Advanced Web Design

Instructor Tonya D. Wright

Unit 2

STEP 1

WATCH VIDEO: Building the Header Section

Now that we have our navigation, let's move on to the rest of our header section.

We need to add:

- an anchor tag for the logo
- a div container to hold the hero area
- inside of the hero area, we're going to have an h1 tag and an anchor tag

Within our code, we need to add an anchor tag with a class of logo, for the logo at the top of the page. Having the logo in an anchor tag will enable the user to click on it and go back to the home page. I used Google for a link in this spot, you can use any site you wish. We need to add a link to prove that our hyperlinke code is working

WORKING IN INDEX.HTML

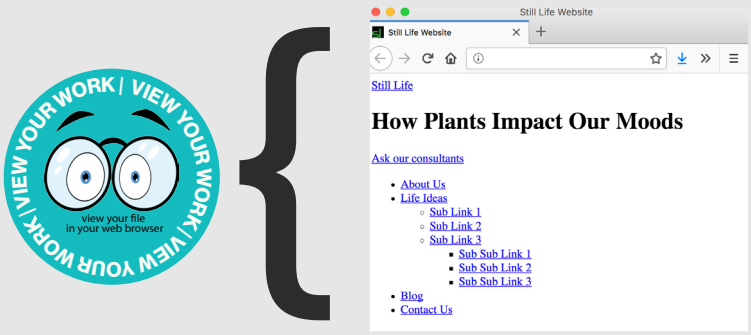
In the middle of your <header></header> tags add this code:

```
39 ▼        <header>
40              <a class="logo" title="Still Life" href="http://google.com">Still Life</a>
41 ▼            <div class="hero">
42                  <h1>How Plants Impact Our Moods</h1>
43                  <a class="btn" title="Ask our consultants" href="#">Ask our consultants</a>
44              </div>
45          </header>
```

Notice we have used a class with the title of hero, we have also identified our button class. We will see these class names again as we apply CSS rules to them.

We also added in some content with an <h1> tag.

Save your index.html file



Page looks good, let's keep moving!

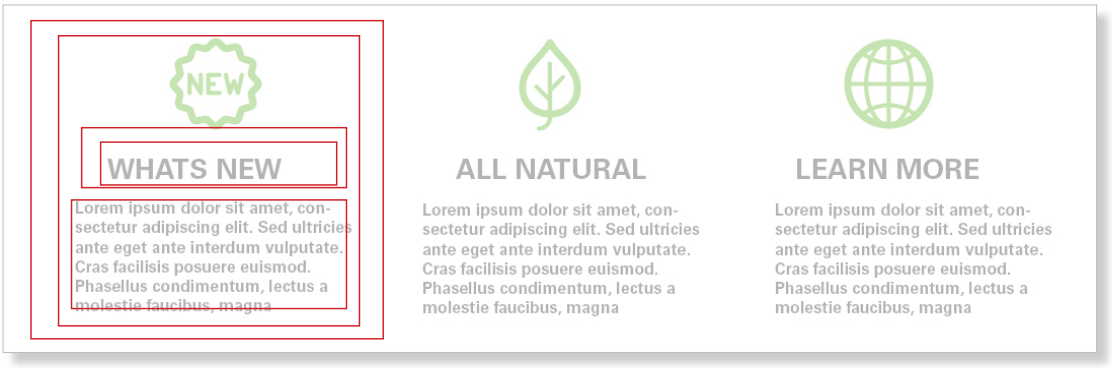
STEP 2

WATCH VIDEO: Building the Main Section

Believe it or not, our header is structurally done. We will dress it up with CSS in the next unit. For now, we need to finish the structure of the rest of our elements. Let's move on to the section elements.

SECTION MAIN
Our main content section consists of three columns of information. In order to create these three columns, we're going to put each one of these columns into an element called an **aside**.

Each aside will have a nested div element and an h3 element with an anchor tag inside, so you can click on it, and lastly, a paragraph element containing the description body of content.



<aside> <h3> <a> <p>

The <aside> tag defines some content aside from the content it is placed in. The aside content should be related to the surrounding content.

FILLER CONTENT
We have our page structure set up. Let's fill in the blanks with some content. But guess what, our client has not given us any content just yet. We want to keep designing and we also want to send our client some ideas for how much content is needed for each section. We can insert filler content into these spaces.

Filler content utilizes Lorem Ipsum.

You can add in this filler content from various websites. Here are a few sites you can choose from to get your Lorem Ipsum.

15 Funny Lorem Ipsum Generators to Shake Up Your Design Mockups

https://www.shopify.com/partners/blog/79940998-15-funny-lorem-ipsum-generators-to-shake-up-your-design-mockups

Here is what I did:

- Go to http://www.lipsum.com/ or whichever lorem ipsum generator you choose.
- Scroll down to select how many paragraphs you will need, we only need 1.
- Copy about **2** lines of that paragraph and insert it into your code.

```
35          <!-- Markup goes here -->
36      <div id="page">
37
38
39          <header> <a class="logo" t...
46
47
48          <nav> <ul> <li><a title="A...
69
70
71          <section class="main">
72              <!-- First Aside -->
73              <aside>
74                  <div class="content new">
75                      <h3><a href="#">What&#39;s New</a></h3>
76                          <p>consectetur adipiscing elit. Suspendisse a leo finibus, consequat ex et, interdum eros. Donec erat
77                          magna, varius sit amet ultricies atconsectetur adipiscing elit. Suspendisse a leo.</p>
78                  </div>
79              </aside>
```

NOTES

- I did not cover each aside so try the next two on your own. Make sure each <h3> tag is titled by its proper section name:

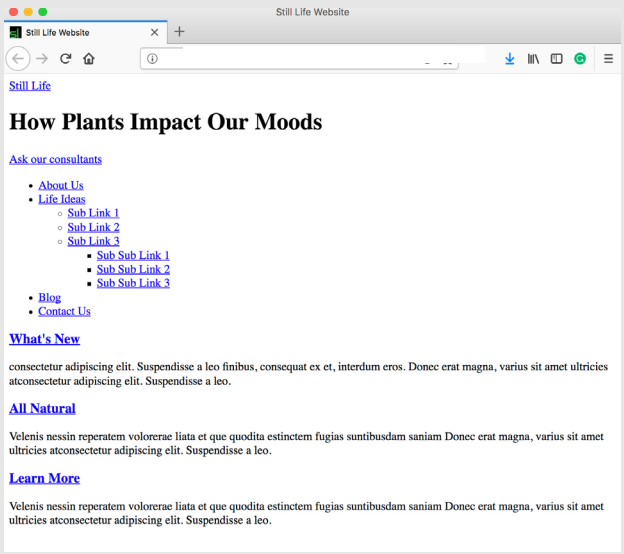
  What's New | All Natural | Learn More

- Don't forget to name your div class for each section

  content new | content natural | content learn

- Be mindful to add markup comments <!-- First Aside -->

GOOD LUCK - I BELIEVE IN YOU



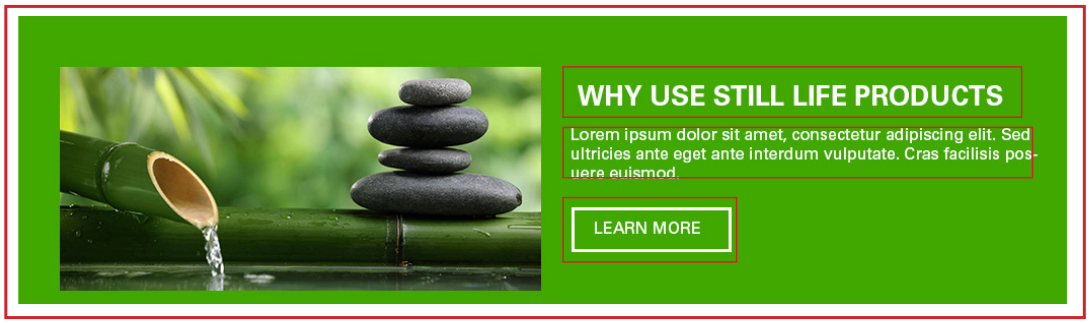
☑ Page looks good, let's keep moving!

STEP 3

WATCH VIDEO: Building The Story Section

SECTION STORY

For this section, we are using a class of story. Below are the tags we are going to create for this section. Note that all the styling will come from our CSS file.



<article> <h2> <p> <a>

Start your code in the section tag marked with a class of story. We will start the code by adding in an <article></article> element.

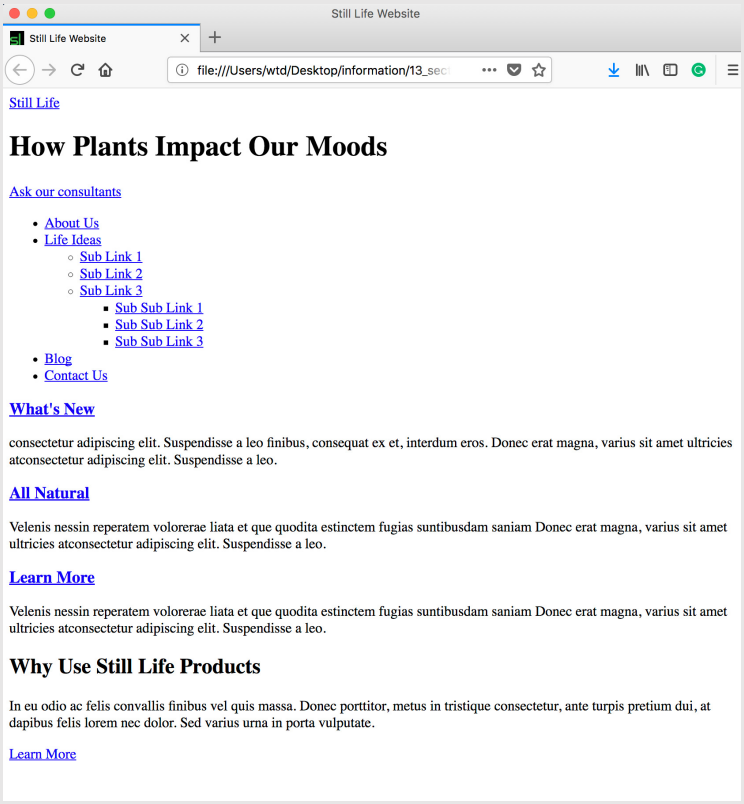
The <article> tag specifies independent, self-contained content.

An **article** should make sense on its own and it should be possible to distribute it independently from the rest of the site. Potential sources for the <article> element: Forum post, Blog post, News story, Comment

Your code should look like this:

```
35          <!-- Markup goes here -->
36      <div id="page">
37
38
39          <header> <a class="logo" t...
46
47
48          <nav> <ul> <li><a title="A...
69
70
71          <section class="main"> <!-...
97
98          <section class="story">
99              <article>
100                     <h2>Why Use Still Life Products</h2>
101                         <p>In eu odio ac felis convallis finibus vel quis massa. Donec porttitor,
                            metus in tristique consectetur, ante turpis pretium dui, at dapibus felis
                            lorem nec dolor. Sed varius urna in porta vulputate.</p>
102                         <a class="btn" title="Why Use Still Life Products" href="#">Learn More</a>
103                 </article>
104         </section>
105
106         <section class="info"></section>
107
108         <footer></footer>
```

DON'T FORGET TO CHECK YOUR TAG PAIRS!



☑ Page looks good, let's keep moving!

STEP 4

WATCH VIDEO: Building The Info Section

SECTION INFO

- For this element, we're going to put two items on the left into an aside.
- Inside of the aside we're going to put our div element.
- We will use image tags for the pictures.
- The next step is to use h4 tags, paragraph and anchor links.
- Finally, we're going to use a blockquote for the quote area, with two paragraph items inside.

<aside> <img> <h4> <p> <a> <blockquote> <p>

Start your code in the section tag marked with a class of info. We will start the code by adding in an <aside></aside> element.

Your code should look like this:

```
35          <!-- Markup goes here -->
36      <div id="page">
37
38
39          <header> <a class="logo" t...
46
47
48          <nav> <ul> <li><a title="A...
69
70
71          <section class="main"> <!-...
97
98          <section class="story"> <a...
105
106         <section class="info">
107             <aside>
108                 <div class="content">
109                     <img alt="5 Ways To Meditate" src="images/img_spa1.jpg">
110                     <h4>5 Ways to Meditate</h4>
111                     <p>molestie eu justo sed, ultricies mollis eros. Integer fermentum pharetra
                        euismod. Proin pharetra urna nunc, eget rhoncus sapien pulvinar sit amet.
                        Quisque iaculis sapien sed diam fermentum placerat. </p>
112                     <a class="btn" title="5 Ways to Meditate"
                        href="http://www.peaceyogabelair.com/">Learn More</a>
113                 </div>
114             </aside>
115
116             <aside>
117                 <div class="content">
118                 <img alt="Peaceful Surroundings" src="images/img_spa3.jpg">
119                     <h4>Peaceful Surroundings</h4>
120                     <p>molestie eu justo sed, ultricies mollis eros. Integer fermentum pharetra
                        euismod. Proin pharetra urna nunc, eget rhoncus sapien pulvinar sit amet.
                        Quisque iaculis sapien sed diam fermentum placerat. </p>
121                     <a class="btn" title="Peaceful Surroundings"
                        href="http://www.peaceyogabelair.com/aboutus.html">Learn More</a>
122                 </div>
123             </aside>
124
125         </section>
126
127         <footer></footer>
```

If your images are not showing up. Make sure they are in the images folder.

I added in a live hyperlink so it can be tested for accuracy.

You can make your link to direct to whatever you would like.

Within the blockquote, we have two paragraph elements. Each has a class one with “quote” and the other “credit”.

Let’s do just a bit more to the structure of the credit class within the HTML.

- Author Name: bold
- Break between name and title
- Title: Italic
- Break between name and title
- Company name

```
35          <!-- Markup goes here -->
36      <div id="page">
37
38
39          <header> <a class="logo" t...
46
47
48          <nav> <ul> <li><a title="A...
69
70
71          <section class="main"> <!-...
97
98
99          <section class="story"> <a...
106
107         <section class="info">
108             <aside>
109                 <div class="content">
110                     <img alt="5 Ways To Meditate" src="images/img_spa1.jpg">
111                     <h4>5 Ways to Meditate</h4>
112                     <p>molestie eu justo sed, ultricies mollis eros. Integer fermentum pharetra euismod. Proin pharetra urna nunc,
113                     eget rhoncus sapien pulvinar sit amet. Quisque iaculis sapien sed diam fermentum placerat. </p>
114                     <a class="btn" title="5 Ways to Meditate" href="http://www.peaceyogabelair.com/">Learn More</a>
115                 </div>
116             </aside>
117
118             <aside>
119                 <div class="content">
120                 <img alt="Peaceful Surroundings" src="images/img_spa3.jpg">
121                     <h4>Peaceful Surroundings</h4>
122                     <p>molestie eu justo sed, ultricies mollis eros. Integer fermentum pharetra euismod. Proin pharetra urna nunc,
123                     eget rhoncus sapien pulvinar sit amet. Quisque iaculis sapien sed diam fermentum placerat. </p>
124                     <a class="btn" title="Peaceful Surroundings" href="http://www.peaceyogabelair.com/aboutus.html">Learn More</a>
125                 </div>
126             </aside>
127             <blockquote>
128                 <p class="quote">Vivamus convallis tempor dui, sit amet imperdiet dolor lobortis ut. Ut a sem lorem. Vivamus
129                 convallis tempor dui, sit amet imperdiet dolor lobortis ut. Ut a sem lorem.</p>
130                 <p class="credit"><strong>Marina Stalkwaters,</strong> <br> <em>Account Director</em>
131                 <br> Pursky Fundamentals
132                 </p>
133             </blockquote>
134         </section>
```

☑ Page looks good, let's keep moving!

STEP 5

WATCH VIDEO: Building The Footer Section

SECTION FOOTER

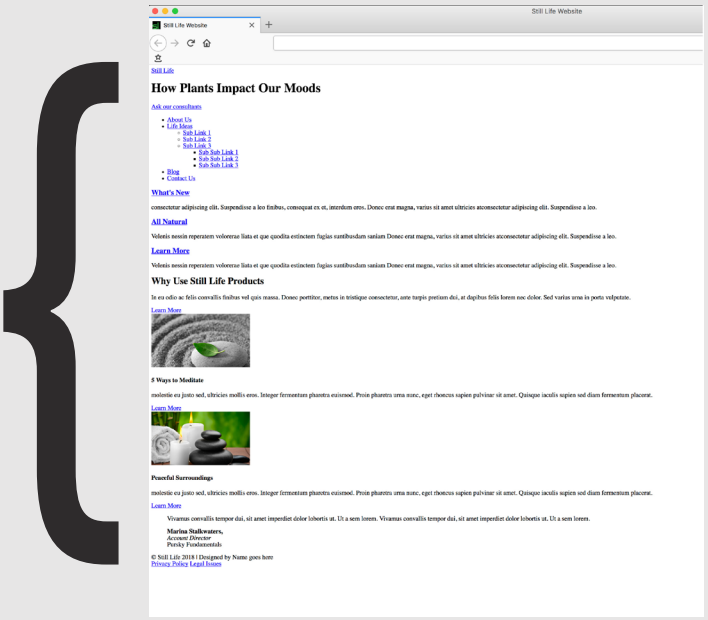
Place your cursor in the footer section and break the tags apart to ensure your content is between the opening and closing tags.

In this section, we will be adding in the copyright information, a few links, and if you choose you can add designed by company information (as the designer of the site).

```
<head> <meta http-equiv="C...
<body>
    <!-- Markup Goes Here -->
    <div id="page">
        <header> <a class="logo" t...
        <section class="main"> <!-...
        <section class="story"> <a...
        <section class="info"> <!-...
        <footer>
            &copy; Still Life XXXX | Designed by TD Wright
            <div class="content">
            <a title="Privacy Policy" href="#">Privacy Policy</a>
             <a title="Legal Issues" href="#">Legal Issues</a>
            </div>
        </footer>
    </div>
</body>
</html>
```

AWESOMESAUCE!
We have completed the structure of our page with HTML. All of our styling which was not included in the HTML file (purposely), will go in our CSS file.

Look over your HTML and notice this strict adherence to structure has left us with a decent looking page in the event something goes wrong with the users' browser and they are only able to view a stripped page view.



☑ Page looks good, let's keep moving!

Advanced Web Design

Instructor Tonya D. Wright

Unit 2.2 Basic Styling

STEP 1

WATCH VIDEO: Importing Google Fonts

Importing Google Fonts

Our page is ready to be transformed with CSS. Our structure (HTML) is in place and now it's time for presentation (CSS).

Locate your CSS folder and open the screen.css file in your text editor. There are no rules in this file as of yet. Remember we created this file so we could embed it into the HTML. Now we will get to creating some rules to style our HTML

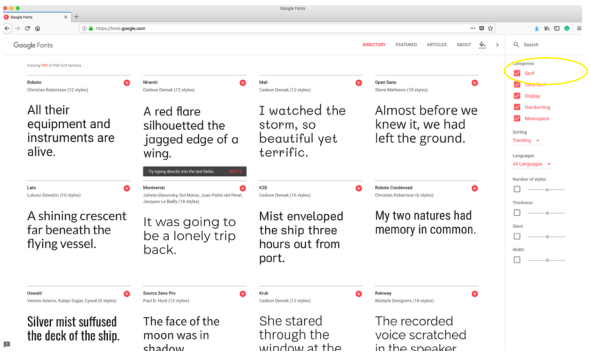
Our first step is to import some web fonts from Google by using @import

It is important to note the @import code needs to be the first line of code in our CSS file.

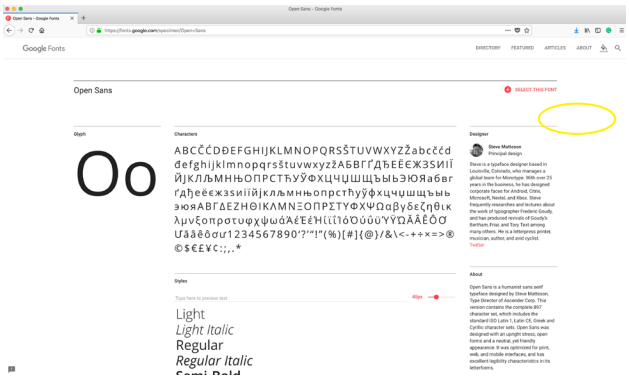
There are many fonts services you can use some are free, some are commercial. Everything on Google Fonts is **free**

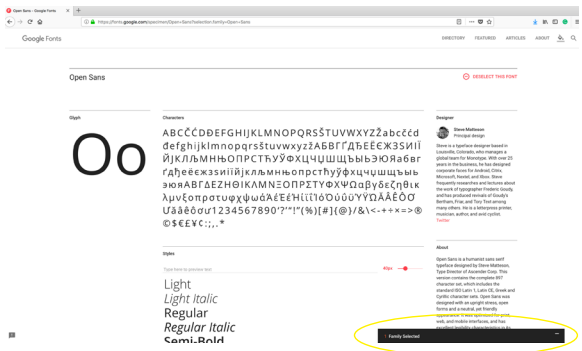
How to get the fonts:
Go to this website: https://fonts.google.com/

1. Search for open sans



2. Select this font



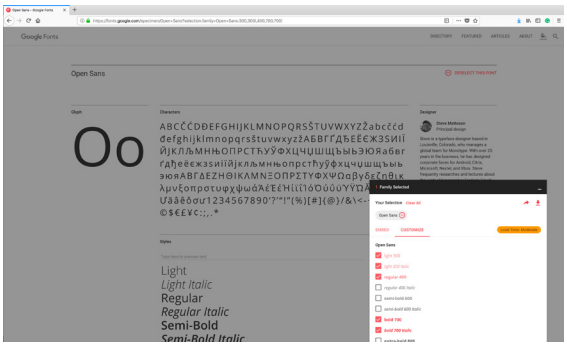


3. You should now see another dialogue box pop up at the bottom of the page.

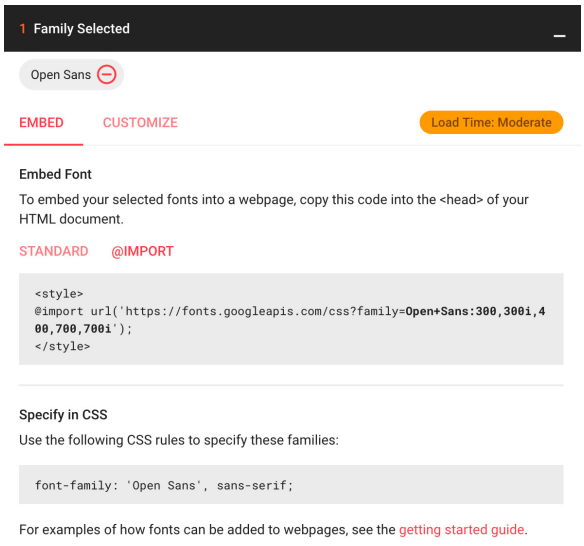
   Click on this box, to expand it.

4. Once the box is expanded go to the customize tab and select the fonts you want to import.

We are importing;
- light
- light italic
- regular
- bold
- bold italic



6. Go back to the embed tab so we can copy the @import command



Don't close the Google Fonts website just yet, we need one more thing from this page.

STEP 2

WORKING IN SCREEN.CSS

Open the screen.css file insert/paste the @import rule into your CSS.

```
1  @charset "UTF-8";
2  /* CSS Document for Screens */
3
4  @import url('https://fonts.googleapis.com/css?family=Open+Sans:300,300i,400,700,700i');
```

- **Go back to the Google Fonts website** and copy the font family information from the dialogue box.
- Copy the name of the font family which is listed as: Open Sans with a specification of Sans Serif.
- We will add a body rule and insert/paste in the information from Google Fonts
- Also add in a font-size of 16px. (16px is our base font size.)

Note: This is the only time we will address the font in pixels, we will be using ems.

Em's are proportional measurements associated with the base font, in our case the base font is 16px. Using em's help when the various devices scale the font for best resolution.

WORKING IN SCREEN.CSS

- add a rule for font-weight. The base weight of 300 is associated with the fonts we imported from Google Fonts.
- Add a property for the color of #282828.
- Finally, add another property to set the margin and padding to 0

Your code should look like this:

```
@charset "UTF-8";
/* CSS Document for Screens */

@import url('https://fonts.googleapis.com/css?family=Open+Sans:300,300i,400,700,700i');

body {
    font-family: 'Open Sans', sans-serif;
    font-size: 16px;
    font-weight: 300;
    color: #282828;
    margin: 0;
    padding: 0;
}
```

CSS can get full very fast. In order to be able to locate rules quickly, we will add in comments. As we continue to develop our CSS we will be coming back to certain spots to add additional code. The comments will help you find that section faster.

Please ensure your rules within the CSS file are ordered properly, exactly in the order you see them in your notes or videos.

Some rules lean very heavily on the rules of precedence or specificity. Read this for more information: https://developer.mozilla.org/en-US/docs/Web/CSS/Specificity

STEP 3

It's time to set our basic styles. We will start by setting h tag properties. The amount of space after an h1 tag is going to be 1em. Specifying 1em is basically saying one times the base font.

Which means approximately 16 pixels - remember by defining ems we're giving mobile devices flexibility to render the type based on their individual screen sizes.

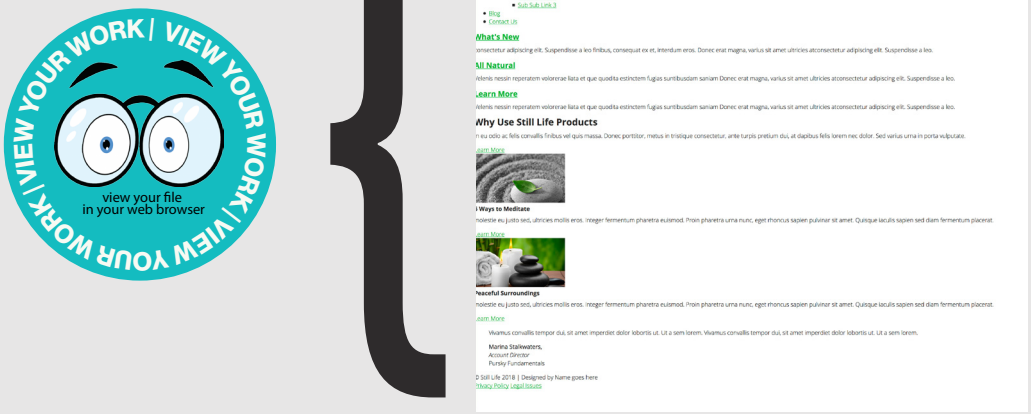
Check this out: PX to Em Converter

WORKING IN SCREEN.CSS

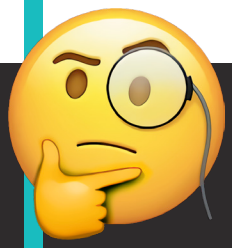
We are going to go down the list and set properties for the following:
- h tags - h1, h2, h3, h4,
- p tag,
- a tag

Pay attention to the parameters set for each property. Your code should look like this:

```
@charset "UTF-8";
/* CSS Document for Screens */

@import url('https://fonts.googleapis.com/css?family=Open+Sans:300,300i,400,700,700i');

body {
    font-family: 'Open Sans', sans-serif;
    font-size: 16px;
    font-weight: 300;
    color: #282828;
    margin: 0;
    padding: 0;
}

/* Basic Text Styles */


h1 {
    margin: 0 0 1em 0;
    font-size: 2.8em;
    font-weight: 700;
}

h2 {
    margin: 0 0 .5em 0;
    font-size: 1.6em;
    font-weight: 700;
    line-height: 1.1em;
}

h3 {
    margin: 0 0 .5em 0;
    font-size: 1.3em;
    font-weight: 700;
}

h4 {
    margin: 0 0 .5em 0;
    font-size: 1em;
    font-weight: 700;
}

p { margin: 0 0 1em 0;}

a { color: #39b54a; }
a:visited { color: #38E261; }
```



☑ Page looks good, let's keep moving!



In Case You Were Wondering

Yes, as you get pretty proficient with CSS you can use shorthand. Shorthand properties are CSS properties that let you set the values of several other CSS properties simultaneously. Using a shorthand property; a Web developer can write more concise and often more readable style sheets, saving time and energy.

For margins and padding, we don't want to have to write down all the variables of margin-top, margin-bottom and so on. So we default to using shorthand.

Remember: TRBL/ **TR**ou**BL**e (Top, Right, Bottom, Left)

THIS: margin { 0 5 4 0; } (Not specifying the unit of measurement, means the measurement is inherent to what we have identified as the measurement in a previous rule.)

IS THE SAME AS:

margin-top: 0
margin-right: 5
margin-bottom: 4
margin-left: 0

Top
Right

TRou**BL**e

Bottom
Left

Advanced Web Design

Instructor Tonya D. Wright

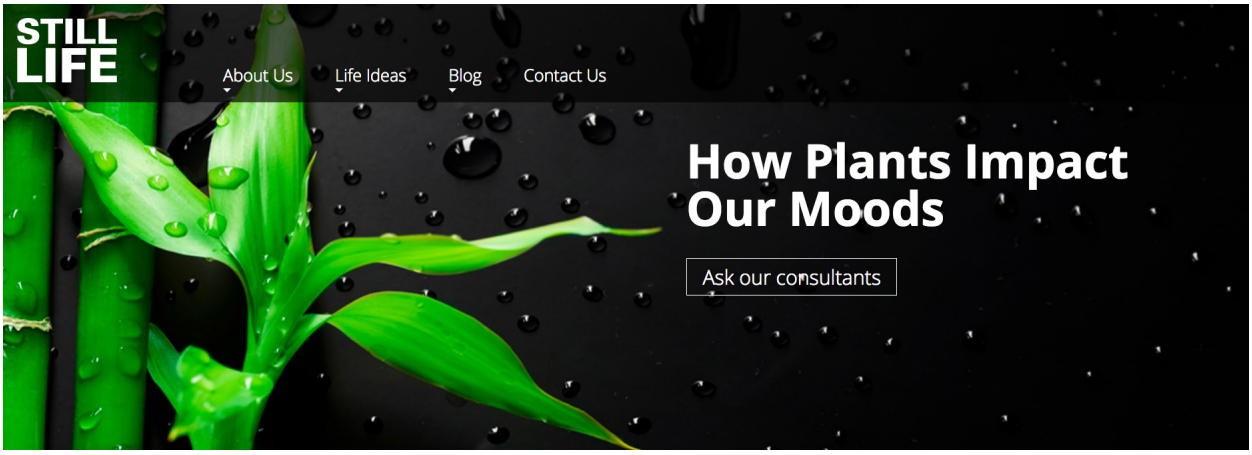
Unit 2.3 Styling the Header

STEP 1

WATCH VIDEO: Styling the Header

Styling the Header

Once we are finished our header will look like this:



WORKING IN SCREEN.CSS

- Add a comment: /* Header */

- Add in properties for the header element; which include height, background color and image, and positioning.

- We are linking our image from within the CSS. If you ever decide to change the header image name, you will need to change it from here.

Your code should look like this:

```
51 ▼ header {
52       height: 430px;
53       background: #000 url(../images/banner_1200.jpg) no-repeat center bottom;
54       position: relative;
55   }
```

Remember folder levels: ../ brings us out to the root folder so we can identify the images folder.

The relative position property will make sure that any items that are positioned inside of the header element will be positioned in relation to the header element itself.

All the various sizes we have for the header image will come into use as we adjust our media queries to accommodate various screen sizes.

STEP 2

FIXING THE HEADER WIDTH

The header image width is 1200 pixels, but elements on the page are allowed to extend further than 1200 pixels. We need to adjust the page to avoid this. We will write a rule that targets the page ID that will limit the width of the items.

CSS PAGE JUMP: Remember earlier, I mentioned we were going to be jumping around the CSS page. Here is our first jump.

Place your cursor right below the body property

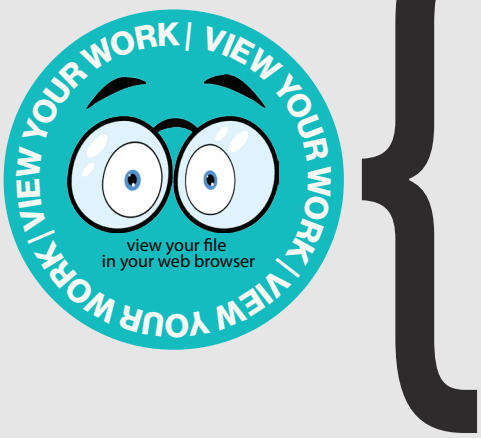
Your code should look like this:

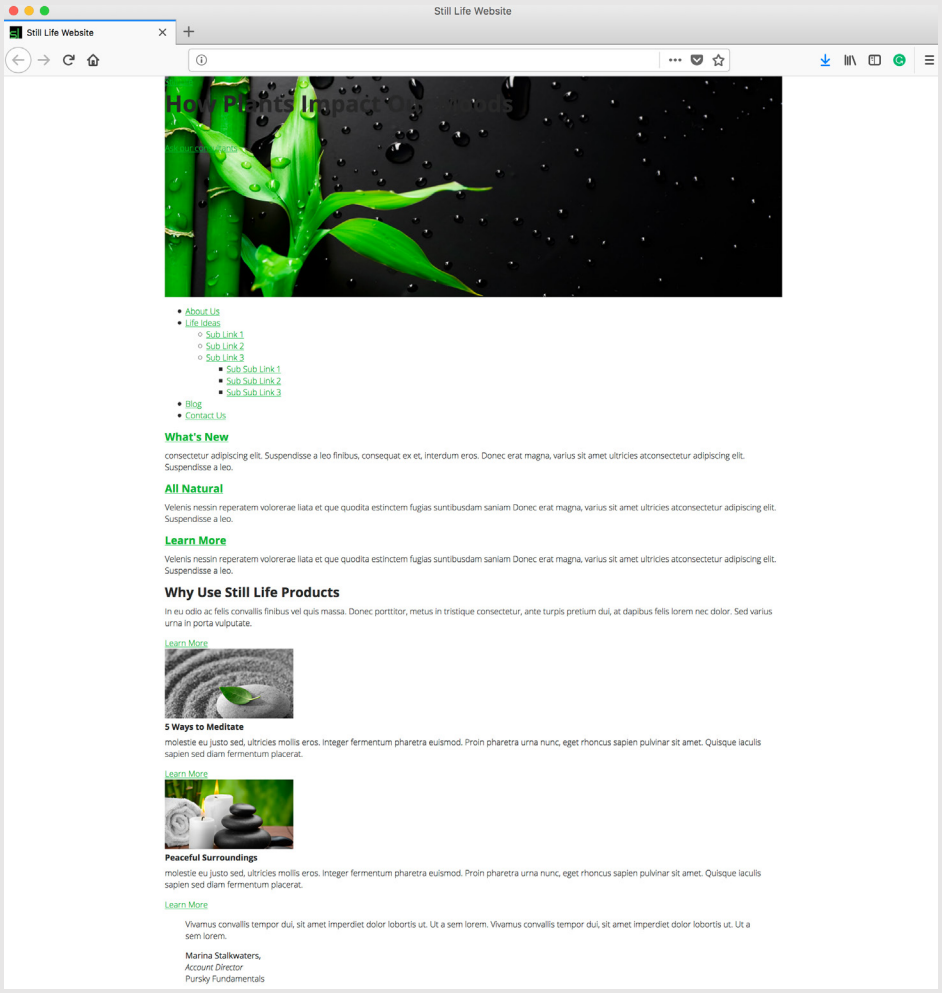
```
@charset "UTF-8";
/* CSS Document for Screens */

@import url('https://fonts.googleapis.com/css?family=Open+Sans:300,300i,400,700,700i');

body {
    font-family: 'Open Sans', sans-serif;
    font-size: 16px;
    font-weight: 300;
    color: #282828;
    margin: 0;
    padding: 0;
}

#page {
    max-width: 1200px;
    margin: 0 auto;
    position: relative;
}

/* Basic Text Styles */


h1 {
    margin: 0 0 1em 0;
    font-size: 2.8em;
    font-weight: 700;
}

h2 {
    margin: 0 0 .5em 0;
    font-size: 1.6em;
    font-weight: 700;
    line-height: 1.1em;
}

h3 {
    margin: 0 0 .5em 0;
    font-size: 1.3em;
    font-weight: 700;
}

h4 {
    margin: 0 0 .5em 0;
    font-size: 1em;
    font-weight: 700;
}

p { margin: 0 0 1em 0;}

a { color: #39b54a; }
a:visited { color: #38E261; }


/* Header */

header {
    height: 430px;
    background: #000 url(../images/banner_1200.jpg) no-repeat center bottom;
    position: relative;
}
```

Shorthand Note: If you only specify two properties like what is listed in the margin, then the first and third properties will be set to top and bottom. The second and fourth are set to right and left, and by setting a margin auto, the content is centered.

The position relative, in this case, means any items positioned inside of the page element will be positioned in relation to this. This will be important to our development because we're going to reposition the navigation element, and the navigation is not inside of the header, which also has a position: relative, so the navigation will position in relation to the page.

If we didn't put in a max-width, when a user opened their browser wider than 1200, the navigation would position outside of the page element, so this makes sure that everything is contained in the main page container.





☑ Page looks good, let's keep moving!

STEP 3

WATCH VIDEO: Styling the Header

Styling the Logo and Hero item
Time to bring the SVG logo into our layout by assigning it to be the background of the anchor tag we added earlier inside of the heading area. Remember this is the main link for the site.

WORKING IN SCREEN.CSS

We will be
- assigning a width of 160px,
- height of 66px,
- adding in the logo.svg file

Explanation
1. The position is set to absolute. This way when we assign positioning properties it will position in relation to this header element because of the position relative on the parent.

2. By default, anchor tags have what's called an inline style, which means they don't have any dimensions, they will stretch to their width. This is because anchor links are text in the page, but in this case, we want to take this anchor link and turn it into a block object.

3. We're going to set the repeat property to no repeat, and then zero position on the x and zero position on the y. This will position the graphic's upper left hand corner to the upper left-hand corner of this logo element.

4. Set a property called background-size to contain, this means the background graphic's going to be sized to fit within the height and width and since we set the height and width to a proportion based on the size of the logo SVG, which is 160 by 66, this will proportionally scale and fill the entire element with the logo.

5. We're going to set a top property, this is in relation to the position absolute, so the top property we're going to set to 15 pixels, which will be down 15 pixels from the top of the header element.

6. A left property of 20 pixels, which will make sure that the left edge of the logo is 20 pixels away from the left-hand side of the page.

```
/* Header */

header {
    height: 430px;
    background: #000 url(../images/banner_1200.jpg) no-repeat center bottom;
    position: relative;
}


header a.logo {
    position: absolute;
    display: block;
    width: 160px;
    height: 66px;
    background: url(../images/logo.svg) no-repeat 0 0;
    background-size: contain;
    top: 15px;
    left: 20px;

}
```

You will notice that the anchor link is over the logo, and the hero text is being covered by the logo

We need to add a span element to the HTML coding. Span is an inline element, just like an anchor tag. Span allows us to put a little bit of extra instruction around a piece of content.

WORKING IN INDEX.HTML

In the header tag of your index.html file add a span class around the text for the logo

```
38
39 ▼        <header>
40              <a class="logo" title="Still Life" href="http://google.com"><span>Still Life</span></a>
41 ▼                <div class="hero">
42                      <h1>How Plants Impact Our Moods</h1>
43                      <a class="btn" title="Ask our consultants" href="#">Ask our consultants</a>
44                  </div>
45          </header>
46
```

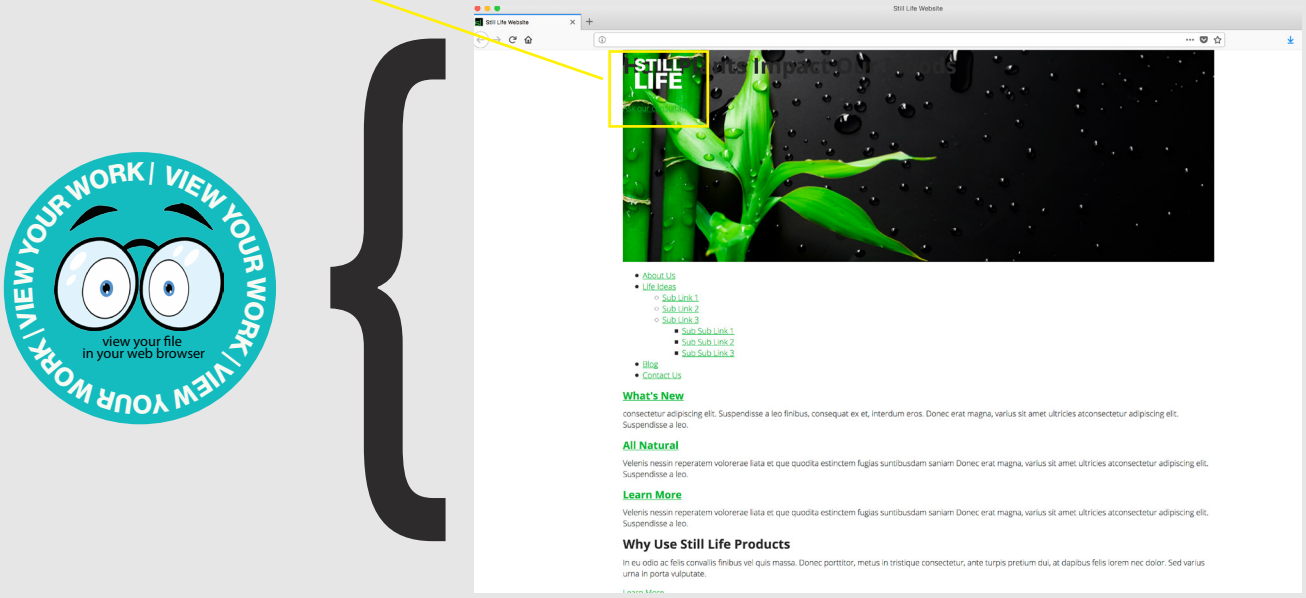
Now that we have a span we can create a CSS rule that will give properties to the span class.

WORKING IN SCREEN.CSS

Title this rule header a.logo span setting the display property to none, which will prevent the user from seeing the anchor link text

```
/* Header */

▼ header {
    height: 430px;
    background: #000 url(../images/banner_1200.jpg) no-repeat center bottom;
    position: relative;
}


▼ header a.logo {
    position: absolute;
    display: block;
    width: 160px;
    height: 66px;
    background: url(../images/logo.svg) no-repeat 0 0;
    background-size: contain;
    top: 15px;
    left: 20px;

}

▼ header a.logo span {
    display: none;
}
```

The link should be gone. Check your page, I don't have a link. Does your page look good?



☑ Page looks good, let's keep moving!

STEP 4

WATCH VIDEO: Styling the Header

WORKING IN SCREEN.CSS

Now let's fix the hero text.
- We are going to absolute position the text and set a width of 42% which will allow this item to be 42% the width of the header element.

- Create a top property of 130 pixels and a left property of 55%.

Your code should look like this:

```
header div.hero {
    position: absolute;
    width: 42%;
    top: 130px;
    left: 55%;
}
```

Wait! before you check your page, let's set a rule for the h1 included within this content and style our button.

We are going to create properties that will affect:
- the line-height which will be set at 1em;
- redefining the margin: 0 0 30px 0. The reason for a specific amount of 30px is because there is only one item
- make the text white #fff

Place this code right below the header div.hero rule
Your code should look like this:

```
88 ▼ header div.hero h1 {
89       line-height: 1em;
90       margin: 0 0 30px 0;
91       color: #fff;
92   }
93
```

By now I hope you are beginning to see how the rules are written to target an element.

STEP 5

WATCH VIDEO: Styling the Header

WORKING IN SCREEN.CSS

We need to create a rule so that our anchor links that have the .btn class style will really look like buttons.

CSS PAGE JUMP: This is a global style so we will jump back up to the top of our css page and put this rule in

Place your cursor right ABOVE the header section, within the Basic Text Styles Section

1. **Font size:** set this to 1.2em, so 20% larger than the base font.
2. **Text decoration:** set this to none, this way anchor links with a class of button will not have an underline.
3. **Color:** white #fff
4. **Border:** use shorthand style for the border. border: 1px solid #fff;
    - Type border
    - First property is going to be the width, so we'll type 1px.
    - Second property is going to be the style, so we'll type solid.
    - The third style is going to be the color, so we're going to type #fff

5. **Padding:** controls the space inside of the element, we don't want the border to touch the text, we want to give a little bit of space. We will use shorthand here as well. padding: 4px 15px;
    - 4px for the top and bottom
    - 15px for the right and left.

Read further to understand the difference between padding and margins:
https://www.w3schools.com/css/css_boxmodel.asp

Your code should look like this:

```
/* Basic Text Styles */

h1 {
    margin: 0 0 1em 0;
    font-size: 2.8em;
    font-weight: 700;
}

h2 {
    margin: 0 0 .5em 0;
    font-size: 1.6em;
    font-weight: 700;
    line-height: 1.1em;
}

h3 {
    margin: 0 0 .5em 0;
    font-size: 1.3em;
    font-weight: 700;
}

h4 {
    margin: 0 0 .5em 0;
    font-size: 1em;
    font-weight: 700;
}

p {
    margin: 0 0 1em 0;
}


a { color: #39b54a; }
a:visited { color: #38E261; }


a.btn{
    font-size: 1.2em;
    text-decoration: none;
    color: #fff;
    border: 1px solid #fff;
    padding: 4px 15px;
}
```

STEP 6

WATCH VIDEO: Styling the Header

Now we need to set the hover state, so when a user rolls over the button there is interactivity. To do this we will create a pseudo-class.

We want to use a semi-transparent green, so instead of using a # sign, which is the hexadecimal value for color, we're going to use the rgba color space. RGBA color values are an extension of RGB color values with an alpha channel - which specifies the opacity for a color.

An RGBA color value is specified with: rgba(red, green, blue, alpha). The alpha parameter is a number between 0.0 (fully transparent) and 1.0 (fully opaque).

Specify red, green, and blue separated by commas, so 6 for red, 180 for green, 43 for blue Then, for the alpha, .7. So this means this will be 70% opaque, or 30% transparent.

Your code should look like this:

```
63
64 ▼ a.btn:hover {
65       background-color: rgba(57,181,74,.7)
66   }
```

For future, you can use this color picker tool to create your perfect RGBA color:
https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Colors/Color_picker_tool

STEP 7

One last touch. Let's animate this button. This is pretty easy.

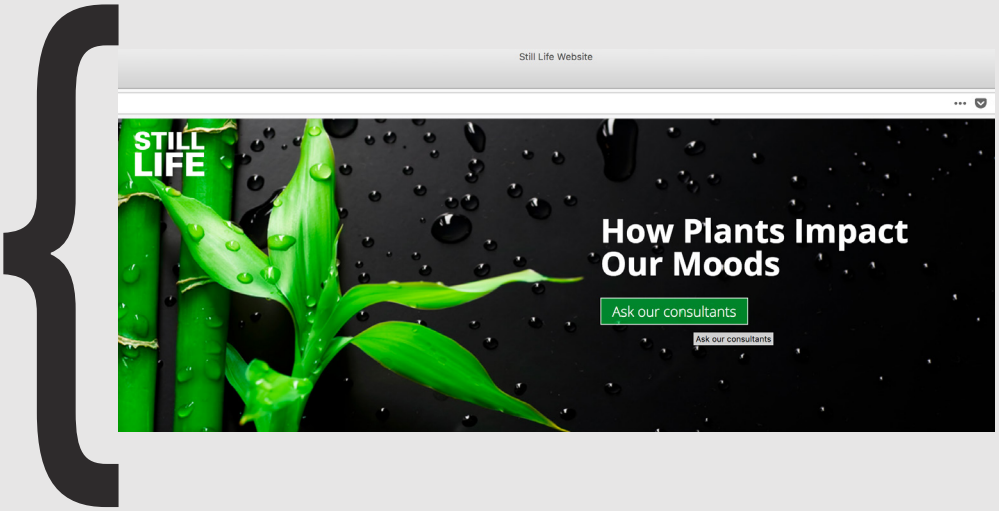
WORKING IN SCREEN.CSS

Return to the a.btn line of code we are going to add in a transition.
We'll use shorthand here, which defines the property and time (5s for 5 seconds):

transition: background-color .5s

```
59 ▼ a.btn{
60       font-size: 1.2em;
61       text-decoration: none;
62       color: #fff;
63       border: 1px solid #fff;
64       padding: 4px 15px;
65       transition: background-color .5s;
66   }
```

By putting transition of background-color in this item, anytime the background color is changed, we're going to get an animated transition when a user hovers over the background goes up to 30% opaque, and then back down to zero when they roll away.



☑ Page looks good, let's keep moving!

Great work! Let's move on to fixing up our navigation.

Advanced Web Design

Instructor Tonya D. Wright

Unit 2.4 Styling The Navigation

STEP 1

WATCH VIDEO: Styling the Navigation

STYLING THE NAVIGATION

WORKING IN INDEX.HTML

Now that we have our header dressed up. I am sure you noticed our navigation is sitting under the header. As you move forward in developing you will include your nav in the header right off. I did it this way to keep our steps separate, and to show you how easy it is to move code around.

Move the navigation inside of the header tags. Place it under the last closing . Make sure when you move the navigation that you have both the starting and closing tags included in your move.

```
<header>
    <a class="logo" title="Still Life" href="http://google.com"><span>Still Life</span></a>
        <div class="hero">
            <h1>How Plants Impact Our Moods</h1>
            <a class="btn" title="Ask our consultants" href="#">Ask our consultants</a>
        </div>

<!-- Nav Starts Here -->
    <nav> <ul> <li><a title="A...
</header>
```

Remember, I have my nav tag toggled close for easy copy and paste. With our nav tag in the right place (between our header tags) let's move on to styling.

STEP 2

WORKING IN SCREEN.CSS

We will create a new section. Add a comment / * Navigation */

Explanation:

- background color, we will use rgba settings so we can set the alpha transparency
- rgba (0, 0, 0, .65)
- position absolute
- top property 0px
- left 0px

  Since the navigation element is sitting inside of the main page container along with the other section elements, header and footer, the navigation element will position itself in relation to the page element. So if we set top 0 and left 0, this will go all the way up in position over-top of the heading area.
- padding 50px, 0 0 0
- width 100%

  The reason we set a width here is by setting an absolute position the navigation element will only be as wide as the content inside of it. And we want to make sure that the navigation element is as wide as the page container. Setting a width of 100 will make sure that the navigation element is as wide as its parent, which again is the div element with an ID of page.

Your code should look like this:

```
111
112   /* Navigation */
113
114 ▼ nav {
115       background-color: rgba(0,0,0,.65);
116       position: absolute;
117       top: 0px;
118       left: 0px;
119       padding: 50px 0 0 0;
120       width: 100%;
121   }
```

No space after rgba

STEP 3

One more thing, we need to ensure the logo is on top of the navigation bar so we need to set a z-index. Z-index allows us to layer elements on top of each other.

Check out this awesome example of CSS and Z-index at work: https://codepen.io/miocene/full/mjLPVp/

WORKING IN SCREEN.CSS

Go up to the header section and look for the header a.logo rule

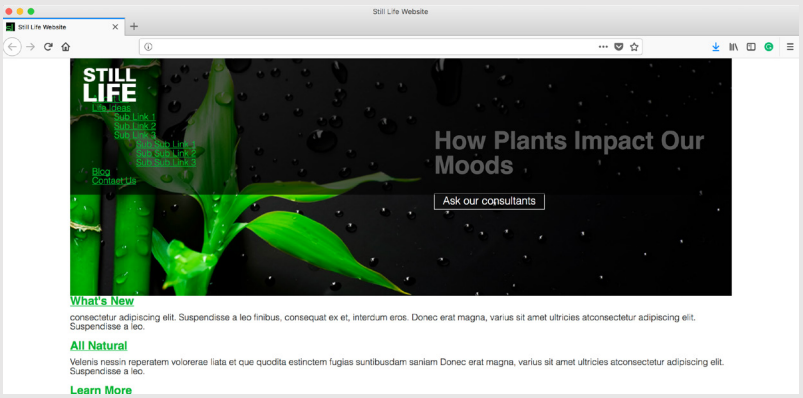
Properties to add:

- z-index: 1

Your code should look like this:

```
▼ header a.logo {
      z-index:1;
      position: absolute;
      display: block;
      width: 160px;
      height: 66px;
      background: url(../images/logo.svg) no-repeat 0 0;
      background-size: contain;
      top: 15px;
      left: 20px;

  }
```

Read up on z-index (Which is a very useful and cool thing)
https://developer.mozilla.org/en-US/docs/Web/CSS/z-index?v=control



☑ Page looks good, let's keep moving!

STEP 4

WATCH VIDEO: Styling the Navigation

We are now going to address the list items within our navigation.

WORKING IN SCREEN.CSS

We need to remove the bullets from our list, and while we are at it we will clear the margins and the padding.

Properties to add:
nav ul

- list-style none
- margin 0 on all four sides
- padding 0 on all four sides

Your code should look like this:

```
/* Navigation */

nav {
    background-color: rgba(0,0,0,.65);
    position: absolute;
    top: 0px;
    left: 0px;
    padding: 50px 0 0 0;
    width: 100%;
}

nav ul {
    list-style: none;
    margin: 0;
    padding: 0;

}
```

Your nav list should now be flushed left with no bullets.

STEP 5

WATCH VIDEO: Styling the Navigation

STYLING THE NAVIGATION

We have reached the point where we can now style the anchor tags, the list items and add pseudo-classes for the hover of our sub links.

WORKING IN SCREEN.CSS

Explanation:

nav ul li a

- display: inline-block
  This will make the links behave more like graphics rather than inline elements
- color: #fff (white)
- padding: 10px 20px
- text-decoration: none
- width: 125px
  Setting a width on these individual anchor tags when they have a display type set to inline-block will allow us to define the width of the submenus.
- position: relative

nav ul li a:visited

- color: #fff

nav ul li a:hover

- background-color:#b6dcae
  this is defining the color of the background on the sublinks

We also want to have the list item change color on hover as well. This way when we hover over a top-level element and then move down, the top level element will still display a hover state.

JUMP Keeping neat code is very important. Adding in new rules in the order of nesting is important for keeping neat code. Let's go back up in our page to above the nav ul li a line.

nav ul li:hover

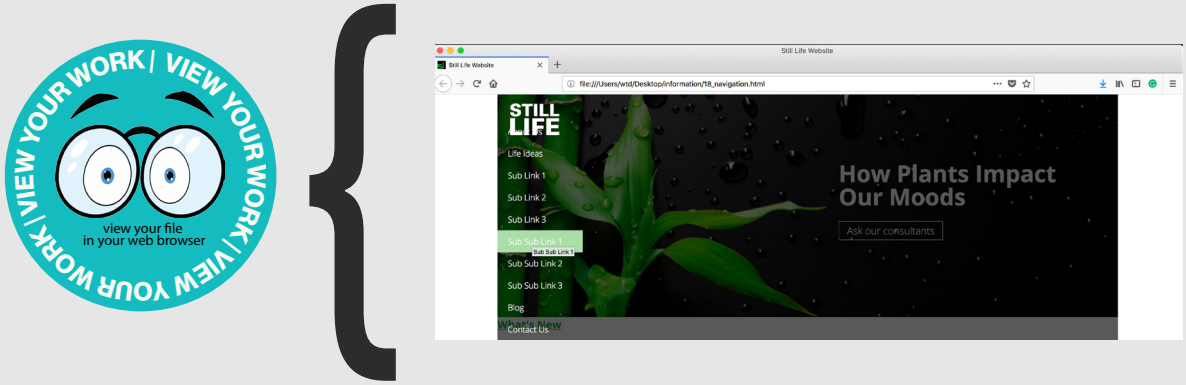
- background-color: #39b54a

Your code should look like this -->

```
/* Navigation */

nav {
    background-color: rgba(0,0,0,.65);
    position: absolute;
    top: 0px;
    left: 0px;
    padding: 50px 0 0 0;
    width: 100%;
}

nav ul {
    list-style: none;
    margin: 0;
    padding: 0;

}

nav ul li a {
    display: inline-block;
    color: #fff;
    padding: 10px 20px;
    text-decoration: none;
    width: 125px;
    position: relative;
}

nav ul li a:visited {
    color: #fff;
}

nav ul li a:hover {
    background-color:#b6dcae;
}
```

You should see your navigation starting to take shape. All the text will still be flushed left, with some space between each. Upon rollover, you should observe the background colors.

Yay! Let's keep on moving.



☑ Page looks good, let's keep moving!

STEP 6

WATCH VIDEO: Styling the Navigation

TOP LEVEL ELEMENTS OF THE NAVIGATION

Selector

The next technique we are going to use is to be specific in defining direct children of a parent element.

> less than selects all direct descendants/children.
greater than > selector will only select all immediate descendants.

Give this article a read. Number 8 is the selector we are referring to in this technique.
https://code.tutsplus.com/tutorials/the-30-css-selectors-you-must-memorize--net-16048

WORKING IN SCREEN.CSS

We will add another comment section /* Navigation - Top Level */

Explanation of Properties to add:

nav > ul - This will only target an unordered list if it is immediately the child of a navigation element.
- padding-left 200px

Still within the same section, we are going to target specifically the list items that are the direct child of ul, which is a direct child of the navigation.

nav > ul > li

- float left

Your code should look like this:

```
/* Navigation - Top Level*/

nav > ul {
    padding-left: 200px;
}

nav > ul > li {
    float: left;
}
```

Note: When we are floating elements we need to add a pseudo element to clear out the floats.

JUMP We want to make sure our code is neat so we will position this next line of code right under the first nav rule in the /* Navigation */ section.

nav::after

- content ' ' (we have to define content but we will not put anything between the tick marks)
- display block
- clear both

Your code should look like this:

```
/* Navigation */

nav {
    background-color: rgba(0,0,0,.65);
    position: absolute;
    top: 0px;
    left: 0px;
    padding: 50px 0 0 0;
    width: 100%;
}

nav::after {
    content:'';
    display: block;
    clear: both;
}

nav ul {
    list-style: none;
    margin: 0;
    padding: 0;

}
```

BE MINDFUL OF WHERE THIS LINE OF CODE IS LOCATED WITHIN YOUR CSS.

STEP 7

WATCH VIDEO: Styling the Navigation

STYLING THE NAVIGATION
Next step is to hide the submenus and then have them show when we hover over the top-level list items. We have three tiers of links and we only want the links to show when they are hovered over. We want the position of the lists to denote the different level of tiers.

WORKING IN SCREEN.CSS

We will add more rules to the nav section, right above the comment for /* Navigation - Top Level */

We will define unordered lists, which are inside of other unordered lists.

Here is a link to read up on ordered and unordered lists:
https://www.w3schools.com/HTML/html_lists.asp

Explanation of properties to add:

nav ul ul
- position absolute
- top 100%
- background-color #39b54a

nav ul ul li
- position relative (this targets a third tier of sublinks in relation to the parent)

nav ul ul ul
- left 100%
- top 0px

Note: We positioned the nested unordered list because we want to make sure that this item will show up directly under the list items, but we don't want the navigation element to wrap around and encompass the entire height of those menus.

We added the pseudo element to the nav element is to make sure that the height of the nav will always encompass the top level items that are set to float.

Your code should look like this:

```
nav ul ul {
    position: absolute;
    top: 100%;
    background-color: #39b54a;

}

nav ul ul li {
    position: relative;
}

nav ul ul ul {
    left: 100%;
    top: 0px;
}
```

STEP 8

WATCH VIDEO: Styling the Navigation

WORKING IN SCREEN.CSS

Before we view the navigation let's finish a bit more. We are going to be jumping around a bit. So watch the placement of your rules. Instructions are on the left, code is on the right.

Go back to nav ul ul
- add in display none

Your code should look like this:

```
nav ul ul {
    position: absolute;
    top: 100%;
    background-color: #39b54a;
    display: none;
}
```

Underneath the nav ul rule
- add in nav ul li:hover
- background-color #39b54a

Your code should look like this:

```
nav ul li:hover {
    background-color: #39b54a;
}
```

Underneath the nav ul li:hover rule
- add in nav ul li:hover > ul
- display block

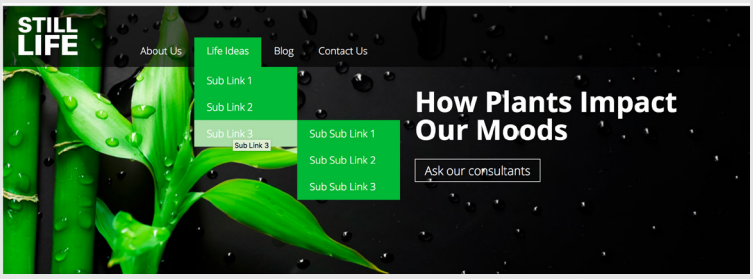
Your code should look like this:

```
nav ul li:hover > ul {
    display: block;
}
```

Go to the /* Navigation - Top Level */ section.
At the bottom of this section add in this rule

Your code should look like this:

```
nav > ul > li > a {
    width: auto;
    padding: 10px 20px 15px 20px;
}
```



☑ Page looks good, let's keep moving!

STEP 9

WATCH VIDEO: Styling the Navigation

Last thing we want to do to our nav is add an indicator arrow that this menu has sub menus. This gets tricky and may be mind-blowing. But I want to do is show you this technique so you can utilize it again in other scenarios.

WORKING IN SCREEN.CSS

Go back to the /* Navigation - Top Level */ section

Follow the lines of this code carefully:

```
nav a[aria-haspopup="true"]::after {
    content:'';
    display: block;
    width: 0px;
    height: 0px;
    position: absolute;
    top: 16px;
    right: 15px;

}
```

WORKING IN INDEX.HTML

Back in our index.html file, we will be adding in the ARIA code to the HTML nav menu items with sublinks This is the code we will be adding in: aria-haspopup="true" (If you copy this code, check your quotes.)

```
<nav>
    <ul>
        <li><a title="About Us" href="#" >About Us</a></li>
        <li><a title="Life Ideas" href="#" aria-haspopup="true">Life Ideas</a>
        <!-- Subnav starts here -->
            <ul>
                <li><a title="Sub Link 1" href="#">Sub Link 1</a></li>
                <li><a title="Sub Link 2" href="#">Sub Link 2</a></li>
                <li><a title="Sub Link 3" href="#" aria-haspopup="true">Sub Link 3</a>
        <!-- Sub Subnav starts here -->
                <ul>
                    <li><a title="Sub Sub Link 1" href="#">Sub Sub Link 1</a></li>
                    <li><a title="Sub Sub Link 2" href="#">Sub Sub Link 2</a></li>
                    <li><a title="Sub Sub Link 3" href="#">Sub Sub Link 3</a></li>
                </ul>
                </li>
            </ul>
        </li>
        <li><a title="Blog" href="#">Blog</a></li>
        <li><a title="Contact Us" href="#">Contact Us</a></li>
    </ul>
</nav>
```

STEP 10

WATCH VIDEO: Styling the Navigation

CSS TRICKS
We are going to use some of the great features of CSS to draw our triangle icon. You can do so many things with CSS, so delve into it. Here are a few links to get you started.

CSS Arrows: https://www.w3schools.com/howto/howto_css_arrows.asp
CSS Shape: https://css-tricks.com/the-shapes-of-css/
CSS Triangle: https://css-tricks.com/snippets/css/css-triangle/

We are going to create a right pointing arrow and a down pointing arrow.

YEP - you could just add in an icon. But what fun is that, let's make it. Plus this is faster!!

WORKING IN SCREEN.CSS

Within the same aria rule

Properties to add:
- border-top: 4px solid transparent;
- border-bottom: 4px solid transparent;
- border-left: 4px solid #fff;

Your code should look like this:

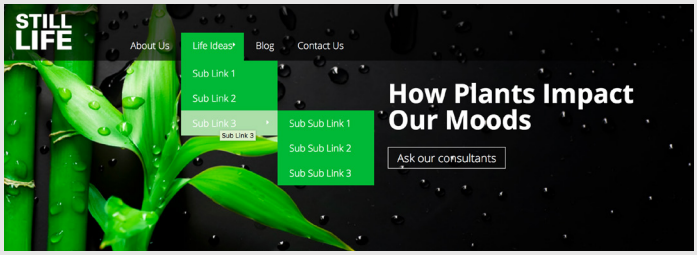
```
nav a[aria-haspopup="true"]::after {
    content:'';
    display: block;
    width: 0px;
    height: 0px;
    position: absolute;
    top: 16px;
    right: 15px;
    /* CSS for creating an arrow */
    border-top: 4px solid transparent;
    border-bottom: 4px solid transparent;
    border-left: 4px solid #fff;
}
```

I put in a comment to indicate this was the code I used to create the triangle

Watch your code as the smallest mistake can produce no results.

The arrows are on the items with sublinks. If you don't see them, check your code or get with me and let's work it out.

If you got it - Let's move on.



☑ Page looks good, let's keep moving!

STEP 11

WATCH VIDEO: Styling the Navigation

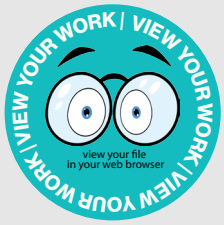
WORKING IN SCREEN.CSS

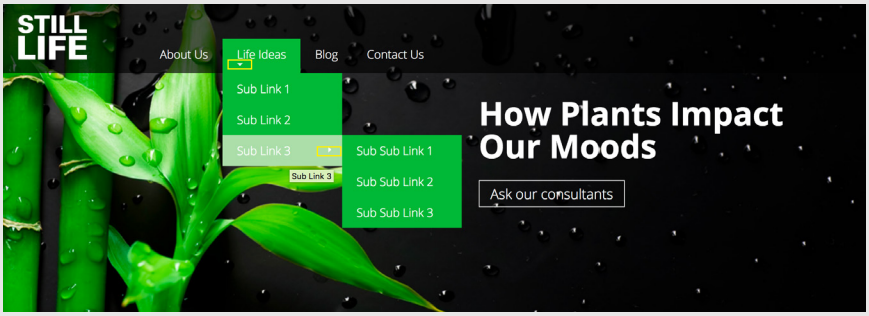
Our goal now is to make the arrows in the top level nav point down. Under the last rule in the /* Navigation - Top Level */ section. We will add:

```
nav > ul > li > a[aria-haspopup="true"]::after {
      /* CSS for creating a down pointing arrow */
      border-left: 4px solid transparent;
      border-right: 4px solid transparent;
      border-top: 4px solid #fff;
      left: 20px;
      right: auto;
      bottom: 6px;
      top: auto;
  }
```

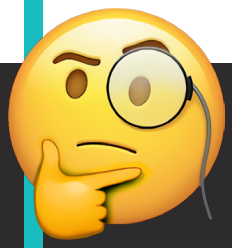
Note: We don't need to define the content element here because it already has a pseudo-element above it. The specificity of this rule allows for us to piggy-back off the pseudo element above it.

We defined the right pointing arrow first, in this case we have more right pointing arrows. This allows us to create those arrows and then just override the arrows on the top level to point them down.





☑ Page looks good, let's keep moving!



In Case You Were Wondering

What is an ARIA?

ARIA stands for Accessible Rich Internet Applications. ARIA can modify existing element semantics or add semantics to elements where no native semantics exist. It can also express semantic patterns that don't exist at all in HTML, like a menu or a tab panel. Often, ARIA lets us create widget-type elements that wouldn't be possible with plain HTML.

The menu we created will also work on Google Android and Apple iOS touch devices. These devices will activate the hover psuedo-class when you tap on the top-level navigation items.

Accessible Rich Internet Applications (ARIA).
https://developer.mozilla.org/en-US/docs/Web/Accessibility/ARIA

Whew, You did it.
Fantastic Work.
Take a break!