Lesson 9: Building Your Site

Welcome Back!

You were introduced to the concept of page layout in the last lesson. We created a layout with a header and nav bar. So far, we have the header and navbar in one page of our site. However, we also externalized our style sheet, so that you can easily add the header and navbar in every page of the site.

Ideally, you want the same header appear at the top of each page in your site. This is done to help people recognize when they're still in your website (and not someone else's site). You also want the nav bar that provides links to all the page in the site to be consistent across all of the pages.

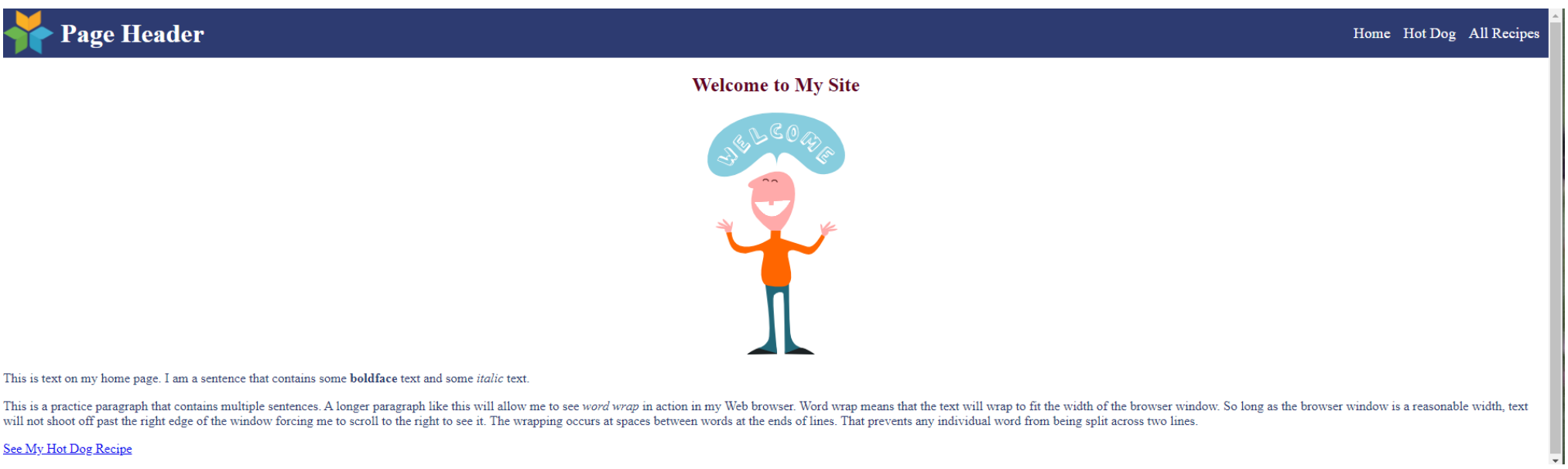
In this lesson we'll continue our journey into page layout design, and make sure that all of your pages are linked to the style sheet.

We've got a lot to cover, so let's jump right into it!

What is a Wrapper Div?

Our page design is looking pretty good. So long as the browser window is a reasonable width:

- If you stretch the browser window extra wide, it looks awfully spread out. That doesn't look so great.



- If you shrink the browser window really narrow, the white text of the navbar drops down to the main body of the page, so it becomes invisible against the white page background. That's not going to work.

To fix this, we can put a limit on how wide, or narrow, the page layout can be by using a ***wrapper div***. A wrapper div provides boundaries for the elements it contains.

Adding a Wrapper Div

Before you can do anything, you need to place all the page content...that is, everything between the <body>...</body> tags... inside a layout block where you can control the width. For this application ... tags are typically used.

Remember

The word *div* is short for division. It just refers to any "chunk" of content to which you want to apply styling to.

However, it's important that we call out this div from other divs in the page. To do this, we need to make a slight adjustment to our opening tag by giving it an **id attribute**. The id attribute is the identification of the tag. This attribute follows the typical *attribute* = "*value*" pair that you've used in other tags.

Tips for Creating Id Names

You can set the value for the id attribute to anything, because the id is a name you make up yourself. While you can use any name you like, you have to follow some rules.Here are some guidelines to keep in mind:

- The first character must be a letter ([A-Z] or [a=z])
- After the first letter any remaining characters can be letters, digits [0-9], underscores (_) or hyphens (-).
- Do not use spaces.
- Names are case-sensitive.
- Each id name on a page must be unique, which means no two elements on one page can have the same id.

Since this is our wrapper div, we'll set the value to "wrapper"for simplicity. The name *wrapper* is a made-up name that's actually fairly common for a div that contains all the page content. Another common name for that is *container*, as in .

Now, let's put all our page content from index.htm into a wrapper div now.

Here are the Steps

1. Open *index.htm* in your code editor.
2. Place your cursor just after the opening <body> tag and press ENTER to create a new line.
3. On the new line, type in the following comment:

```
<!-- Wrapper (container) div for entire page layout -->
```

4. Press **ENTER** again and add the following code to start your wrapper div:

```
<div id="wrapper">
```

5. Now, place your cursor just before the closing </body> tag and press **ENTER** to make a new line.

6. On the new line, type in the following code:

```
</div>
```

7. Press **ENTER** again and add the following closing comment:

```
<!-- End wrapper -->
```

8. Save your changes and close the page.

Now that we have our wrapper div, we'll add some styling to control its width.

Controlling the Layout Width

Don't worry about getting it perfect the first time, you can set and change these widths at any time. It's the kind of thing you can tweak until you find numbers that work for you.

Adding a Wrapper Div Style Rule

Once you give an element (like a div) an id name, it's easy to write a CSS style rule for it. You just use the id as the selector.

Important

To use an id as the selector, you need to lead with the # sign (e.g. #wrapper).

You don't even have to use the word div in this case. Just *#wrapper* is sufficient. Let's try!

Here are the Steps

1. Open the *stylesheet.css* external style sheet in your editor.

> ### Note
>
> You open a style sheet for editing just as you would a web page. Just right-click the stylesheet.css file and choose **Open With**, and select the name of your editor.
>
> 

2. Move the cursor so it's just after the body{} style rule and press **ENTER** to make a new line.

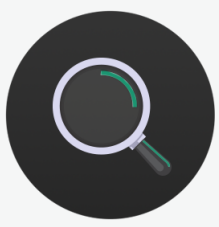
3. One the new line type in the following comment and style rule:

```
/* Wrapper div controls width of entire layout */
```

4. Press **ENTER** and copy/paste the following style rule in the empty line:

```
#wrapper {
  width: 800px;
}
```

5. Save your changes.

> 
>
> ### Take a Closer Look

Have one more look at your code to make sure you got it right. The new #wrapper{} style rule should be below the body{} style rule and above the header style rule.

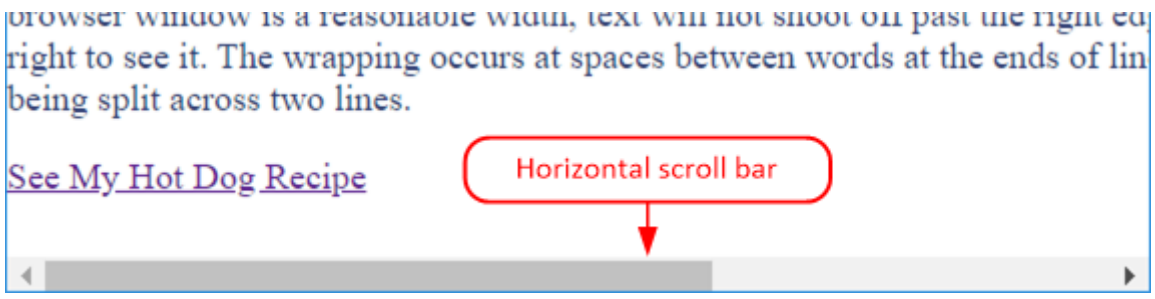
```
/* This is stylesheet.css an external style sheet */
  /* Style whole page */
  body {
  color: #2d3b71;
  margin: 0;
  }
  /* Wrapper div controls width of entire layout */
  #wrapper {
  width: 800px;
  }
  /* Page header at the top of each page */
  header {
  background-color: #2d3b71;
  color: white;
  height: 60px;
  line-height: 60px;
  }
  /* Navigation bar */
  nav {
  width: 250px;
  float: right;
  }
  /* Navigation bar links (<a> tags) */
  nav a {
  text-decoration:  none;
  color: white;
  font-size: 115%;
  }
  /* Level 1 and Level 2 headings */
  h1 {
  margin:0;
  }
  h2 {
  color: #64001f;
  }
```

To see if it worked, make sure you've saved all your work (both in the page and style sheet). Then open the page in a browser. You'll notice the content no longer widens and narrows with the browser window. It stays a fixed with.

- When the browser window is very wide, empty white space appears to the right of the content as you expand the size past the width of the wrapper.

- When the browser window is very narrow, the content doesn't word wrap and cuts off when the window is smaller than the wrapper width.

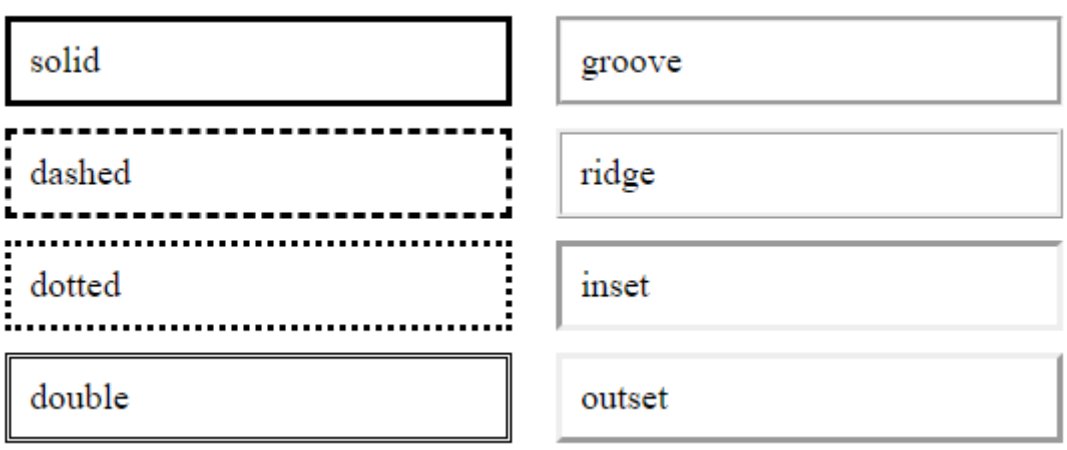
The page content stays the same width, regardless of how wide or narrow the browser window is. When the window is narrower than the wrapper width, a scroll bar will appear at the bottom of the browser window so you can scroll left and right to see the content.



You now have a wrapper div in your page, and the beginnings of a style rule for it in your external style sheet. That's just the start, there's a whole lot more to do.

Choosing a Border Style

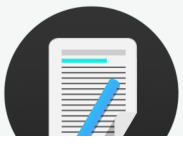
We added the tags for a wrapper div to our index.htm page, but right now it doesn't really look like much. We'll add some more styling to it in stages, starting with a adding a border around the wrapper. CSS allows you to place borders around any block element (such as a div, header, nav bar, paragraph). You can choose from a number of different styles shown below.



To achieve any of these border styles, just create a property:value pairing using the ***border*** property and enter the border style name as the value. For example, if you want a solid border you would code it as follows:

```
border:solid;
```

Note



By default, the border color is *black*. So, the border property puts a *black* border around an element unless you specify otherwise with the ***border-color*** property.

Try it out!

Here are the Steps

1. Open *stylesheet.css* in your editor.

2. Type the following CSS code into the the **#wrapper{}** style rule:

```
border:solid;
```

3. Save your changes to the style sheet.

Sneak Peek

If you typed the code correctly, and put it inside the #wrapper{} style rule, you should see a solid border around your content.

Page Header

Home Hot Dog All Recipes

Welcome to My Site

This is text on my home page. I am a sentence that contains some **boldface** text and some *italic* text.

This is a practice paragraph that contains multiple sentences. A longer paragraph like this will allow me to see *word wrap* in action in my Web browser. Word wrap means that the text will wrap to fit the width of the browser window. So long as the browser window is a reasonable width, text will not shoot off past the right edge of the window forcing me to scroll to the right to see it. The wrapping occurs at spaces between words at the ends of lines. That prevents any individual word from being split across two lines.

See My Hot Dog Recipe

Remember

When you change the code in the style sheet the change isn't automatically reflected on the webpage. To see the effects of the change you made, you first need to save the changes to the style sheet. Then, you need to reload the page in the browser using the browser's **Reload** or **Refresh** button. You can also hard refresh the page by pressing **CTRL** + **F5** for Windows (or **COMMAND + F5** for Mac).

You're welcome to experiment with other border styles such as *border:solid*, *border:double*, *border:dotted*, or some other style. Next, we'll look at the border width.

Styling a Border

Setting the Border Width

You can change the width of the border by adding it's width in pixels to the *border:* property value. You need to add a space between the border style and the border width. Let's change the border to that it's a thick, dashed line.

Here are the Steps

1. Open *stylesheet.css* in your editor.

2. Got to the *#wrapper{}* style rule.

3. Change the border style from solid to dashed:

```
border:dashed;
```

4. Place your cursor between the word dashed and the semicolon and press the **SPACEBAR**.

Important

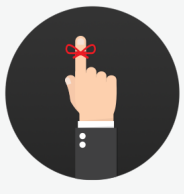
When placing multiple values for a property in a CSS style sheet, the values must be separated by a space.

5. Then set the border width to 10 pixels:

```
border: dashed 10px;
```

Remember

When determining size in pixels, use the abbreviation px after the number.



6. Save the style sheet changes, and refresh the page in the browser.

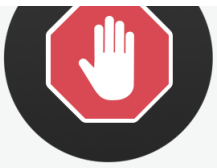
Sneak Peek

You should see a 10-pixel thick dashed border around the content within the wrapper.

You may think this dashed border is less than appealing, as it makes you page look a bit like a coupon. But don't worry, it's temporary. We'll fix it when we add some color.

Coloring the Border

If you don't specify a color for a border, it will be black. If you'd like your border to be a different color, you just need to add a color name or hex code as a value for the *border* property.



`

Important

If you use a hex code, rather than a color name, don't forget the # at the start of the hex code. For example:

```
border: solid 1px #0000ff;
```

The exact order of the values doesn't matter.

When adding values to a property, it doesn’t matter what order you add them in, but there does need to be a space between each value. For example, all six lines of code below display a solid silver border that is 1-pixel in width:

```
border: solid 1px silver;
border: solid silver 1px;
border: silver solid 1px;
border: silver 1px solid;
border: 1px silver solid;
border: 1px solid silver;
```

The border looks exactly the same in the browser regardless of how you order the three values.

For now, let's style our border as a 1-pixel wide, solid, silver line.

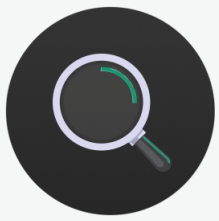
Here are the Steps

1. Open *stylesheet.css* in an editor.

2. Change the *#wrapper{}* style so it displays a solid 1-pixel silver border using the code below.

```
border: solid 1px silver;
```

3. Save your changes.



Take a Closer Look

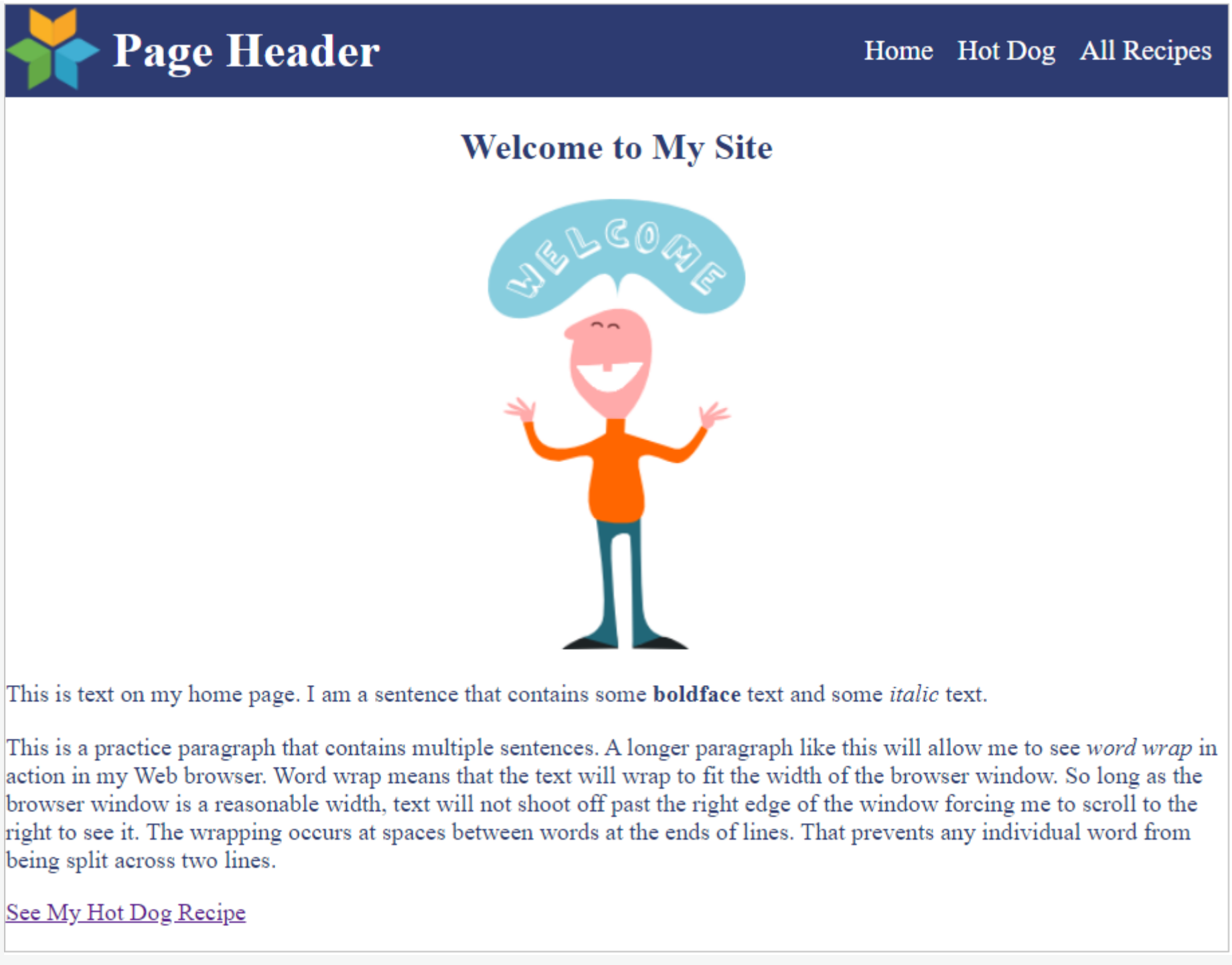
Your #wrapper{} style rules should now match the following:

```
/* Wrapper div controls width of entire layout */
  #wrapper {
  width: 800px;
  border: solid 1px silver;
  }
```

4. Open or Reload/Refresh *index.html* in your browser.

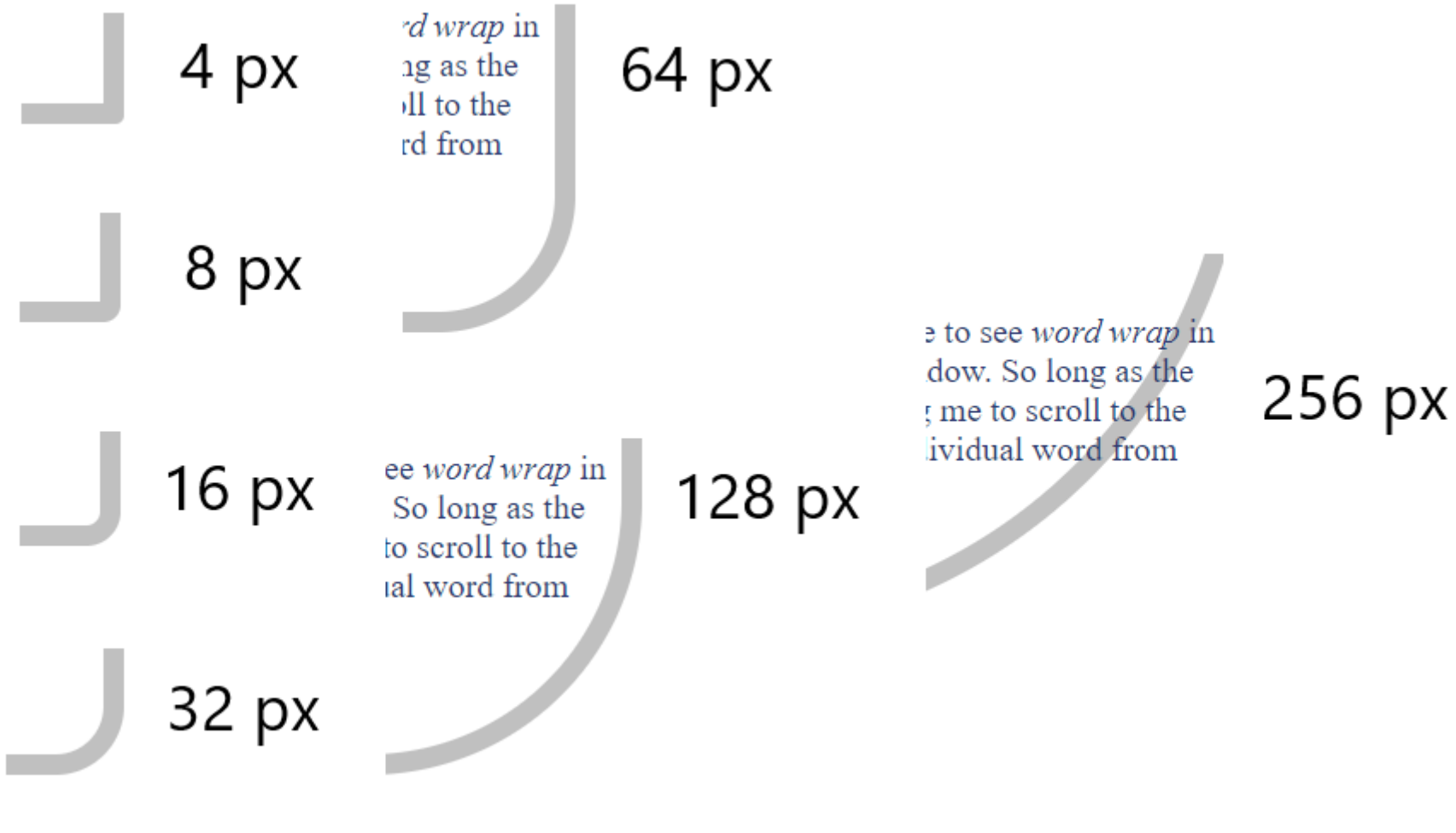
Sneak Peek

The border around your content should now have a thin silver appearance.

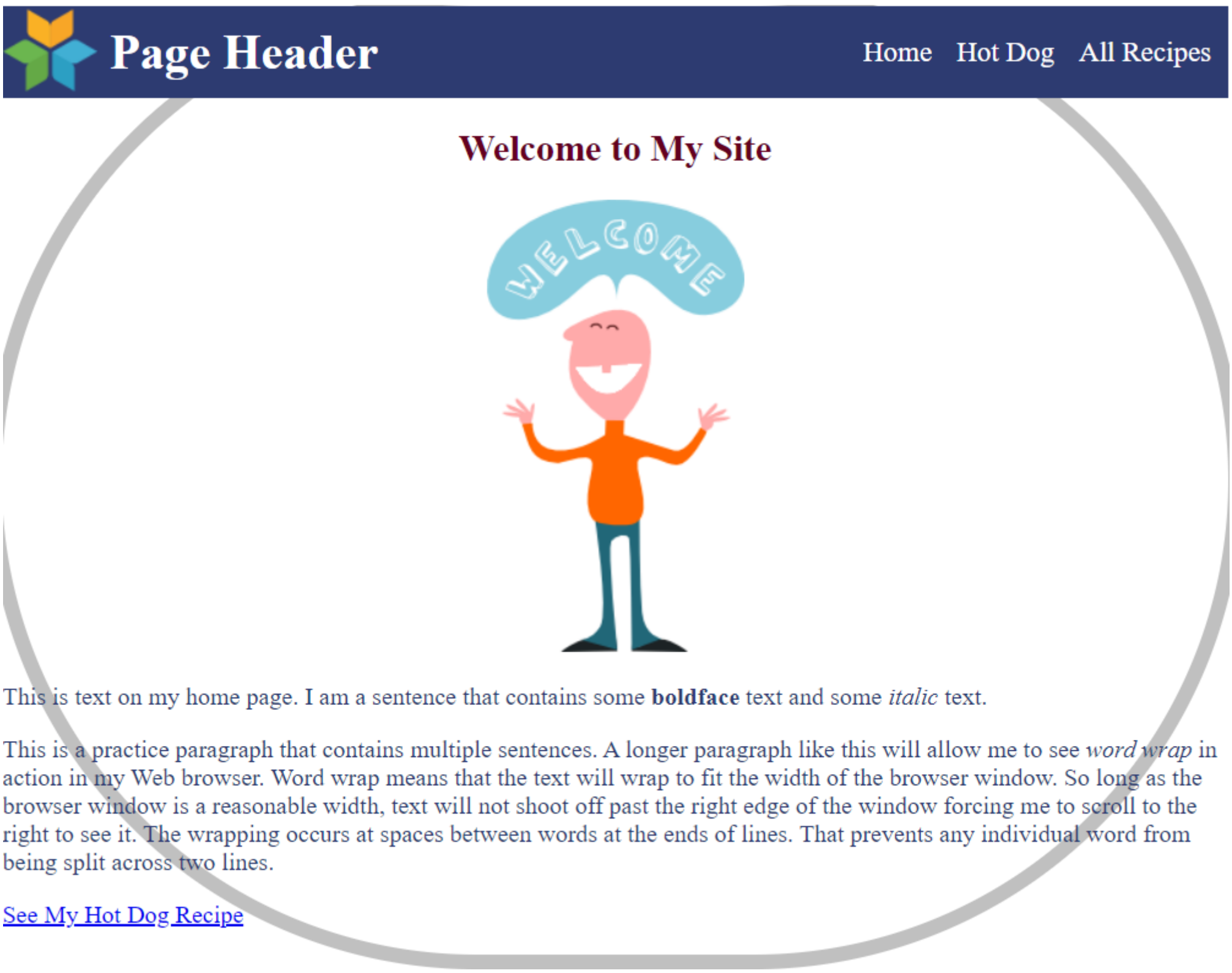


Rounding the Corners

By default, a border has square corners, as you've seen in our page so far. You can round those corners using the CSS ***border-radius*** property. Then you can specify how rounded the border is by defining the size of the radius you want for the corner in pixels.



The lower the size of the radius, the smaller the rounding of the corners. A border-radius of 4px would create a slightly rounded corner, whereas a border-radius of 16 px would create a significantly rounded corner. The bigger the border-radius, the more rounded the corner. A border-radius of 360px would create an almost circular border on your page.



Go ahead and play around with the border-radius to see how it works. We'll set our border-radius to 4px.

Let's give it a try!

Here are the Steps

1. Open *stylesheet.css* in an editor.

2. Go to the **#wrapper{}** style rule.

3. Add the border property:value pairing as a new line, as in the code below:

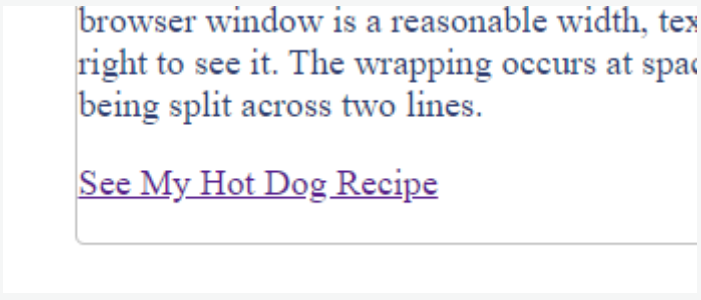
```
#wrapper {
  width: 800px;
  border: solid 1px silver;
  border-radius: 4px;
}
```

4. Save your changes.

5. Open or reload/refresh the *index.htm* page in the browser.

Sneak Peek

You should now see some subtle rounding on the corners. Our border-width is set to 1 px, and the border-radius is set to 4px, and the border-color is set to silver. Feel free to experiment with the radius, and other widths and colors on your own.



By now things are really starting to take shape, but having the whole layout flush against the left side of the screen looks a bit cramped. Particularly on larger monitors. To fix this, we'll adjust the margins around the edges of the wrapper.

Centering the Layout

We've been making great progress on our page. You're probably starting to get the hang of things now, and may even be able to guess the next steps. So, let's keep the momentum, starting with positioning the layout on the page.

We want to center the entire layout. This is different than centering text and pictures inside the layout. To make this happen, we need to center the wrapper div as a whole. You don't use text-align for that. Instead, you have to use the CSS ***margin property***.

Adding and Adjusting Margins

The margin property allows you to set the margins around a given block. A ***margin*** is the empty space *outside* the border. The margin is measured in pixels. However, there are a number of ways to use the margin property.

The most common is to use margin shorthand where you can either set all the margins at once, each one individually, or in groups depending on the syntax.

Text equivalent start.

Topic	Information

margin: *all*;	This syntax allows you to set all margins as the same size.
	
	This code sets all margins to 10px:
	
	`margin: 10px;`
	
	While this code centers the block on the page horizontally and vertically:
	
	`margin: auto;`

margin: *vertical horizontal;*	This syntax allows you to set the top and bottom margins as the same value, and the left- and right-side margins as the same value. This code sets the top and bottom margins to 10px, and centers the block on the page horizontally: `margin: 10px auto;`

Topic	Information
margin: *top horizontal bottom;*	This syntax allows you to set the top and bottom margins as separate values, and the left- and right-side margins as the same value. This code sets the top margin to 10px, the bottom margin to 5px, and side margins to 20px: `margin: 10px 20px 5px;`
margin: *top right bottom left;*	This syntax allows you to set the top and bottom margins as separate values, and the left- and right-side margins as separate values. This code sets the top margin to 10px, the right-side margin to 20px, the bottom margin to 5px, and the left-side margin to 2px: `margin: 10px 20px 5px 2px;`

Read the topic in the first column. Then read the second column for the information.

Text equivalent stop.

Let's look at it another way, just to help you get your head wrapped around it:

Example	Top Margin	Left-Side Margin	Right-Side Margin	Bottom Margin
margin: 10px;	10px	10px	10px	10px
margin: 10px auto;	10px	auto	auto	10px
margin: 10px 20px 5px;	10px	20px	20px	5px
margin: 10px 20px 2px 5px;	10px	20px	2px	5px

Margin Settings

You don't need to spend a lot of time memorizing this. As with anything, you can always do a web search (try the search phrase *css margins*) at any time to get more info.

Let's try it out!

Here are the Steps

1. Open *stylesheet.css* in your editor.

2. Go to the **#wrapper{}** style rule.

3. Set the vertical margin to *10px* and set the horizontal margin to *auto* to as below:

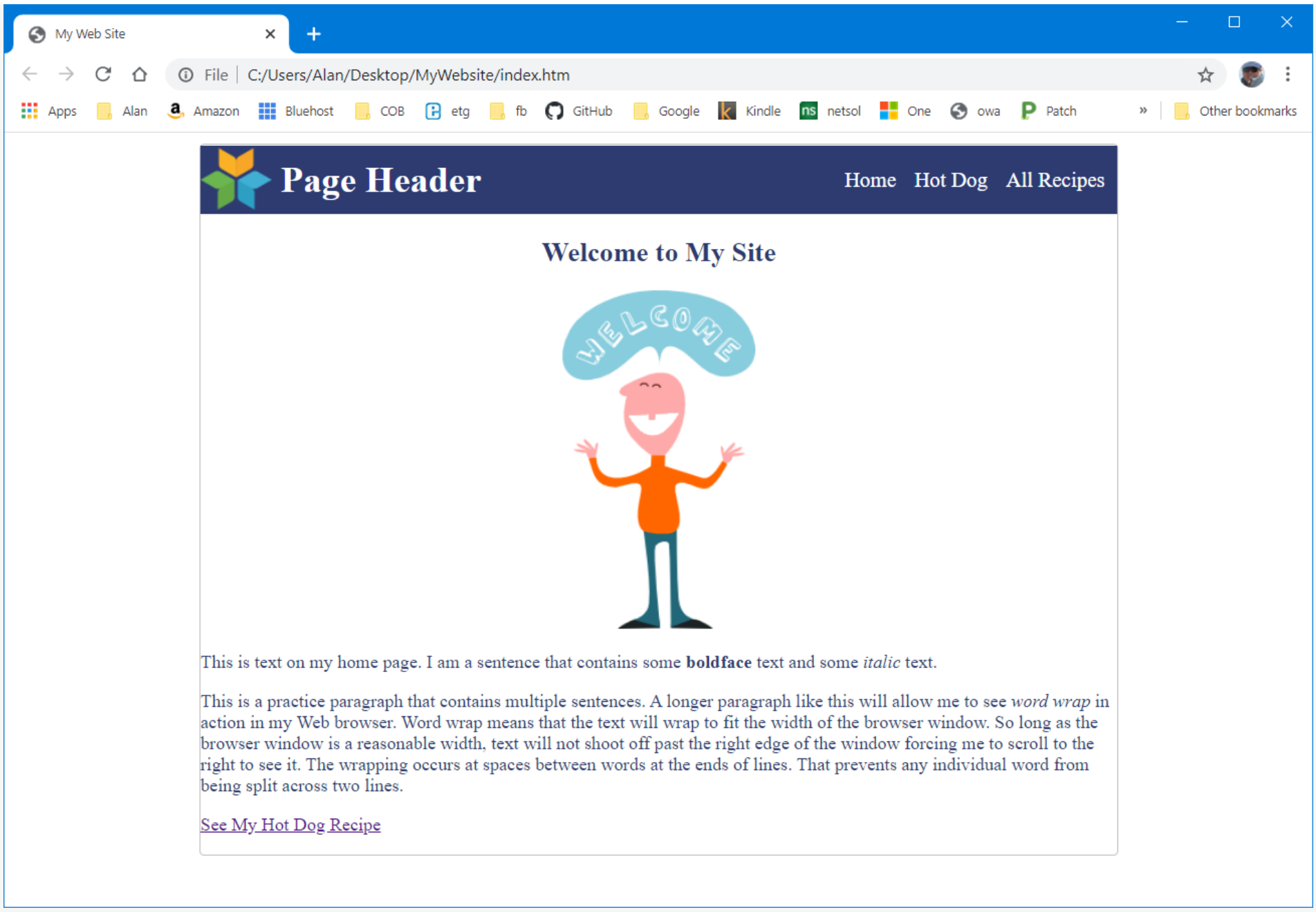
```
#wrapper {
  width: 800px;
  border: solid 1px silver;
  border-radius: 4px;
  margin: 10px auto;
  }
```

4. Save your changes.

5. Open or reload/refresh the page in the browser.

Sneak Peek

Expand your browser window so that it's wider than the page layout. This should make the content in your wrapper appear horizontally centered from left to right and you should see a small strip of white at the top of the page.



The layout is centered because the browser automatically calculated the left and right margins to be equal in size, which causes the layout to be centered horizontally. The strip of white above the header is the 10 pixels margin you set for the top and bottom margin.

To better help you visualize and resize the space inside and outside your wrapper, we're going to add some background colors.

Visualizing Your Layout

Coloring the Page

All that white space you see surrounding the layout in our page is the actually the page *body*. If you want that to be some other color, you can use the CSS background-color property to give it a different color. The syntax is simple:

```
background-color: color;
```

Then, all you need to do is replace *color* with any valid color name or hex code. Let's give it a try!

Here are the Steps

1. Open *stylesheet.css* in your editor.

2. Go to the **body{}** style rule.

3. Add a light blue background-color as below:

```
body {
  background-color: lightblue;
  color: #2d3b71;
  margin: 0;
  }
```

4. Save the style sheet and reload/refresh index.htm in the browser.

Sneak Peek

When you view the page in your browser, you'll notice the background is now light blue. Unfortunately, so is the main content area of the page, but we'll fix that in a moment.

Coloring the Wrapper

The wrapper div is still transparent, so you can see right through the main section to light blue body background. The header has its own dark blue background color, so you can't see the body background through that. So, let's make the wrapper background white, so it's no longer transparent. That way, anything that currently inside the wrapper section will have a white background, rather than transparent.

Here are the Steps

1. Open *stylesheet.css* in an editor.

2. Go to the wrapper{} style rule.

3. Set the background-color to white as in the code below:

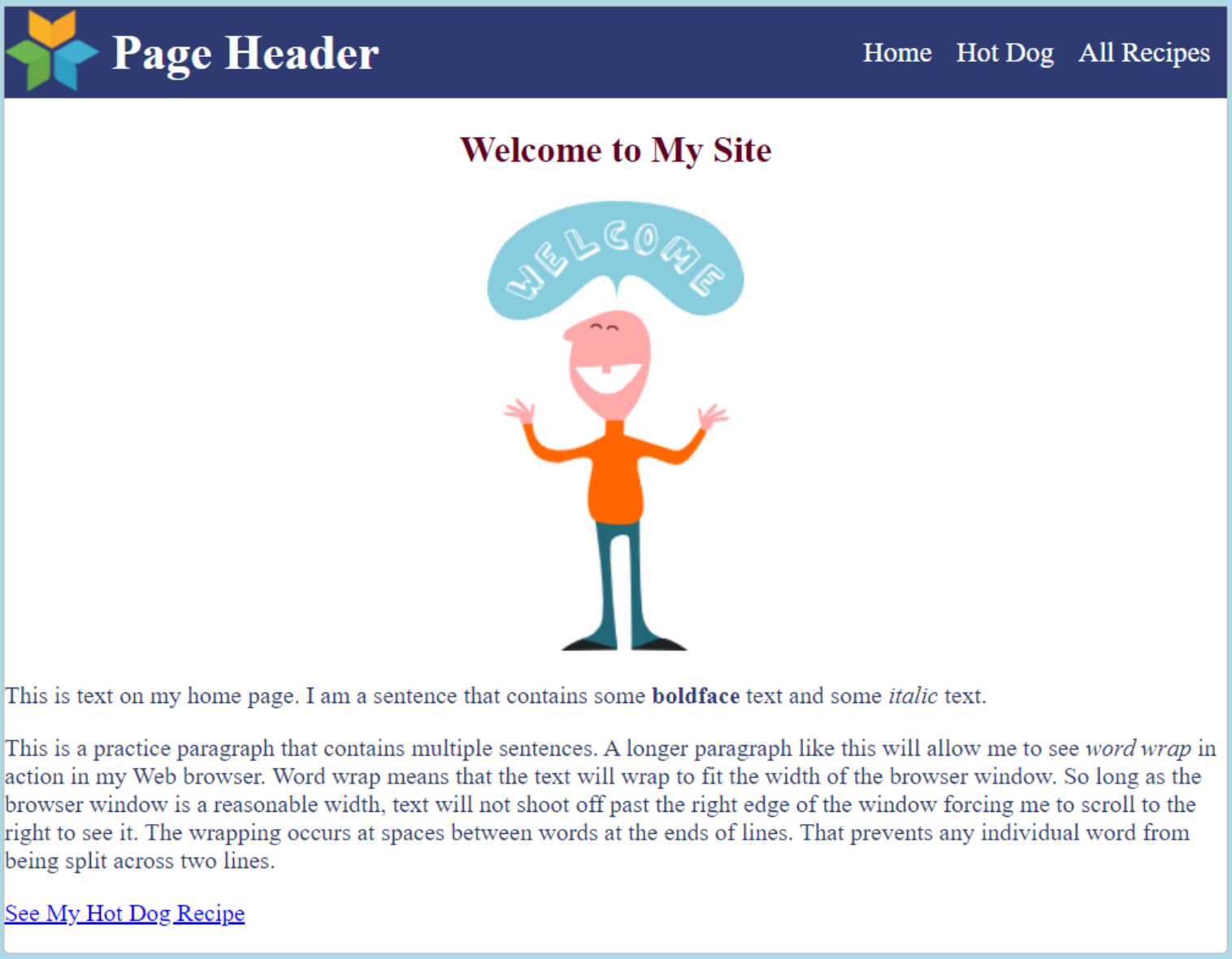
```
/* Wrapper div controls width of entire layout */
  #wrapper {
  width: 800px;
  background-color:white;
  border: solid 1px silver;
  border-radius: 4px;
  margin: 10px auto;
  }
```

4. Save your changes.

5. Then open or reload/refresh the *index.htm* page in a browser.

Sneak Peek

Your page wrapper, which contains your main content, should now have a white background.



Styling the Main Content

Below the header and navbar in our page is the main section of the page, we can identify that content by adding <main>...</main> tags in *index.htm*. Right now, the text in the main content div (the wrapper div) is flush against the left border. It's more visually pleasing to have some space between the text and the border. For that, we need to add some ***padding*** to that section.

Defining Your Main Content

We want to use the <main>...</main> tags to define the main content section, this way we can apply some styling to this section. Let's do that now.

Here are the Steps

1. Open index.htm in your editor.

2. Place your cursor after the </header> tag and press **ENTER**.

3. Add the following comment:

```
<!-- Start of main page content -->
```

4. Press **ENTER** and add the <main> tag.

5. Place your cursor just after the </p> that closes the *See My Hot Dog Recipe* link.

6. Press **ENTER** and add the </main> tag.

7. Save your changes.

This really won't change the way your page looks, but we haven't added any styling to the main div yet. When you're done, your index.html code should look like so:

```
<!DOCTYPE html>
  <html>
  <head>
  <!-- Title in browser window -->
  <title>My  Website</title>
  <!-- Link to external CSS style sheet -->
  <link rel="stylesheet" href="stylesheet.css">
  </head>
  <body>
  <!--  Wrapper (container) div for entire page layout -->
  <div id="wrapper">
  <!-- Main page header -->
  <header>
  <img src="logo.png" style="float:left; height:58px;" alt="">
  <h1 style="float:left;"> Page Header</h1>
  <!-- Navigation bar -->
  <nav>
     <a href="index.htm">Home</a>
     <a href="recipe.htm">Hot Dog</a>
     <a href="https://allrecipes.com/">All Recipes</a>
  </nav>
  </header>
  <!-- Start of main page content -->
  <main>
  <!-- Center page title and image -->
  <div style="text-align:center;">
  <!-- Main page title -->
  <h2>Welcome to My Site</h2>
  <p><img src="welcome.png" alt="Welcome"/></p>
  </div>
  <p>This is text on my home page. I am a sentence that contains some  <strong>boldface</strong> text and
some <em>italic</em>  text.</p>
  <p>This is a practice paragraph that contains multiple sentences. A longer paragraph like this will allow
me to see <em>word wrap</em> in action in my  Web browser. Word wrap means that the text will wrap to fit
the width of the  browser window. So long as the browser window is a reasonable width, text will  not shoot
off past the right edge of the window forcing me to scroll to the right to see it. The wrapping occurs at
spaces between words at the ends of lines. That prevents any individual word from being split across two
lines.</p>
  <p>
  <a href="recipe.html">See My Hot Dog Recipe</a>
  </p>
  </main>
  </div>
  <!-- End  wrapper -->
  </body>
  </html>
```

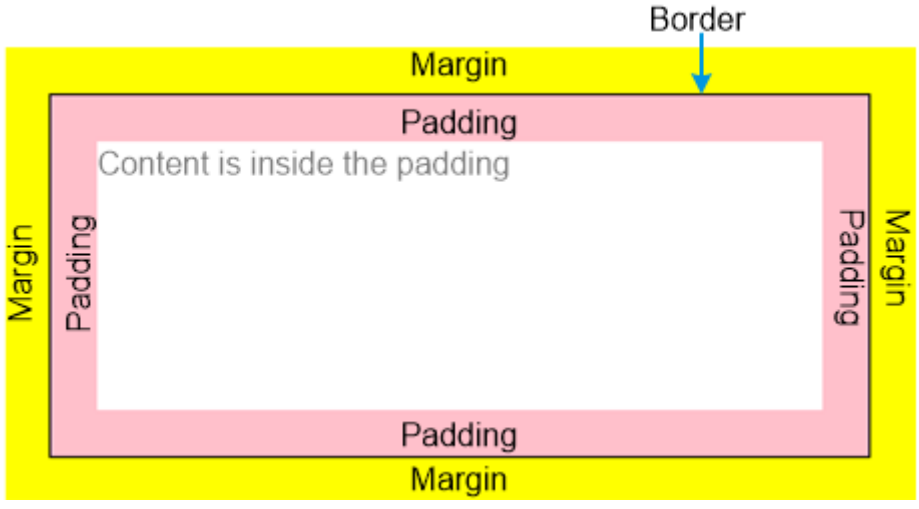
Adding and Adjusting Padding

Padding is not the same as a margin:

- A margin is empty space between the edge of the block and the browser window. It's the space *outside* the border.

- Padding is empty space between the edge of the block and the elements inside it. It's the space *inside* the border.

When we were working with tables, the padding was the space between the edge of the cell and the content. We made the analogy that when you put something fragile in a box for shipping you add padding *inside* the box. So, in this case CSS padding is space around the content *inside* the borders of the box.



You can apply padding similar to the way you apply a margin, except you use the word *padding* rather than *margin* as the property name. As with margins you can use the shorthand syntax to set all padding at once:

```
padding:8px;
```

Or you can add it in pieces:

```
padding-top: 8px;
padding-left: 8px;
padding-right: 8px;
padding-bottom: 8px;
```

Typically, you just want to set the padding for all four sides at once. For example, padding:8px puts eight pixels of padding all around the inner border. Let's give it a try.

Note

We haven't created a style rule for the main div yet. So, in our style sheet we need a style rule with *main* as the selector. Having main{} as the selector tells us that this styling goes to content between the <main>...</main> tags in the page.

Here are the Steps

1. Open *stylesheet.css* in an editor.

2. Place your cursor just before the h1{} style rule and press **ENTER** to make some space.

3. Type or copy/paste in, the following comment into the space you just created.

```
/* Main content below the header and navbar */
```

4. Press ENTER and type the following main{} style rule.

```
main{
  padding: 8px;
  }
```

5. Save the style sheet.

6. Open *index.htm* in your browser.

Sneak Peek

Now, the text in the main content division is no longer right up against the border on the left side. That's because there's an additional 8 pixels of padding between the border and the content. There's actually 8 pixels of padding all the way around the content, but it's most noticeable at that left side of the main content division.

This is text on my home page. I am a sentence that contains some **boldface** text and some *italic* text.

This is a practice paragraph that contains multiple sentences. A longer paragraph like this will allow me to see *word wrap* in action in my Web browser. Word wrap means that the text will wrap to fit the width of the browser window. So long as the browser window is a reasonable width, text will not shoot off past the right edge of the window forcing me to scroll to the right to see it. The wrapping occurs at spaces between words at the ends of lines. That prevents any individual word from being split across two lines.

See My Hot Dog Recipe

We all know people use different size screens to access websites, ranging from small phone screens to large monitors. So, we really want a page that looks good on a variety of screen widths this is called ***responsive web design***.

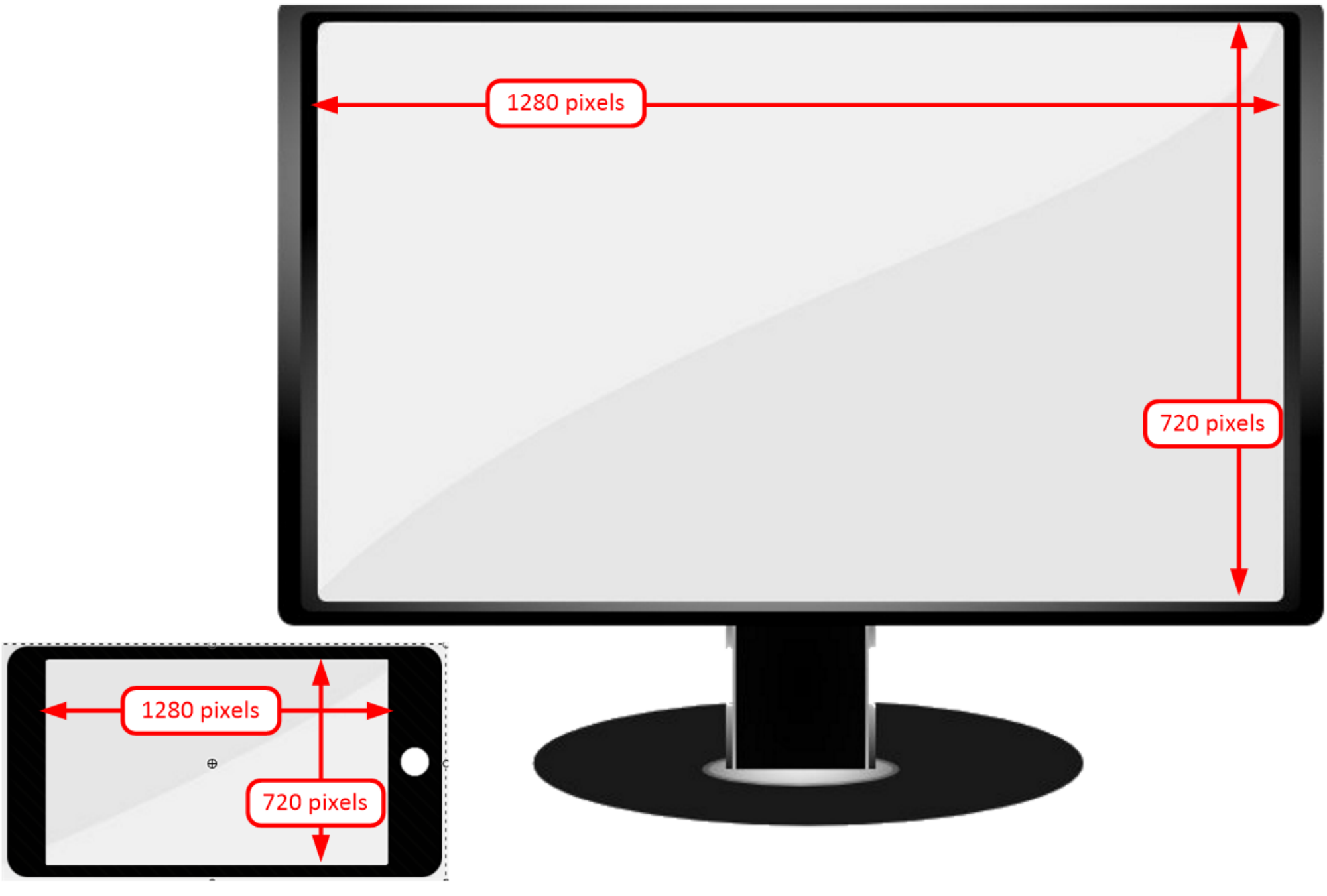
Creating a Responsive Page

Understanding Screen Dimensions

We tend to measure how wide or tall a screen is in terms of *inches* (or *centimeters*). But from the computer's perspective, a screen is measured is in *pixels.* A pixel is one tiny dot on the screen. You don't see individual pixels, they're too small. However, if you looked at your computer screen with a power magnifying glass, you might be able to see that the image on the screen is actually created by thousands of very tiny square dots.

To give you a sense of how small one pixel is, a single pixel is approximately 1/96 inch (0.26 mm). The image below shows three red squares. The one on the left is 3x3 pixels, the next one is 2x2 pixels, and the third one to the right is exactly one pixel.

While we tend to think of screen size in terms of it's physical height and width in inches (or centimeters), CSS measures screen size in the number of pixels it takes to fill the screen. This measurement is expressed as *width* x *height*.



The physical size of the screen is irrelevant.

An iPhone or Android phone can have just as many pixels as a full-size computer monitor. So, rather than width x height, it may be easier to think of this as *longest dimension x shortest dimension*. Here are some common screen sizes:

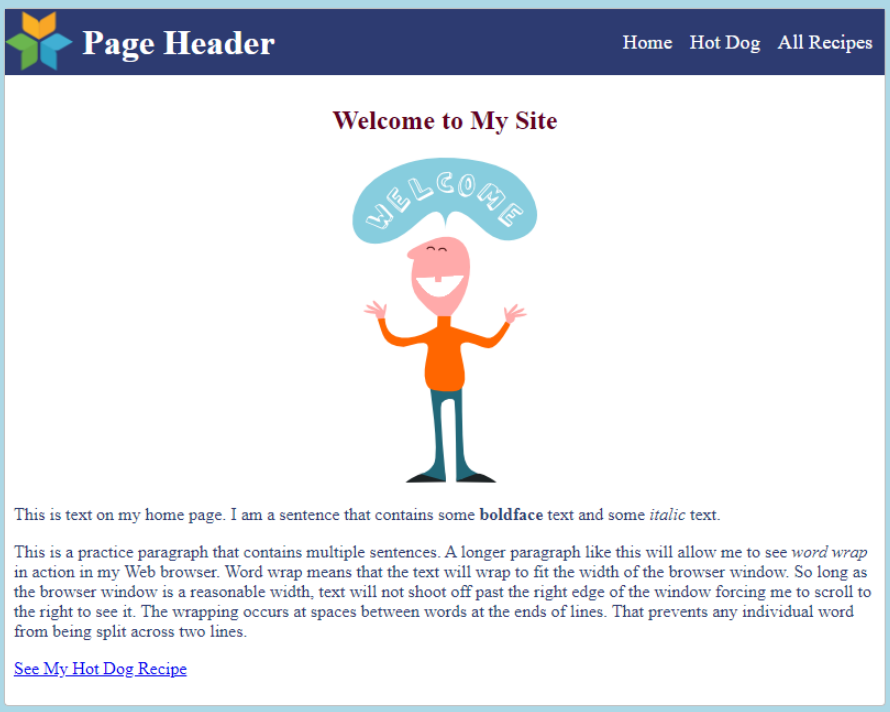
Size	Meaning	Description

Size	Meaning	Description
1024 x 768	1,024 pixels wide, 768 pixels tall	Older computer monitor
1280 x 720	1,280 pixels wide, 730 pixels tall	HD (High Definition)
1600 x 900	1,600 pixels wide, 900 pixels tall	HD+ (High Definition Plus)
1920 x 1080	1,920 pixels wide, 1080 pixels tall	Full HD (Full High Definition)

Common Screen Sizes

Adding Responsive Dimensions

Currently we have our wrapper width set to 800 pixels. This narrow width might be good for a small cell phone screen, so the user can see the whole page without scrolling left and right, but on a wide screen it leaves a lot of empty space to the left and right of the wrapper.



To make the site a little more responsive, allowing the width of the layout to better adapt to the screen size, we can set the width of the wrapper as a percent (%) rather than in pixels.

Example

If you want your layout width to take up half the width of the screen, you would set it to 50%:

```
width: 50%
```

So, if you wanted to fill the screen width, it would be 100%:

```
width: 100%
```

To try it out, let's change the wrapper width to 90%.

Here are the Steps

1. Open *stylesheet.css* in your editor.
2. Go to the #wrapper {}style rule
3. Change width:800px; to width:90%;
4. Save the changes in the stylesheet.
5. Open or refresh *index.htm* in a browser

Sneak Peek

Your #wrapper style rule should look like this when you're finished:

```
/* Wrapper div controls width of entire layout */
  #wrapper {
  width: 90%;
  background-color: white;
  border: solid 1px silver;
  border-radius: 4px;
  margin: 10px auto;
  }
```

Since the width of your wrapper is now 90%, the layout width will automatically adjust to 90% of the browser window.

This new layout design makes much better use of the available screen width, however we are back to the original issue we had earlier: if the window gets too narrow, the nav links move down to the main content area and become invisible. To fix this, we can put a limit on how narrow it's allowed to go using the CSS property ***min-width***.

Limiting the Width Dimension

In CSS, there are two properties that can be useful in providing additional parameters for the width property:

- min-width: This allows you to set a minimum width.
- max-width: This allows you to set a maximum width.

The maximum and minimum widths should be expressed in pixels. Figuring out the appropriate min-width and max-width, depends on the element you're applying it to and the overall layout of the page. We did the guesswork around determining the number of pixels that work in our current layout, but feel to experiment with different widths. For right now, we'll apply a range of 580 pixels to 1400 pixels wide.

#IRL: The dimensions depend on the design.

The dimensions we are using is what works well for the *index.htm* sample page. It's not an industry-wide best practice. In real life people design their websites and choose their widths based on what looks good with the design of their specific website.

Anyway, let's go ahead and code the layout width so that it doesn't go below 700 pixels or above 1400 pixels.

Here are the Steps

1. Open *stylesheet.css* in your editor.

2. Go to the #wrapper{} style rule.

3. Place your cursor just after the width: 90%; and press **ENTER**.

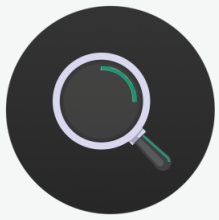
4. On the new line set the minimum width to 580 pixels by adding the following code:

```
min-width: 580px;
```

5. Press **ENTER** and on the new line set the maximum width to 1400 pixels by adding the following code:

```
max-width: 1400px;
```

6. Save your changes.



Take a Closer Look

When you've finished, your #wrapper{} style rule should look like the code below:

```
/* Wrapper div controls width of entire layout */
  #wrapper {
  width: 90%;
  min-width: 580px;
  max-width: 1400px;
  background-color: white;
  border: solid 1px silver;
  border-radius: 4px;
  margin: 10px auto;
  }
```

The exact order of lines within the style rule isn't important, but all of the property:value pairings should be there.

7. Open or reload/refresh *index.htm* in the browser.

8. Widen and narrow the browser window.

You'll notice that the layout width will still adjust, based on the size that you set the browser window to. However, the layout stops getting smaller once it hits that 580 pixel minimum and stops getting bigger once it hits the 1400 pixel maximum. This way the nav links never drop out of the header and disappear, and the main content text stays fairly compact on the page.

- If the browser window is narrower than 580 pixels, a horizontal scroll bar will appear at the bottom of the browser window so you can scroll left and right.
- If the browser window is wider than 1400 pixels the layout stops getting wider.

We've come up with a nice design here, and have a fairly robust external style sheet going. Let's head over to Chapter 5 now to review what we discussed in this lesson.

Lesson 9 Review

We covered a lot in this lesson! While there's no rule that says you must create a page layout for your site, doing so can add some professional polish to the effort. Let's take a look at what you did in this lesson:

- You've added a header and navbar to *index.htm*, the home page for our sample site.
- You placed the entire layout inside a wrapper div, which allows you to control over how wide, or how narrow we want the layout to be.

- You discovered that setting your side margins to *auto* centers your layout in the browser window.

- You learned that stating the width as a percentage (%), rather than in pixels, allows the width of the layout be responsive, adjusting to the width of the screen being used to view the site.

Congratulations! You've made some great progress. Pretty soon you'll be publishing your site for all the world to see. Up next, you'll learn a bit about ***SEO (Search Engine Optimization)***, which uses HTML tags to make it easier for people to find your website when the search with Google, Bing, or any other search engine.

Lesson 9 Assignment

Our index.htm page has a nice design with page header and nav bar, however right now only index.htm has the new layout features. We haven't done anything with recipe.htm yet. Don't worry, we're going to do some work on recipe.htm in the assignment.

Learning Check

Styling Layout with CSS

Text equivalent start.

Question	Choices	Feedback
When adjusting the width of a block, which of the following property:value pairings will ensure that the width is responsive?	a. min-width: 600px; b. max-width: 90%; c. width:80%; d. width: 1800px;	a. Incorrect. b. Incorrect. c. Correct! d. Incorrect.
If you wanted to apply custom formatting to your wrapper div, which selector would you use?	a. #wrapper b. #div c. div d. wrapper	a. Correct! b. Incorrect. c. Incorrect. d. Incorrect.
Which property-value pairing would adjust the space between the border and the content?	a. border-radius: 6px; b. border: solid 5px; c. padding: 10px; d. margin: 20px auto;	a. Incorrect. b. Incorrect. c. Correct! d. Incorrect.

Instructions: Read the questions in the first column and consider the answer choices in the second column. Check the third column to reveal the correct answers.

Text equivalent stop.