

Fall 2017 — ar589.github.io

Week 2

Interactive Design

Intro to Terminal

Let's Git Going

Layout with HTML
& CSS

Intro to **Terminal.app**

OMG. Why?

- Command Line Interfaces are way faster than Graphical User Interfaces.
- We want our computers to act like a web server while we're working.
- It comes with a great Git client.

We only need to know a little bit.

- Move around the file system.
- Make files and directories.
- Save our work with Git.

**Open up
Terminal.app!**

The Prompt

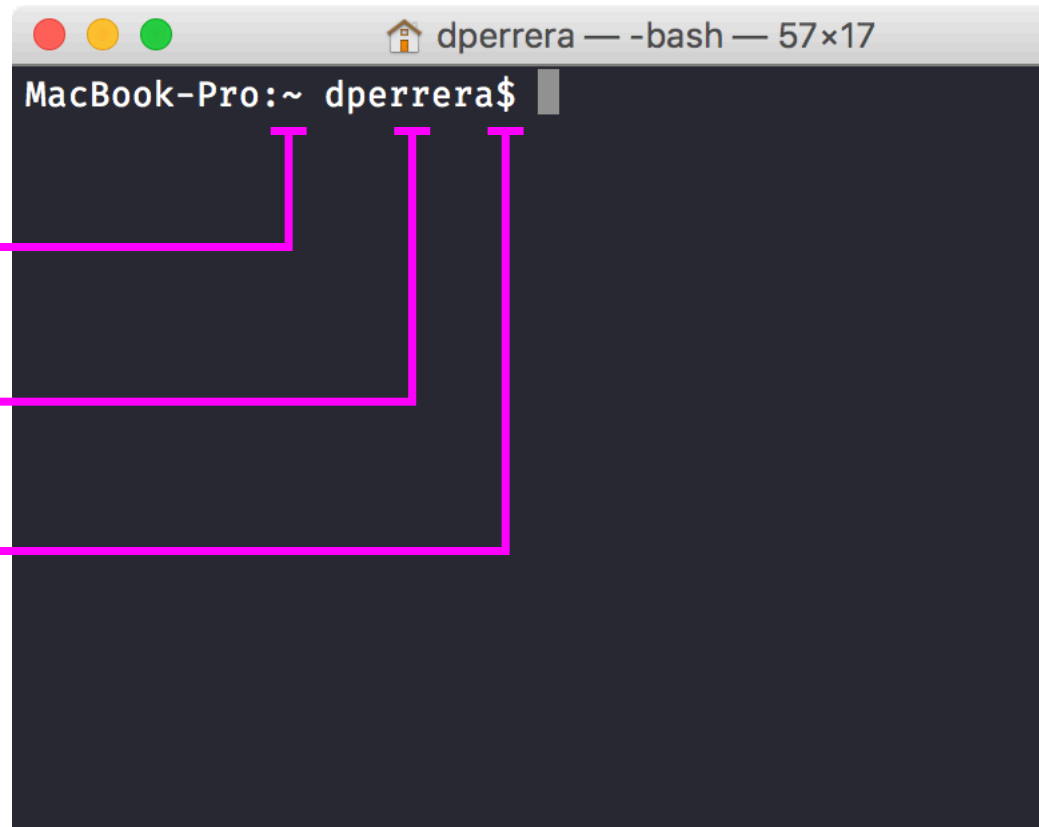
By default the command line gives you information about who you are and where you are.

Computer

Current Directory

User

End of the Prompt



```
MacBook-Pro:~ dperrera$
```

The terminal window shows the prompt 'MacBook-Pro:~ dperrera\$'. Pink lines connect the labels on the left to the corresponding parts of the prompt: 'Computer' points to 'MacBook-Pro', 'Current Directory' points to '~', 'User' points to 'dperrera', and 'End of the Prompt' points to '\$'.

The Anatomy of a Command

Every command starts with the program and can be followed by “arguments” and “flags.”

```
$ program arguments -f
```

Install Oh My Zsh.
<http://ohmyz.sh>

Attractive tools are nicer to use.

Moving Around

List Stuff

List the contents of a directory.

```
$ ls
```

Change Directory

.. means the parent directory.

~ is shorthand for your home directory.

```
$ cd Desktop
```

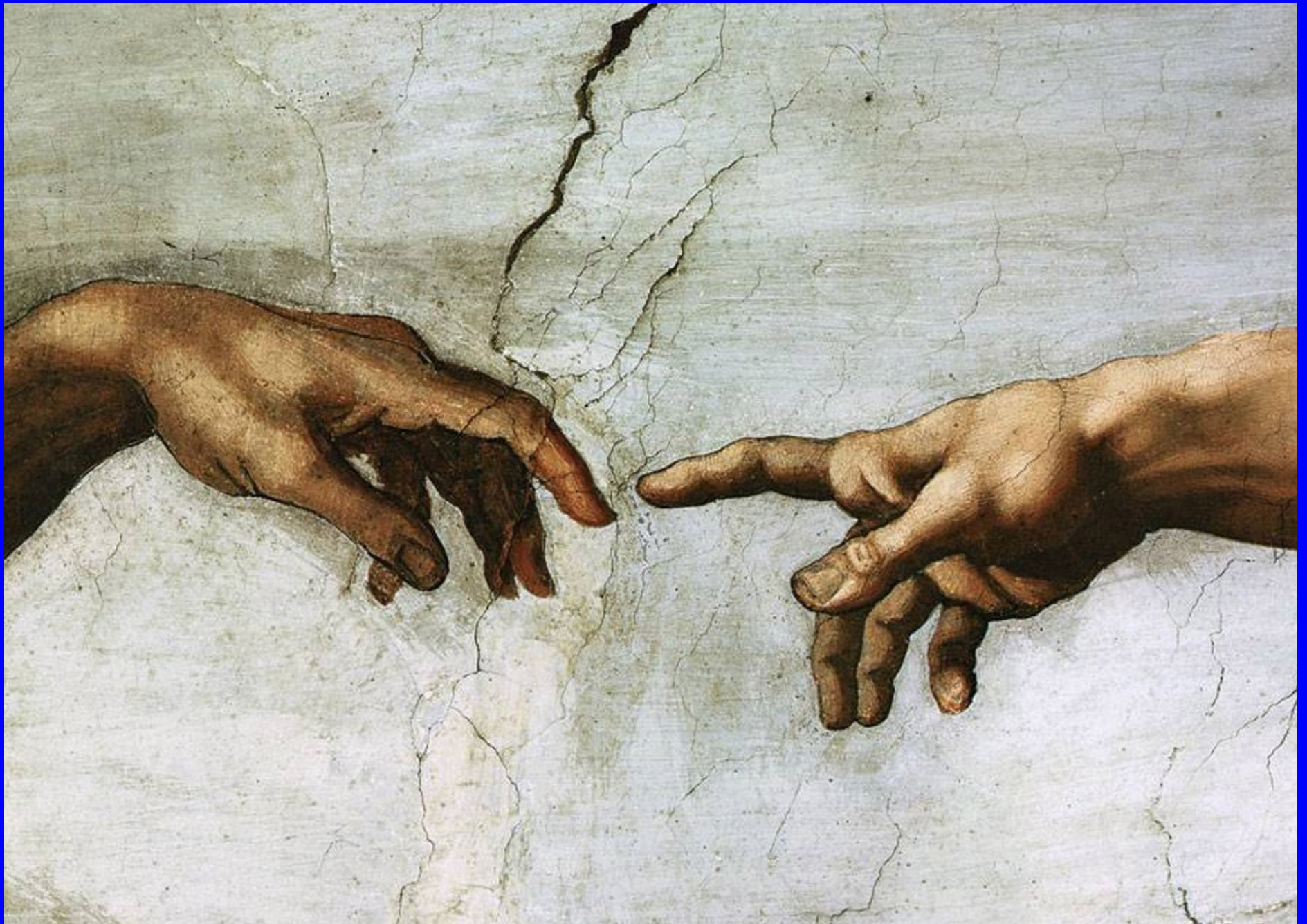
```
$ cd ..
```

```
$ cd ~
```

Making Files & Directories

Create a File

```
$ touch index.html
```



Make a Directory

```
$ mkdir sites
```

Start a Local Development Server

Start a Server

Tell your computer to serve any directory at `http://localhost:8000`. Stop the server by hitting Control-C.

```
$ python -m SimpleHTTPServer
```

Let's Git Going

What is Git?

- A distributed **version control system** that keeps track of your files.
- We can only interface with Git via the command line.

How it Works

- You tell Git to keep track of your files.
- When you do, your folder becomes a **Git repository**.
- You can tell a folder is a Git repo if it contains a **.git folder** (which you won't touch).

Turn a Folder into a Git Repo

```
$ git init
```

The Git Workflow

- **Stage** your changes.
- **Commit** your changes.
- **Push** your changes to your remote repo.

The Commit Dance!

Stage everything that changed

```
$ git add --all
```

Commit your changes

```
$ git commit --message "Added index.html"
```

Send commits to GitHub

```
$ git push
```

**Go to GitHub.com
in Chrome.**

Challenge 1

Review

Layout with HTML & CSS

HTML Elements: div & span

Non-semantic elements.

<div> is a block element. is an inline element.

```
<div>
  <h1>Dog Care and Maintenance</h1>
  <p>
    The <span>ultimate</span>
    guide to cuddling doggos.
  </p>
</div>
```

The HTML Class Attribute

Any HTML element can have the class attribute.

```
<div class="header">
  <h1>Dog Care and Maintenance</h1>
  <p>
    The <span class="highlight">ultimate
    </span> guide to cuddling doggos.
  </p>
</div>
```

The CSS Class Selector

```
.header {  
    background: black;  
    color: white;  
    padding: 50px;  
}  
.highlight {  
    background: yellow;  
    color: black;  
}
```

Let's give it a try!

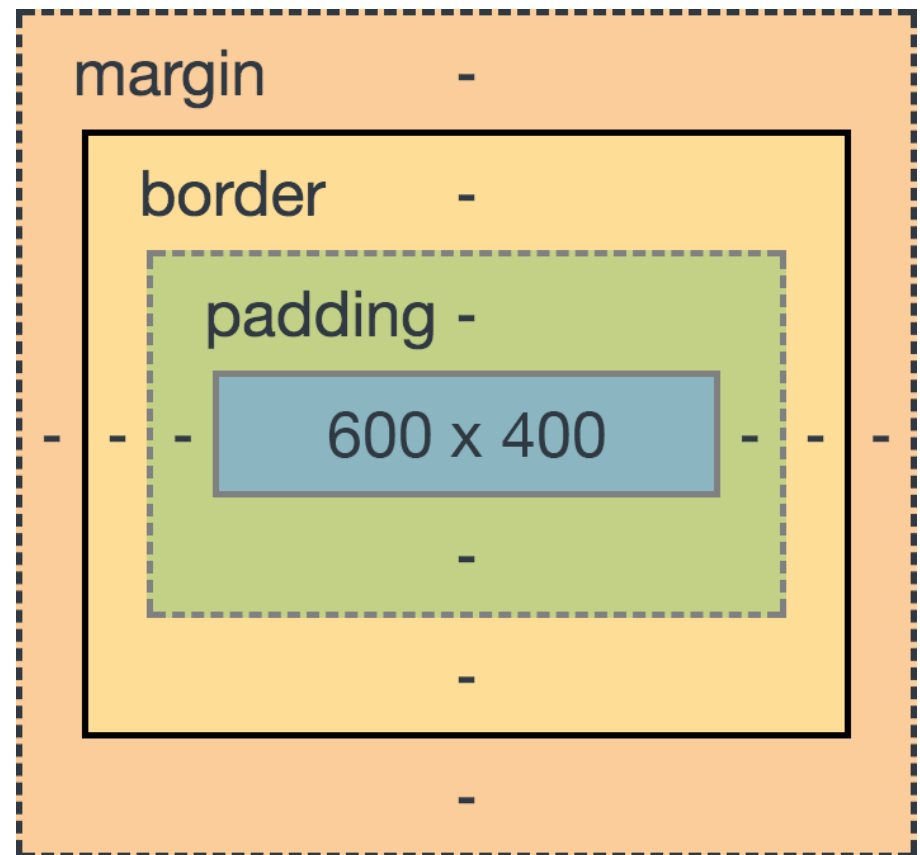
Open up Terminal, Atom, and Chrome.

The CSS Box Model

Every element on the screen is rendered as a box.

The **Box Model** is how browsers figure out the size of each element.

The way it measures space is a little weird by default.



The Box-Sizing Property

Makes the Box Model behave how we expect it to.

```
Box width measured as  
600px :(  
.header {  
  padding: 50px;  
  width: 500px;  
}
```

```
Box width measured as  
500px :)  
.header {  
  box-sizing: border-box;  
  padding: 50px;  
  width: 500px;  
}
```

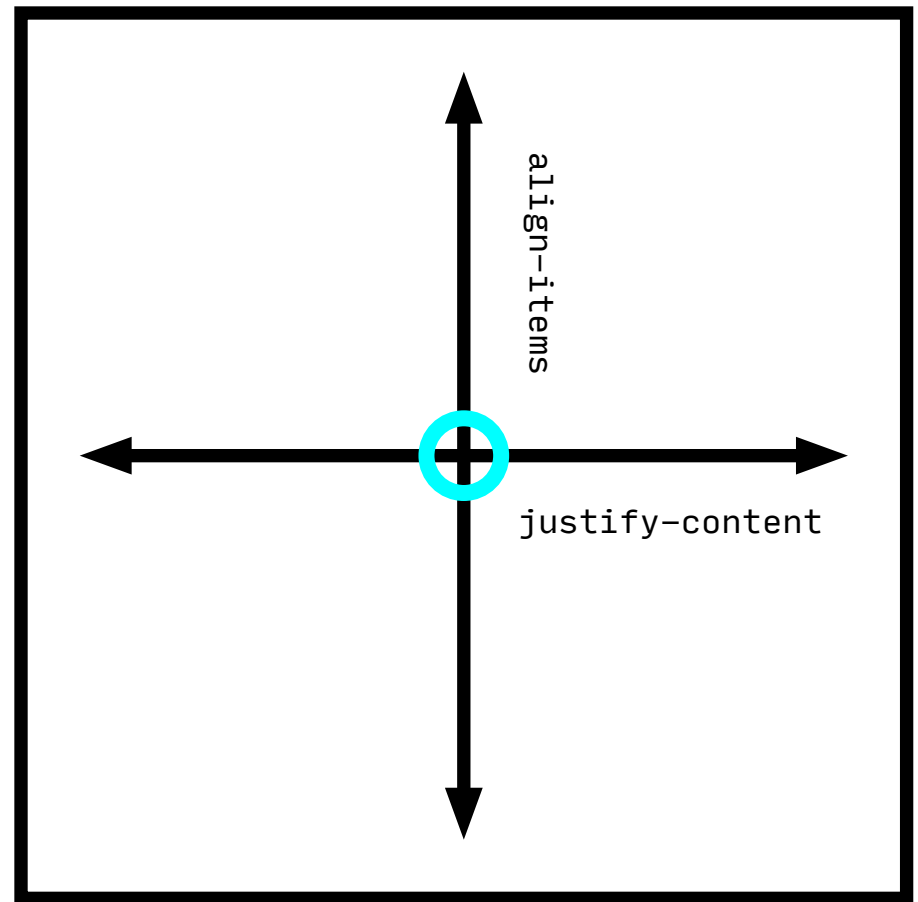

The Display Property

```
.myElement {  
  display: none;  
  display: block;  
  display: inline;  
  display: inline-block;  
  display: flex;  
}
```

Flexbox

Gives an element layout powers over it's direct children.

```
.myElement {  
  display: flex;  
  align-items: center;  
  justify-items: center;  
}
```



Common Flexbox Properties

Some properties only work on the parent element and some only work on child elements.

Parent Properties

```
.parentElement {  
  display: flex;  
  flex-direction: row;  
  flex-wrap: wrap;  
  align-items: center;  
  justify-content: center;  
}
```

Child Properties

```
.childElement {  
  flex-basis: 100px;  
  flex-grow: 1;  
  order: 2;  
}
```

Challenge 2

Improve your Recipe

Add divs and classes to your HTML and use CSS to create a grid layout, preferably with Flexbox.

Assignment 1

Art-Directed Article

Find a longform article and create a design using HTML and CSS to enhance the viewer's understanding of the content.

Due October 6

Resources

<https://try.github.io>

<http://cssreference.io/flexbox>