WEEK 2

8.30.21

Agenda

THIS WEEK *(2)*

Discussion / Share Work

Modern Web Development
Clients & Servers
Stacks
Tools
Languages: HTML / CSS

Github, GIthub Pages

The Terminal

Assignment 2

NEXT WEEK *(3)*

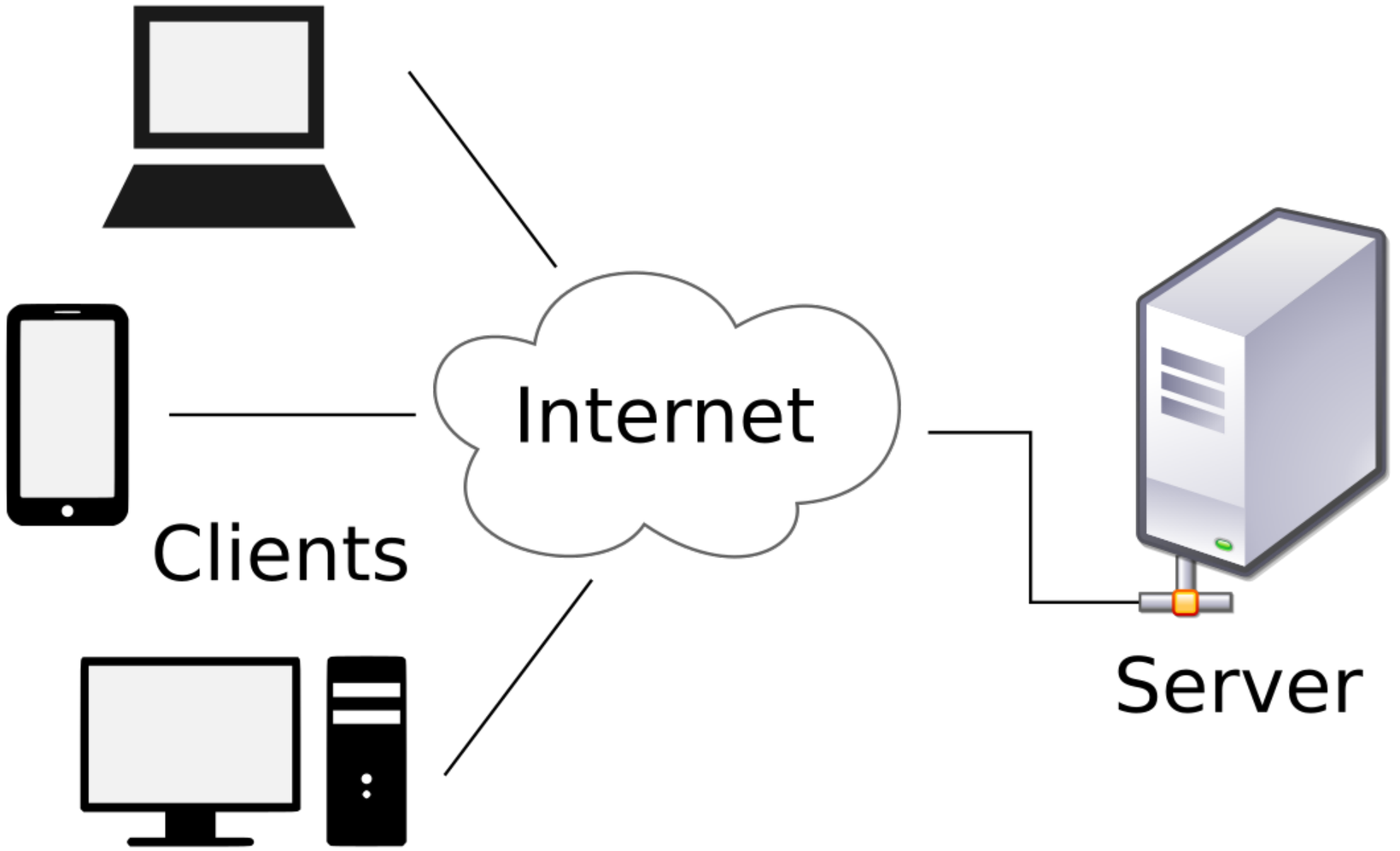
Labor Day
No Class

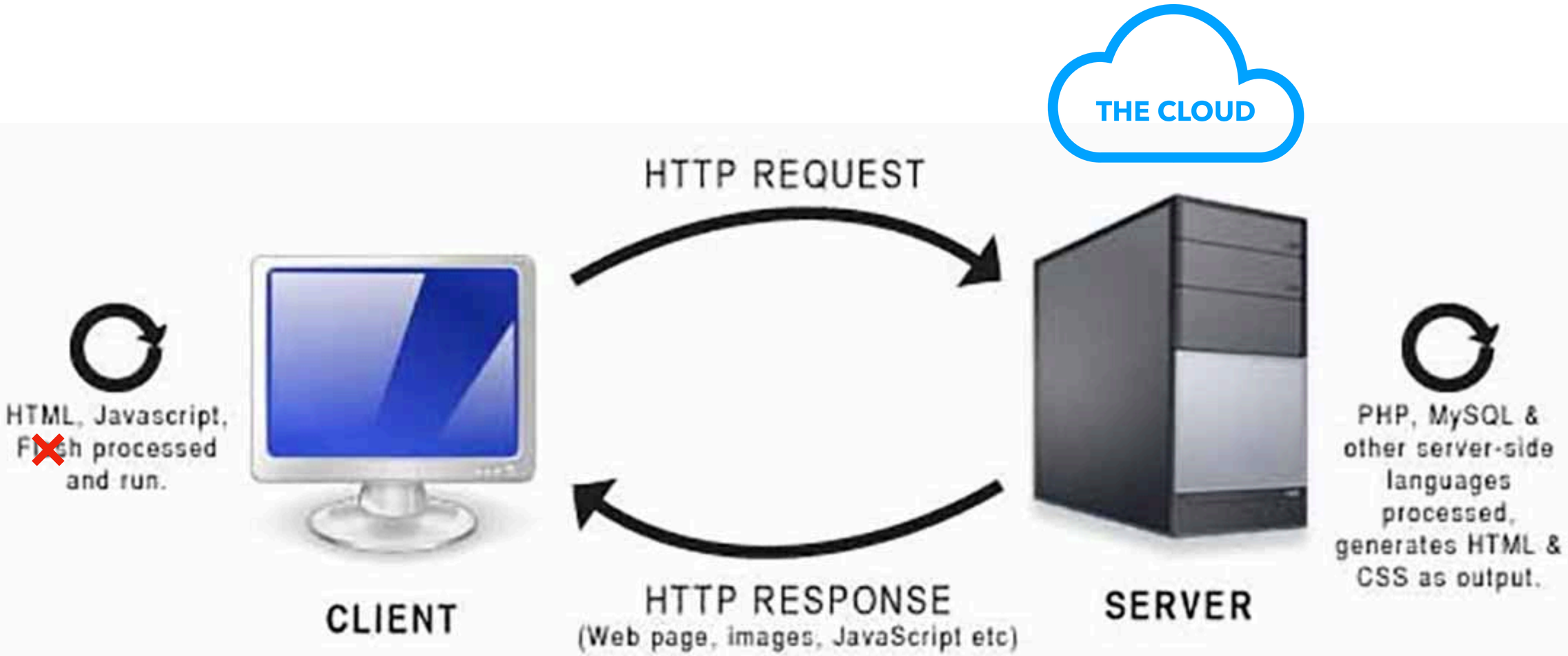
. . .but I recommend spending time working on Assignment 2 TBD.

Objectives

- Understand the basics of clients and servers and their role on the Internet
- Understand what tools and technologies developers generally use to do their job
- Understand what HTML and CSS are and how to build a simple webpage from scratch
- Understand how to create a github repo, clone a repo, and push changes to a repo
- Understand how to deploy (publish) a basic website using Github Pages (project site)

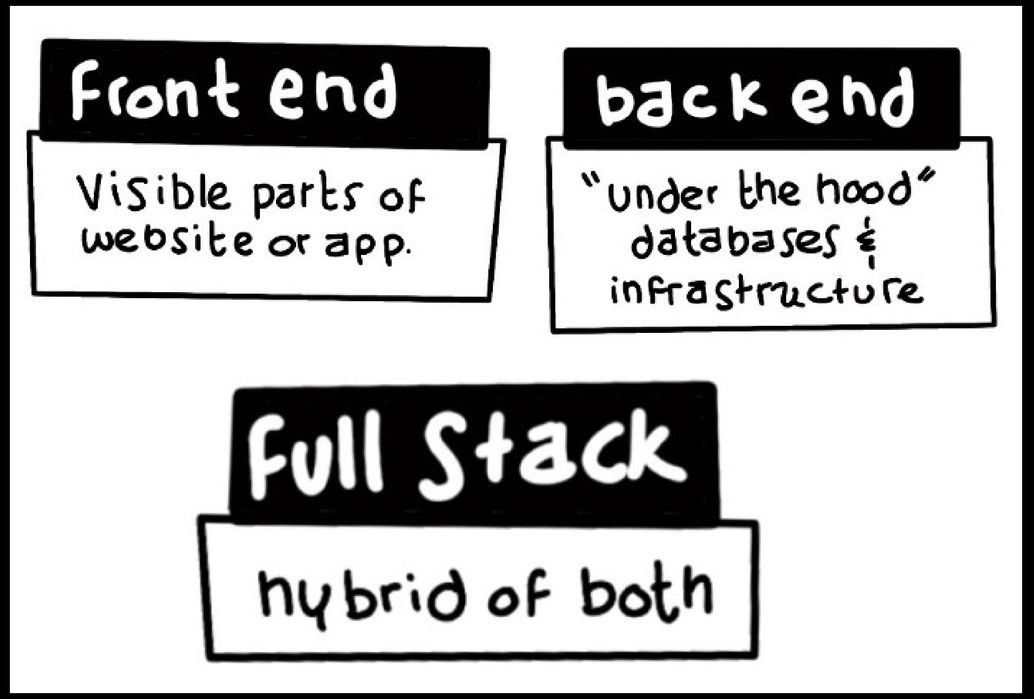
CLIENTS & SERVERS

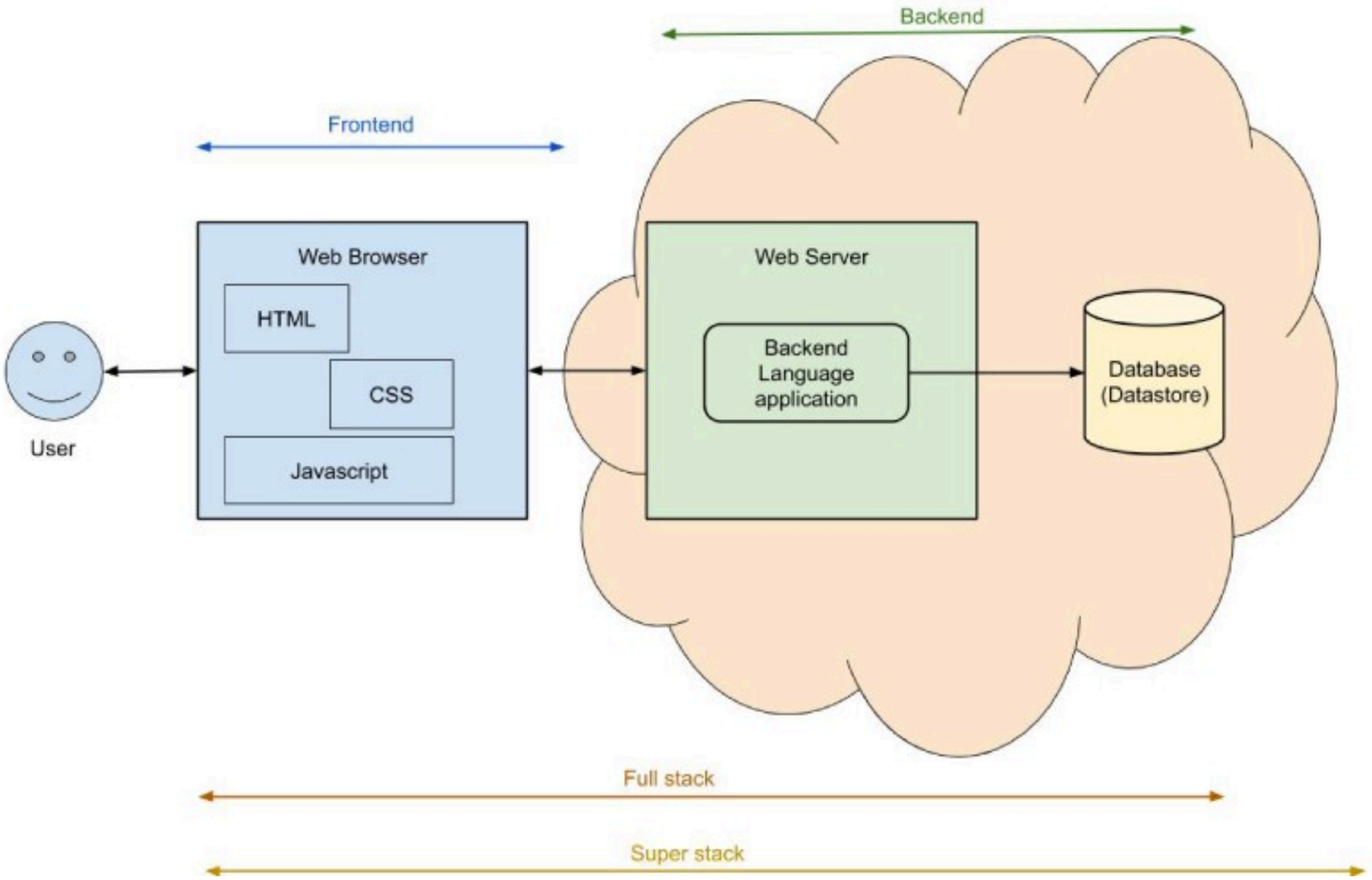

Internet
Clients
Server



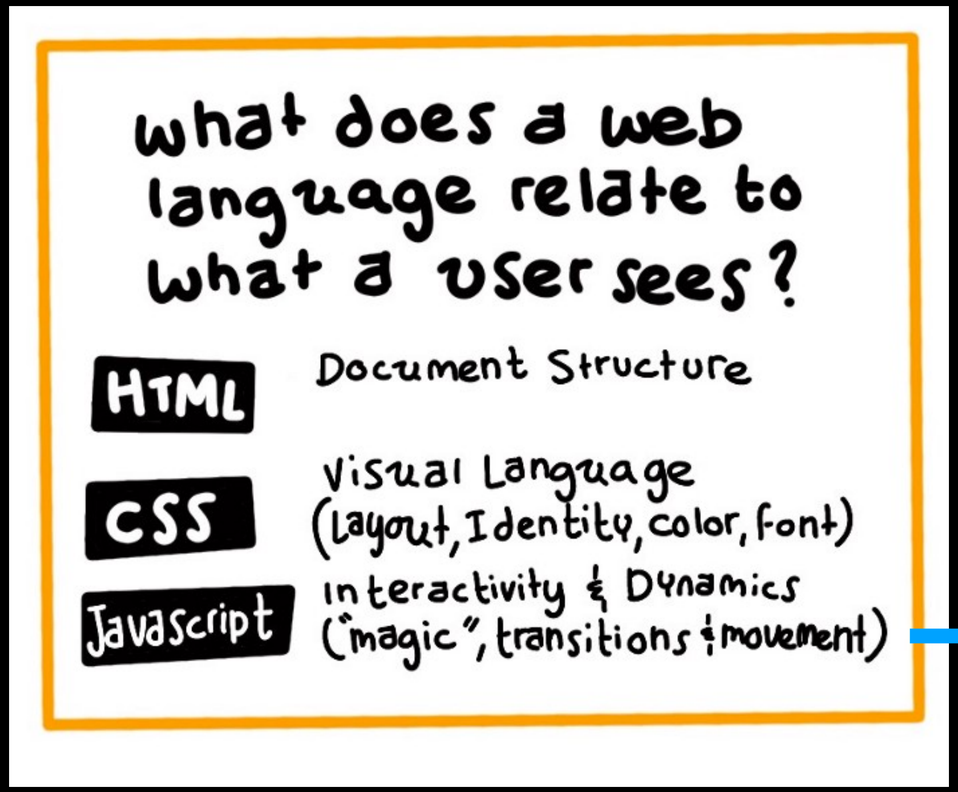
The cloud refers to software
and services that run on the Internet,
instead of locally on your computer.

STACKS


Front end
Visible parts of website or app.
back end
"under the hood" databases & infrastructure
Full Stack
hybrid of both


Backend
Frontend
Web Browser
HTML
CSS
Javascript
User
Web Server
Backend
Language
application
Database
(Datastore)
Full stack
Super stack

HTML & CSS



we will talk about javascript next week

HTML & CSS

HTML *(Document Structure)*

```
<html>
  <head>
  </head>
  <body>
    <h1>Hello World</h1>
    <p>This is my website!</p>
  </body>
</html>
```

CSS *(Look & Feel)*

```
...
<style>
  h1 {
    color: red;
  }
  p {
    color: #00FF00;
  }
</style>
...
```

index.html

The most basic webpage ever consists of a single index.html file with html, head, and body tags in it.

```
<html>
  <head>
  </head>
  <body>
  </body>
</html>
```

index.html

Something slightly more advanced will have more tags and may include CSS inside the style tag.

```
<html>
  <head>
  </head>
  <style>
    h1 {
      color: red;
    }
    p {
      color: #00FF00;
    }
  </style>
  <body>
    <h1>Hello World</h1>
    <p>This is my website!</p>
  </body>
</html>
```

Language Resources

- https://www.w3schools.com/

TOOLS

Tools

Tools we will setup:

- Text Editor: Visual Studio Code aka VSCode
- Local Development Server: Live Server Extension
- Version Control / Deployment: Github, Github Pages
  - Need to setup Github account
- Utilities: macOS terminal or Windows Git bash terminal

Terminal

macOS *(Terminal)*

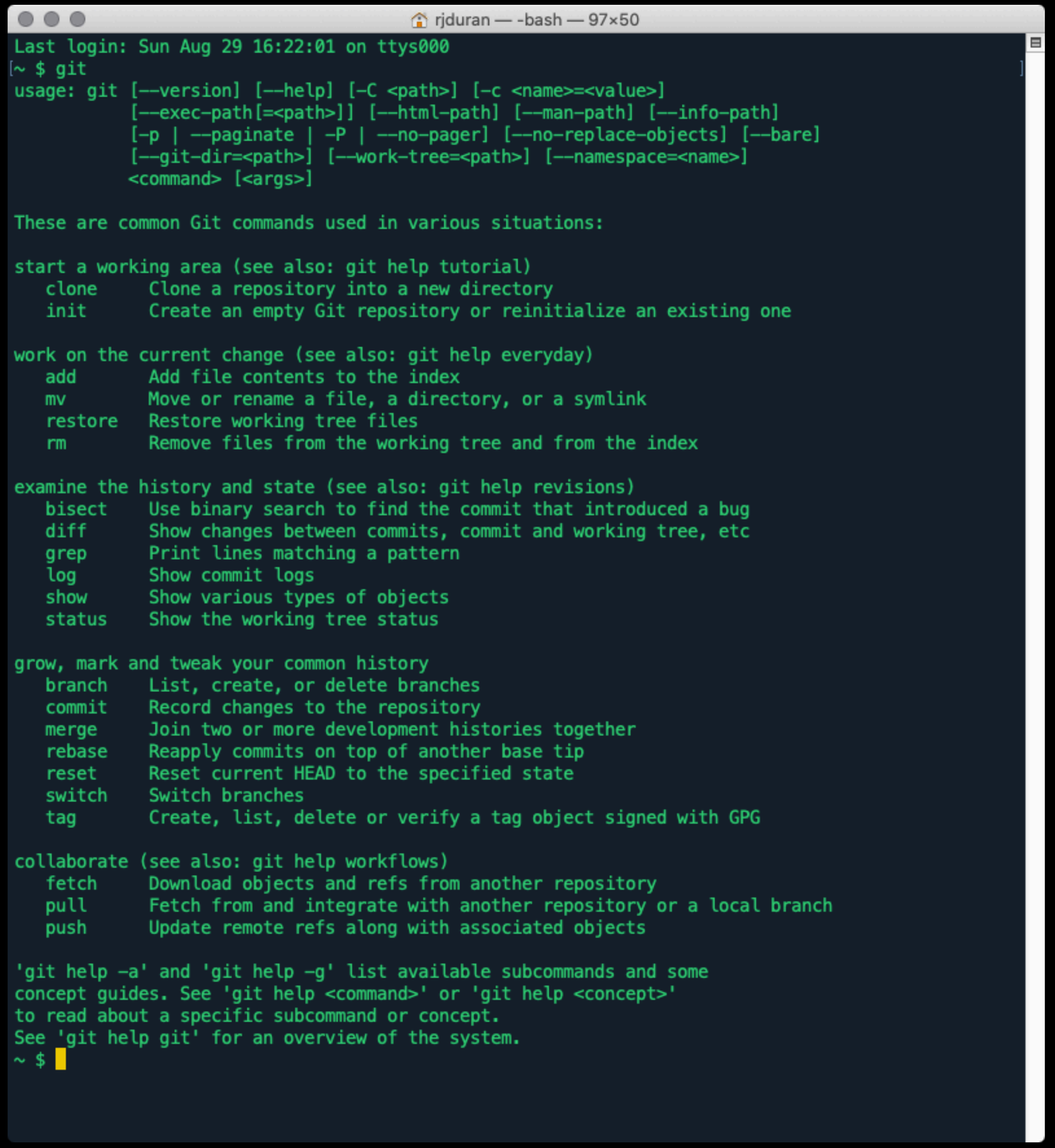
Applications > Utilities > Terminal.app

Install Git:
Type *git* into the terminal.
What happens? It may ask to install developer tools first.
Once installed, you should see a list of options for the command.

Windows *(Git bash)*

Download and install 64-bit Git for Windows

Git bash is a terminal emulator for Windows that offers a command line interface similar to macOS and Linux.


rjduran — -bash — 97×50
Last login: Sun Aug 29 16:22:01 on ttys000
~ $ git
usage: git [--version] [--help] [-C <path>] [-c <name>=<value>]
[--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
[-p | --paginate | -P | --no-pager] [--no-replace-objects] [--bare]
[--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
<command> [<args>]
These are common Git commands used in various situations:
start a working area (see also: git help tutorial)
clone Clone a repository into a new directory
init Create an empty Git repository or reinitialize an existing one
work on the current change (see also: git help everyday)
add Add file contents to the index
mv Move or rename a file, a directory, or a symlink
restore Restore working tree files
rm Remove files from the working tree and from the index
examine the history and state (see also: git help revisions)
bisect Use binary search to find the commit that introduced a bug
diff Show changes between commits, commit and working tree, etc
grep Print lines matching a pattern
log Show commit logs
show Show various types of objects
status Show the working tree status
grow, mark and tweak your common history
branch List, create, or delete branches
commit Record changes to the repository
merge Join two or more development histories together
rebase Reapply commits on top of another base tip
reset Reset current HEAD to the specified state
switch Switch branches
tag Create, list, delete or verify a tag object signed with GPG
collaborate (see also: git help workflows)
fetch Download objects and refs from another repository
pull Fetch from and integrate with another repository or a local branch
push Update remote refs along with associated objects
'git help -a' and 'git help -g' list available subcommands and some
concept guides. See 'git help <command>' or 'git help <concept>'
to read about a specific subcommand or concept.
See 'git help git' for an overview of the system.
~ $

BREAK

GITHUB

How to Use Github

1. Signup for a Github account. Choose a username that you want to type ALL THE TIME.
I recommend something short and sweet like your name or initials or hacker username.

2. Create a repository

3. Clone a repository *(git clone)*

4. Made some edits to your source files using VSCode

5. Stage the changes *(git add)*

6. Commit the changes *(git commit -m "message")*

7. Push the changes *(git push)*

8. Repeat the process starting at #4

DEPLOY

Deploy (Publish)

Follow the process outlined at https://pages.github.com/. Be sure to choose the **“Project Site”** workflow and **“Start from scratch”**.

Objectives

- Understand the basics of clients and servers and their role on the Internet
- Understand what tools and technologies developers generally use to do their job
- Understand what HTML and CSS are and how to build a simple webpage from scratch
- Understand how to create a github repo, clone a repo, and push changes to a repo
- Understand how to deploy (publish) a basic website using Github Pages (project site)

QUESTIONS?