

---

---

# START SCREENCAST!

---

## LET'S GET EVERYTHING SET UP!

---

1. Navigate to the FEWD 42 Dashboard ([saraheholden.com/fewd\\_dashboard/](https://saraheholden.com/fewd_dashboard/)) and download the Lesson 6 starter code and slides. You'll want to keep the dashboard open for other links and resources we'll be referencing in class.
2. Move the starter code and slides from your Downloads folder to the **fewd** folder on your desktop.
3. Double-click on `starter_code_lesson_6.zip` to unzip it
4. After you've unzipped, be sure to delete the original .zip file!
5. Open the entire **fewd** folder with Sublime Text (either drag and drop the folder on the Sublime icon in the dock on Mac, or open Sublime and go to file > open... and select the **fewd** folder).
6. Log in to the FEWD 42 Slack ([fewd42.slack.com](https://fewd42.slack.com)) and join the class6 channel.

---

## WEEKLY OVERVIEW

---

### WEEK 2

Advanced CSS — Box Model and Layout

### WEEK 3

CSS Lab Week / CSS Positioning

### WEEK 4

Intro to Programming / jQuery Basics

---

**FEWD**

---

# FINAL PROJECTS

---

## FINAL PROJECTS

---

### WHERE CAN I FIND MORE INFO AND WHEN EACH MILESTONE IS DUE?

- Final Project page on the [FEWD 42 Dashboard](#)

### WHERE CAN I GET SOME INSPIRATION FROM WHAT PAST STUDENTS HAVE DONE?

- Visit the General Assembly [Gallery](#)

### WHERE SHOULD I BE RIGHT NOW?

- By the end of this week you should have looked through the gallery and jotted down a few ideas. Wireframes/project proposals will be due at the end of Week 5!



# STARTUP MATCHMAKER PART TWO

*Sarah Holden*

# LEARNING OBJECTIVES

- Practice web development by transforming a design comp into a webpage.

# AGENDA

---



- Review
- Lab — Startup Matchmaker Pt. 2
- Advanced CSS Positioning (if time permits)



---

**REFACTOR**

---

**REVIEW**

---

## EXIT TICKET QUESTIONS

---

- Why are single page applications more popular than click thru pages?
- What is SASS? Should we be using it?
- [Is there only one place to insert google font link?](#)
- What should I do when I get stuck?

---

# PREPROCESSORS

---

## CSS PRE-PROCESSORS: SASS AND LESS

- ▶ Use variables in CSS!
- ▶ Nest styles!
- ▶ Define mixins (similar to JS functions)
- ▶ Mathematical functions
- ▶ Operational functions (such as "lighten" and "darken")

## \$VARIABLES

```
$baseTextColor: #e51b24;  
  
h1 {  
    color: $baseTextColor;  
}
```

## NESTING

```
article {  
    margin-bottom: 3em;  
    h1 {  
        font-size: 2em;  
    }  
}
```

---

## EXIT TICKET QUESTIONS

---

- ▶ Can I apply `max-width: 100%;` to the background image?

*Hint: look up `background-size: cover`*

- ▶ How do you [position the background-image](#)?

- ▶ How to create button styles?

*Hint: you can use a box-shadow. I like to use a [box-shadow generator](#) to experiment.*

- ▶ How do I add a background-image?

*Hint: You'll need to look up the `background-image`, `background-position`, `background-size` and `background-repeat` properties. You'll need to use **all four**!*

---

**REFACTOR/REVIEW**

---

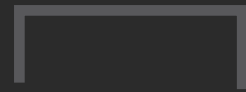
# CSS SELECTORS

---

## CSS SYNTAX

---

Selector



h1 {

color: yellow;

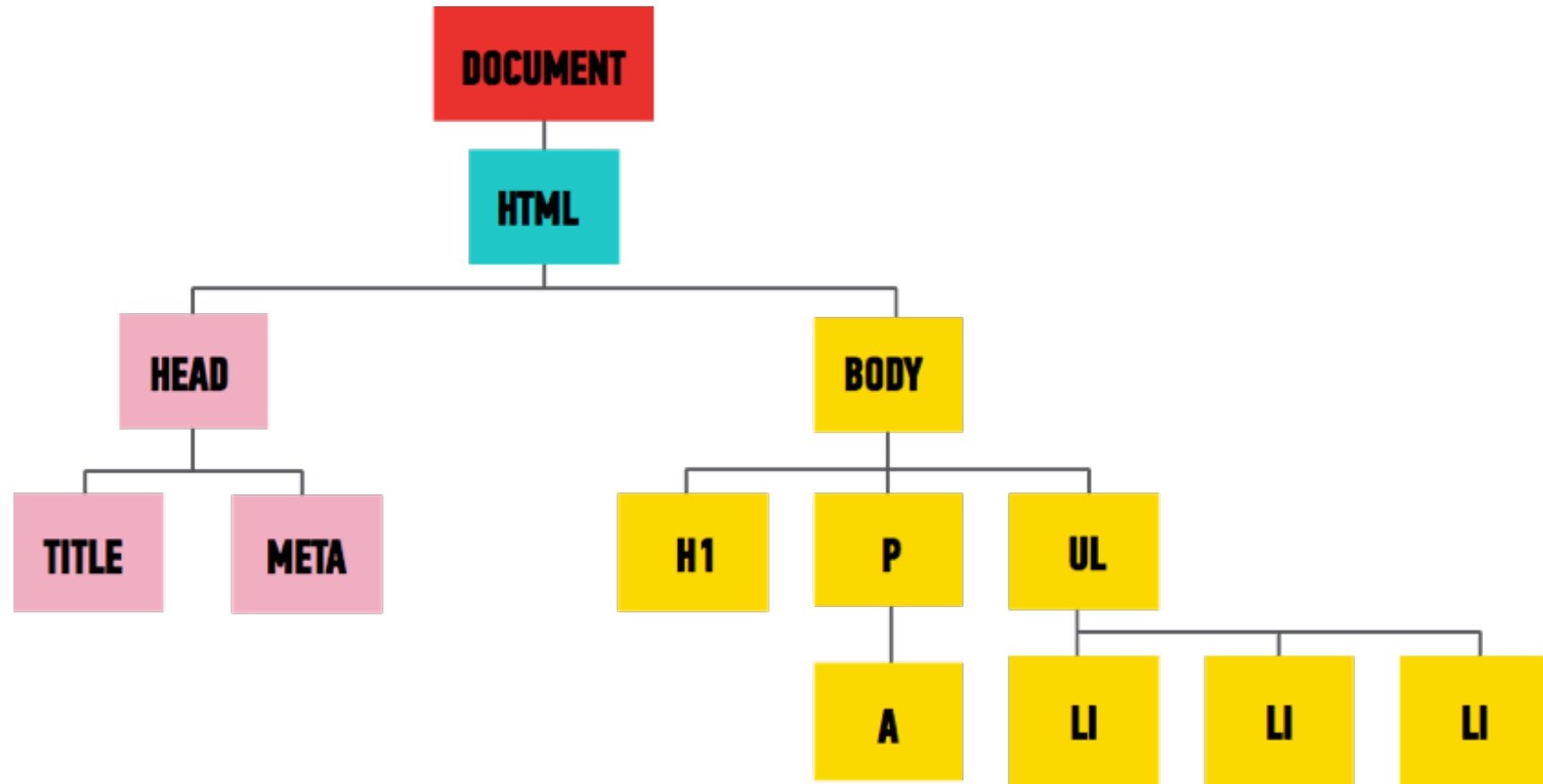
}



---

# DOM TREE

---



SELECTORS

SELECTOR:

	MEANING:	EXAMPLE:
UNIVERSAL	Applies to all elements in the document	* {}
TYPE	Matches element names	h1, h2, h3 {}
CHILD	Matches an element that is a direct child of another element	p>a {}
DESCENDANT	Matches an element that is a descendent (not just a child) of another element	p a {}
ADJACENT SIBLING	Matches the element that is directly after another element	p+a {}
GENERAL SIBLING	Matches the element that is a sibling of another	p~a {}

---

**REFACTOR/REVIEW**

---

# FLOATS

---

## MATCH IT! — WHICH GOES WITH WHICH?

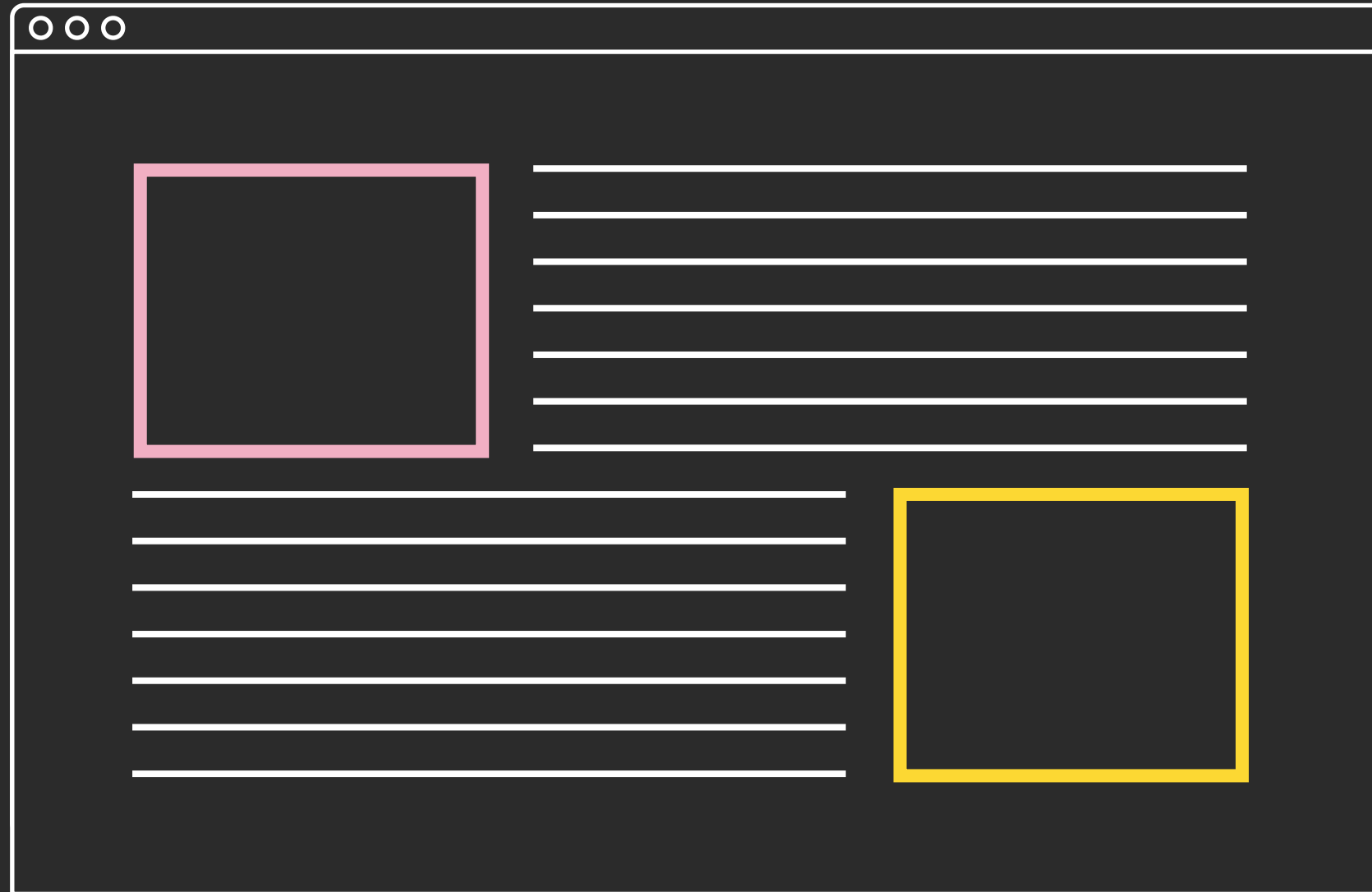
---

1. Make sure an element starts on a new line
2. Fixes "collapsed" parent when all children are floated.
3. Take an element from the normal flow and place it along the left side of its container, where text and inline elements will wrap around it

?

- A. float: left
- B. clearfix
- C. clear:both

# CSS — FLOATS



# CSS CLEAR PROPERTY

## PROBLEM:

### CSS Clear Property

#### Main Content Section

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Rem, eligendi voluptatum cupiditate nisi molestias nemo ipsum odio, laudantium blanditiis vitae a! Sapiente dolor, obcaecati voluptatibus harum unde in doloreque sint. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Rem, eligendi voluptatum cupiditate nisi molestias nemo ipsum odio, laudantium blanditiis vitae a! Sapiente dolor, obcaecati voluptatibus harum unde in doloreque sint.

- Something funny
- Click on me
- Sidebar item
- Another great point
- Wolfman ipsum

Copyright. I should be at the bottom of the page.



*Footer is floating up!*

## SOLUTION:

### CSS Clear Property

#### Main Content Section

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Rem, eligendi voluptatum cupiditate nisi molestias nemo ipsum odio, laudantium blanditiis vitae a! Sapiente dolor, obcaecati voluptatibus harum unde in doloreque sint. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Rem, eligendi voluptatum cupiditate nisi molestias nemo ipsum odio, laudantium blanditiis vitae a! Sapiente dolor, obcaecati voluptatibus harum unde in doloreque sint.

- Something funny
- Click on me
- Sidebar item
- Another great point
- Wolfman ipsum

Copyright. I should be at the bottom of the page.

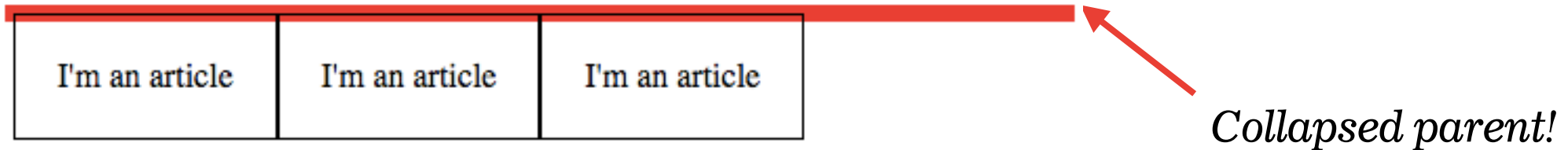
```
footer {  
  clear: both;  
}
```



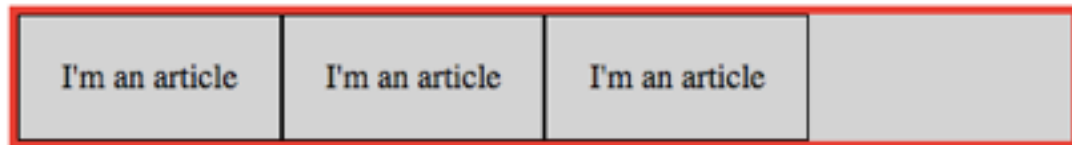
## PARENTS OF FLOATED ELEMENTS

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

### PROBLEM:



### SOLUTION:



### PT. 1 — ADD CSS CLASS:

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

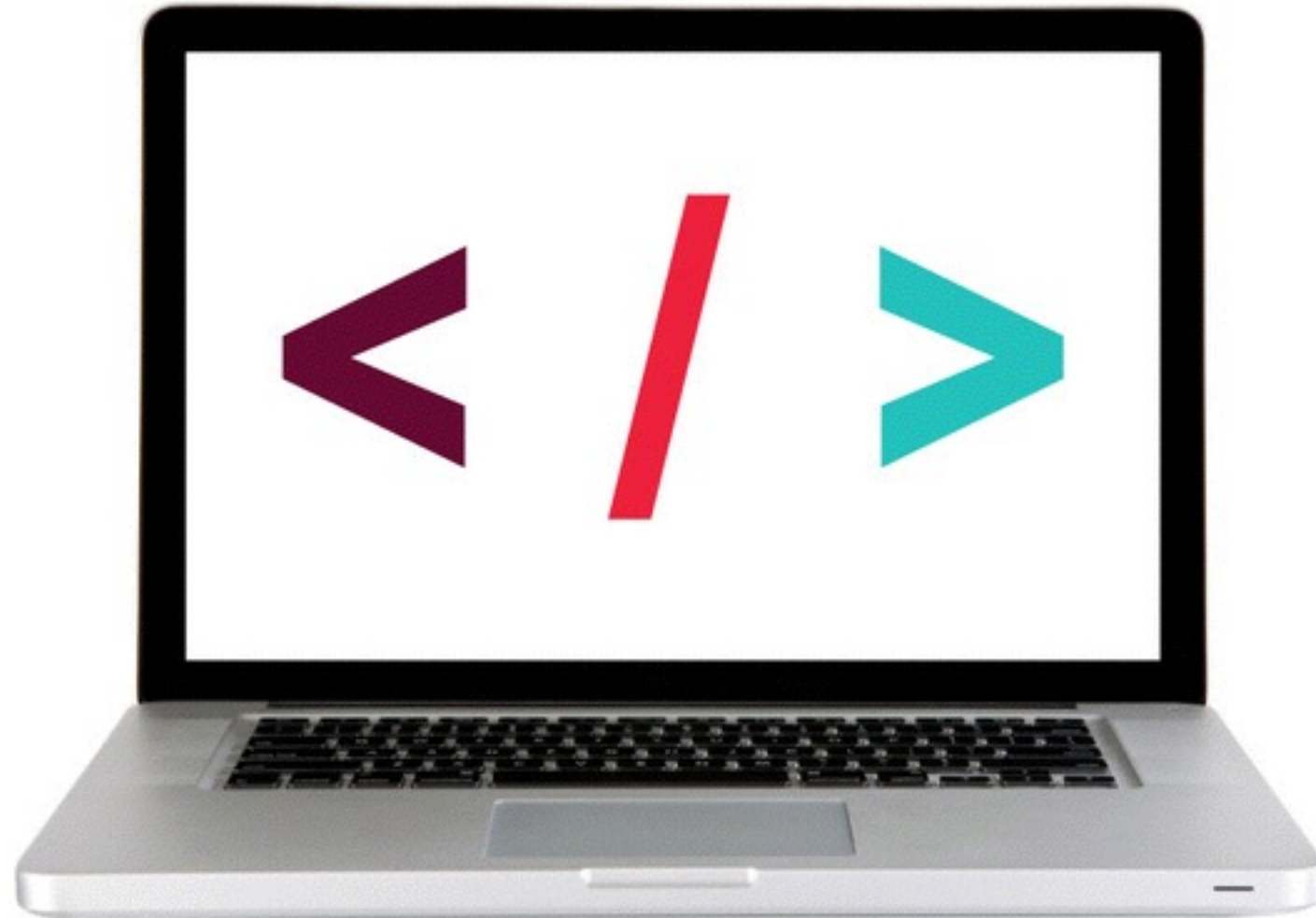
### PT. 2 — ADD CLASS TO HTML:

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

---

**LET'S TAKE A LOOK**

---



# CONFUSING NAMES — KEEPING THINGS STRAIGHT

## CLEAR: BOTH;

*Make sure an element starts on a new line*

CSS Clear Property

Main Content Section

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Rem, eligendi voluptatum cupiditate nisi molestias nemo ipsam odio, laudantium blanditis vitae at Sapiente dolor, obcaecati voluptatibus harum unde in doloreque sint. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Rem, eligendi voluptatum cupiditate nisi molestias nemo ipsam odio, laudantium blanditis vitae at Sapiente dolor, obcaecati voluptatibus harum unde in doloreque sint.

- Something funny
- Click on me
- Sidebar item
- Another great point
- Wolfman ipsum

Copyright. I should be at the bottom of the page.



CSS Clear Property

Main Content Section

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Rem, eligendi voluptatum cupiditate nisi molestias nemo ipsam odio, laudantium blanditis vitae at Sapiente dolor, obcaecati voluptatibus harum unde in doloreque sint. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Rem, eligendi voluptatum cupiditate nisi molestias nemo ipsam odio, laudantium blanditis vitae at Sapiente dolor, obcaecati voluptatibus harum unde in doloreque sint.

- Something funny
- Click on me
- Sidebar item
- Another great point
- Wolfman ipsum

Copyright. I should be at the bottom of the page.

## CLEARFIX:

*Fixes collapsed parent*

1	2	3	4
5	6	7	8



1	2	3	4	
5	6	7	8	

# ACTIVITY

---



## EXERCISE

### KEY OBJECTIVE

---

- Differentiate between floats, clear and clearfix.

### TYPE OF EXERCISE

---

- Individual/Partner

### WHERE

---

- Exercise is in starter\_code\_lesson\_6 > floats\_clear\_clearfix

### INSTRUCTIONS

---

*5 min*

1. Float each paragraph within the .blocks-wrapper to the left.
2. Make sure the footer stays at the bottom of the page!
3. Fix things so that we still have a nice pink background behind the blocks.

---

## STEPS TO ACHIEVE A MULTI-COLUMN LAYOUT

---

**1. MAKE SURE EACH COLUMN HAS A WRAPPER AROUND IT IN YOUR HTML**

**2. ADD BOX-SIZING: BORDER-BOX; TO EVERYTHING (USE THE \* CSS SELECTOR)**

**3. GIVE A WIDTH TO EACH COLUMN (PREFERABLY IN %)**

**4. FLOAT EACH COLUMN TO LEFT**

**5. USE PADDING TO ADD SPACE BETWEEN COLUMNS**

**6. CLEAR ANYTHING UNDERNEATH YOUR COLUMNS I.E. A FOOTER  
USING THE CSS CLEAR PROPERTY (CLEAR: BOTH;)**

---

**FEWD**

---

# REFACTORING



---

**ANIMATION**

---

# TRANSITIONS

---

## TRANSITIONS

---

- Provide a way to control animation speed when changing properties
- Instead of having property changes take effect immediately, you can have them take place over a period of time.

```
.example {  
  transition: [transition-property] [transition-duration] [transition-timing-function] [transition-delay];  
}
```

---

## TRANSITIONS

---



**PROPERTIES  
TO ANIMATE**

Which properties  
to animate



**DURATION**

How long the  
transition will last



**TIMING  
FUNCTION**

How the transition  
will run



**DELAY**

When the animation  
will start

## TRANSITIONS – TRANSITION-PROPERTY

PROPERTIES  
TO ANIMATE

- ▶ Can specify a specific property to transition or "all" to transition all properties
- ▶ *Default:* all

```
div {  
  transition: opacity 0.5s;  
}
```

```
div {  
  transition: all 0.5s;  
}
```

```
div {  
  transition: height 0.5s;  
}
```

```
.example {  
  transition: [transition-property] [transition-duration] [timing-function] [transition-delay];  
}
```

## TRANSITIONS – TRANSITION-DURATION

### DURATION

- ▶ A time value, defined in seconds or milliseconds

```
div {  
  transition: all 0.5s;  
}
```

```
div {  
  transition: all 350ms;  
}
```

```
div {  
  transition: all 3s;  
}
```

```
.example {  
  transition: [transition-property] [transition-duration] [timing-function] [transition-delay];  
}
```

# TRANSITIONS

## TIMING FUNCTION

- Describes how a transition will proceed over its duration, allowing a transition to change speed during its course.
- Timing functions: ease, linear, ease-in, ease-out, ease-in-out

```
div {  
  transition: opacity 0.5s ease;  
}
```

```
div {  
  transition: opacity 0.5s ease-in-out;  
}
```

```
.example {  
  transition: [transition-property] [transition-duration] [timing-function] [transition-delay];  
}
```



# TRANSITIONS

**DELAY**

- ▶ Length of time before the transition starts

```
div {  
  transition: background-color 0.5s ease 2s;  
}
```

```
.example {  
  transition: [transition-property] [transition-duration] [timing-function] [transition-delay];  
}
```

---

## MORE FUN WITH TRANSITIONS — CODROPS

---

Fun CSS button styles: [Creative buttons](#)

Icon hover effects: [Icon Hover Effects](#)

Modal dialogue effects (advanced): [Dialogue Effects](#)

---

## ACTIVITY — BUTTON LAB

---



### EXERCISE

#### KEY OBJECTIVE

---

- Practice using CSS transitions

#### TYPE OF EXERCISE

---

- Individual/Partner Lab

#### TIMING

---

*6 min*

1. Add :hover styles and transition to the button

---

**FEWD**

---

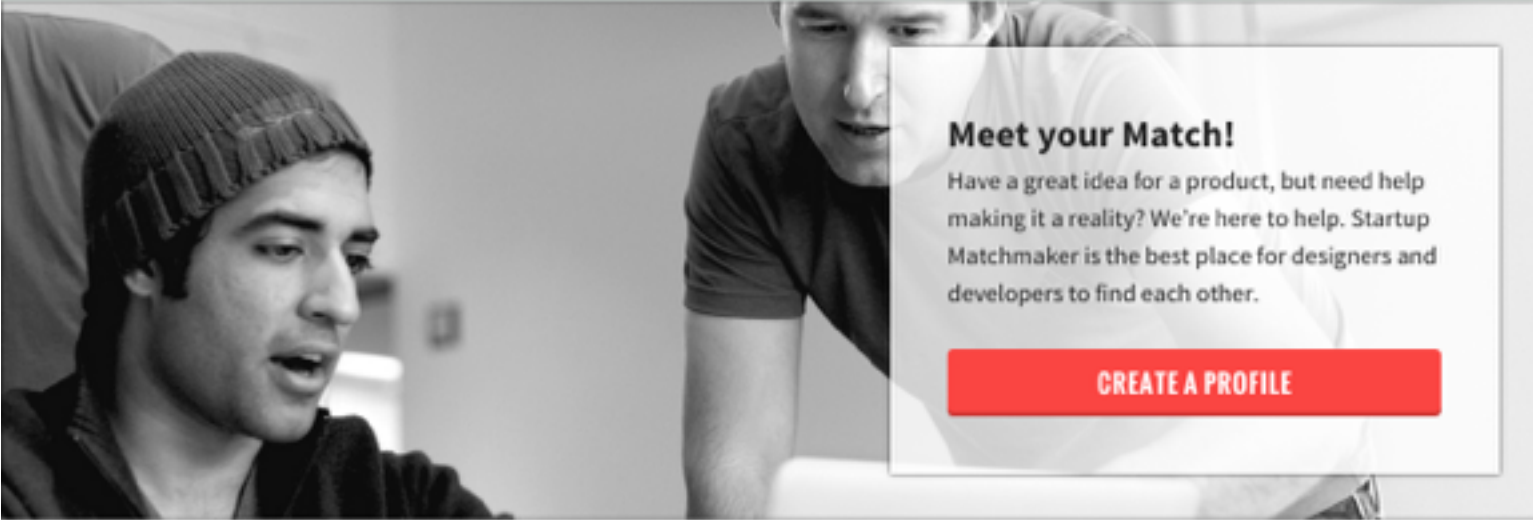
# STARTUP MATCHMAKER PT. 2

# LAB — STARTUP MATCHMAKER

## Startup Matchmaker

DEVELOPERS DESIGNERS How it Works Our Team Blog

*Because two heads are better than one.*



### Meet your Match!

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.

[CREATE A PROFILE](#)

### Create a Profile

Are you a Designer? Put yourself out there so that others can find you!

[SIGN UP NOW](#)

### Find a Developer

Looking for a developer to work with on the next big thing? Look no further.

[START YOUR SEARCH](#)

### Find a Designer

Need someone who can make a product intuitive and appealing? Get ready.

[START YOUR SEARCH](#)

© 2013 Startup Matchmaker. Made in NY.

# LAB – STARTUP MATCHMAKER



## KEY OBJECTIVE

- Demonstrate the ability to plan and build a website

## LAB SESSION

- |            |   |
|------------|---|
| 10 min     | 1. Hook up Google Fonts (use styleguide.png for reference).   |
|            | 2. Use a color picker (Sip) to pick main colors (use styleguide.png for reference).   |
| 25 min     | 3. Get everything where it needs to be! Add styles for page structure (columns, floats, inline-block, clear: both, clarifix). Use " <i>Steps to achieve a multi-column layout</i> " in review section for reference |
| 15 min     | 4. Look up background-image, background-position, and background-repeat properties in CSS (Sarah recommends MDN). Implement the background image and "overlay".   |
| 5 min      | 5. Add base Styles (base fonts, colors, etc.)   |
| 20 min     | 6. Add base styles for headers, anchors, text   |
| Until 8:50 | 7. Style everything else!   |
|            | 8. <b>Super bonus:</b> Build pages for designers and developers (use pngs in starter code folder for reference)   |

# LAB – STARTUP MATCHMAKER (HACKER EDITION)

---



## EXERCISE

### KEY OBJECTIVE

---

- Demonstrate the ability to plan and build a website

### LAB SESSION

---

*Until 8:50*

1. Materials for the advanced version are in the `startup_matchmaker_advanced` folder.
2. Set up folder/document structure for project (feel free to use the `html_template` as a starter!)
3. Gather content (in `copy.txt`) and add HTML Markup
4. Focus on layout styles first (getting everything in place)
5. Then add colors, text styles, fonts, etc.
6. Tidy things up
7. **Super bonus:** Build pages for designers and developers (use pngs in starter code folder for reference)

---

# LET'S GIT IT!

---

## IN SUBLIME TEXT:

Make sure all your changes are saved.

## IN THE GITHUB APP:

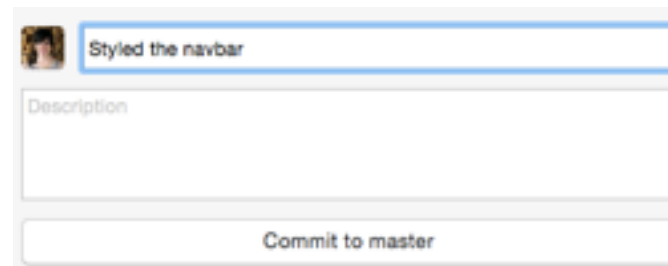
**ADD**

Make sure the boxes next to each file you've edited are checked.

☒ travel\_blog\_part1/images/ad.gif

**COMMIT**

Enter a commit message and click "commit to master."

A screenshot of the GitHub web interface for committing changes. It shows a text input field with the commit message "Styled the navbar", a "Description" text area, and a "Commit to master" button at the bottom.


Styled the navbar

Description

Commit to master

**PUSH**

Click "sync" in upper right corner.

 Sync



---

**FORM BASICS**

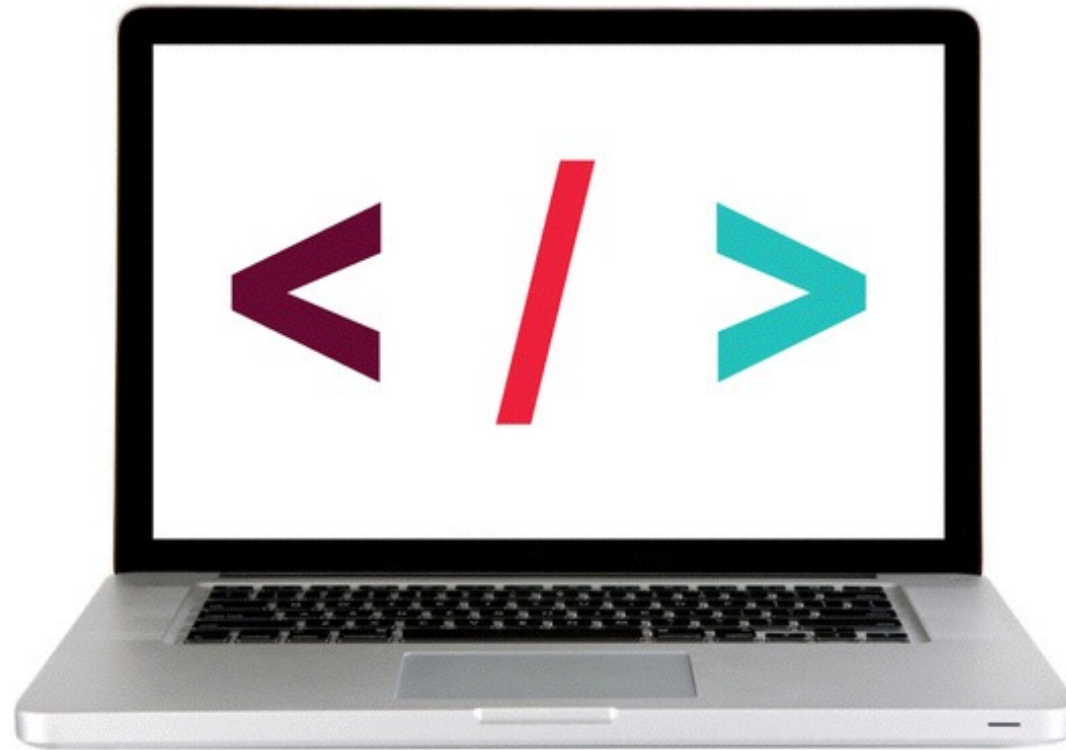
---

# ADVANCED CSS POSITIONING

---

## LET'S TAKE A CLOSER LOOK — POSITIONING 101

---



---

## 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

---

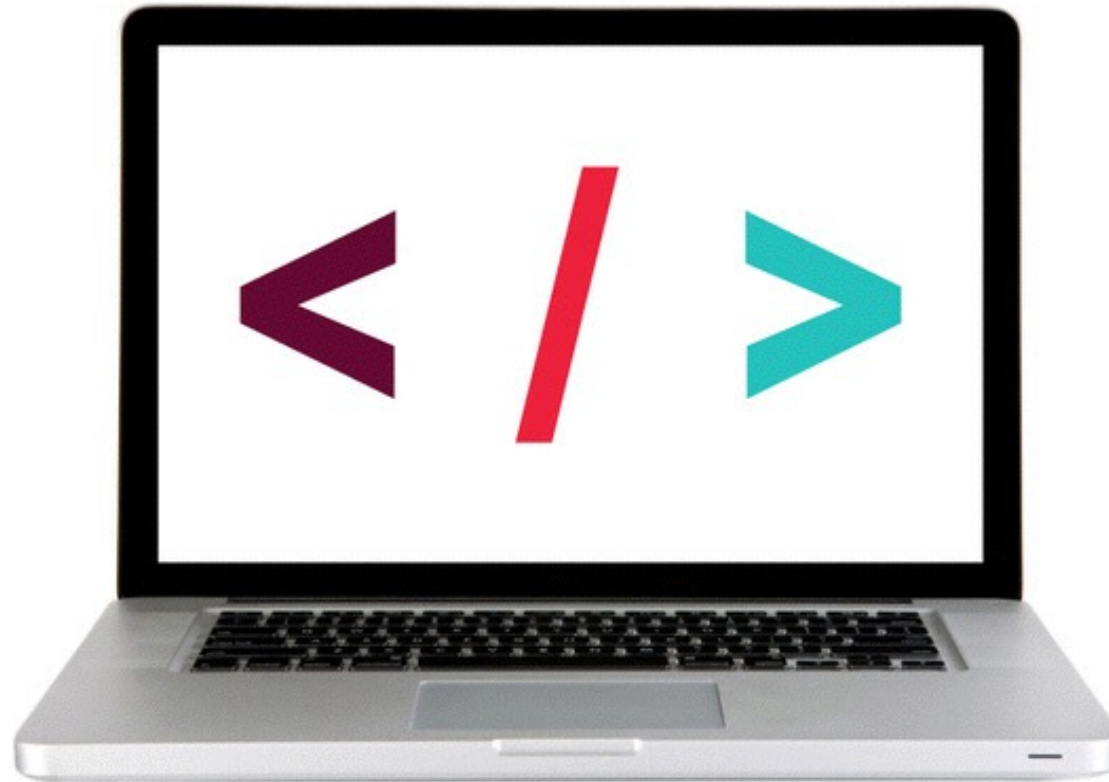
- When using relative, fixed or absolute positioning, elements can overlap.
- When elements overlap, the elements that appear later in the HTML code sit on top of those that appear earlier in the page.
- If you want to control which elements are layered on top of each other, you can use the z-index property.
- 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;  
}
```

---

## LET'S TAKE A CLOSER LOOK

---



```
starter_code_lesson_6 > positioning_fun
```



---

## WANT TO LEARN MORE?

---

Resources for more info/examples:

- Textbook (CSS & HTML): Pages 363 - 369
- A List Apart: [CSS Positioning 101](#)

# LEARNING OBJECTIVES

- Practice web development by transforming a design comp into a webpage.

---

**LAB**

---

# **HOMEWORK**

---

## **HOMEWORK**

---

*Read the specs on the FEWD 42 Dashboard.  
Don't forget the videos/reading!!*

# EXIT TICKETS

<http://goo.gl/forms/vPhCOlfESf>