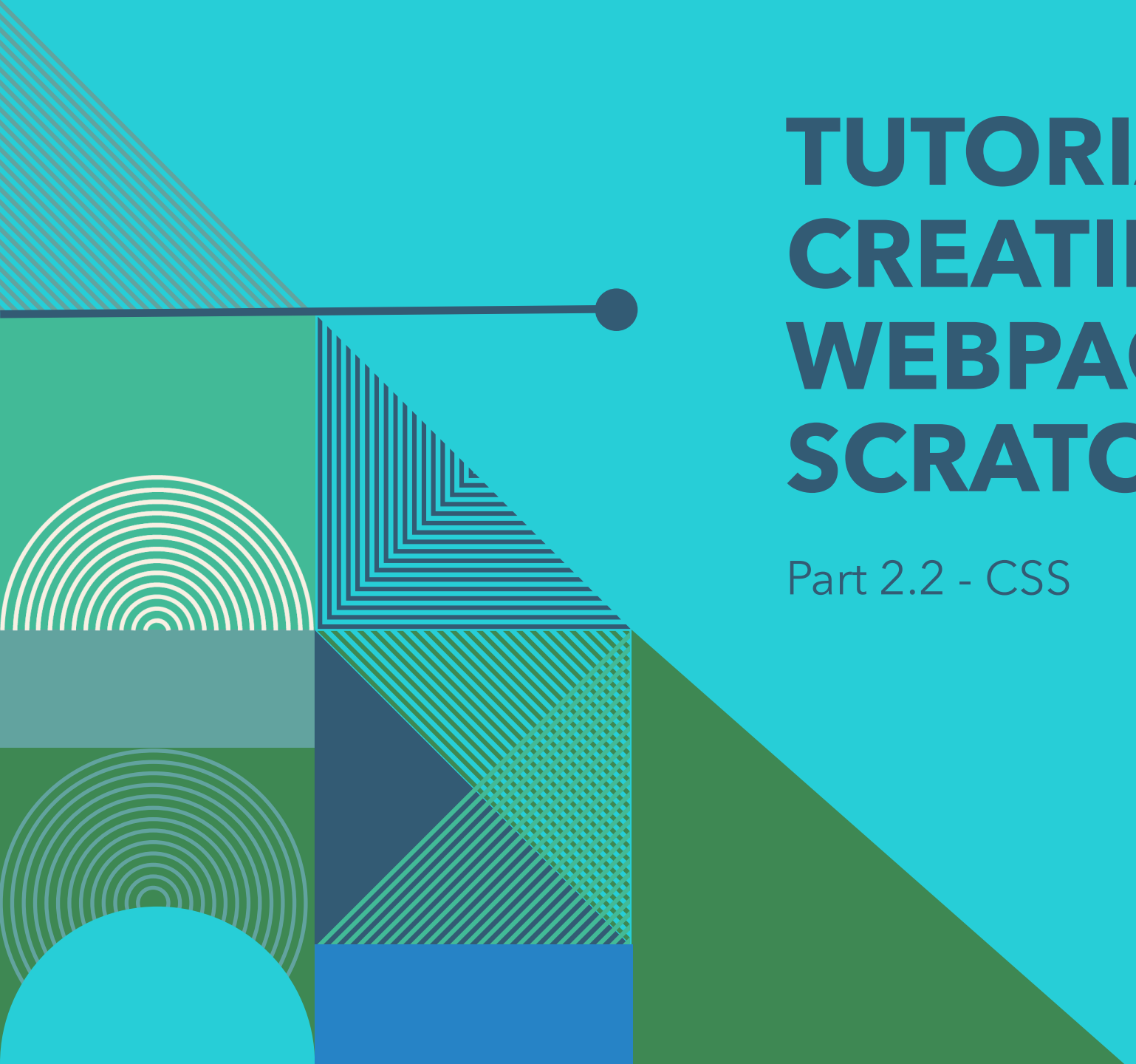


TUTORIAL: CREATING A WEBPAGE FROM SCRATCH

Part 2.2 - CSS

WHY USE CSS?

You've used HTML to build your webpage, but it doesn't have a lot of style yet. Let's add some with CSS.

CSS stands for cascading style sheet, a somewhat confusing name for a language that you use to format your page so that it looks just the way you want.

CSS is used to define styles for your web pages, including the design, layout and variations in display for different devices and screen sizes.

CSS can get super-complicated, but the basic principles aren't hard to master.

CSS SOLVED A BIG PROBLEM

HTML was NEVER intended to contain tags for formatting a web page!
HTML was created to describe the content of a web page, like:

```
<h1>This is a heading</h1>
<p>This is a paragraph.</p>
```

When tags like <font>, and color attributes were added to the HTML 3.2 specification, it started a nightmare for web developers. Development of large websites, where fonts and color information were added to every single page, became a long and expensive process.

To solve this problem, the World Wide Web Consortium (W3C) created CSS.

CSS removed the style formatting from the HTML page!

SELECTORS, PROPERTIES, & VALUES

For each selector there are “properties” inside curly brackets, which simply take the form of words such as color, font-weight or background-color.

A value is given to the property following a colon (NOT an “equals” sign). Semi-colons are used to separate the properties

```
body {
   font-size: 14px;
   color: navy;
}
```

This will apply the given values to the font-size and color properties to the body selector.

So basically, when this is applied to an HTML document, text between the body tags (which is the content of the whole window) will be 14 pixels in size and navy in color.

Whereas HTML has tags, CSS has selectors. Selectors are the names given to styles in internal and external style sheets. In this CSS Tutorial we will be concentrating on HTML selectors, which are simply the names of HTML tags and are used to change the style of a specific type of element.

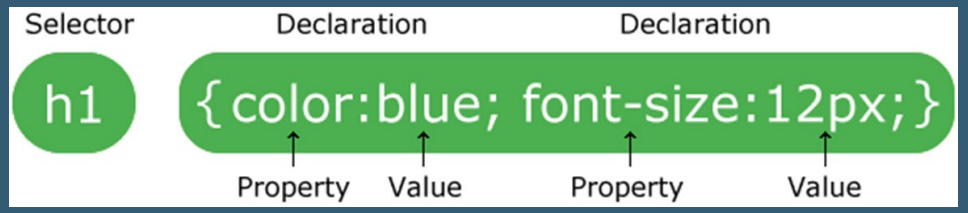
CSS SYNTAX

The selector points to the HTML element you want to style.

The declaration block contains one or more declarations separated by semicolons.

Each declaration includes a CSS property name and a value, separated by a colon.

Multiple CSS declarations are separated with semicolons, and declaration blocks are surrounded by curly braces.



Example

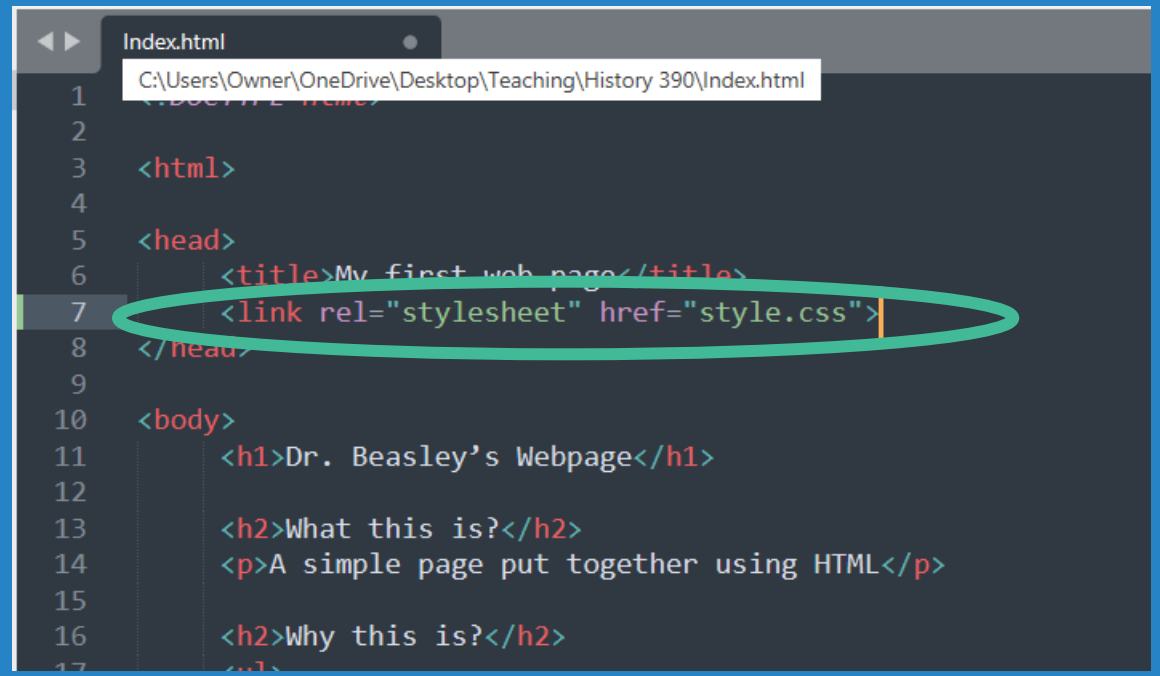
In this example all <p> elements will be center-aligned, with a red text color:

```
p {
  color: red;
  text-align: center;
}
```

Example Explained

- p is a selector in CSS (it points to the HTML element you want to style: <p>).
- color is a property, and red is the property value
- text-align is a property, and center is the property value

LINK YOUR HTML DOCUMENT TO A CSS DOCUMENT



You can add CSS styles to your HTML document in a few different ways, but I like to keep all of my CSS rules in a separate document.

Open a new text document in Sublime and save it in the same place as your html document, with the name style.css.

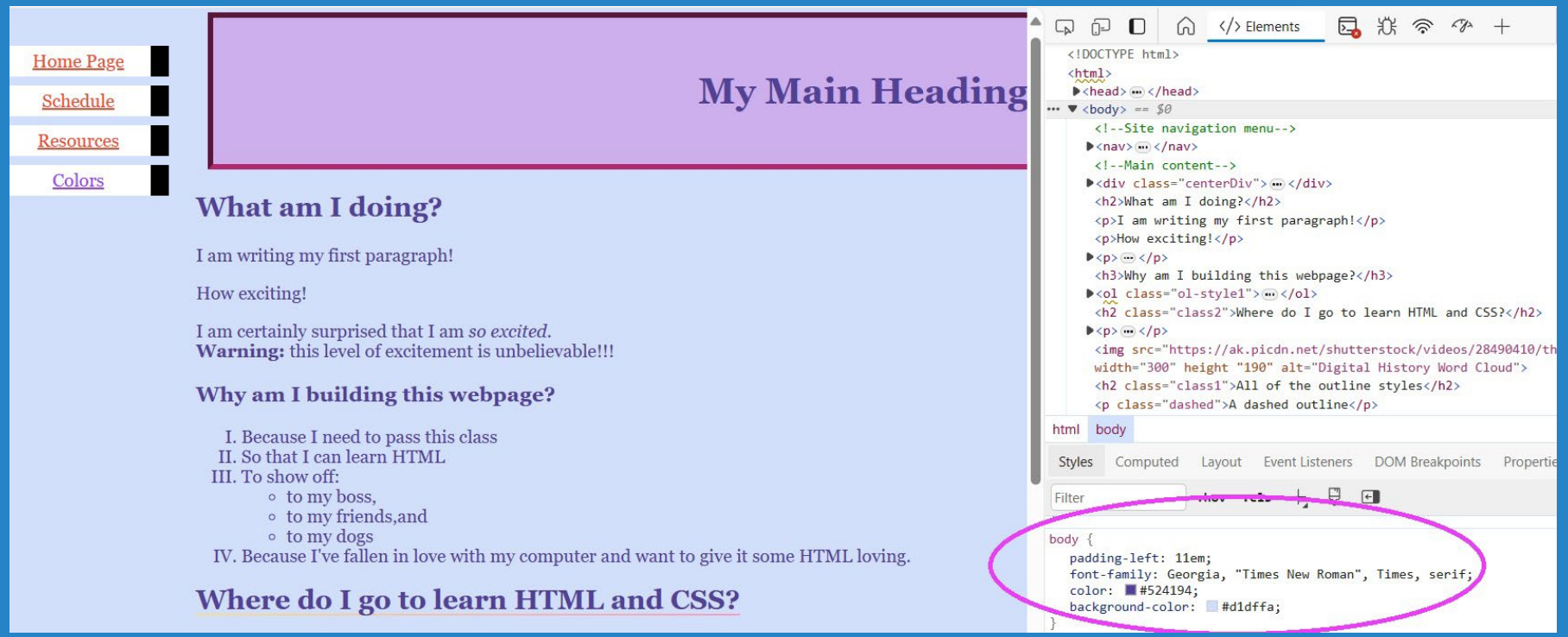
(You can call your stylesheet whatever you want, but style is customary.)

Now we have to tell the HTML document to look for the CSS document in order to receive information about styles. Luckily, that's not too hard. Inside the<head> tags on your html document, type:

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

Now you should be linked!

HOW CSS STYLE WORKS



1. The basic rule is that you specify the html tag you'd like your rule to affect and then say what you want to do to the content inside the tag. Then all of the content inside of that tag will be affected.

2. In the example above, I've specified that all of the content inside <body> tags should be made blue and transformed into the Georgia font. Notice that the navigation tags aren't affected. That's because the navigation menu is not inside the <body> tags.

3. As you can see, your rules go inside angle brackets, which look like this { } and are separated by semicolons.

LENGTHS AND PERCENTAGES

There are many property-specific units for values used in CSS, but there are some general units that are used by a number of properties and it is worth familiarizing yourself with these before continuing.

px (such as font-size: 12px) is the unit for pixels.

em (such as font-size: 2em) is the unit for the calculated size of a font. So "2em", for example, is two times the current font size.

pt (such as font-size: 12pt) is the unit for points, for measurements typically in printed media.

% (such as width: 80%) is the unit for… wait for it… percentages.

Other units include pc (picas), cm (centimeters), mm (millimeters) and in (inches).

When a value is zero, you do not need to state a unit. For example, if you wanted to specify no border, it would be border: 0.

1. "px" in this case, doesn't actually necessarily mean pixels - the little squares that make up a computer's display - all of the time. Modern browsers allow users to zoom in and out of a page so that, even if you specify font-size: 12px, or height: 200px, for example, although these will be the genuine specified size on a non-zoomed browser, they will still increase and decrease in size with the user's preference. It's a good thing. Trust me.

ADDING A NAVIGATION MENU

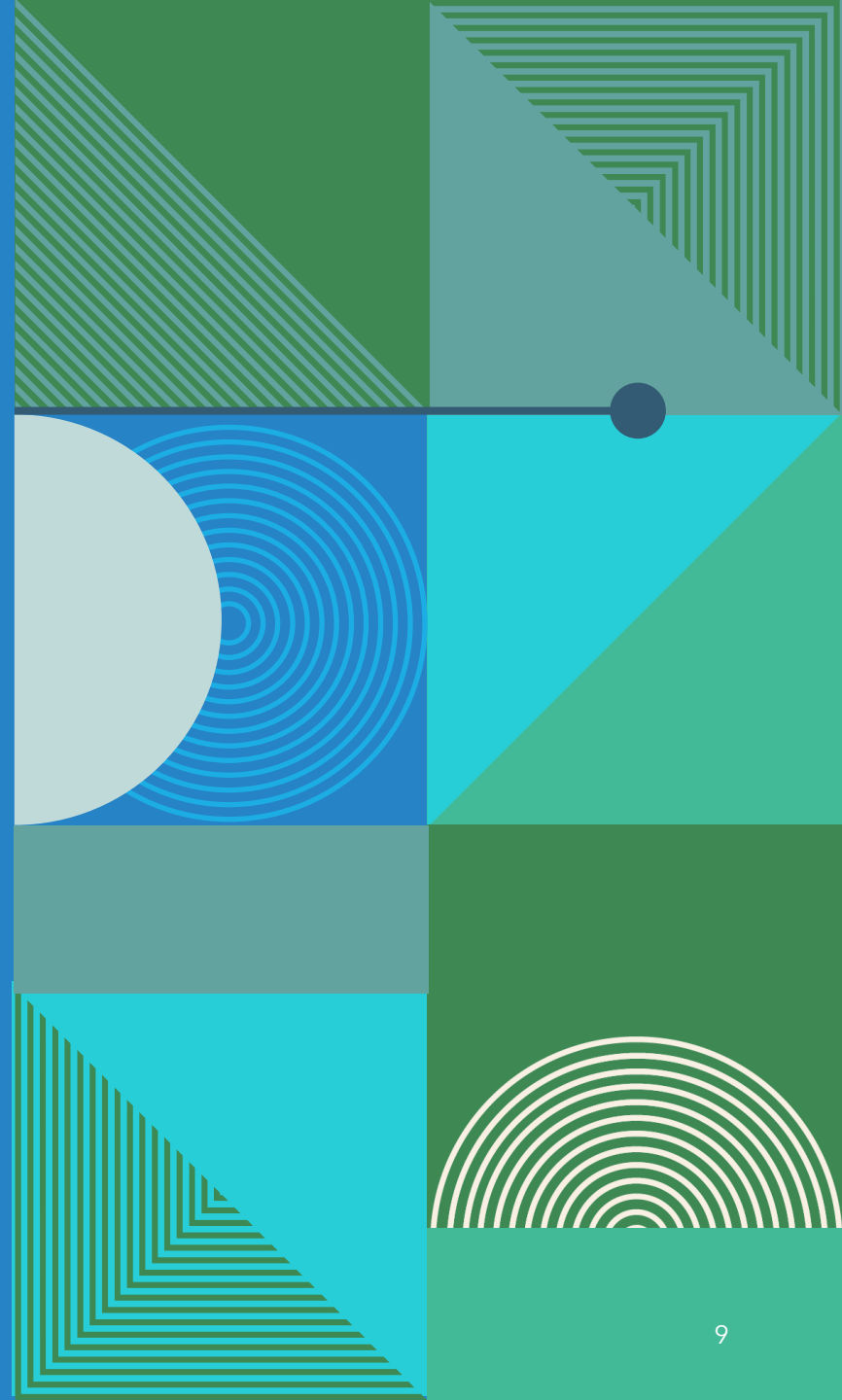
The list at the top of the HTML page is meant to become a navigation menu. Many Web sites have some sort of menu along the top or on the side of the page and this page should have one as well. We will put it on the left side, because that is a little more interesting than at the top…

The menu is already in the HTML page. It is the <ul> list at the top. The links in it don't work, since our "Web site" so far consists of only one page, but that doesn't matter now. On a real Web site, there should not be any broken links, of course.

So we need to move the list to the left and move the rest of the text a little to the right, to make room for it.

The CSS properties we use for that are 'padding-left' (to move the body text) and 'position', 'left' and 'top' (to move the menu).

There are other ways to do it, but this one is OK for our purposes.



ADDING A NAVIGATION MENU, CONT.

In the editor window, add the following lines to the HTML file:

```
body {
    padding-left: 11em;
    font-family: Georgia, "Times New Roman",
        Times, serif;
    color: purple;
    background-color: #d8da3d }
  ul.navbar {
    position: absolute;
    top: 2em;
    left: 1em;
    width: 9em }
```

If you save the file again and reload it in the browser, you should now have the list of links to the left of the main text. That already looks much more interesting, doesn't it?

A NAVIGATION MENU, EXPLAINED

```
body {
  padding-left: 11em;
  font-family: Georgia, "Times New Roman",
      Times, serif;
  color: purple;
  background-color: #d8da3d }
ul.navbar {
  position: absolute;
  top: 2em;
  left: 1em;
  width: 9em }
h1 {
  font-family: Helvetica, Geneva, Arial,
      SunSans-Regular, sans-serif }
```

1. The 'position: absolute' says that the ul element is positioned independently of any text that comes before or after it in the document and the 'left' and 'top' indicate what that position is. In this case, 2em from the top and 1em from the left side of the window.

2. '2em' means 2 times the size of the current font. E.g., if the menu is displayed with a font of 12 points, then '2em' is 24 points. The 'em' is a very useful unit in CSS, since it can adapt automatically to the font that the reader happens to use.

3. Most browsers have a menu for increasing or decreasing the font size: you can try it and see that the menu increases in size as the font increases, which would not have been the case, if we had used a size in pixels instead.

STYLING LINKS

The navigation menu still looks like a list, instead of a menu. Let's add some style to it. We'll remove the list bullet and move the items to the left, to where the bullet was. We'll also give each item its own white background and a black square. (Why? no particular reason, just because we can.)

Styling the Navigation, Part 1

```
body {
padding-left: 11em;
font-family: Georgia, "Times New Roman",
Times, serif;
color: purple;
background-color: #d8da3d }
ul.navbar {
list-style-type: none;
padding: 0;
margin: 0;
position: absolute;
top: 2em;
left: 1em;
width: 9em }
```

```
h1 {
   font-family: Helvetica, Geneva, Arial,
        SunSans-Regular, sans-serif }
ul.navbar li {
   background: white;
   margin: 0.5em 0;
   padding: 0.3em;
   border-right: 1em solid black }

[etc.]
```

STYLING LINKS

You can style links within the body of your webpage.

In HTML, hyperlinks are created with <a> elements, so to specify the color, we need to add a style rule for "a".

To differentiate between visited and unvisited links, CSS provides two "pseudo-classes" (:link and :visited). They are called "pseudo-classes" to distinguish them from class attributes, that appear in the HTML directly, e.g., the class="navbar" in our example.

In addition, links can be styled differently depending on what **state** they are in.

The four links states are:

a:link - a normal, unvisited link	a:visited - a link the user has visited	a:hover - a link when the user mouses over it	a:active - a link the moment it is clicked

When setting the style for several link states, there are some order rules:

a:hover MUST come after a:link and a:visited	a:active MUST come after a:hover

```
…
/* unvisited link */
a {
  color: hotpink;
}

/* visited link */
   a:visited {
      color: #ff00ff;
   }

   /* mouse over link */
   a:hover {
      color: #00ff00;
      border-bottom: none;
   }

   /* selected link */
   a:active {
      color: #00ffff;
   }

[Etc.]
```

CONGRATULATIONS!

Hey, you're done! Nice work. If you have extra time, take a look at some of the more advanced ways you can use CSS.