Lecture 1: Course Overview

What We'll Be Building

Throughout this course we're going to learn HTML and CSS by building out the One Million Lines site. In the process of making the site you will learn everything you need to know about HTML and CSS so that you can create any site you want.

I think you are going to be surprised at how easy it is to build great looking sites fast and hopefully you have flashes of feeling like you have a new super power.

Download Resources

- Sublime Text
- Google Chrome Browser
- Bootstrap

Lesson 3: Tips For Course

Use these suggestions to crush this course and be building awesome sites in no time.

1. Go all the way through the course the first time following along **typing in the code yourself**.
2. After the first time, then go to create your own site - any type of site it doesn't have to be the same style and structure as the One Million Lines site - and use these videos, source codes, and other learning materials provided in the lessons as guides
3. **QUANTITY over quality.** What I mean by that is to not worry about creating the perfect site, with the perfect code and don't worry about messing up. Instead just focus on creating sites as fast as you can. In the process of pumping out sites you'll find yourself doing the same actions such as adding images or creating 3 columns as well as many others and they'll become second nature to you over the time.
4. **Practice in shorts bursts.** I don't recommend you going through this course in its entirety in one sitting. Instead spread out the learning over a couple days with hour long sessions to keep you excited and wanting more. During those hours sessions create breaks and break up the 1-2 hours into short bursts. Use e.ggtimer to set a timer and practice in 20 minute bursts with 5 minute breaks afterwards.
5. I can't stress this enough, but **repetition, repetition, repetition.** Keep creating, keep building sites and it will be second-nature in no time

6. Code snippets will be included in lessons so you can easily copy and paste or type the code. If you're ever having trouble getting the code to work or look the same in your browser as it does in the lesson you can always find the zipped source code for that file at the bottom of each lessons page. Click on the link to download, unzip the file, and you will have all the files as they should be at the end of that video.

Lesson 4: Hello World

Let's dive into bootstrap and get your first site up.

1. Download Bootstrap here. Make sure to choose Download Bootstrap option.

2. Unzip the bootstrap folder and rename "dist" folder as "first-site" folder

NOTE: IF you are a PC user when you zip/extract the Bootstrap zip file it will take you to a folder titled `bootstrap-3.1.1-dist` or with a number similar to that - 3.1.1 is the Bootstrap version number. Take that file and rename it `first-site`. Then copy the file and paste it on your desktop. Once you've done that open the `first-site` folder in Sublime as directed in the next step.

3. Open Bootstrap in sublime

On a mac you can grab the first-site folder and drag it over Sublime app icon (if the Sublime app icon is in your bottom app doc) to open. Otherwise open Sublime and click File at the top, then Open Folder or Open if on Mac. Locate the first-site folder, highlight it and then click open.

4. Create and save an index file

Right click on the "first-site" folder and choose new file. Highlight the code below (or on Bootstrap site) and insert it into the document. Save the file and name it "index.html".

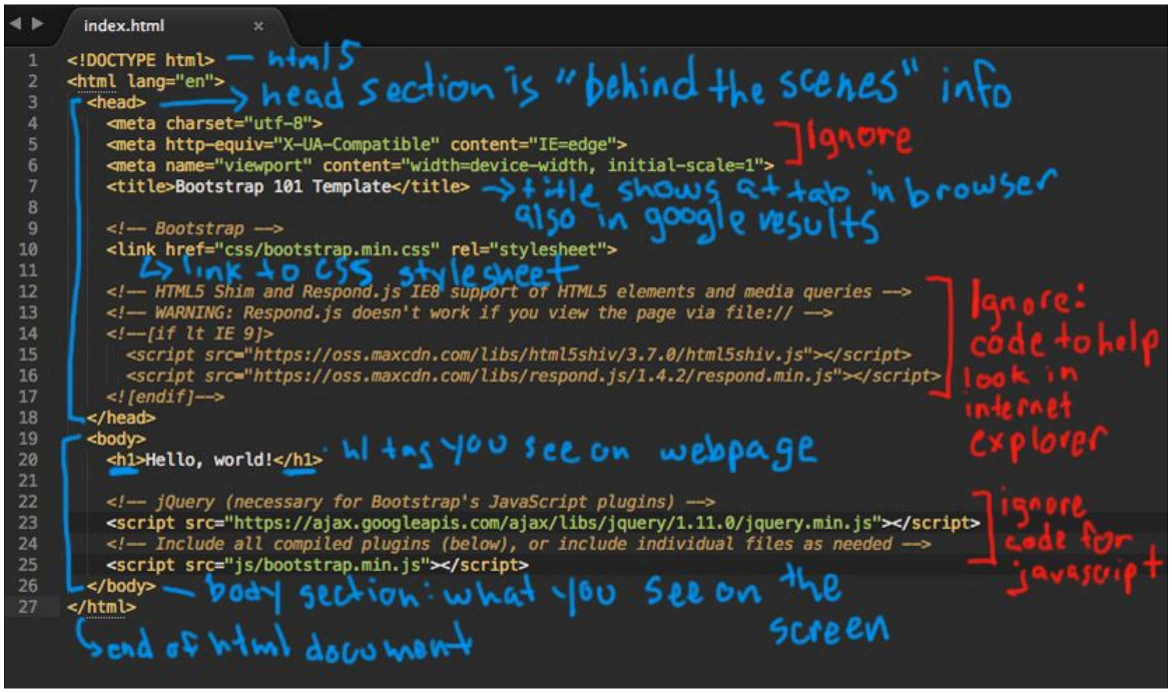
5. Open the index.html file in google chrome to view your first ever web site!

Find your file (in finder for mac, in your folders for pc) and right click on it selecting "open with" > "Google Chrome". Nerdy tidbit: Hello World is common practice among coders to do as their first program or output in any new coding language. Welcome to our world!

```html
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>Bootstrap 101 Template</title>

    <!-- Bootstrap -->
    <link href="css/bootstrap.min.css" rel="stylesheet">

    <!-- HTML5 Shim and Respond.js IE8 support of HTML5 elements and media queries -->
    <!-- WARNING: Respond.js doesn't work if you view the page via file:// -->
    <!--[if lt IE 9]>
      <script src="https://oss.maxcdn.com/libs/html5shiv/3.7.0/html5shiv.js"></script>
      <script src="https://oss.maxcdn.com/libs/respond.js/1.4.2/respond.min.js"></script>
    <![endif]-->
  </head>
  <body>
    <h1>Hello, world!</h1>

    <!-- jQuery (necessary for Bootstrap's JavaScript plugins) -->
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.0/jquery.min.js"></script>
    <!-- Include all compiled plugins (below), or include individual files as needed -->
    <script src="js/bootstrap.min.js"></script>
  </body>
</html>
```

Lesson 5: An HTML Document

Become familiar with the different types of tags throughout this course and how to open and close each. Below is a one-pager for you to refer back to throughout the course. Also use W3Schools.com as a reference throughout the course.


index.html
— html 5
head section is "behind the scenes" info
Ignore
→ title shows at tab in browser also in google results
↳ link to css stylesheet
Ignore: code to help look in internet explorer
h1 tag you see on webpage
ignore code for javascript
body section: what you see on the screen
↳ end of html document

```html
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>Bootstrap 101 Template</title>

    <!-- Bootstrap -->
    <link href="css/bootstrap.min.css" rel="stylesheet">

    <!-- HTML5 Shim and Respond.js IE8 support of HTML5 elements and media queries -->
    <!-- WARNING: Respond.js doesn't work if you view the page via file:// -->
    <!--[if lt IE 9]>
      <script src="https://oss.maxcdn.com/libs/html5shiv/3.7.0/html5shiv.js"></script>
      <script src="https://oss.maxcdn.com/libs/respond.js/1.4.2/respond.min.js"></script>
    <![endif]-->
  </head>
  <body>
    <h1>Hello, world!</h1>

    <!-- jQuery (necessary for Bootstrap's JavaScript plugins) -->
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.0/jquery.min.js"></script>
    <!-- Include all compiled plugins (below), or include individual files as needed -->
    <script src="js/bootstrap.min.js"></script>
  </body>
</html>
```

Lesson 7: Let’s Bootstrap!

Bootstrap offers a handful of templates for you to jumpstart your site's creation process. We'll learn how to grab any of these templates we want.

1. Go to Bootstrap's getting started section and grab the Jumbotron template at 0:35

After you click to open the Jumbotron template then right click on the page and select "view page source". Copy the entire HTML document and replace all the code in your "index.html" file currently with the copied code. Save the new index.html file.

2. Edit the CSS path so that it is correct

index.html

change

```
<link href="../../dist/css/bootstrap.min.css" rel="stylesheet">
```

to

```
<link href="css/bootstrap.min.css" rel="stylesheet">
```

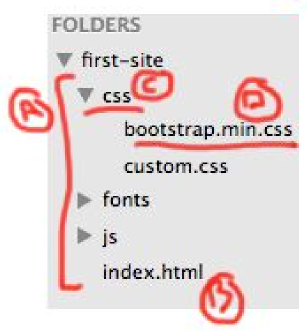
Link Notes

We're creating the link so that it is relative to the current file. In other words we're telling the browser how to get to (or the path to) the correct CSS document from the index.html file. So "css/bootstrap.min.css" is telling the browser to look for the "css" folder that is in the current folder - which is "first-site" - then go into it and find the "bootstrap.min.css" file.

LINKS / HREF

We're in index.html and we want to link to bootstrap.min.css

```
<link href="css/bootstrap.min.css" rel="stylesheet">
```



① This tells the browser to look for the "css" folder in the same directory (in this case the first-site folder Ⓐ) as the current file. (Ⓑ index.html)

② Once "css" folder is found go into it. Ⓒ

③ Then find "bootstrap.min.css" Ⓓ

3. Include our own custom.css file

View page source on the jumbotron template page if you don't still have it open. Then click on the jumbotron.css link. Copy all of the text in the document. Go to sublime and right click on the CSS folder. Click on New File, paste all the text in the document and save it as"*custom.css*" in the CSS folder. Then change the current "*jumbotron.css*" CSS link in our index.html file to our newly created "*custom.css*"file.

CONGRATS! You just created your first CSS file

Extra Lesson Notes

1. In this lesson we are linking to the CSS file via a relative path from the file we are currently in (index.html). In a future lesson we'll discuss how to do absolute links like to http://www.google.com.
2. Comments in HTML documents (or any coding documents for that matter) are for the human eye only. The code that indicates the beginning and the ending of a comment tells the browser just to skip over that section and not worry about the code in there. Comments begin with "<!--" and end with "-->". For example: `<!-- This is where you put the comment -->`
3. You've also seen CSS comments in this video. Check out the first line of your custom.css file where it says: `/* Move down content because we have a fixed... */`. Comments in CSS start with "/*" and end with "*/".
4. Make sure to include the custom.css file lower in our "*index.html*" file. This way the browser reads the custom.css file second after bootstrap's CSS file. Doing it this way makes sure that the CSS rules we include in our "*custom.css*" file will take precedence over the same ones in bootstraps's "*bootstrap.min.css*" file. Don't worry if you don't understand this right now. You will see it in action soon and it will soon become second-nature.
5. **Bonus**** You don't need to know this, but is an interesting tidbit. A "min" file is a minimized version of a file. So instead of linking to a .css file in this lesson we are linking to a .min.css file. Browsers

have to read the entire CSS file to interpret it. What a minimized version of a file does is take out all the unnecessary spaces and line breaks so that the browser can read the file faster and thus your site loads faster. Every extra line and space in a document makes it longer to for the browser to interpret it. Minimized files eliminate these unnecessary wastes.

Lesson 8: Editing the Navbar

In this lesson we'll explore how to combine templates, use Chrome's dev tools, and utilize Bootstrap's native components to start putting together the site fast.

1. Grab the Bootstrap's Starter Template navbar

Copy the navbar div (as seen in video) of the Starter Template and replace the current navbar div in our "*index.html*" with it.

index.html

change

```
<div class="navbar navbar-inverse navbar-fixed-top" role="navigation">
  <div class="container">
    <div class="navbar-header">
      <button type="button" class="navbar-toggle" data-toggle="collapse"
data-target=".navbar-collapse">
        <span class="sr-only">Toggle navigation</span>
        <span class="icon-bar"></span>
        <span class="icon-bar"></span>
        <span class="icon-bar"></span>
      </button>
      <a class="navbar-brand" href="#">Project name</a>
    </div>
    <div class="navbar-collapse collapse">
      <form class="navbar-form navbar-right" role="form">
        <div class="form-group">
          <input type="text" placeholder="Email" class="form-control">
        </div>
        <div class="form-group">
          <input type="password" placeholder="Password" class="form-control">
        </div>
        <button type="submit" class="btn btn-success">Sign in</button>
      </form>
    </div><!--/.navbar-collapse -->
  </div>
</div>
```

to

```
<div class="navbar navbar-inverse navbar-fixed-top" role="navigation">
  <div class="container">
    <div class="navbar-header">
      <button type="button" class="navbar-toggle" data-toggle="collapse" 
data-target=".navbar-collapse">
        <span class="sr-only">Toggle navigation</span>
        <span class="icon-bar"></span>
        <span class="icon-bar"></span>
        <span class="icon-bar"></span>
      </button>
      <a class="navbar-brand" href="#">Project name</a>
    </div>
    <div class="collapse navbar-collapse">
      <ul class="nav navbar-nav">
        <li class="active"><a href="#">Home</a></li>
        <li><a href="#about">About</a></li>
        <li><a href="#contact">Contact</a></li>
      </ul>
    </div><!--/.nav-collapse -->
  </div>
</div>
```

2. Change the navbar-brand to be One Million Lines

index.html

change

```
<a class="navbar-brand" href="#">Project name</a>
```

to

```
<a class="navbar-brand" href="#">One Million Lines</a>
```

3. Move the links in the Navbar to the right

In the bootstrap components we find that there is a "navbar-right" class to do just that. Add the "navbar-right" class to our "ul" element.

index.html

```
<ul class="nav navbar-nav navbar-right">
```

4. Change the links in the navbar

index.html

change

```
<ul class="nav navbar-nav navbar-right">
  <li class="active"><a href="#">Home</a></li>
  <li><a href="#about">About</a></li>
  <li><a href="#contact">Contact</a></li>
</ul>
```

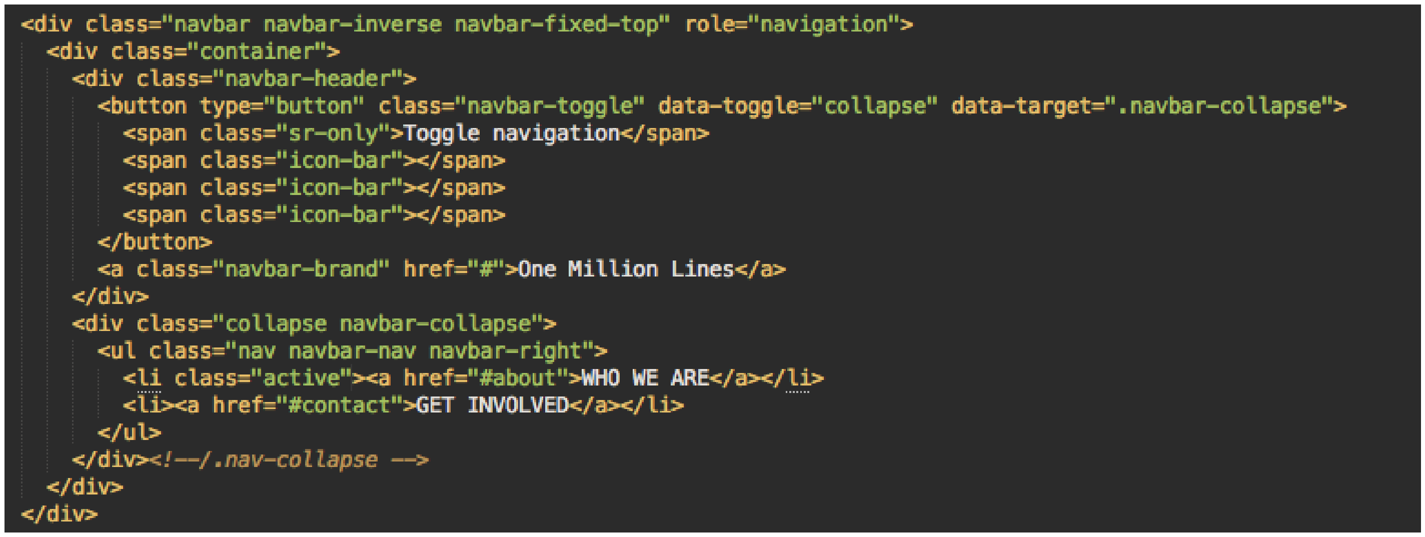
to

```
<ul class="nav navbar-nav navbar-right">
  <li class=”active”><a href="#about">WHO WE ARE</a></li>
  <li><a href="#contact">GET INVOLVED</a></li>
</ul>
```

NOTE: We'll worry about the href in the above code later in the course

Extra Lesson Notes

1. **Child and parent elements.** An HTML document’s structure can be looked like a family tree. Each element has a parent element. An element can have a child (or children) element if it contains other HTML elements. Indentation within a document does not determine whether an element is a child or a parent element. Instead the indentation just keeps the document organized so we can see it easier visually. In the example below the div with class navbar is the parent element of the div with class of container. The *ul* is a child element of the div with class collapse and navbar-collapse. At the same time the *ul* is the parent element of the two *li* HTML elements



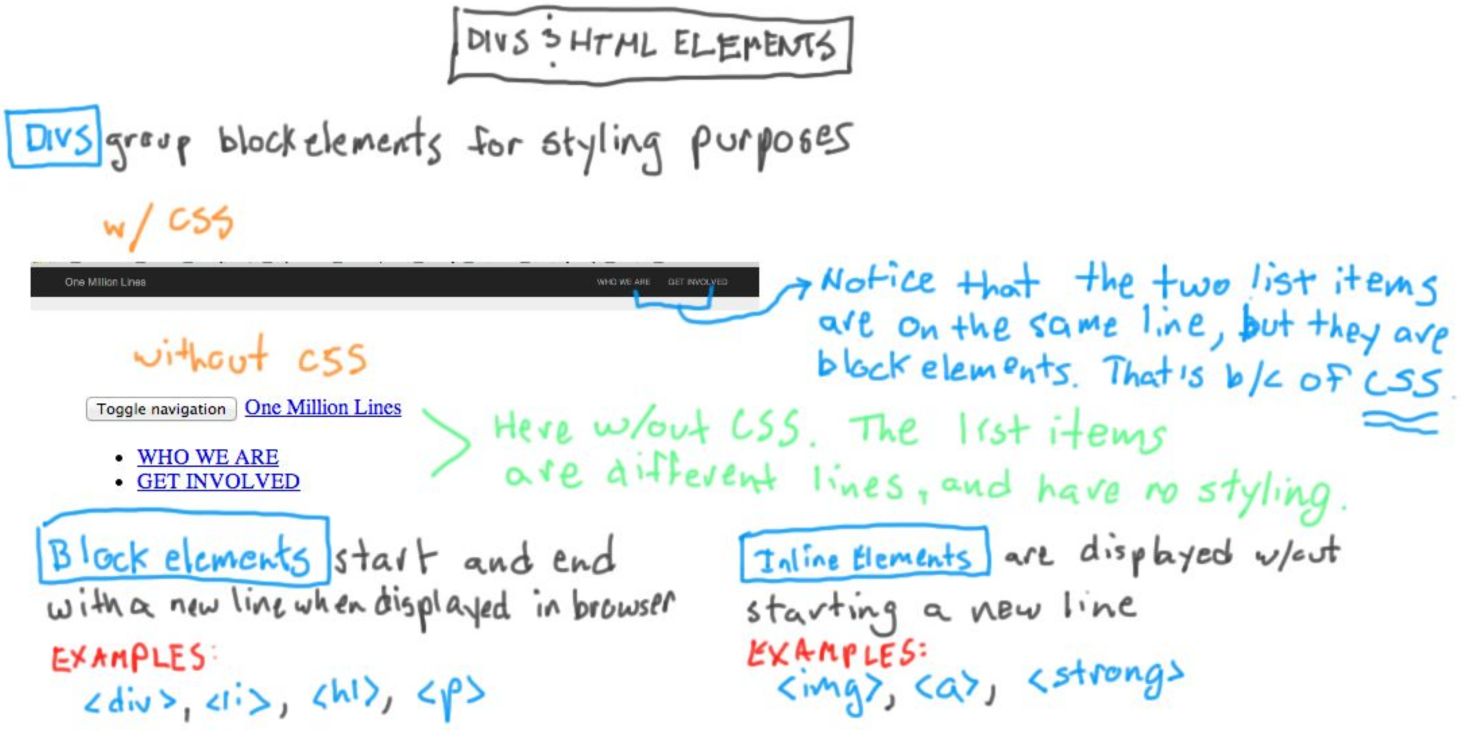
```
<div class="navbar navbar-inverse navbar-fixed-top" role="navigation">
  <div class="container">
    <div class="navbar-header">
      <button type="button" class="navbar-toggle" data-toggle="collapse" data-target=".navbar-collapse">
        <span class="sr-only">Toggle navigation</span>
        <span class="icon-bar"></span>
        <span class="icon-bar"></span>
        <span class="icon-bar"></span>
      </button>
      <a class="navbar-brand" href="#">One Million Lines</a>
    </div>
    <div class="collapse navbar-collapse">
      <ul class="nav navbar-nav navbar-right">
        <li class="active"><a href="#about">WHO WE ARE</a></li>
        <li><a href="#contact">GET INVOLVED</a></li>
      </ul>
    </div><!--/.nav-collapse -->
  </div>
</div>
```

2. The *Inspect Element* feature of Google Chrome is an awesome tool to experiment and use as your sandbox to try new things and learn about how changes in CSS and HTML will look live on your site. You'll become much more familiar with this tool throughout the course.

Lesson 9: What the Div?!

Divs are all over HTML documents. Understand divs and you'll be like an HTML magician. Feel the power!

Lesson Notes



DIVS & HTML ELEMENTS

Divs group block elements for styling purposes

w/ CSS

One Million Lines WHO WE ARE GET INVOLVED

Notice that the two list items are on the same line, but they are block elements. That is b/c of CSS

without CSS

Toggle navigation One Million Lines

- WHO WE ARE
- GET INVOLVED

Here w/out CSS. The list items are different lines, and have no styling.

Block elements start and end with a new line when displayed in browser

EXAMPLES:
<div>, <li>, <h1>, <p>

Inline Elements are displayed w/out starting a new line

EXAMPLES:
<img>, <a>, <strong>

- A div defines a division or section in an HTML document

- A div is used to group block elements so that you can format them with CSS. Their main benefits are organization and for styling purposes. As you can see in the image above of the difference between an unordered list with and without CSS
- The class *.container* keeps all of our site content lined up within a certain width
- Divs are also block elements, which means that there is a line-break before and after the element by default. You can change this through CSS as Bootstrap did for us in the Navbar.
- Unordered lists - `<ul>` - create bulleted lists, ordered lists - `<ol>` - create numbered lists. Use `<li>` for each list item in both types of lists. Learning to style these lists as Bootstrap does for us will be for a future lesson. `<ul>`, `<ol>`, and `<li>` are all block elements as you can see when we took away the CSS.
- Block elements start and end with a line-break. Inline elements are displayed without starting a new line - an example we've seen so far is the `<a>` tag. This is why you can have the `<li>` and `<a>` tags together in the code and they're on the same line. Also in the example you see the button and the link are on the same line when CSS is not present. This is because they are both inline elements.

Lesson 10: Your First CSS Rule

CSS controls the style of your site. Make your first CSS rule and start transforming the site from boring text to sexy startup quality.

1. Update the h1 tag

index.html

change

`<h1>Hello, world!</h1>`

to

`<h1>Our goal is to inspire <br>Tallahassee to write 1,000,000<br> lines of code_</h1>`

NOTE: `<br>` is a line break in HTML. Notice on the web page that after each `<br>` the text goes to a new line

2. Update the paragraph

index.html

change

```
<p>This is a template for a simple marketing or informational website. It includes a
large callout called a jumbotron and three supporting pieces of content. Use it as a
starting point to create something more unique.</p>
```

to

```
<p>All over the country people are taking the <strong>HOUR OF CODE</strong> challenge
issued by <strong>CODE.org</strong>. Millions of lines of code are being written. In
the capital of Florida, Tallahassee, the community is taking the challenge and our
goal is to write 1,000,000 lines of code_</p>
```

NOTE: `<strong>` boldens the text that the opening and closing strong tags wrap.

3. Center all the text in the div with a class of "jumbotron"

css/custom.css

```
.jumbotron {
  text-align: center;
}
```

NOTE: The above code translates to "for any div with a class 'jumbotron' align all text in the center." This is your first CSS rule. Make sure you follow the syntax exactly.

About the inspect element feature: When you have the proper HTML element selected you can type CSS declarations for that HTML element on the right in the element.style section. For example, in this part of the video I have selected the div with a class of jumbotron so whatever CSS declarations I put in the element.style section will apply to that div only. To reset the style just reload the page. This feature is your sandbox to have fun!

4. Make the button red and change its text

Here we take advantage of Bootstrap's prewritten CSS classes for buttons

index.html

change

```
<a class="btn btn-primary btn-lg" role="button">Learn more »</a>
```

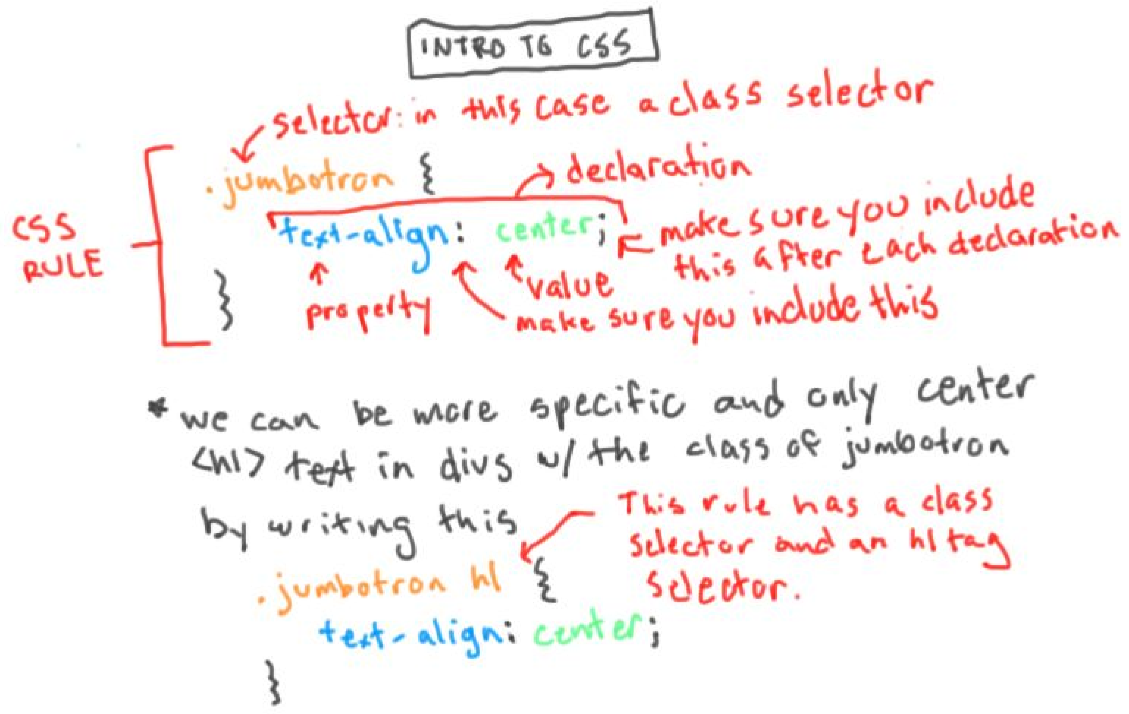
to

```
<a class="btn btn-danger btn-lg" role="button">Get Involved</a>
```

Lesson Notes

- In addition to `<br>` and `<strong>` an opening and closing `<em>` tag makes the text it surrounds display in italics.

- Giving an HTML element a *class* is just a naming convention so that we can know what to refer to it as and assign it a style in CSS. There is also an *id* naming convention which we will cover later including when to use which one (id or class).
- You denote a class in CSS by putting a period in front of the class name. For the example, in the video we defined styling for the jumbotron class in our CSS document by typing `.jumbotron`
- Look at the image below to learn more about CSS syntax and the different parts of a rule


INTRO TO CSS
selector: in this case a class selector
CSS RULE
.jumbotron {
declaration
text-align: center;
make sure you include this after each declaration
property
value
make sure you include this
}
* we can be more specific and only center <h1> text in divs w/ the class of jumbotron by writing this
This rule has a class selector and an h1 tag selector.
.jumbotron h1 {
text-align: center;
}

Lesson 11: More CSS Fun

Lets take a quick moment to learn more about CSS

1. Make the button bigger

Here we take advantage of Bootstrap's prewritten CSS classes for buttons

css/custom.css

```
.btn-lg {
  font-size: 36px;
}
```

NOTE: After you change this above, inspect the button as I did in the video. Notice on the right how there is a `.btn-lg` CSS rule from the "jumbotron.min.css" document that makes a declaration `font-size: 18px`. Also notice how that declaration is crossed out and the CSS rule you made in "custom.css" is above it and the declaration `.btn-lg { font-size: 36px; }` is not crossed out. This is because we included the custom.css file lower in our HTML document than the bootstrap.min.css file. So now when any declarations or rules conflict between Bootstrap's and our custom.css file the one in our custom.css file takes priority.

Lesson Notes

- 3 ways to include CSS
    - External - how we've been doing with custom.css
    - Internal - in the header as shown in the video
    - Inline - using the style attribute within the element's opening tag
- External and internal style sheets have the same priority (assuming the rules have the exact same HTML selectors) so whichever is read last (in other words included in the head section further down in the document) will be the CSS rule that takes place. This is why we are able to overwrite Bootstrap's CSS with our own custom.css - because we include the custom.css lower in the HTML document.
- So the above bullet as to do with the *cascading* part of CSS. Now let's talk about *specificity.* If there are two competing style rules for an element the CSS rule that had the more specific selectors will take priority. In other words if one rule says all text in a div with a class of jumbotron should be red, but then there is another rule that says all text of any p that is in a div with a class of jumbotron should be blue. Which color will the p text be? Well which is more specific? In the first rule it applies to any HTML element in a div with that class, but in the second rule it is limiting it only p's in a div with that class. Thus the second rule is more specific so the p text will be blue and other html elements their text will be red.
- Inline styling is the most specific styling and will take highest priority so that the inline style rule will occur. This is because when you style an individual HTML element you can't get more specific than that because you a declaring a CSS rule for that HTML element alone and no other.
- Use External Stylesheets so that you can make small CSS changes across your site quickly
- In this lesson we made a CSS rule with the font-size property. You can learn more about the font-size property and other CSS font properties here. I like to use pixels when I refer to font-size because it gives the browser an exact size, but some developers use *em* or *%* which are relative measurements.

Lesson 12: Get Your Font On

If you've ever heard of Steve Jobs you know that fonts are critical in the design of a product. Let's learn to put any font you want into your website.

1. Find and include the Arvo font

You can find the Arvo font at the Google Fonts page here.

index.html

include

```
<link href='http://fonts.googleapis.com/css?family=Arvo' rel='stylesheet' 
type='text/css'>
```

2. Make CSS rule for ul.nav and .jumbotron h1 to have Arvo font

css/custom.css

```
ul.nav, .jumbotron h1 {
  font-family: ‘Arvo’, courier, serif;
}
```

This will change the font-family of an ul with a class of nav and any h1 in a div with the class "jumbotron".

NOTE: You could include multiple font-family names after Arvo for fallbacks (as we did here) in case Arvo doesn't load correctly from Google. Learn more about how to do that here.

Lesson Notes

- So we’ve seen 2 new types of selectors in CSS in this lesson. Last lesson we did `.jumbotron` to define styling in any HTML element with a class of jumbotron.
    - However in this less we used `.jumbotron h1` to style only h1 tags in our div with the jumbotron class. So this is saying any HTML element with a class of jumbotron that has an h1 within it (another way to say would be that has an h1 child element) style those h1’s this way.
    - Then we also used `ul.nav` in this lesson. This says any *ul* that has a class of *nav* style it this way. So you may be asking the question well we didn’t put div before *.jumbotron*. We could have. So we could have said `div.jumbotron` and it would be referring to only *divs* with a class of *jumbotron*, but since in our HTML we have only given a class of *jumbotron* to divs and we haven’t given any *uls* or any other type of HTML element the class of *jumbotron* then it doesn’t matter.
    - So notice the difference in the two naming conventions of the above examples with the selectors. A selector with a class and then an HTML tag (ex. `.jumbotron h1`) refers to the

styling of the h1 child elements of *divs* (or any HTML element for that matter since div wasn't specified) with a class of jumbotron. In contrast to an HTML tag followed with a class in a selector refers to only HTML elements that are the tag mentioned and has the class mentioned, for example `p.lead` refers only to paragraphs with a class of lead (the HTML would look like this `<p class="lead">content</p>`).

- Also we've seen in this video that you can combine multiple selectors in one CSS rule. To do this all you have to do is separate the additional selectors with a comma and a space - for example: `ul.nav, .jumbotron h1 { CSS rule content here }`.

Lesson 14: The Box Model

Every html element is a rectangular box. Understand this along with what margin, padding, and border and you're on your way to being a website ninja

1. Change the list items `<li>` in the unordered list `<ul>` to be smaller

css/custom.css

```
ul.nav {
    font-size: 13px;
  }
```

2. Include correct margin for the .jumbotron `<h1>` and `<p>`

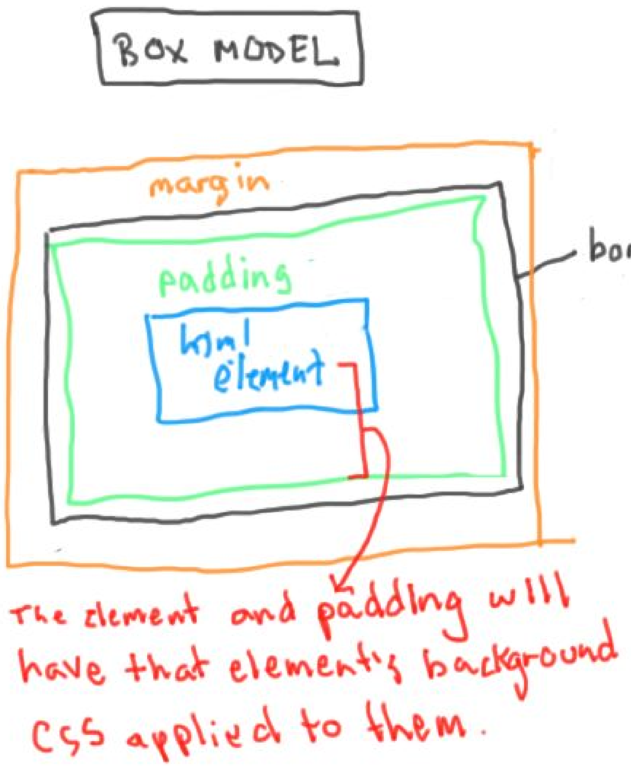
css/custom.css

```
.jumbotron h1 {
    margin-top: 0px;
    margin-bottom: 50px;
  }

  .jumbotron p {
    margin-bottom: 30px;
  }
```

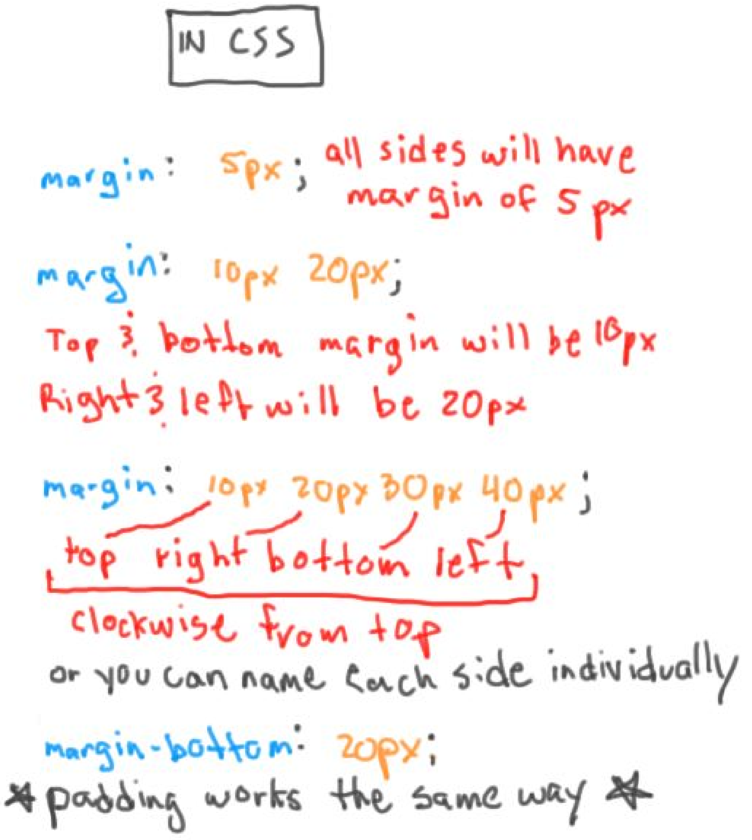
Lesson Notes

- Every HTML element is in the shape of a box and has the box model applied to it. Inspect elements on the page and highlight various HTML elements in order to see this in action.

- When inspecting HTML elements in Google Chrome blue is the HTML element itself, green is padding, and orange is the margin.
- The HTML element and the padding is within the border. Margin is outside the border. Use padding if you want to add spacing within the border; in other words if you want to expand the background color of that particular element. Use margin if you want to add spacing outside of the border. With margin the spacing will have the same background as that particular elements parent HTML element.
- You can visualize what an element's parent is by indentation in the inspect element area. In our video example the `<body>` element is the parent of the `<div class="jumbotron">` element. See how the `<div class="jumbotron>` is indented one tab in and under the`<body>` element. Then when you click on the arrow to the left of the `<div class="jumbotron>` to open it you'll see`<div class="container>` appear which is the child element of the `<div class="jumbotron>` element.
- There are multiple ways to declare the padding and margin for an HTML element. We went over them in the video and you can use the image below to as a reference.



The element and padding will have that element's background CSS applied to them.


IN CSS

margin: 5px; all sides will have margin of 5 px

margin: 10px 20px;
Top & bottom margin will be 10px
Right & left will be 20px

margin: 10px 20px 30px 40px;
top right bottom left
clockwise from top
or you can name each side individually

margin-bottom: 20px;
* padding works the same way *

Lesson 15: Bringing In The Images

Lets go through how to add images to your site so that it really pops, both through HTML and CSS. And now we'll be done with the top section. Pat yo' self on the back!

1. Include the Code.org image

You can find the images in the lesson description/download tab to the right. Download the images. Then create a new folder within your "first-site" folder titled "images". Save the images in that folder.

index.html

include

```
<img src="images/code.png" title="Code.org logo" alt="Code.org">
```

on the line after the button

The `alt` attribute is what will show if the image loads incorrectly and the `title` attribute is what will show if you hover over the image long enough with your mouse. These are optional, but can help your site's accessibility say if the image doesn't load correctly or if someone who is visually impaired comes to your site - the screen reader will read what image should be there since they can't see it. **Every HTML element has a set of HTML attributes that go along with it. You can always find them for each element at w3schools.com.**

2. Insert background image

You can get image in the download/lesson description tab. Save it as hero_image.jpg.

css/custom.css

```
.jumbotron {
  text-align: center;
  background-image: url("../images/hero_image.jpg");
}
```

- We were introduced to links earlier. The only thing new we're adding here is `../` which tells the browser to exit the current folder. In other words, we're in the custom.css file so we're in the CSS folder. We tell the browser to exit the CSS folder so then its in the first-site folder. From there we tell it to find the images folder `images`. To enter the images folder `/` and then to find the `hero_image.jpg` image.
- In this lesson we've use the background image. There are also other background properties you can learn about here.

3. Change font color

css/custom.css

```
.jumbotron {
  text-align: center;
  background-image: url("../images/hero_image.jpg");
  color: white;
}
```

3. Correct font weight, font size and spacing (padding and margin)

css/custom.css

```
.jumbotron h1 {
  margin-top: 0px;
  margin-bottom: 50px;
}

.jumbotron p {
  margin-bottom: 30px;
  font-weight: 100;
  padding: 0 50px;
}
```

4. Add One Million Lines image

You can get image in the download/lesson description tab. Save it as top_logo.png in the images folder.

index.html

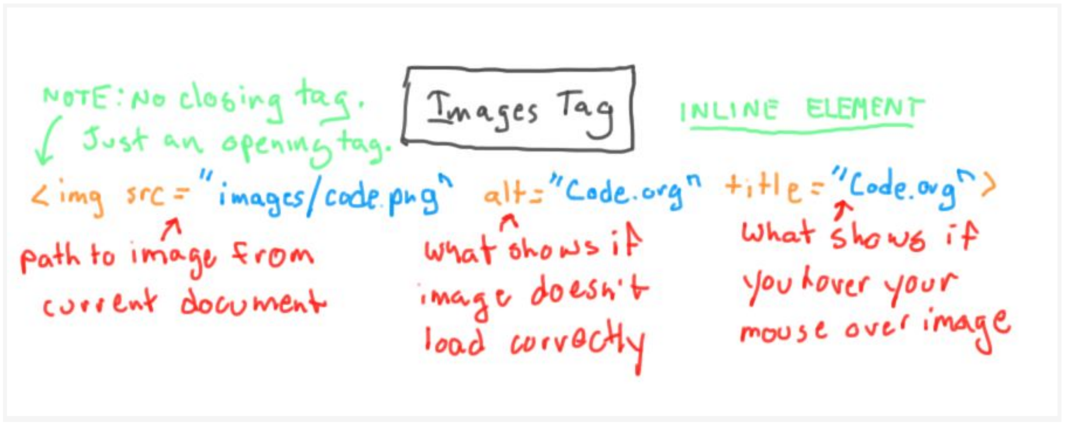
change

```
<a class="navbar-brand" href="#">One Million Lines</a>
```

to

```
<a class="navbar-brand" href="#"><img src="images/top_logo.png" alt="One Million 
Lines"></a>
```

Use the image below as a reference when creating new image tags.

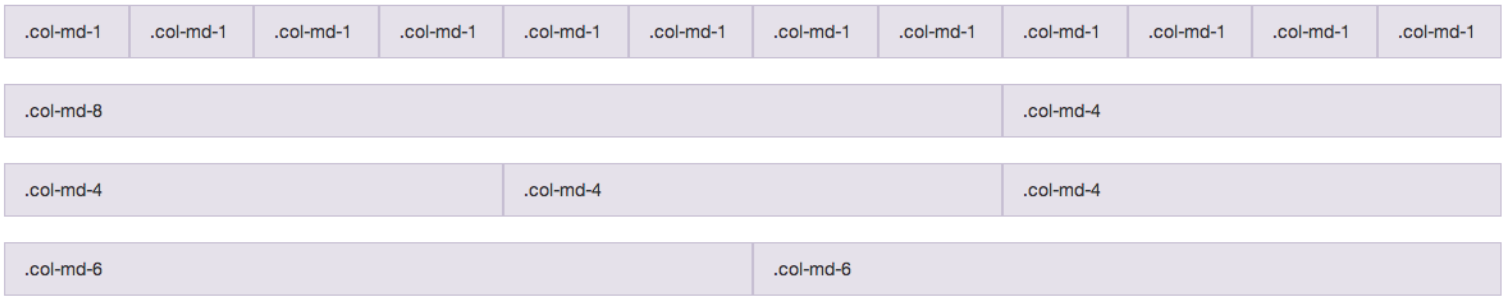

NOTE: No closing tag.
Just an opening tag.
Images Tag
INLINE ELEMENT
<img src="images/code.png" alt="Code.org" title="Code.org">
path to image from current document
what shows if image doesn't load correctly
What shows if you hover your mouse over image

Lesson 16: The Grid System

Lets go into bootstrap's grid system so you can learn to make the site look exactly however you want it. With the added understanding of floats you can build **any** complex-looking site.

Lesson Notes

- In `.col-md-4` the number represents the width, more specifically how many columns wide of the 12-column layout. In other words that div will be 4 columns wide. Since it is out of 12 it will be 1/3 of the page's width. Look at the image below for a quick cheat sheet on the grid system and div widths.



- **Rules**
  - Rows are placed w/in a container div
  - Only columns may be immediate children of rows

Lesson 17: Linking It Up

Use links to control the flow of users from and through your site. In this lesson we cover the different types of links

1. Update content

index.html

change

```
<div class="row">
  <div class="col-md-4">
    <h2>Heading</h2>
    <p>Donec id elit non mi porta gravida at eget metus. Fusce dapibus, tellus ac
cursus commodo, tortor mauris condimentum nibh, ut fermentum massa justo sit amet
risus. Etiam porta sem malesuada magna mollis euismod. Donec sed odio dui. </p>
    <p><a class="btn btn-default" href="#" role="button">View details &raquo;</a></p>
  </div>
  <div class="col-md-4">
    <h2>Heading</h2>
    <p>Donec id elit non mi porta gravida at eget metus. Fusce dapibus, tellus ac
cursus commodo, tortor mauris condimentum nibh, ut fermentum massa justo sit amet
risus. Etiam porta sem malesuada magna mollis euismod. Donec sed odio dui. </p>
    <p><a class="btn btn-default" href="#" role="button">View details &raquo;</a></p>
 </div>
  <div class="col-md-4">
    <h2>Heading</h2>
    <p>Donec sed odio dui. Cras justo odio, dapibus ac facilisis in, egestas eget
quam. Vestibulum id ligula porta felis euismod semper. Fusce dapibus, tellus ac
cursus commodo, tortor mauris condimentum nibh, ut fermentum massa justo sit amet
risus.</p>
    <p><a class="btn btn-default" href="#" role="button">View details &raquo;</a></p>
  </div>
</div>
```

to

```
<div class="row">
  <div class="col-md-4">
```

```
    <h2>Students</h2>
    <p>Want to learn how to code? Want to help us get to 1,000,000 lines?  Click the 
button below and we'll let you know how to get involved.</p> 
    <p><a class="btn btn-default" href="#" role="button">Start Learning 
&raquo;</a></p>
  </div>
  <div class="col-md-4">
    <h2>Educators</h2>
    <p>Want to bring this initiative to your school or institution?  Awesome!  Click 
the button below and we'll make it happen.</p> 
    <p><a class="btn btn-default" href="#" role="button">Join The Iniative 
&raquo;</a></p>
 </div>
  <div class="col-md-4">
    <h2>Sponsors</h2>
    <p>We love all the support we get to help host more events, and empower more 
lives with the knowledge of coding.  Click to get involved.</p> 
    <p><a class="btn btn-default" href="#" role="button">Give Support &raquo;</a></p> 
  </div>
</div>
```

<a> Notes

- Use the `href` attribute to determine the URL (aka the address) of what page or website the browser will go to when the user clicks on a link
- **Absolute link/URL:** Starting a href with `http://` will direct the user to the exact site destination specified afterwards. In other words I used `http://google.com` as an example in the lesson. Users will be directed to google.com when they click on that link.
- **Relative link/URL:** Take away `http://` and you can link to other pages within your site. The destination page is relative from the current page the link is in and you declare the path the same way we did earlier with images. In other words if you are currently in the `index.html` file and you want to create and link to another page in the "first-site" folder titled `grid.html` you would do `href="grid.html"`. If the grid.html page was in a folder called "about" you would do `href="about/grid.html"`.
- Use `target="_blank"` to open link in a new tab.
- **Added Bonus**: you can use `href="mailto:youremailaddress@domainname.com"` so that if someone clicks they will be set up to email you.

- You can also see more special HTML character symbols here.

Lesson 18: BONUS: Google Forms

Learn how to bring in Google Forms. An awesome, free tool to get feedback from you site users and put a sense of interactivity in your site

NOTE: You can also embed the form into your site so that you can style it to look like your site. We will cover this when we discuss embedding iframes into your site.

Lesson 19: Font Awesome Is Awesome

Font-awesome allows you to include professional quality icons in your site free and easy to give it that extra pop.

1. Download Font-awesome here. Click download.

2. Unzip the font-awesome.zip and rename the resulting unzipped folder as "font-awesome".

3. Take the font-awesome folder and drag or paste it into the fonts folder within your first-site folder

On a mac you can grab the font-awesome folder and drag it into the "first-site/fonts" folder as in video. With PCs you may have to highlight the font-awesome folder and copy it. Then find the "fonts" folder (wihin your "first-site" folder), highlight it, and then click paste and the font-awesome folder should now be within the fonts folder.

4. Include font-awesome in your index.html file

index.html

add

```
<link rel="stylesheet" href="fonts/font-awesome/css/font-awesome.min.css">
```

5. Put an icon in your index.html file to see if we included font-awesome correctly

index.html

add

`<i class="fa fa-camera-retro"></i>`

above `<h2>Students</h2>`

NOTE: If you have any trouble including it correctly check the source files to see how to include it correctly.

6. Once you have font-awesome included correctly put in correct icons

index.html

change

```
<div class="row">
  <div class="col-md-4">
    <h2>Students</h2>
    <p>Want to learn how to code? Want to help us get to 1,000,000 lines?  Click the 
button below and we'll let you know how to get involved.</p>
    <p><a class="btn btn-default" href="#" role="button">Start Learning 
&raquo;</a></p>
  </div>
  <div class="col-md-4">
    <h2>Educators</h2>
    <p>Want to bring this initiative to your school or institution?  Awesome!  Click 
the button below and we'll make it happen.</p>
    <p><a class="btn btn-default" href="#" role="button">Join The Iniative 
&raquo;</a></p>
  </div>
  <div class="col-md-4">
    <h2>Sponsors</h2>
    <p>We love all the support we get to help host more events, and empower more 
lives with the knowledge of coding.  Click to get involved.</p>
    <p><a class="btn btn-default" href="#" role="button">Give Support &raquo;</a></p>
  </div>
</div>
```

to

```
<div class="row">
  <div class="col-md-4">
    <span class="fa-stack fa-4x">
```

```
      <i class="fa fa-circle fa-stack-2x"></i>
      <i class="fa fa-user fa-stack-1x fa-inverse"></i>
    </span>
    <h2>Students</h2>
    <p>Want to learn how to code? Want to help us get to 1,000,000 lines?  Click the
button below and we'll let you know how to get involved.</p>
    <p><a class="btn btn-default" href="#" role="button">Start Learning
&raquo;</a></p>
  </div>
  <div class="col-md-4">
    <span class="fa-stack fa-4x">
      <i class="fa fa-circle fa-stack-2x"></i>
      <i class="fa fa-pencil fa-stack-1x fa-inverse"></i>
    </span>
    <h2>Educators</h2>
    <p>Want to bring this initiative to your school or institution?  Awesome!  Click
the button below and we'll make it happen.</p>
    <p><a class="btn btn-default" href="#" role="button">Join The Iniative
&raquo;</a></p>
  </div>
  <div class="col-md-4">
    <span class="fa-stack fa-4x">
      <i class="fa fa-circle fa-stack-2x"></i>
      <i class="fa fa-money fa-stack-1x fa-inverse"></i>
    </span>
    <h2>Sponsors</h2>
    <p>We love all the support we get to help host more events, and empower more
lives with the knowledge of coding.  Click to get involved.</p>
    <p><a class="btn btn-default" href="#" role="button">Give Support &raquo;</a></p>
  </div>
</div>
```

Lesson 20: Playing With Spans

Spans let you style inline-elements. Use it if you want to make certain words, phrases, or other inline elements stand out with a different color or style.

In this example we used span to change the font color and size, but you could use it to do a variety of other things with the font. Go ahead and play around with different font and text properties, combine them, and see what they do. You can find a list of different font properties here.

Lesson 21: Styling The Get Involved Section

Through working on the get-involved section we learn and revisit many text-based CSS properties as well as see how to give the same CSS rule to multiple different HTML elements.

1. Update Get Involved section's HTML content and give it a class

index.html

and change

```
<div class="container">
  <!-- Example row of columns -->
  <div class="row">
    <div class="col-md-4">
      <span class="fa-stack fa-4x">
        <i class="fa fa-circle fa-stack-2x"></i>
        <i class="fa fa-user fa-stack-1x fa-inverse"></i>
      </span>
      <h2>Students</h2>
      <p>Want to learn how to code? Want to help us get to 1,000,000 lines?  Click 
the button below and we'll let you know how to get involved.</p>
      <p><a class="btn btn-default" href="#" role="button">Start Learning »</a></p>
    </div>
    <div class="col-md-4">
      <span class="fa-stack fa-4x">
        <i class="fa fa-circle fa-stack-2x"></i>
        <i class="fa fa-pencil fa-stack-1x fa-inverse"></i>
      </span>
      <h2>Educators</h2>
      <p>Want to bring this initiative to your school or institution?  Awesome!
Click the button below and we'll make it happen.</p>
      <p><a class="btn btn-default" href="#" role="button">Join The Iniative
»</a></p>
    </div>
    <div class="col-md-4">
      <span class="fa-stack fa-4x">
        <i class="fa fa-circle fa-stack-2x"></i>
        <i class="fa fa-money fa-stack-1x fa-inverse"></i>
      </span>
      <h2>Sponsors</h2>
      <p>We love all the support we get to help host more events, and empower more
lives with the knowledge of coding.  Click to get involved.</p>
      <p><a class="btn btn-default" href="#" role="button">Give Support »</a></p>
    </div>
  </div>

  <hr>

  <footer>
    <p>© Company 2014</p>
  </footer>
</div> <!-- /container -->
```

to

```
<div class="container homepage">
  <h2>Get Involved</h2>
  <!-- Example row of columns -->
  <div class="row">
    <div class="col-md-4">
      <span class="fa-stack fa-4x">
        <i class="fa fa-circle fa-stack-2x"></i>
        <i class="fa fa-user fa-stack-1x fa-inverse"></i>
      </span>
      <h3>Students</h3>
      <p>Want to learn how to code? Want to help us get to 1,000,000 lines?  Click
the button below and we'll let you know how to get involved.</p>
```

```
      <p><a class="btn btn-default" href="#" role="button">Start Learning »</a></p>
    </div>
    <div class="col-md-4">
      <span class="fa-stack fa-4x">
        <i class="fa fa-circle fa-stack-2x"></i>
        <i class="fa fa-pencil fa-stack-1x fa-inverse"></i>
      </span>
      <h3>Educators</h3>
      <p>Want to bring this initiative to your school or institution?  Awesome! 
Click the button below and we'll make it happen.</p>
      <p><a class="btn btn-default" href="#" role="button">Join The Iniative 
»</a></p>
    </div>
    <div class="col-md-4">
      <span class="fa-stack fa-4x">
        <i class="fa fa-circle fa-stack-2x"></i>
        <i class="fa fa-money fa-stack-1x fa-inverse"></i>
      </span>
      <h3>Sponsors</h3>
      <p>We love all the support we get to help host more events, and empower more 
lives with the knowledge of coding.  Click to get involved.</p>
      <p><a class="btn btn-default" href="#" role="button">Give Support »</a></p>
    </div>
  </div>

  <hr>

  <footer>
    <p>© Company 2014</p>
  </footer>
</div> <!-- /container -->
```

2. Center align the font and style the h2 and h3 tags in the Get Involved section classes

css/custom.css

add

```
.homepage {
  text-align: center;
}

.homepage h2, .homepage h3 {
  font-family: ‘Arvo’, courier, serif;
  color: #e74c3c;
  text-transform: uppercase;
}

.homepage h2 {
  font-size: 30px;
}

.homepage h3 {
  font-size: 26px;
}

.homepage p {
  font-size: 21px;
  font-weight: 300;
}

.homepage a {
  color: #e74c3c;
}
```

Lesson Notes

- You can give HTML elements multiple classes by separating them with a space as we did in this video `<div class="container homepage">`. This div has 2 classes: homepage and container.
- Let's talk about **colors.** The color of your site is very important to the styling and feeling a user gets when visiting. In the past lessons I have been using word values for color; **I do not recommend this for final product sites.** The pre-assigned color words are harsh on user's eyes. Instead I recommend using hexadecimal values as in this video. Hexadecimal values give you millions of different color options with all types of shading. You can find different color schemes all over the internet and you can even grab different colors from sites you love now that you know how to use the ***inspect element*** feature. One of my favorite sites to grab colors from is flatuicolors.com. Go

to the site, click on the color you want to use, and that color's hexadecimal value will be copied for you to paste into your code.

Lesson 22: Centering Block Elements

Let's have some fun learning how to center block elements

1. Include the hr

index.html

add

`<hr>`

underneath `<h2>Get Involved</p>`

css/custom.css

add

```
.homepage hr {
  border-top: 1px solid #e74c3c;
  width: 150px;
  margin-top: 15px;
}
```

Lesson Notes

- If you want to center a block element in the middle of its parent element make sure to give it a left and right margin value of auto for example `margin: 10px auto`. In this case that HTML element would have a top and bottom margin of 10 pixels and would be centered.
- Also in this lesson you saw two different types of values for width: **%** and **px**. Percentage (&) is relative to the parent element, in the case shown in the video the **<hr>** % width was relative to the container div. That is why it became smaller as decreased the width of the page's viewing area. A pixel width is absolute and stays the same length without regard of the page's viewing area or the parent element. As stated with font-size which value to use changes per the occasion and the intended result. If you are trying to style a div and you know you want that div to be half the size of that div's container div then give it a width of 50%. If you are styling a line like we are here and we want it to be an exact width then use pixels.

Lesson 23: Seeing The Benefits Of Using Classes

In this lesson you'll see the power of assigning classes to elements when you want multiple elements to have a similar style. It's a beautiful thing!

1. Put in Who We Are section

index.html

```
<div class="container homepage">
  <h2>Who We Are</h2>
  <hr>
  <p>We are <a href="http://www.massiveacademy.us" target="_blank">MASSIVE 
Academy</a>. We aim to improve education through both method - effective 
project-based learning - and material - by teaching skills that are applicable to 
improving your life today.</p>
</div>
```

2. Put in image

You can grab image from the lesson's download tab.

index.html

```
<img src="images/academy_brand_med.png" title="MASSIVE Academy" alt="MASSIVE 
Academy">
```

Lesson 24: Hello Hover

We're rockin' and rollin' now as we style links for when users hover over them and get an introduction to pseudo-classes.

1. Put in styling for links when a user hovers over them

css/custom.css

add

```
.homepage a:hover {
  color: #e74c3c;
  text-decoration: none;
  opacity: .8;
}
```

Lesson Notes

- The opacity property is the transparency of the HTML element. Its on scale from 0 to 1 with zero being completely transparent to 1 being not transparent at all. A value of .8 would be about 20% transparent. I like to use opacity with links when they're hovered to show a slight change in color as shown in the video - neat little trick.
- **Bonus**: This wasn't covered in the video, but you can combine opacity and color by giving an html element an rgba value. So you can convert hexadecimal values to rgb values here. By doing this our hexadecimal salmon color (`color: #e74c3c;`) can also be styled in CSS with the following code `color: rgb(247, 76, 60);`. Then to give it the slight change in color with the opacity when someone hovers you could use the following CSS code `color: rgba(247, 76, 60, .8);`. Some people like to use the rgba value as a background-color. Say for example you wanted your header to be slightly transparent for an effect you could use **rgba**.

Lesson 25: Beautification Through Background Colors

Lets learn how to style full-width background colors in order to give our some subtle design beauty

1. Change the background color

index.html

Wrap the div `<div class="container homepage">` with another div `<div id="get-involved">`. Also move the closing tag for`<div class="container homepage">` so that it doesn't wrap the `<hr>` and `<footer>`. Don't forget to indent all the lines in between the opening and closing tags of `<div class="container homepage">`.

css/custom.css

add

```
#get-involved {
  background-color: #f5f5f5;
}
```

2. Put in some padding to make the two sections look good

css/custom.css

add

```
.homepage {
  text-align: center;
  padding-bottom: 50px;
  padding-top: 40px
}
```

Lesson Notes

- Notice the naming convention for `id` versus `class`. Where we have been identifying classes by putting a `.` before the class name, to identify an id we put a `#` before the id name. Ex: `#get-involved`.
- There will also be an added benefit to using the `id` tag later when we create links to certain parts of our page from the navbar.

Lesson 26: IDs Versus Classes

In this lesson we'll learn about the benefits of IDs and when to use them versus classes

1. Link to the get-involved div

index.html

change
```
<li><a href="#">GET INVOLVED</a></li>
```

to

```
<li><a href="#get-involved">GET INVOLVED</a></li>
```

2. Wrap the WHO WE ARE section with a div with an id of who

index.html

Wrap the div `<div class="container homepage">` (of the WHO WE ARE section) with another div `<div id="who">`. Don't forget to indent all the lines in between the opening and closing tags of `<div class="container homepage">`.

3. Link to the get-involved div

index.html

change

`<li><a href="#">WHO WE ARE</a></li>`

to

`<li><a href="#who">WHO WE ARE</a></li>`

Lesson Notes

- When to use an ID vs a class? Use an ID when it is a unique element that will have its own type of styling that no other element on your site will have or you can use ID to name an element that you want to link to take a user to that section when a certain link is clicked. Use class when you have multiple HTML elements that you want to style in a similar manner.
- IDs are more specific than classes so they will take priority. In other words if we wanted the h2 in the get-involved section to have a different color than the h2 color we specified by coding `.homepage h2 { color: #ef4c3c; }` we could have put in the rule `#get-involved h2 { color: black; }`. In this case the h2 in the who we are section would still be that specified by the homepage class, but the h2 in the get-involved section would be black since IDs are more specific than classes. This is exactly like what we did with the "educator" ID example in the video.

Lesson 27: Finish Off With The Footer

A quick styling of the footer section and you have yourself your first website. Do a dance!

1. Put image in Footer section

You can grab image from this lesson's downloads tab.

index.html

```
<footer>
  <div class="container">
    <p><img src="images/one_million_bottom_brand.png" alt="One Community_ One Million
Lines_"></p>
  </div>
</footer>
```

Also delete the `<hr>` above the footer section.

2. Style the footer section correctly

css/custom.css

```
footer {
  background-color: #e74c3c;
  padding-top: 150px;
  padding-bottom: 30px;
}
```

also change CSS for the body

```
body {
  padding-top: 50px;
}
```

Lesson 28: Making Your Site Look Good Across All Devices

We'll make the site look good across all devices with some responsive design and your bonus intro to the awesome world of Javascript.

1. Make yo' site responsive

css/custom.css

```
/* Responsive Styling */

@media (max-width: 1199px) {

  .jumbotron h1 {
    font-size: 56px;
    padding: 30px;
  }

}

@media (max-width: 991px) {

  .jumbotron h1 {
    font-size: 44px;
  }
  .jumbotron p {
    padding: 0 10px;
  }

  #get-involved .col-md-4 {
    padding-top: 20px;
    padding-bottom: 20px;
  }

}

@media (max-width: 767px) {

  .jumbotron h1 {
    font-size: 24px;
  }

  .jumbotron p {
    font-size: 16px;
    padding: 0;
  }

  .btn-lg {
    font-size: 18px;
  }

  .homepage p {
    font-size: 18px;
  }

  footer img {
    width: 80%;
  }

}
```

2. Correct the link to Bootstrap's built in Javascript document

index.html

change
```
<script src=”../../dist/js/bootstrap.min.js”></script>
```

to

```
<script src=”js/bootstrap.min.js”></script>
```

3. Change our page's title

index.html

change
```
<title>Jumbotron Template For Bootstrap</title>
```

to

```
<title>One Million Lines</title>
```

Lesson Notes

- In this lesson we fix the link to Bootstrap's Javascript. Javascript allows us to add functionality to our site. You can see this in the video once we correct the link to the .min.js file as the button now works - when clicked you see the drop-down menu appear. Pretty awesome right?! Through javascript we can make our sites do some extremely cool things and build in functionality our users will love. We'll talk more about javascript and its capabilities in future lessons.

Lesson 29: Getting Your Site Live Online With Dropbox

Lets get yo' site live and online with Dropbox for the world to see.

You can get dropbox here.

1. Make Google Font load over a secure server

index.html

change

```
<link href='http://fonts.googleapis.com/css?family=Arvo' rel='stylesheet' 
type='text/css'>
```

to

```
<link href='https://fonts.googleapis.com/css?family=Arvo' rel='stylesheet' type='text/css'>
```

Lesson Notes

- Dropbox is a good service to use if you are working on a site and want to have it live from the beginning for people to see - say if you have a client you are building a site for. Just transfer your site's folder into the public Dropbox folder when you start building the site and everytime you make changes in Sublime or any other text editor the changes will be shown live on the web for anyone to see.
- Push: putting the site live online for people to access.

Lesson 30: What's Next

Congratulations on completing the course!

At this point you've come a long way in a short amount, possibly with no knowledge of HTML and CSS to now knowing the building blocks of HTML and CSS, knowing how to put sites up and even having your own site up.

So where do you go from here? Well, although you've made great strides there is still a healthy amount of HTML and CSS for you to learn till you're a true website building pro.

We cover all of this in my second course "Learn to Build Beautiful Websites with HTML5 and CSS3 in 1 Month". In this course you will take the step to being a website building pro and not only able to build websites, but websites that wow the user.

And since you are a repeating student I want to thank you and give you a chance to enter the course at 50% off what any other student coming to the course would pay.

So go ahead and take that next step towards completing your knowledge of HTML and CSS and start building beautiful websites today at 50% off. Click here :)