



**1) HAVE SLACK OPEN**

**2) PARTY**

---

**FRONT-END WEB DEVELOPMENT**

---

**SNACKS & DESIGN**

**TODAY!**

**ANNA MATRAS & JIM HOWES**

**(GOOGLE SHEET IS PINNED IN SLACK)**

---

**FEWD**

---

**Q & A**

*“Github. How’d it go?”*

*“Making any css for the Navigation structure”*

*“Can you animate background images with CSS?”*

*“Can you set classes to an <img> or does it need to sit inside of a wrapper?”*

*“What is normalize.css exactly?”*



---

## RESET.CSS

---

You can use a **reset** file to give you a blank slate for css styles.

1. [Normalize.css](#) makes browsers render all elements more consistently and in line with modern standards. It precisely targets only the styles that need normalizing
2. The common [reset from MeyerWeb](#) will take away all the styles for every HTML tag.

To add to your project:

1. Include the stylesheet (either `normalize.css` or `reset.css`) in your css directory
2. Add a `<link>` to the stylesheet in the head of your HTML. You'll want to include it **above** any other stylesheets so that your styles will be able to override the defaults.

*“How do older browser versions interpret the HTML5 semantic tags (<nav>, <article>... etc.)?”*

*<https://modernizr.com/>*

*(Do a little reading up on this now, we'll dig in more later...)*

*“Can we do a refresher on the position absolute and position relative at the start of the next class, please?”*

---

**FORM BASICS**

---

# ADVANCED CSS POSITIONING

---

## STATIC POSITIONING

---

- This is the normal flow of the document, the **default**
- Elements render in order, as they appear in the document flow.

```
.my-class {  
  position: static;  
}
```

---

## RELATIVE POSITIONING

---

- Relative positioning moves an element *relative to where it would have been in normal flow*.
- For example, "left: 20px" adds 20px to an element's **left** position
- **Creates a coordinate system for child elements.**

```
.my-class {  
  position: relative;  
  top: 20px;  
  left: 30%;  
}
```

---

## ABSOLUTE POSITIONING

---

- When the *position* property is given a value of *absolute*, an element is taken out of the normal flow of the document.
- This element no longer affects the position of other elements on the page (they act like it's not there).
- You can add the *right*, *top*, *left* and *bottom* properties to specify where the element should appear relative to its first positioned (not static) ancestor element

```
.my-class {  
  position: absolute;  
  top: 0;  
  left: 500px;  
}
```

---

## FIXED POSITIONING

---

- When the *position* property is given a value of *fixed*, the element is positioned in relation to *the browser window*
- When the user scrolls down the page, it stays in the same place.
- You can add the *right*, *top*, *left* and *bottom* properties to specify where the element should appear in relation to the browser window.

```
.my-class {  
  position: fixed;  
  top: 0;  
  left: 500px;  
}
```



---

## OVERLAPPING ELEMENTS — Z-INDEX

---

- When using relative, fixed or absolute positioning, elements can overlap.
- When elements overlap, the elements that appear later in the HTML code sit on top of those that appear earlier in the page.
- If you want to control which elements are layered on top of each other, you can use the z-index property.
- This property takes a number — the higher the number the closer that element is to the front.
- Similar to 'bring to front' and 'send to back' in programs like *Adobe Illustrator*.

```
.my-class {  
  z-index: 10;  
}
```

---

## WANT TO LEARN MORE?

---

Resources for more info/examples:

- ▶ A List Apart: [CSS Positioning 101](#)

*“Clears are something I need more work on”*

## PARENTS OF FLOATED ELEMENTS

- ▶ If a containing element **only contains floated elements**, some browsers will treat it as if it is zero pixels tall.

### PROBLEM:

1	2	3	4
5	6	7	8

*Collapsed parent!*

### SOLUTION:

1	2	3	4	
5	6	7	8	

#### PT. 1 — ADD CSS CLASS:

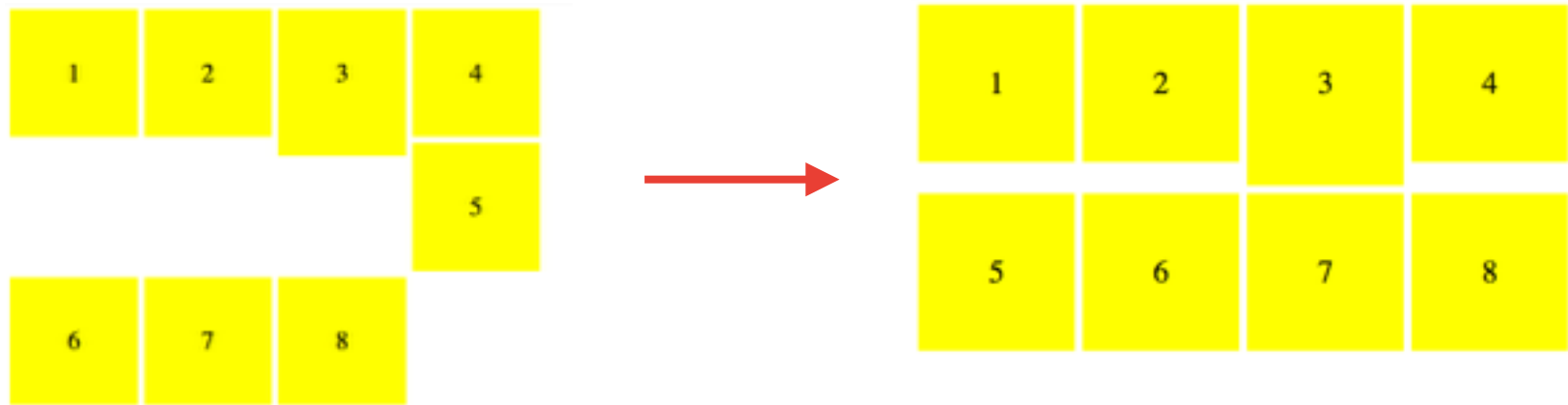
```
.clearfix:after {  
  content: "";  
  display: table;  
  clear: both;  
}
```

#### PT. 2 — ADD CLASS TO HTML:

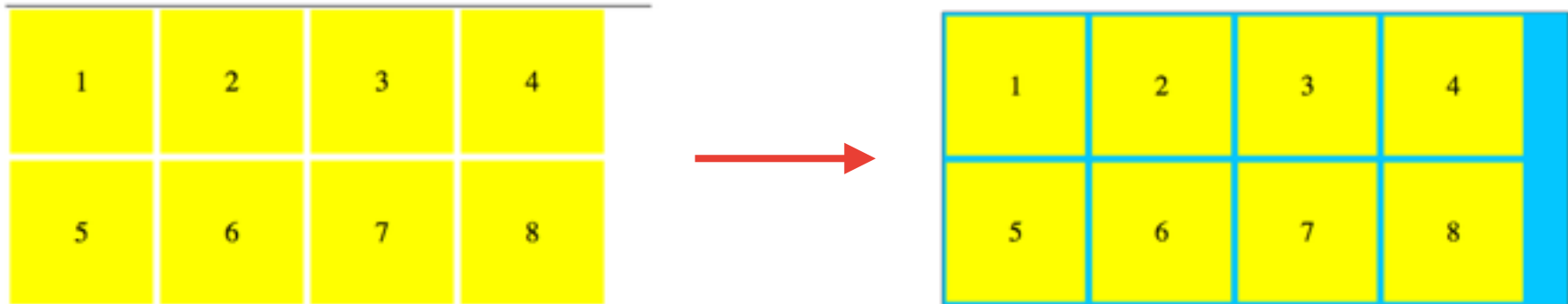
```
<div class="clearfix">  
  <p>1</p> <!-- float: left -->  
  <p>2</p> <!-- float: left -->  
  <p>3</p> <!-- float: left -->  
</div>
```

# CONFUSING NAMES — KEEPING THINGS STRAIGHT

**CLEAR: BOTH;**  
*Make sure an element starts on a new line*



**CLEARFIX:**  
*Fixes collapsed parent*



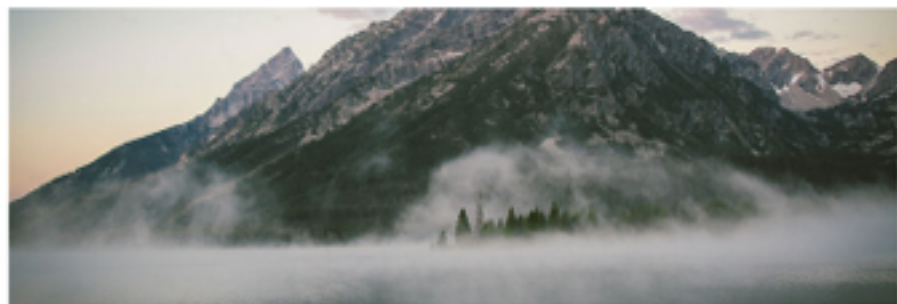
*“Could you walk through part of a lab on the screen?  
Would love to see how you  
would go about creating something! “*

## On the Road The Adventure of a Lifetime

[Blog](#)[Events](#)[Snapshots](#)[Store](#)[Contact](#)[About](#)

11/12/15

### Camping by Loon Lake



Cray culpa stumptown, flexitarian ex Odd Future do fugiat Wes Anderson proident 3 wolf moon officia bitters small batch. Et consequat do, nulla you probably haven't heard of them High Life scenester. Vinyl fugiat High Life, kogi do VHS in aliqua lo-fi leggings gentrify Neutra tumblr. Odio vegan PBR, Marfa forage blue bottle try-hard readymade meggings retro delectus Echo Park. Ugh consectetur farm-to-table forage, laboris blue bottle McSweeney's tattooed excepteur yr. Magna ut Schlitz flexitarian, vinyl craft beer proident yr forage 8-bit ethical sustainable placeat retro. Officia pickled beard, adipisicing gluten-free sint craft beer quis thundercats id 3 wolf moon fashion axe.

[Continues...](#)

#### About Us

On the Road is u salvia, fixie mumblecore ex aesthetic qui minim blog cliché. Retro disrupt keytar PBR, delectus consectetur flexitarian fingerstache selfies nostrud Schlitz. Tempor Wes Anderson banh mi bicycle rights. Eu occaecat Williamsburg yr letterpress, biodiesel plaid tote bag cliché messenger bag lomo bespoke sapiente next level.

[More...](#)

Ad

11/11/15

### A Morning Hike in Big Sur



#### Popular Posts

- [Breakfast in the Bay](#)
- [Morning Crescent](#)
- [Above the Clouds](#)
- [Camper update - Kitchen Rehaul](#)

# COMING UP...

~~MARCH 3: LAYOUT LAB (W/ ADVANCED CSS ~ RESETS)~~

**MARCH 8: INTRO TO PROGRAMMING (JS)**

**MARCH 10: PROGRAMMING CONT. (JS)**



# INTRO TO PROGRAMMING

*Eric Boyer*

---

**FEWD**

---

# REVIEW

---

## REVIEW

---

1. ["Lines" \(borders\) between sections](#)
2. [Background-image](#)
3. Hover states
4. Why use anchor instead of button elements?
5. Shadow on bottom of buttons
6. Text-transform
7. Pixels/percentages/responsive
8. Reuse things like box-shadow

---

## REVIEW — MULTIPLE CLASSES ON ONE ELEMENT

---

An element can have multiple classes. Multiple classes are separated by a space in the HTML.

```
<section class="banner clearfix"></section>
```

---

# PROCESS

---

---

## How to draw an Owl.

---

*"A fun and creative guide for beginners"*

---



Fig 1. Draw two circles



Fig 2. Draw the rest of the damn Owl

---

---

## PROCESS

---

1. Write your HTML (use [HTML5 Flowchart](#) for guidance in picking elements)
2. Get things into place. (Add floats, clear: both, get columns set up, etc.)
3. Add base styles. (Font-family/color for the body, remove text-decoration under anchors and add a base anchor color, remove bullets for list items, etc.)
4. Work through the page section by section and start "filling in the details." Resist the urge to be a perfectionist at this point.
5. Polish things up! Compare the design with your page. Are you using the right fonts? Colors? Is any of the spacing off?

# PROCESS — STEP 2 (LAYOUT STYLES)

<div>DevelopersDesignersHow it WorksOur TeamBlog</div>		
<div>Startup Matchmaker</div>		
Because two heads are better than one.		
	<div>Meet your Match!</div>	
	Have a great idea for a product, but need help making it a reality? We're here to help. Startup Matchmaker is the best place for designers and developers to find each other.	
	<div>Create a Profile</div>	
<div>Create a Profile</div>	<div>Find a Developer</div>	<div>Find a Designer</div>
Are you a Designer? Put yourself out there so that others can find you!	Looking for a developer to work with on the next big thing? Look no further.	Need someone who can make a product intuitive and appealing? Get ready.
<div>Sign up Now</div>	<div>Start Your Search</div>	<div>Start Your Search</div>
© 2013 Startup Matchmaker. Made in NY.		

\*Pro tip: adding a border to everything on the page can help during this process:  
\* {border: 1px solid black; }

---

## PROCESS

---

1. Write your HTML
2. Get things into place. (Add floats, clear: both, get columns set up, etc.)
3. Add base styles. (Font-family/color for the body, remove text-decoration under anchors and add a base anchor color, remove bullets for list items, etc.)
4. Work through the page section by section and start "filling in the details." Resist the urge to be a perfectionist at this point.
5. Polish things up! Compare the design with your page. Are you using the right fonts? Colors? Is any of the spacing off?



## PROCESS — STEP 3 (BASE STYLES)

<div>Startup Matchmaker</div>		<div>Developers</div>		<div>Designers</div>	<div>How it Works</div>	<div>Our Team</div>	<div>Blog</div>
<div>Because two heads are better than one.</div>							
				<div>Meet your Match!</div>			
				<div>Have a great idea for a product, but need help making it a reality? We're here to help. Startup Matchmaker is the best place for designers and developers to find each other.</div>			
				<div>Create a Profile</div>			
<div>Create a Profile</div>		<div>Find a Developer</div>			<div>Find a Designer</div>		
<div>Are you a Designer? Put yourself out there so that others can find you!</div>		<div>Looking for a developer to work with on the next big thing? Look no further.</div>			<div>Need someone who can make a product intuitive and appealing? Get ready.</div>		
<div>Sign up Now</div>		<div>Start Your Search</div>			<div>Start Your Search</div>		
<div>© 2013 Startup Matchmaker. Made in NY.</div>							

---

## PROCESS

---

1. Write your HTML
2. Get things into place. (Add floats, clear: both, get columns set up, etc.)
3. Add base styles. (Font-family/color for the body, remove text-decoration under anchors and add a base anchor color, remove bullets for list items, etc.)
4. Work through the page section by section and start "filling in the details." Resist the urge to be a perfectionist at this point.
5. Polish things up! Compare the design with your page. Are you using the right fonts? Colors? Is any of the spacing off?

# LEARNING OBJECTIVES

- Practice programmatic thinking by writing pseudo code to solve a basic problem.
- Define web site behavior and the practical uses of JavaScript.
- Predict DOM output / changes by reading JS code.

---

# AGENDA

---



- ~~Q&A~~
- ~~Travel Blog CSS Review~~
- Intro to Programming
- Intro to Pseudo Code
- Intro to JS
- Reading JS
- Lab

---

**FEWD**

---

# INTRO TO PROGRAMMING

---

# PROGRAMMING

---



## WHAT IS A PROGRAM?

- ▶ A program is a set of instructions that you write to tell a computer what to do

## WHAT IS PROGRAMMING?

- ▶ Programming is the task of writing those instructions in a language that the computer can understand.

# WHAT IS A PROGRAM?



## chocolate chip cookies

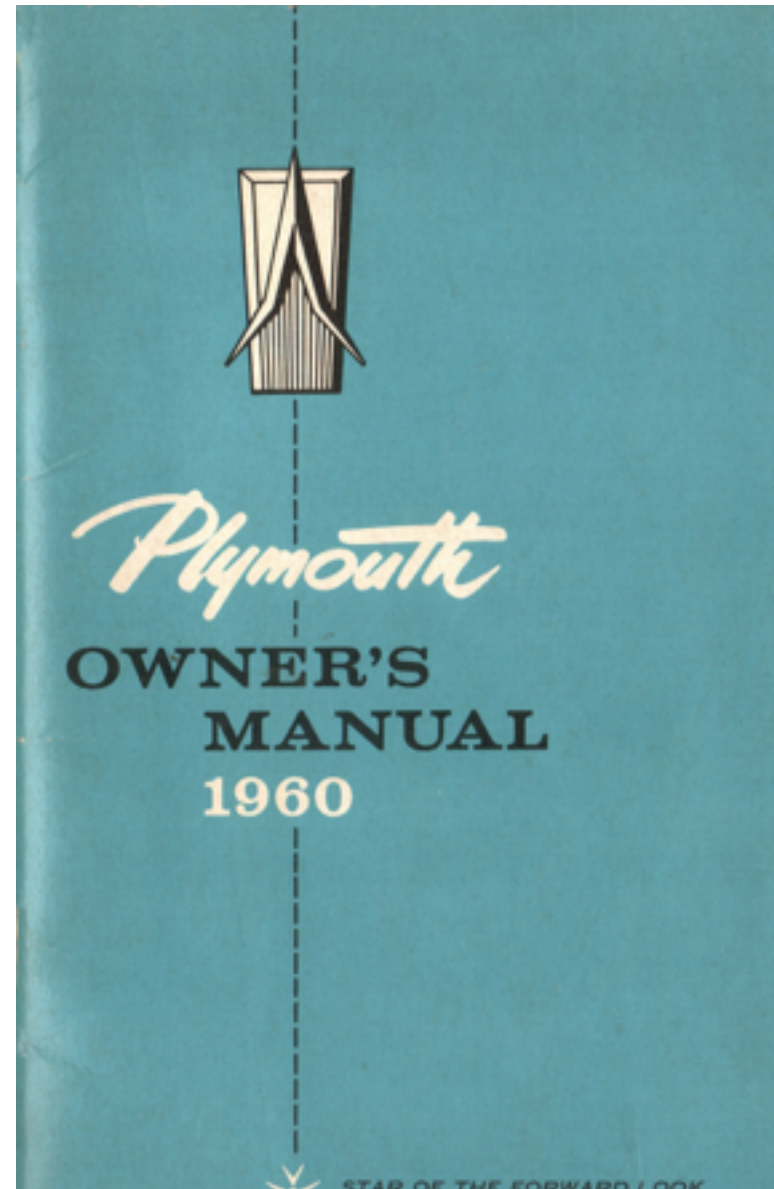
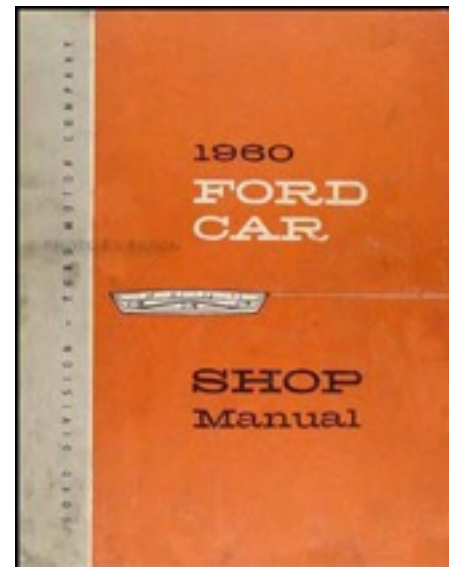
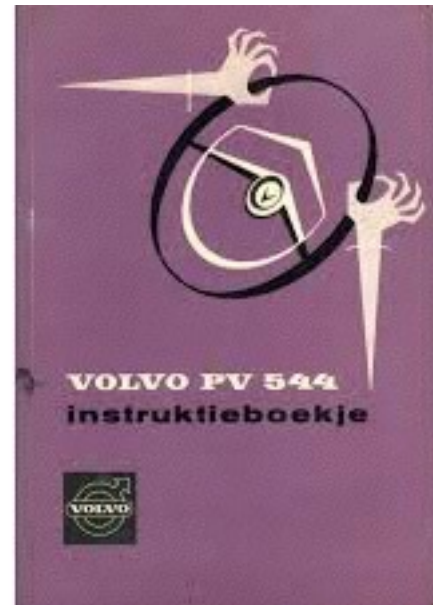
**ingredients**

- 2 cups minus 2 tablespoons cake flour
- 1 2/3 cups bread flour
- 1 1/4 teaspoons baking soda
- 1 1/2 teaspoons baking powder
- 1 1/2 teaspoons coarse salt
- 2 1/2 sticks unsalted butter
- 1 1/4 cups light brown sugar
- 1 cup plus 2 tablespoons granulated sugar
- 2 large eggs
- 2 teaspoons natural vanilla extract
- 1 cup dark chocolate chips
- 1 cup milk chocolate chips
- 1 teaspoon sea salt

Adapted from New York Times  
Preparation Time: 25 minutes, plus at least 24 hours chilling time  
Cooking Time: 20 minutes  
Yield: 2 dozen 3-inch cookies.

The secret to richer Chocolate Chip Cookies with a more sophisticated flavor is letting the dough rest for 24 to 36 hours before baking.

# WHAT IS A PROGRAM?





---

## BECOMING A PROGRAMMER

---

*It isn't about the programming language!!!  
It is about changing how you think.*

---

## HOW COMPUTERS 'THINK'

---

- ▶ Short answer — they don't think!
- ▶ While computers don't think, they *act as if they do*, by sequentially executing simple instructions.
- ▶ The only things a computer knows are the things we tell it.
- ▶ A computer doesn't learn to perform tasks like you and I — it needs to follow instructions every time it performs the task.

---

**FEWD**

---

# INTRO TO PSEUDO CODE

---

## PSEUDO CODE

---

- When we write a program, we need to figure out a way to translate the ideas that are in our heads into code
- Pseudo code is a way to 'plan out' your program before coding it
- **Pseudo code** is a detailed yet readable description of what a computer program must do, expressed in plain english rather than in a programming language

---

# THE IMPORTANCE OF PLANNING

---



Image credit: [Minecraft HD Wallpapers](#)

---

## PSEUDO CODE — THERMOSTAT

---

**Goal:** *Write pseudo code for an application that would monitor the room temperature and adjust it so the room remains at a certain temperature.*



---

## LAB — ROCK PAPER SCISSORS

---



# LAB — ROCK PAPER SCISSORS

---



## EXERCISE

### KEY OBJECTIVE

---

- Practice programmatic thinking by writing pseudo code to solve a basic problem

### TYPE OF EXERCISE

---

- Group of 3-4

### TIMING

---

*30 min*

1. Write pseudo code to program a computer to play the game 'rock paper scissors'
2. Write each line of instruction onto a post it
3. Put the post its in order to form the program
4. If you finish early, walk around and view what other groups came up with



---

**FEWD**

---

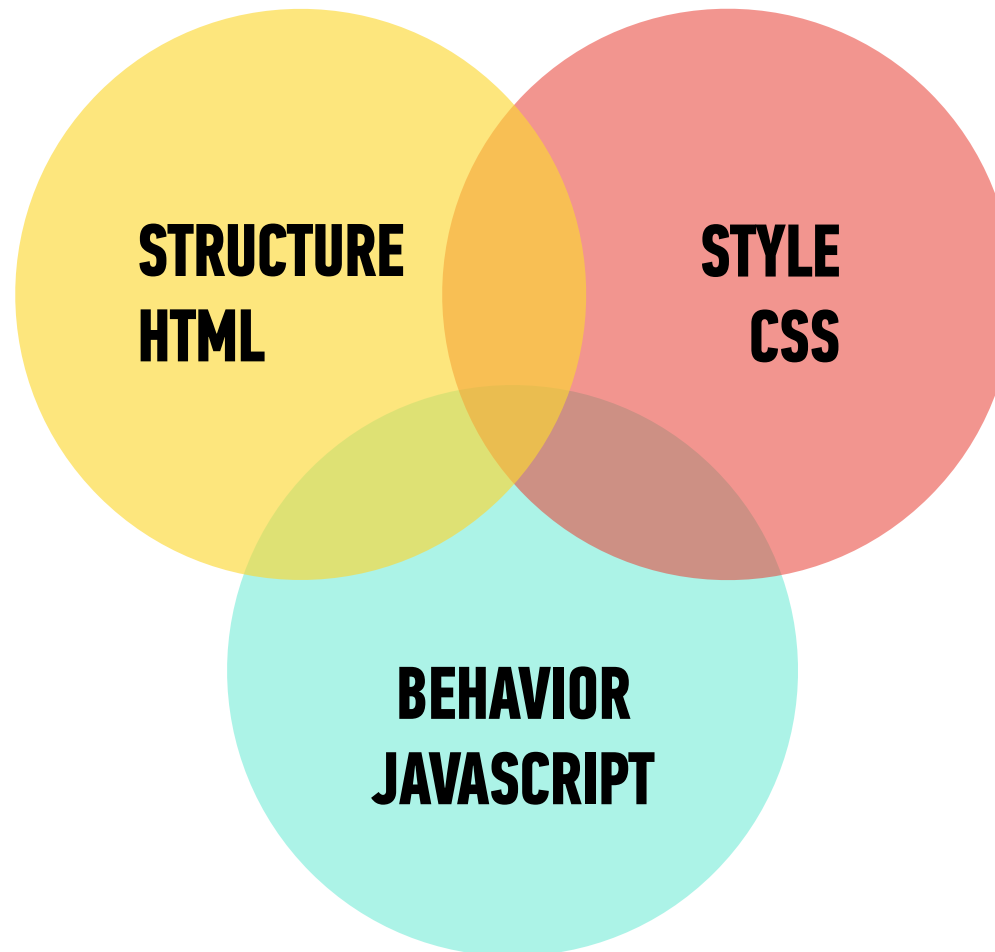
# INTRO TO JS

---

## THE THREE AMIGOS: STRUCTURE, STYLE, BEHAVIOR

---

- HTML = Noun
- CSS = Adjective
- Javascript = Verb



# JAVA VS. JAVASCRIPT

---

*Just a quick note! We're learning **JavaScript** in this class, not Java. Java and JavaScript are actually two different languages.*



!=



---

## WHAT JAVASCRIPT CAN DO!

---

**1**

Access  
Content

**2**

Modify  
Content

**3**

Program  
Rules

**4**

React to  
Events

---

## WHAT JAVASCRIPT CAN DO!

---

1

Access  
Content

2

Modify  
Content

3

Program  
Rules

4

React to  
Events

*You can use JS to select any element, attribute or text from an HTML page.*

For example:

- Select the text inside all the `<p>` elements on a page
- Select the element that has the id attribute with a value of **email**
- Find out what the user entered into a text input when they submit a form

---

## WHAT JAVASCRIPT CAN DO!

---

1

Access  
Content

2

Modify  
Content

3

Program  
Rules

4

React to  
Events

*You can use JS to add elements, attributes and text to the page (or remove them)*

For example:

- Add an error message below a form
- Change the size, position, color, or other styles for an element
- Add or remove a class from elements to trigger new CSS rules for those elements

# WHAT JAVASCRIPT CAN DO – MODIFYING CONTENT

Please Enter Your Details

REQUIRED

Some fields below require your attention

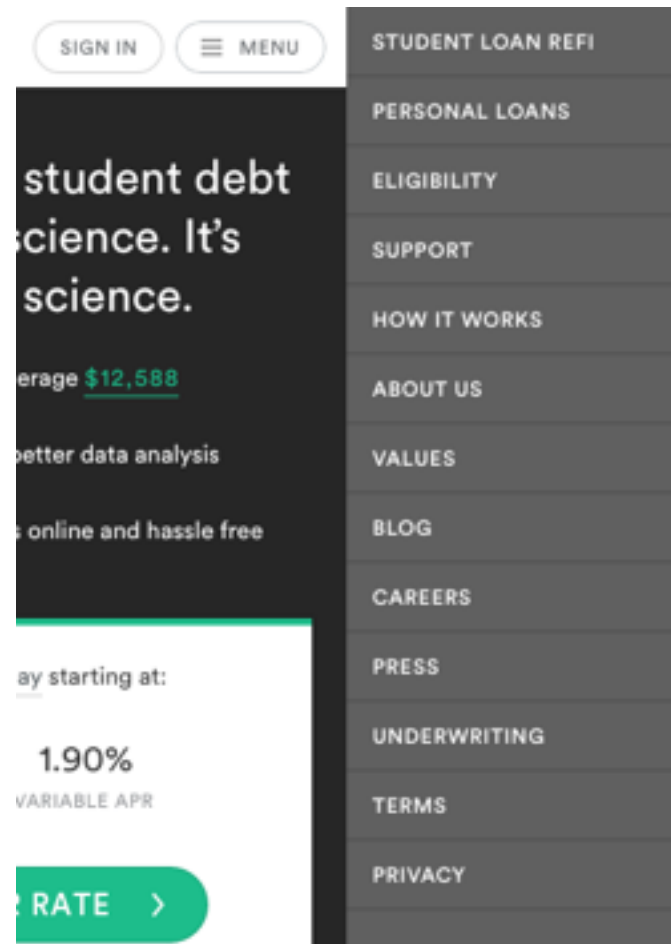
FIRST NAME	Sarah
LAST NAME	Holden
HIGHEST DEGREE	Choose One
SCHOOL	Enter School Name
EMPLOYER	Enter Employer Name
JOB TITLE	Enter Job Title
ANNUAL INCOME	\$ Enter Your Annual Income (Not Household)
BANKING + INVESTMENT TOTAL	\$ Enter Estimated Total
STUDENT LOAN BALANCE	\$ Approximate Amount (\$5,000 Minimum)
STREET ADDRESS	Enter Street Address

Add an error message  
(and styles) to a form

---

## WHAT JAVASCRIPT CAN DO – MODIFYING CONTENT

---



Change the size, position, color, or other styles for an element



---

## WHAT JAVASCRIPT CAN DO!

---

1

Access  
Content

2

Modify  
Content

3

Program  
Rules

4

React to  
Events

*You can specify a set of steps (instructions) for the browser to follow.*

For example:

- Have images/text fade in as the user scrolls down the page
- Check to make sure the user has entered a valid email address into a form and display an error message if not
- Open a chat panel when the user clicks on a 'Chat with Us' button
- Filter data when the user selects a filter

# WHAT JAVASCRIPT CAN DO – PROGRAM RULES

Filters

Clear All

Filter

Clear All

Cuisine

Clear

☐ Dinner • 28

☐ Lunch • 19

☐ Asian • 7

☐ Sandwiches • 7

☐ Healthy • 6

☐ Mediterranean • 6

☒ Thai • 5

☐ Vegetarian • 5

☐ Japanese • 4

☐ Latin American • 4

+ See all

Rating


★ ★ ★ ★ ★

& up

Pick A Restaurant

7 Restaurants nearby

Sort By Default



Trike Thai Noodles & Sushi

Japanese, Asian

★★★★★

75 Ratings

\$20


Min

\$3

Delivery

60-70 m

Est. Wait



New China Chinese Restaurant

Asian, Dinner

★★★★★

222 Ratings

\$15


Min

\$3 - \$5

Delivery

60-70 m

Est. Wait



Dante's Pizzeria (MILWAUKEE)

Dinner, Sandwiches

★★★★★

40 Ratings

\$18

Min

\$3

Delivery

90-100 m

Est. Wait

Filter data when the user selects a filter

---

## WHAT JAVASCRIPT CAN DO!

---

1

Access  
Content

2

Modify  
Content

3

Program  
Rules

4

React to  
Events

*You can specify that a script should run when an event occurs*

For example:

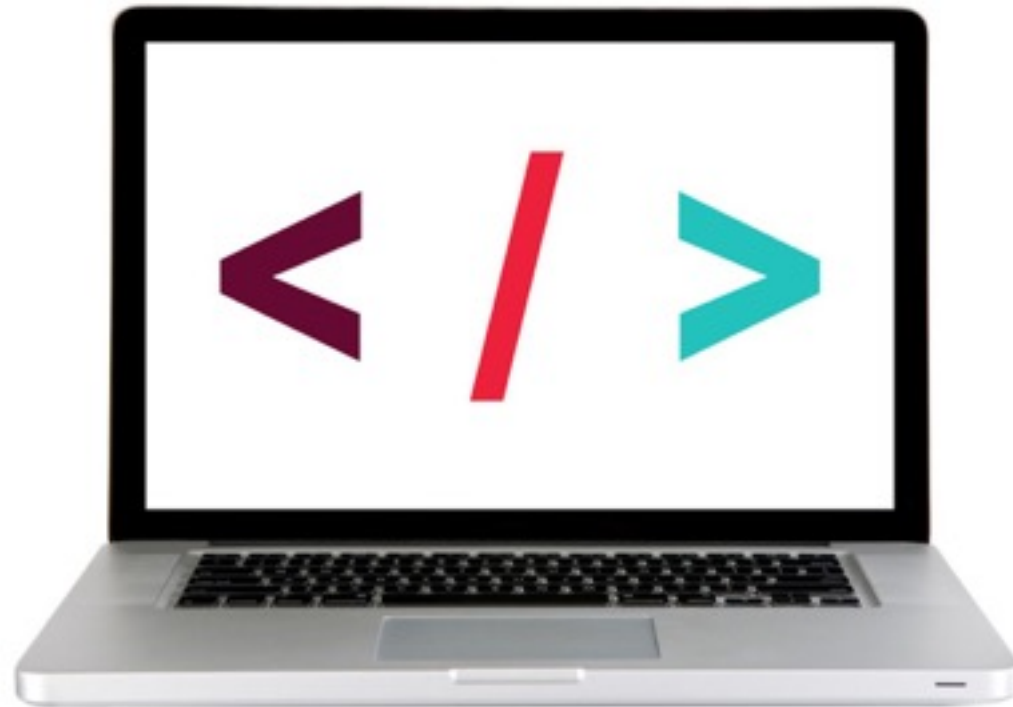
- When a button is clicked
- When the cursor hovers over an element
- When the user types information into a form
- When a page has finished loading
- When the user hits enter to submit a form

GET YOUR RATE >

---

## LET'S TAKE A LOOK

---



<https://kinhr.com/>

---

**FEWD**

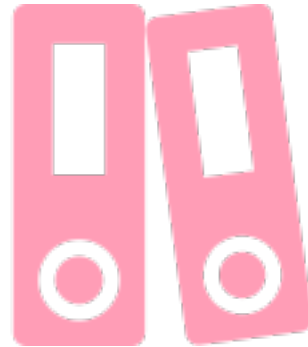
---

# READING JS

## READING JS

---

- When you are a child you learn to speak and read before you learn to write
- We learned to 'speak' JS with the discussion, video, and pseudo code



---

## READING JS — COLOR SWITCHER WALK THROUGH

---



[Color Switcher CodePen](#)

---

## LAB — TRAFFIC LIGHT

---





# LAB — TRAFFIC LIGHT

---



## EXERCISE

### KEY OBJECTIVE

---

- Predict DOM output / changes by reading JS code.

### TYPE OF EXERCISE

---

- Partner

### TIMING

---

*30 min*

1. Take a look at the [Traffic Light](#) code in Codepen
2. The yellow button changes the bulb to purple and the green light does not work.
3. Make some minor changes to the code so that the traffic light works correctly.

---

**FEWD**

---

# WIREFRAMES

---

## GETTING STARTED

---



**IDEA**

**PROTOTYPE**

**DESIGN**

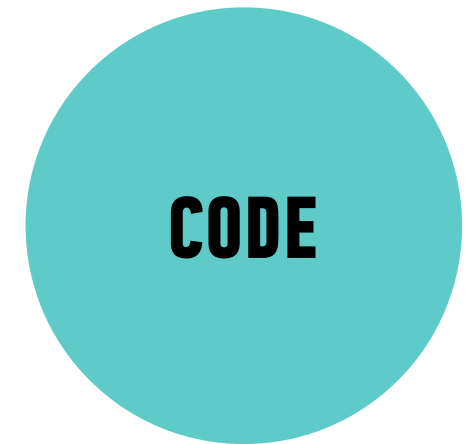
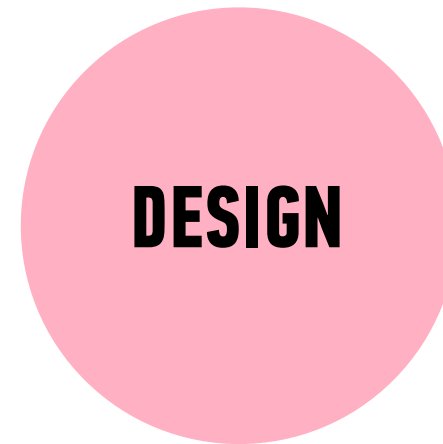
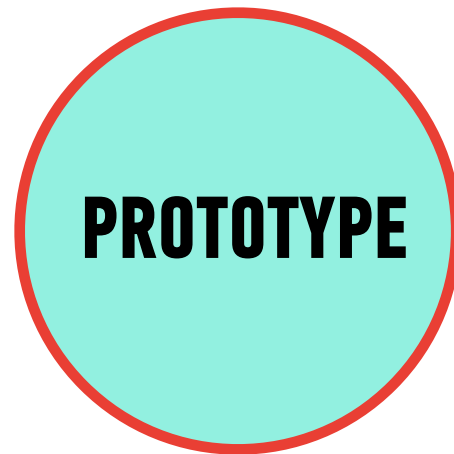
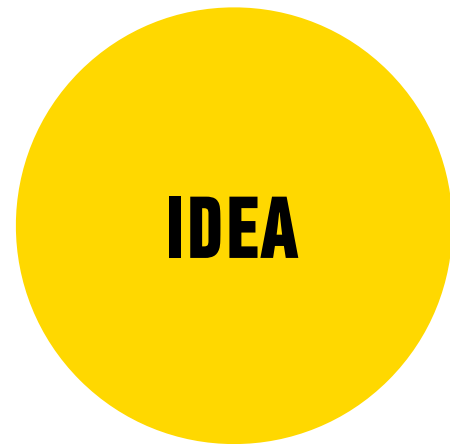
**CODE**

### FIND A COMPELLING PROBLEM:

- What problem are you solving? And for whom?
- How is your “customer” solving this problem now?
- Why is your proposed solution better?

## GETTING STARTED

---



### A PICTURE IS WORTH 1000 WORDS:

- › How will users most likely access my site?
- › What type of content/information is most valuable? How will I structure things in a way that users will be able to easily find the most important information?
- › How will users navigate the site? Will it be a single-page or multi-page site?
- › What types of organizational structures are possible?

---

## GETTING STARTED

---



**IDEA**

**PROTOTYPE**

**DESIGN**

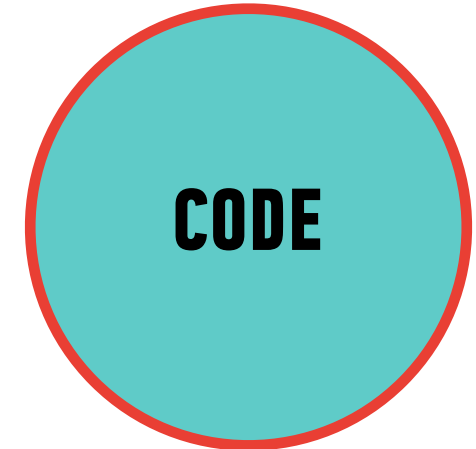
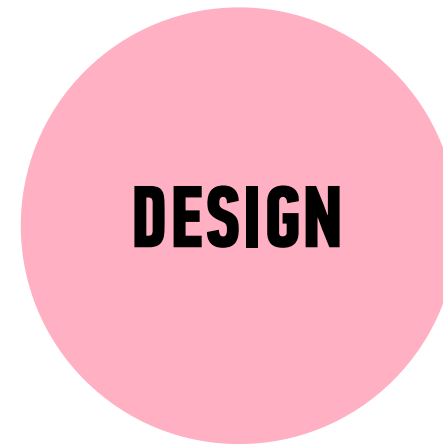
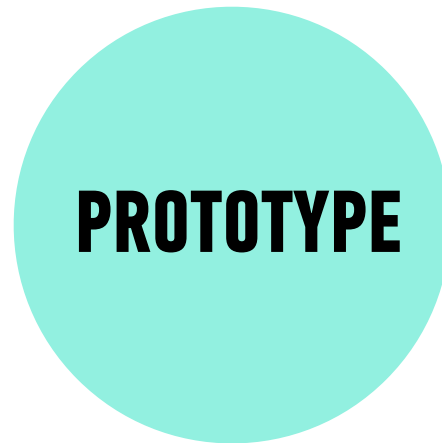
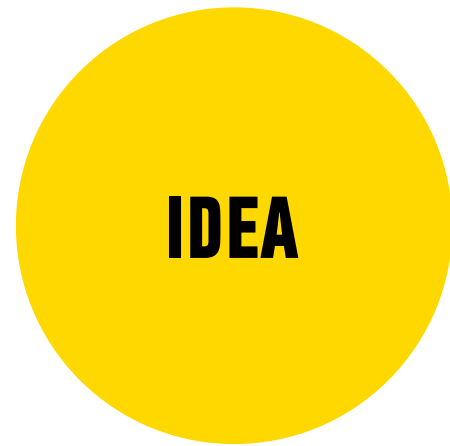
**CODE**

## FILL IN THE DETAILS:

- › I often find it helpful to create a "styleguide" for the site
- › Pick 4-5 colors for the site
- › Look through Google Fonts/Font Squirrel and pick out 2-3 fonts for the site
- › Use sites like <https://unsplash.com/> for high-resolution stock images
- › Sites like <http://www.siteinspire.com/> , <http://www.awwwards.com/> and <http://www.webdesign-inspiration.com/> are great for inspiration!

## GETTING STARTED

---



### PLAN MORE, CODE LESS:

- › Create a features/functionality list
- › Start with the structure (HTML), then add styles (CSS), then work on interactions (pseudo code and JavaScript).

---

## TIMELINE

---

Milestone 1      3/13/16      Project Proposal / Wireframes

Milestone 2      TBD

# LAB — TRAFFIC LIGHT

---



## EXERCISE

### KEY OBJECTIVE

---

- ▶ Start on wireframes/project proposals

### TYPE OF EXERCISE

---

- ▶ Individual/Partner

### TIMING

---

- Until 8:50*
1. Look at <http://gallery.ga.co/> for inspiration
  2. Start on wireframes/project proposals



# LEARNING OBJECTIVES

- Practice programmatic thinking by writing pseudo code to solve a basic problem.
- Define web site behavior and the practical uses of JavaScript.
- Predict DOM output / changes by reading JS code.

---

**PSEUDO CODE**

---

# **HOMEWORK**

---

## **HOMEWORK**

---

## **HOMEWORK**

- Keep thinking about Final project milestone #1 (Due Sunday!)
- An additional homework assignment will be added on Thursday. Keep this in mind.

**FRONT-END WEB DEVELOPMENT**

---

**SNACKS & DESIGN**

**THURSDAY MAR 10TH**

**CHRISTOPHER ZALEK & COLIN QUIRK**

**(GOOGLE SHEET IS PINNED IN SLACK)**

---

**PSEUDO CODE**

---

**EXIT TICKETS**