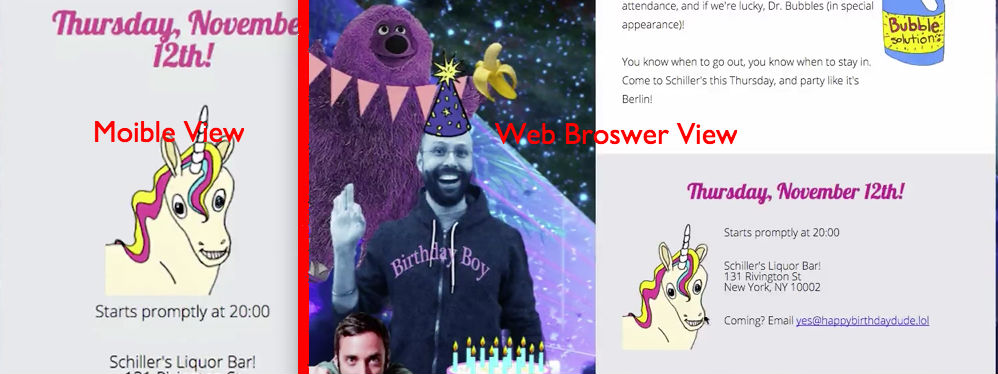
**Challenge: Get Your Mobile Revving**

[Week 3: Build a Responsive Website for iPhone and Android](https://onemonth.com/courses/html#learn-responsive-web-design)· Lesson 2 ·

* [Lesson Notes](https://onemonth.com/courses/html/steps/get-your-mobile-revving?autoplay=1#step-notes)
* [Discussion (7)](https://onemonth.com/courses/html/steps/get-your-mobile-revving?autoplay=1#discussion)

In this lesson, I’ll introduce you to the concept of “mobile-first.” Mobile first, as the name suggests, means that we start developing an optimized version of the web application for mobile devices, which have more restrictions than expand its features to create a tablet and desktop version. Coding mobile first lets you think philosophically about how to make your site as simple and legible as possible. The lesson ends with you're challenge!

Compare the difference between the desktop and mobile version of our festive birthday invitation:



Notice how on the mobile (lefthand) version, the image and information are centered for easy access, while the wider desktop version has room for extra imagery and different alignment configurations, with the unicorn to the left of the text.

Coding for the least common denominator - coding mobile first - lets you think philosophically about how to make your site as simple and legible as possible. From that solid baseline, you can expand out and get fancier on the web.

You can find all the necessary text in the read.me (which is in this week's project download). Play around placing it inside the div container for the body!

<body>

<div class="container">

<h1>Alexis's Birthday Party!</h1>

<h2>Thursday, November 12th!</h2>

<p>Starts promptly at <time>20:00</time>

</div>

</body>

</html>

**Challenge Answers: What's In The Box?**

[Week 3: Build a Responsive Website for iPhone and Android](https://onemonth.com/courses/html#learn-responsive-web-design)· Lesson 3 ·

* [Lesson Notes](https://onemonth.com/courses/html/steps/whats-in-the-box?autoplay=1#step-notes)
* [Discussion](https://onemonth.com/courses/html/steps/whats-in-the-box?autoplay=1#discussion)

Let’s go over the solution to the last lecture’s challenge.

**The answers to the challenge**:

1. Make sure the different containers for the text are different divs. You can differentiate them with class tags. Like so:

*<div class="intro"></div>* and *<div class="contact"></div>*

2. Add <p> tags for paragraphs, adding <h1> and <h2> tags to set off important titles and dates.

3. Include the images from the project folder.

You're code should look (more or less) like this! If not, just self-correct and let's keep moving. If you keep moving you'll succeed!

<div class="container">

<div class="intro">

<h1>Alexis's Birthday Party!</h1>

<p>Alexis' birthday is coming up! We hope you intend to come to our magical party for a most lovely friend. Their will be 99 bottles of wine, a dozen in attendance, and if we're lucky, Dr. Bubbles (in special appearance)!</p>

<p>You know when to go out, you know when to stay in. Come to Schiller's this Thursday, and party like its Berlin!</p>

</div>

<div class="contact">

<h2>Thursday, November 12th!</h2>

<p>Starts promptly at <time>20:00</time>

<p>Schillers's Liquor Bar!

131 Rivigton St

New York, NY </p>

<p>Coming? Email yes@happybirthdaydude.lol</p>

</div>

Don't worry if your code looks slightly different. We'll go over how to streamline it next.

**Have Your Cake And Make It Flexible Too**

[Week 3: Build a Responsive Website for iPhone and Android](https://onemonth.com/courses/html#learn-responsive-web-design)· Lesson 4 ·

* [Lesson Notes](https://onemonth.com/courses/html/steps/have-your-cake-and-make-it-flexible?autoplay=1#step-notes)
* [Discussion (6)](https://onemonth.com/courses/html/steps/have-your-cake-and-make-it-flexible?autoplay=1#discussion)

In this lesson, we learn how to make our images 100% flexible whether they are on the desktop, mobile or in an email. This is the first truly responsive code you're going to write. By the end of the lesson, you'll know how to add responsive images to your pages that automatically adjust their size with regards to the size of the screen.

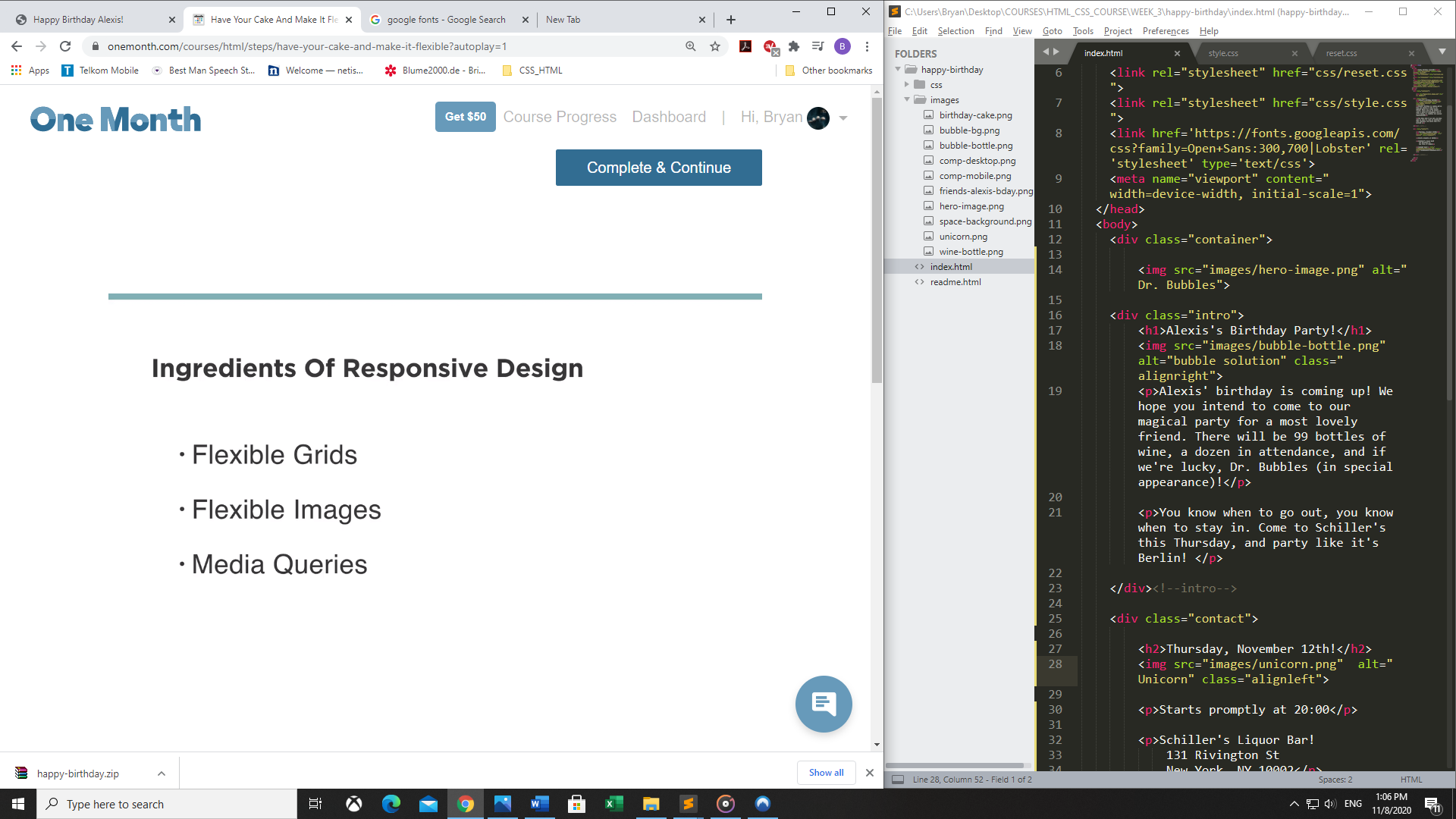
Making your images flexible presents a couple of challenges. You can't simply change the image width to a percentage in CSS because that would change *all* the images. So the first step is to make our "hero" or top image a discrete div with its own class, and then specify the percentage for that class.

So your code should look like this:

<div class="hero">

<img src="images/hero-image.png" alt="Dr.. Bubbles">

</div>



**Own The Newest Style: Media Queries and CSS Specificity**

[Week 3: Build a Responsive Website for iPhone and Android](https://onemonth.com/courses/html#learn-responsive-web-design)· Lesson 5 ·

* [Lesson Notes](https://onemonth.com/courses/html/steps/own-the-newest-style?autoplay=1#step-notes)
* [Discussion (9)](https://onemonth.com/courses/html/steps/own-the-newest-style?autoplay=1#discussion)

Welcome to the wonderful world of media queries! In this lesson, we use conditional CSS rules to keep your images beautiful at every size. We also cover the principle of specificity. Specificity is what your browser uses to decide the value (or importance) of conflicting CSS property values. Let’s learn more!

**Specificity** - Here, it means that that choices you specify farther down you're style sheet override the choices above them. If you make the background grey, and then later on in the cascade make your background blue, the background will look blue. It defers to the most recent code in you're CSS.

But the neat things about **media queries** is that they allow you to switch between different style sheets, so that one only overrides the other when certain conditions are met. Here's how it should look:

<head>

<title>Happy Birthday Alexis!</title>

<meta name="description" content="Happy Birthday Alexis!">

<link rel="stylesheet" href="css/reset.css">

<link rel="stylesheet" href="css/style.css">

<link rel="stylesheet" href="css/small.css" media="(max-width: 630px)">

<link href='https://fonts.googleapis.com/css?family=Open+Sans:300,700|Lobster' rel='stylesheet' type='text/css'>

<meta name="viewport" content="width=device-width, initial-scale=1">

</head>

Here, we're dealing with a size condition. When the image is *larger* than 630 pixels, the second sheet doesn't load at all. When the image is smaller than 630 pixels, the second CSS sheet knows to get off the couch and actually do something.

**Remember**: there's no one right answer. If it makes more sense to you to define a media query by the minimum width, go for it!

It's important to input your style sheets in the correct order: style.css and than small.css. Because, small.css doesn't outright replace the first CSS sheet. It simply overrides duplicate conditions. This allows you to code for different size screens.

**Tip**: Keep toggling back and forth between large and small views of your invitation as you play around. Make sure the features that look good in one view but not the other only appear at the right time, in the right size.

**HTML vs. HTML5: What’s the difference?**

[Week 3: Build a Responsive Website for iPhone and Android](https://onemonth.com/courses/html#learn-responsive-web-design)· Lesson 6 ·

* [Lesson Notes](https://onemonth.com/courses/html/steps/html-vs-html5?autoplay=1#step-notes)
* [Discussion (11)](https://onemonth.com/courses/html/steps/html-vs-html5?autoplay=1#discussion)

What’s the difference between HTML and HTML5? In this lesson, we look at the updated elements specific to the HTML5 language. We then keep the magic rolling and figure out how to float images in both our mobile and desktop views using some advanced CSS floats.

**Why and when should we use an *aside*?**In HTML5, we were given new elements that allowed us to give more sophisticated "meaning" to our code. That classifications more specific then <div> makes your code easier to read. Thus, we now have elements like the header, footer, section, and aside.  
  
The aside element is used when we have content that is related to our main content, but can be considered separate from that content. This would be very similar to creating a div with a class of "aside." By streamlining this process with a built-in element, its that much easier to "say" exactly what we mean with our code and have it's meaning be immediately evident to both our fellow humans and robots.   
  
**Don't worry TOO MUCH about the "semantic meaning" of your code**. Its a fiercely debated topic and it does have some benefits. It's much more important to practice writing code then to worry if we're being semantic enough.

**Tip**: You can always use the Web Inspector to figure out what's going on. Play around. Don't know what this is? Ask in the community and we're here to help you.

**What is an Example of an iFrame in HTML?**

[Week 3: Build a Responsive Website for iPhone and Android](https://onemonth.com/courses/html#learn-responsive-web-design)· Lesson 7 ·

* [Lesson Notes](https://onemonth.com/courses/html/steps/example-of-an-iframe-in-html?autoplay=1#step-notes)
* [Discussion (9)](https://onemonth.com/courses/html/steps/example-of-an-iframe-in-html?autoplay=1#discussion)

We use an iframe (inline frame) to embed a map onto our webpage. What’s an iFrame tag used for? For example, if you'd like to add Google Maps to a website, you can easily do simply by copying and pasting the iframe code from Google into your source code. The best part is that once you know how to configure iframes, you can steal this code from Google Maps, YouTube, Vimeo and thousands of cool sites around the web.

This step is actually pretty straightforward. But if you'd like a second look at the code, the first step is to grab the embeddable iframe from Google Maps. Than you add an additional section at the bottom of you're page here:

</section>

<iframe src="https://www.google.com/maps/embed?pb=!1m14!1m8!1m3!1d12095.715959922774!2d-73.98471989416502!3d40.71957968260492!3m2!1i1024!2i768!4f13.1!3m3!1m2!1s0x0%3A0x6455254f2d644a86!2sSchiller&#39;s+Liquor+Bar!5e0!3m2!1sen!2sus!4v1446874459782" width="600" height="450" frameborder="0" style="border:0" allowfullscreen></iframe>

</section>

</div>

# Best Practices: Keeping Your Code Ship-Shape

[Week 3: Build a Responsive Website for iPhone and Android](https://onemonth.com/courses/html#learn-responsive-web-design)· Lesson 8 ·

* [Lesson Notes](https://onemonth.com/courses/html/steps/keeping-code-clean#step-notes)
* [Discussion (11)](https://onemonth.com/courses/html/steps/keeping-code-clean#discussion)

In this coding tutorial, we’ll make some improvements to the website by changing the background color with CSS, adding whitespace (padding or margin), adjusting the text size and using descriptive, meaningful code (e.g. for email links, address, etc.)

Make sure you're taking the time to practice coding yourself before checking out the answers. The only way to get better is to run into roadblocks and problem-solve for yourself. There's a number of issues to tackle in order to get this invitation page polished, and you should absolutely feel free to attack them in the order that makes the most sense to you.

**Things you should have addressed:**

1. Background

2. Padding, Margins, And Line Height

3. Text Size

4. Descriptive, meaningful code (ie. time, address, links)

So your final project should look ROUGHLY something like this:

<body>

<div class="body">

<img src="images/hero-image.png" alt="Dr. Bubbles">

</aside>

<section class="intro">

<h1>Alexis's Birthday Party!</h1>

<aside class="right">

<img src="images/bubble-bottle.png" alt="Bubble Solution">

</aside>

<p>Alexis' birthday is coming up! We hope you intend to come to our magical party for a most lovely friend. They're will be 99 bottles of wine, a dozen in attendance, and if we're lucky, Dr. Bubbles (in special appearance)!</p>

<p>You know when to go out, you know when to stay in. Come to Schiller's this Thursday, and party like its Berlin!</p>

</section>

</section>

<section class="contact">

<h2>Thursday, November 12th!</h2>

<aside class="left">

Be sure to check and double check all the adjustments you made in the different size views!

# Pixels, Ems, and Rems (Oh My!)

[Week 3: Build a Responsive Website for iPhone and Android](https://onemonth.com/courses/html#learn-responsive-web-design)· Lesson 9 ·

* [Lesson Notes](https://onemonth.com/courses/html/steps/pixels-ems-rems?autoplay=1#step-notes)
* [Discussion (2)](https://onemonth.com/courses/html/steps/pixels-ems-rems?autoplay=1#discussion)

Pixels vs. Ems vs. Rems what’s the difference? In this lesson, we take a moment to go over the differences between Pixels, Ems, and Rems, and talk about which unit is the most useful for creating responsive code.

**Pixel -**An absolute unit of measurement - the smallest area of illumination on a display screen (although the physical size of a pixel varies depending on the size of the display screen). They have support from every browser.

**Em -** A multiple of the font-size of their parent element. That means that each em will adjust the size of you're font by a factor of ten, based on the size of the element it's inside.

**Rem -**A multiple of the root element. That means that each rem multiples its value by the default built into either you're code or your browser. Because different browsers and devices have different defaults, a REM allows you're content to automatically adjust and keep its shape.

### **Why Choose One Over The Other:**

When we say that pixels are an absolute unit of measurement, that doesn't mean that 1px is always the same size in each environment. Think about what happens when you hook your computer up to a projector. You have the same number of pixels on each display, but a pixel on the projector takes up more physical size than that on your laptop. What we mean instead is that a pixel does not change size based on any other measurements on the page.

This is not the case for ems and rems, which are relative units. Hold on. We'll explain what that means.

Now, about **ems**. Let's say you have a div with a stated font-size of 10px and let's say that inside that div you have a paragraph. If you set that paragraph's font-size to 2rem, it multiplies by the parent font-size of 10px, giving the paragraph an equivalent size of 20px. Set the paragraph font-size to 1.6em and the equivalent size of 16px, and so on.

The **rem** unit of measurement stands for "root em." While an em is a multiple of the font-size of the parent, a rem is a multiple of the root element, which is the html element that wraps all of our html.  
  
Rems are the newest of the three, so some older browsers don't support rems. This means if you have to support super old browsers like IE7 or IE8, you have to choose whether to use rems and provide pixel values as a fallback or to go without rems altogether. We think the first option is the way to go, but it does add some extra work for you.

**Some links for farther reading:**

* [Comprehensive Guide To Rems And Ems](http://webdesign.tutsplus.com/tutorials/comprehensive-guide-when-to-use-em-vs-rem--cms-23984)
* [What's The Deal With Rems And Ems](https://codemyviews.com/blog/whats-the-deal-with-em-and-rem)
* [Confused About Rems And Ems](https://css-tricks.com/confused-rem-em/)
* [Confused About Rems And Em?](https://j.eremy.net/confused-about-rem-and-em/)

# Divide And Conquer The Rems

[Week 3: Build a Responsive Website for iPhone and Android](https://onemonth.com/courses/html#learn-responsive-web-design)· Lesson 10 ·

* [Lesson Notes](https://onemonth.com/courses/html/steps/divide-and-conquer-rems?autoplay=1#step-notes)
* [Discussion (4)](https://onemonth.com/courses/html/steps/divide-and-conquer-rems?autoplay=1#discussion)

Rems are useful for creating responsive typography. To use them effectively, you’ll need to know the guiding principles for calculating rems. In this lesson, I’ll show you how to calculate rem sizes and directs you to an online tool that'll make your life easier when it comes to coding with rems.

Rems are especially useful to adjust typography. However, it does require some math in order to achieve an exact size you (or your designer, or your team) may have in mind for a header, or a body font. If you have a target font, you need to divide it by the default. So for instance, to reach size 50 in a header in Chrome, you'd need to divide 50 by 16, and then specify 3.125rem.

If you need any help, access this handy-dandy [REM Calculator](https://offroadcode.com/prototypes/rem-calculator/).

# Rem Review And Get Ready

[Week 3: Build a Responsive Website for iPhone and Android](https://onemonth.com/courses/html#learn-responsive-web-design)· Lesson 11 ·

* [Lesson Notes](https://onemonth.com/courses/html/steps/rem-review?autoplay=1#step-notes)
* [Discussion (4)](https://onemonth.com/courses/html/steps/rem-review?autoplay=1#discussion)

Here, we review some of the ways you could have adjusted your code to include rems, as well as how to set the background image for your desktop view.

Again, there's no one right answer to how your code should look, but if you'd like to see an example of how to incorporate rems and the background image, behold:

/\*

\* Aside Styles

\*/

aside.left {

float: left;

}

aside.right {

float: right;

}

aside {

margin-right: 1.5rem;

margin-bottom: 0.625rem;

}

And if you were wondering how to make the background image stretch, you can lay it out like this:

/\*

\* General Styles

\*/

body {

background: url('../images/space-background.png');

font-family: 'Open Sans', arial, sans-serif;

/\* Full Page Background Receipe \*/

background-size: cover;

background-position: center;

background-attachment: fixed;

}

\* {

-moz-box-sizing: border-box; /\* Firexfox \*/

-webkit-box-sizing: border-box; /\* Safari/Chrome/iOS/Android \*/

box-sizing: border-box; /\* IE \*/

}

.clearfix:after {

content: "";

display: table;

clear: both;

}

# Week 3 Challenge!

[Week 3: Build a Responsive Website for iPhone and Android](https://onemonth.com/courses/html#learn-responsive-web-design)· Lesson 12 ·

* [Lesson Notes](https://onemonth.com/courses/html/steps/a-very-merry-live-birthday#step-notes)
* [Discussion (53)](https://onemonth.com/courses/html/steps/a-very-merry-live-birthday#discussion)

Congratulations! You made it to the end of the project AND the week. Now its time to create a responsive birthday site all on your own. Keep watching for the instructions!

# ****Create your own Birthday Site!****

Your homework is to create a birthday website of your own which meets the following criteria:

**1. Use nth-child for the background colors on the sections.** (Use Google for some hints on this, or check out the [bonus video I made for nth-child](https://onemonth.com/courses/html/steps/css3-nth-child-for-zebra-stripes) in the bonus section - or just ask me and I'll help you!).   
  
**2. Add a section at the bottom** as a footer letting your friend know that you coded this yourself using One Month! Like - "I coded this myself for you" (and if you like One Month feel free to put a little link to this class <a href="www.onemonth.com" target="\_blank">One Month</a>")  
  
**3. Solve this problem**: When there is only the unicorn image in the <section> the layout breaks. Why?  
  
**4. View the site on at least one mobile device**. What's working? What might still be improved? Ask questions and be curious.