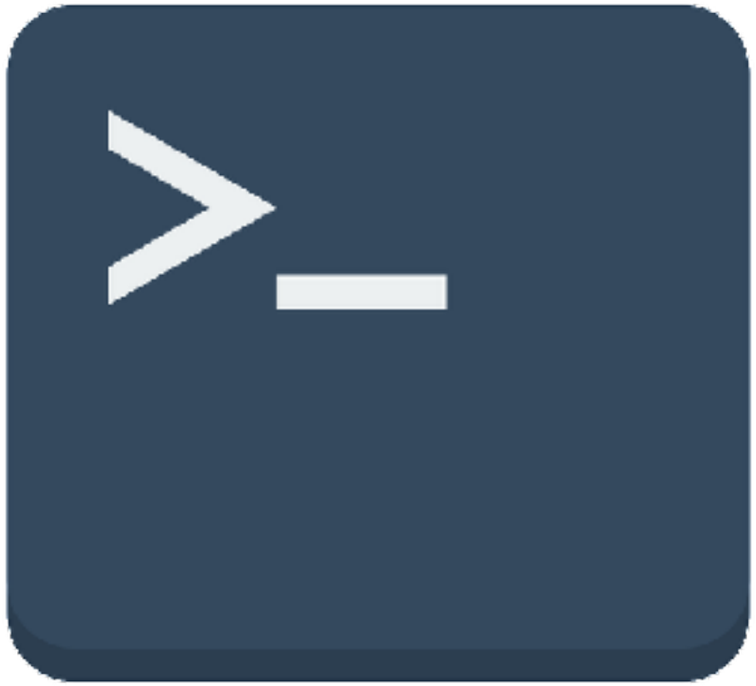

2013
BACKBONE
SO HOT RIGHT NOW


2014
ANGULAR
SO HOT RIGHT NOW


2015
REACT
SO HOT RIGHT NOW

The Terminal

INTRODUCTION TO THE TERMINAL



- Terminal allows you to interact with your computer faster
- Terminal === Command Line === Console

UNIX

UNIX

- Family of operating systems, including all Linux systems and OS X/macOS

SHELL

- A generic name for the primary program that runs inside a terminal

BASH

```
Sashas-MacBook-Pro:JS-SF-10 sasha$
```

- Bourne-Again Shell: a specific shell program

ANATOMY OF THE TERMINAL

```
Sashas-MacBook-Pro:JS-SF-10 sasha$
```

ANATOMY OF THE TERMINAL

Host (computer) name

```
Sashas-MacBook-Pro:JS-SF-10 sasha$
```

ANATOMY OF THE TERMINAL

Working directory (current folder)

```
Sashas-MacBook-Pro:JS-SF-10 sasha$
```

ANATOMY OF THE TERMINAL

Username

```
Sashas-MacBook-Pro:JS-SF-10 sasha$
```

ANATOMY OF THE TERMINAL

Bash prompt

```
Sashas-MacBook-Pro:JS-SF-10 sasha$
```

ANATOMY OF THE TERMINAL

Command (program)

```
Sashas-MacBook-Pro:JS-SF-10 sasha$ ls
```

ANATOMY OF THE TERMINAL

Argument (input)

```
Sashas-MacBook-Pro:JS-SF-10 sasha$ ls 00-installfest
```

ANATOMY OF THE TERMINAL

Option

```
Sashas-MacBook-Pro:JS-SF-10 sasha$ ls -a 00-installfest
```

ANATOMY OF THE TERMINAL

Output

```
Sashas-MacBook-Pro:JS-SF-10 sasha$ ls -a 00-installfest
.                .DS_Store        index.html       slides.md
..               img              install.md
Sashas-MacBook-Pro:JS-SF-10 sasha$
```

THE COMMAND LINE

Command line codealong

For Mac

Open the Terminal app (Applications > Utilities > Terminal)

For Windows

Open the Git BASH application

LAB — COMMAND LINE

EXERCISE

KEY OBJECTIVE

- Use the most common commands to navigate and modify files / directories via the terminal window.

TYPE OF EXERCISE

- Individual/Pairs

TIMING

10 min Follow the instructions posted on the class website to navigate and modify files and directories using the command line.

EXERCISE — COMMAND LINE

EXERCISE

KEY OBJECTIVE

- Use the most common commands to navigate and modify files / directories via the terminal window.

TYPE OF EXERCISE

- Whole class brainstorm

TIMING

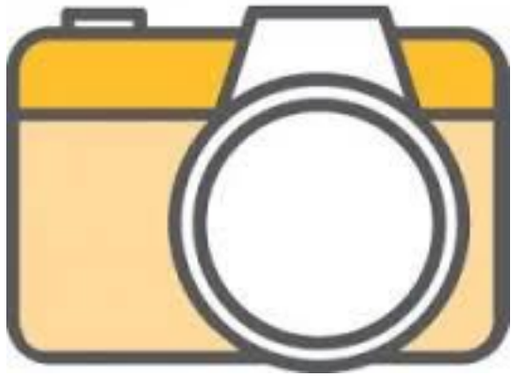
2 min

1. Name a command line command and explain what it does. Let's hear from everyone at least once!

Introduction to Git/GitHub

GIT

- A **version control** program that saves the state of your project's files and folders
- Basically, it takes a "snapshot" of what all your files look like at a moment and stores a reference to that "snapshot"



GITHUB IS A WEB APP/PLATFORM THAT

- **Platform** that makes it easy to manage git repositories.
- Similar to Dropbox or Google Drive, but for code.
- Stores a history of files and the changes that happen within each changed document.
- Hosts files on the cloud so you can share the finished product with other people.
- **Git** - the technology that Github is based on top of - was designed to allow for multiple engineers to work on the same project.

GitHub

Why use GitHub?

HISTORY

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

COLLABORATION

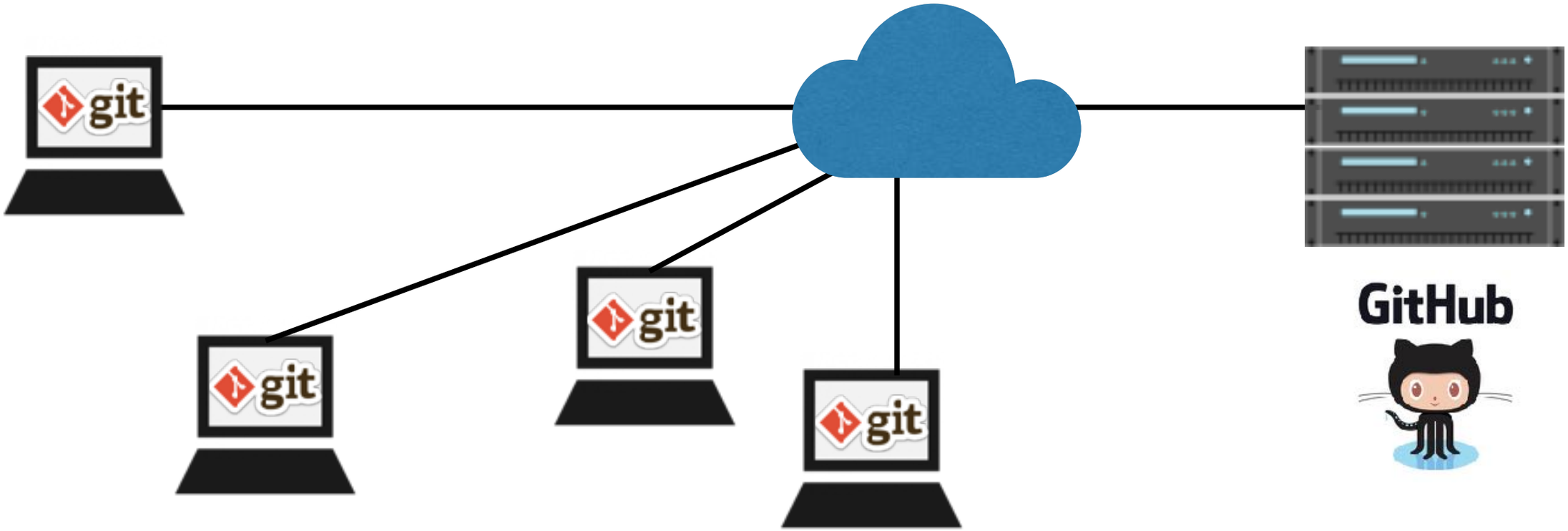
- Allows multiple developers to work on the same project. Much like Google Drive lets multiple people collaborate on the same document, GitHub allows this for code.
- You can see who worked on what.

FEEDBACK

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

Git vs GitHub

- **Git** is version control software
- **GitHub** is a website and platform for utilizing Git in a collaborative way



Git/GitHub Vocabulary

- Repository
- Clone
- Commit
- Push
- Pull

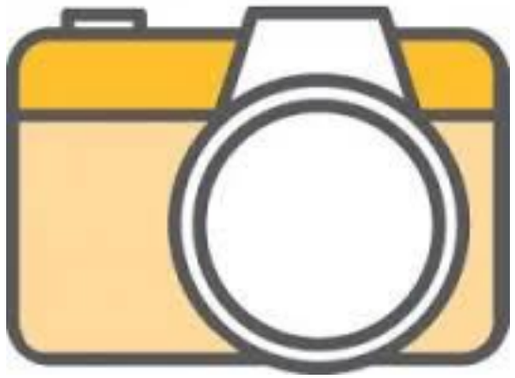
What is a repository (repo)?

- Basic element of GitHub
- Contains all of a project's files (all the code)
- One or more users can contribute to a single repository
- Repositories are either public or private
- By the end of class today, you will create your own repo

`clone`

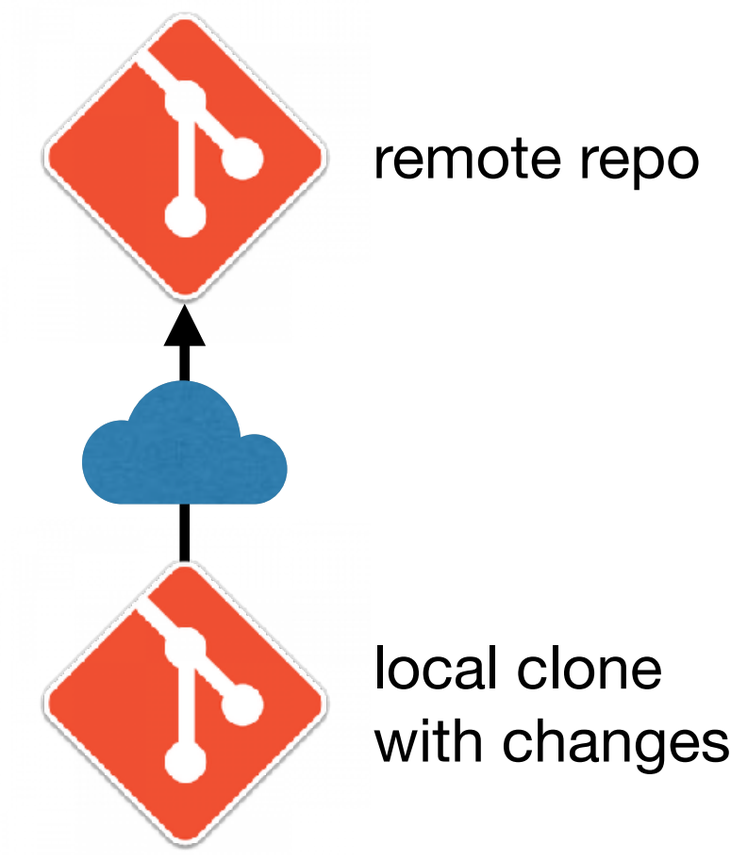
- Git command that copies/clones a **remote** repo to your machine
- This copy/clone is called a **local** repo
- Changes to the **local** repo will not affect the **remote**

`commit`



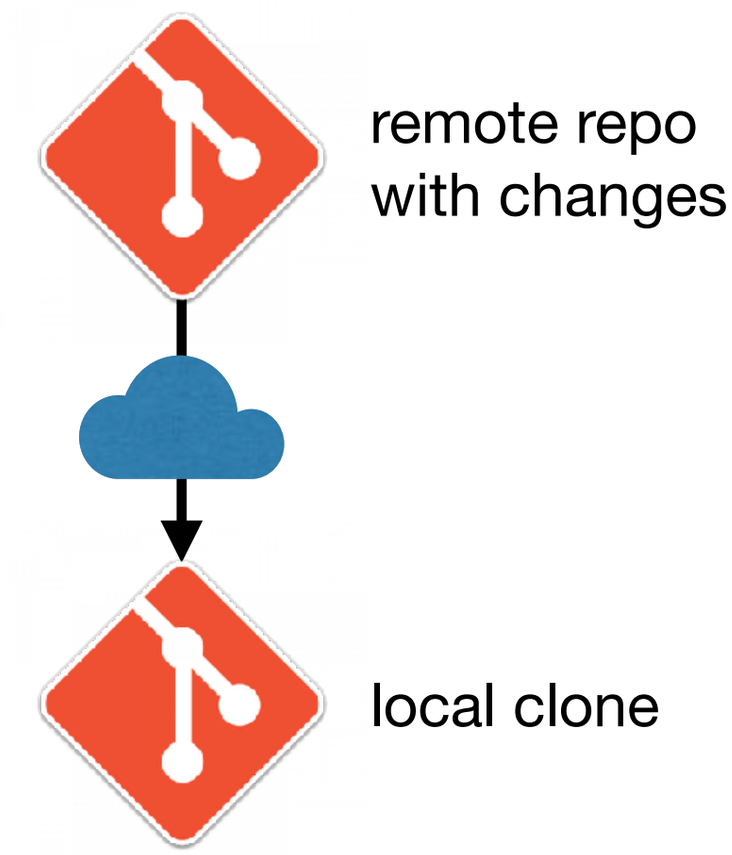
- Git command that creates a snapshot of changes to a repo
- Think of it as saving your changes with a timestamp
- Contains a message describing the changes made

push



- Git command that sends your commits (saved changes) to a **remote** repository
- Allows other developers to see your changes and copy ("pull") them to their own local repos

`pull`



- Git command that copies (pulls) changes by other developers from a remote repository to your local clone
- Allows you to see changes made by other developers and incorporate them into your local clone

How will we use GitHub in JSD10?

`JS-SF-10-resources`

- contains start and solution files
- you will pull changes at the start of each class

`JS-SF-10-homework`

- currently empty
- you will push your completed homework and receive feedback here

You will create your own additional repos for the 3 projects during this course.

GIT COMMANDS

THE COMMAND LINE

EXERCISE — GIT/GITHUB

EXERCISE

KEY OBJECTIVE

- Understand how to initialize a local Git repository and push/pull changes to a remote Git repository.

TYPE OF EXERCISE

- Pairs

TIMING

2 min

1. What command do you use to initialize a local Git repository? (Hint: Check the handout.) What does initializing do?
2. What command do you use to push changes to a remote Git repository? What does pushing do?
3. What command do you use to pull changes from a remote Git repository? What does pulling do?
4. BONUS: Draw a diagram illustrating all 3 commands

Intro to Node.js and command line JS

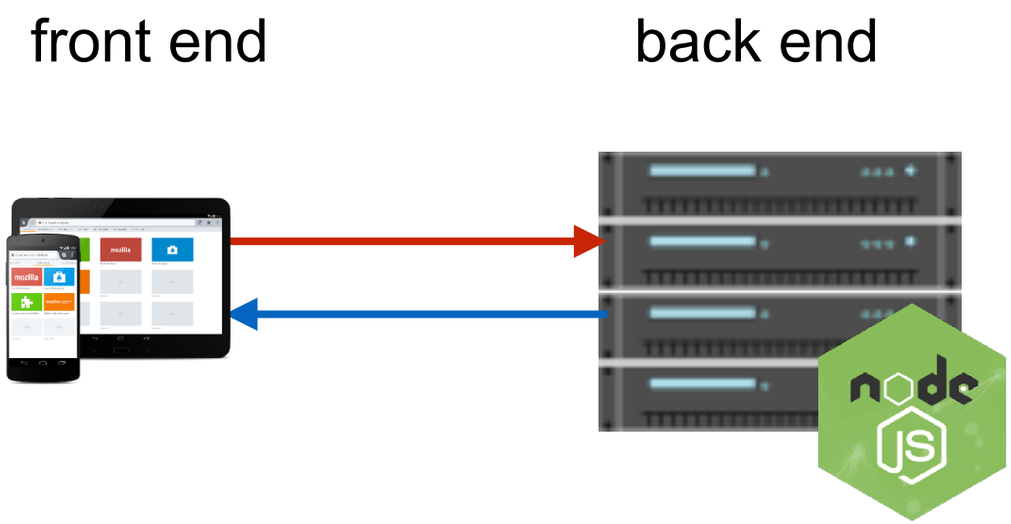

WENGER

How is Node different from JS in the browser?

- No browser-specific functionality
- Same JS engine as Chrome

What is Node good for?

- Creating a backend server for a web application
- Running a script to do data analysis
- File management
- Making command line programs



Ways to run commands in Node

Interactive command line

Your command
Node's response

```
> 5 + 2
< 7
```

Run a file

You

```
> script.js
```

Node loads the file script.js and executes its contents

Node

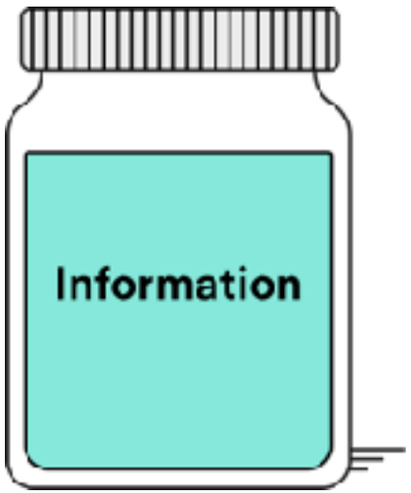
```
< 7
```

Executing JavaScript code

Let’s write some JavaScript!

Variables

- Containers that allow us to store values
- Let us tell our program to remember values for us to use later on
- The action of saving a value to a variable is called **assignment**



Declaring a variable

```
let age;
```

Assigning a value to a variable

```
age = 29;
```

Declaring and assigning in a single statement

```
let age = 29;
```

Printing things out for our own inspection

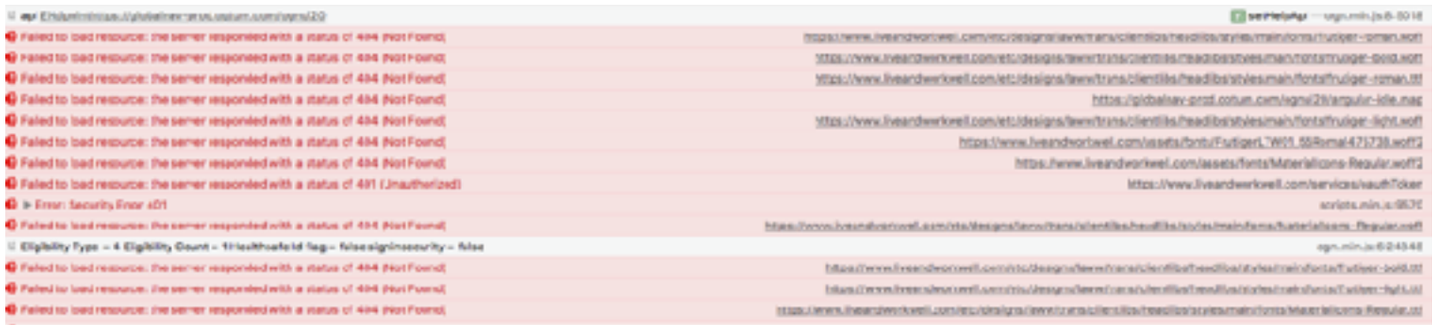
```
console.log("Hello!");
```

Printing a variable value out for our own inspection

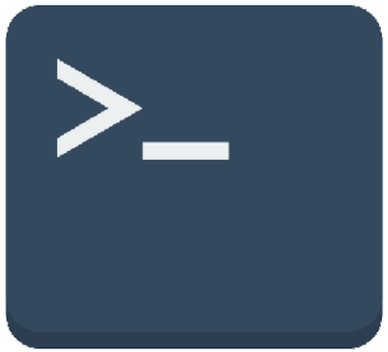
```
console.log(age);
```

When do you use console.log?

- When you are developing a program and need help figuring out what's going on (aka debugging)
- When you want to print things to the command line



browser developer tools



command line

THE COMMAND LINE

Exit the Node console

Node prompt

```
>
```

`control` + `C` twice

BASH prompt

```
$
```

EXERCISE — NODE

EXERCISE

KEY OBJECTIVE

- Run basic JavaScript code on the command line using Node.

TYPE OF EXERCISE

- Turn and talk

TIMING

2 min

1. What is Node?
2. What did we use it for today?
3. BONUS: How else can it be used?

Exit Tickets!

(Class #1)

LEARNING OBJECTIVES - REVIEW

- Use the most common commands to navigate and modify files / directories via the terminal.
- Initialize a local Git repository and push/pull changes to a remote Git repository.
- Run basic JavaScript code on the command line using Node.

Next class preview: Data Types

- Describe the concept of a "data type" and how it relates to variables.
- Declare, assign to, and manipulate data stored in a variable.
- Create arrays and access values in them.
- Iterate over and manipulate values in an array.

Q&A