Introduction to Bootstrap Training

with examples and
hands-on exercises

WEBUCATOR

Version: 2.1.0

The Authors

Chris Minnick

Chris Minnick, the co-founder of WatzThis?, has overseen the development of hundreds of web and mobile projects for customers from small businesses to some of the world's largest companies. A prolific writer, Chris has authored and co-authored books and articles on a wide range of Internet-related topics including HTML, CSS, mobile apps, e-commerce, e-business, Web design, XML, and application servers. His published books include Adventures in Coding, JavaScript For Kids For Dummies, Writing Computer Code, Coding with JavaScript For Dummies, Beginning HTML5 and CSS3 For Dummies, Webkit For Dummies, CIW E-Commerce Designer Certification Bible, and XHTML.

Nat Dunn (Editor)

Nat Dunn is the founder of Webucator (www.webucator.com), a company that has provided training for tens of thousands of students from thousands of organizations. Nat started the company in 2003 to combine his passion for technical training with his business expertise, and to help companies benefit from both. His previous experience was in sales, business and technical training, and management. Nat has an MBA from Harvard Business School and a BA in International Relations from Pomona College.

Follow Nat on Twitter at @natdunn and Webucator at @webucator.

Class Files

Download the class files used in this manual at https://static.webucator.com/media/public/materials/classfiles/BTS101-2.1.0.zip.

Errata

Corrections to errors in the manual can be found at https://www.webucator.com/books/errata/.

Table of Contents

LESSON 1

Getting Started with Bootstrap Basics

Topics Covered

☑ The purpose of Bootstrap.

☑ Getting Bootstrap.

☑ A basic Bootstrap template.

☑ Bootstrap classes.

☑ The benefits of responsive and mobile-first design.

Introduction

Bootstrap is one of the most widely-used and powerful tools in any web developer's toolbox. It's currently used by millions of websites for everything from styling buttons to creating entire websites that will work and look great on mobile, tablet, and desktop browsers. In this lesson, you'll learn about the terminology and how to get started with Bootstrap.

1.1. Introduction to Bootstrap

❖ 1.1.1. What Is Bootstrap?

Bootstrap is a framework for creating responsive, mobile-first websites. That sentence contains a lot of lingo, so let's break it down into pieces so we can understand what Bootstrap is and when you may want to use it.

❖ 1.1.2. What Is a Framework?

A framework is a collection of pre-built pieces of code that can be assembled or used together with other code to create custom software or websites. The word "framework" is often used interchangeably with the word "library," although it's generally understood that frameworks are larger and more all-encompassing than libraries, which may be very narrowly focused.

❖ 1.1.3. What Is Responsive Design?

When we talk about websites being "responsive" we're talking about their ability to be used on different sized devices, such as desktop, tablet, and smartphone, and to adapt (or "respond") to each device.

❖ 1.1.4. What Is Mobile-first Design?

Mobile-first is a strategy of designing websites to work well on mobile devices and to scale upward for larger and more capable devices. This strategy represents an inversion of the traditional way of designing websites that held that sites should be designed for the ideal browser and then "gracefully degrade" for smaller or less capable devices.

❖ 1.1.5. Major Changes in Bootstrap 5

In this course, we'll be using Bootstrap 5.2, which is the latest version at the time of this writing. However, many people and companies still use the earlier version of Bootstrap, Bootstrap 4, and you'll find that most of the content here is relevant to either Bootstrap 4 or Bootstrap 5.

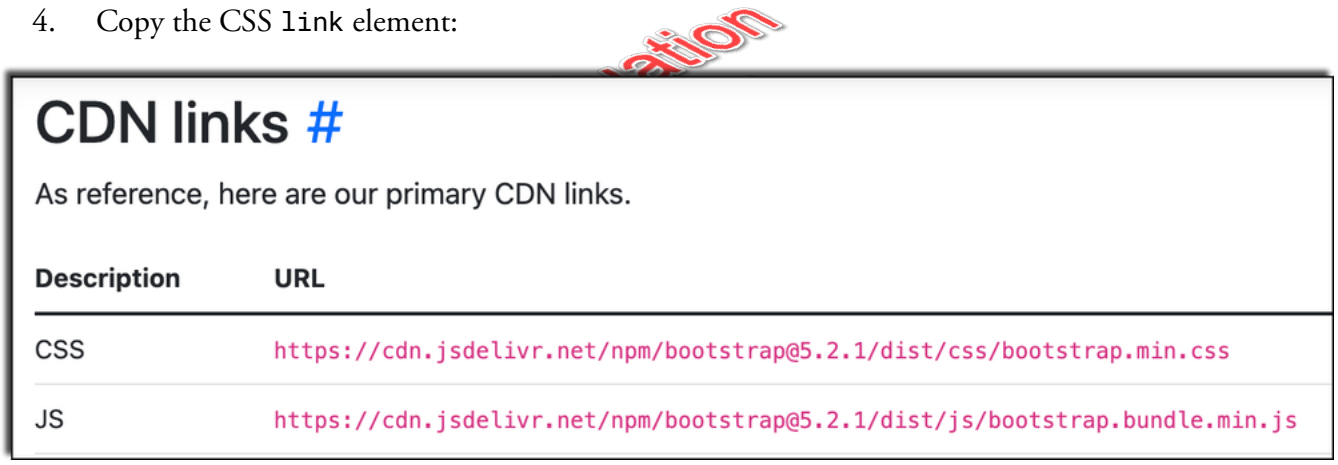
The Bootstrap website contains a migration guide to help you upgrade from previous versions. You can find that guide at `https://getbootstrap.com/docs/5.2/migration/`.

Exercise 1: Creating a Basic HTML Template with Bootstrap

5 to 10 minutes

For your first demonstrations of what Bootstrap can do, you don't even need to download Bootstrap. You can get started using Bootstrap with nothing more than a code editor and a browser. In this exercise, you're going to create a basic Bootstrap template that links to Bootstrap on a Content Delivery Network (CDN).

1. Open `getting-started/Exercises/bootstrap-template.html`.
2. Open this HTML file in a web browser to see what it looks like. This file doesn't yet use Bootstrap. To use Bootstrap, you'll need to include the Bootstrap CSS file and JavaScript file.
3. Go to `https://getbootstrap.com/docs/5.2/getting-started/introduction/#cdn-links` in your web browser.
4. Copy the CSS `link` element:



CDN links

As reference, here are our primary CDN links.

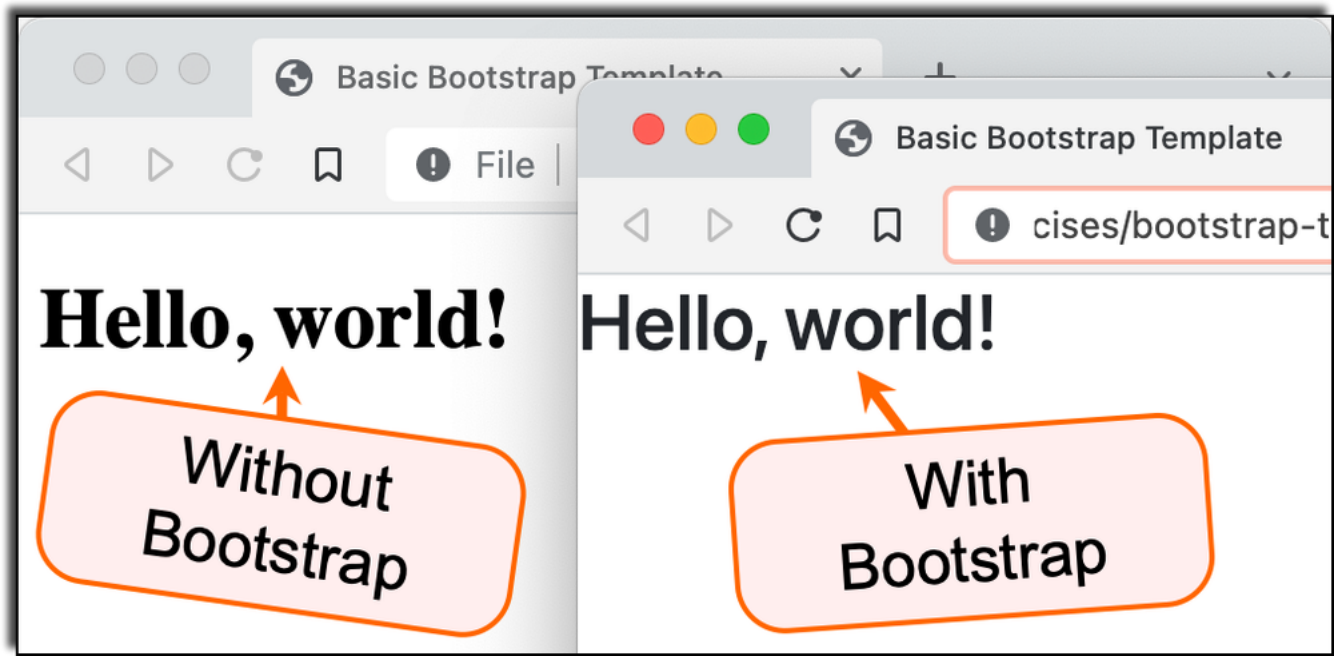
Description	URL
CSS	https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/css/bootstrap.min.css
JS	https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/js/bootstrap.bundle.min.js

   A. **Note**: The JavaScript file is only required if your page makes use of one (or more) of Bootstrap's JavaScript Components.

5. In your code editor, use the CSS link to create an HTML `link` element right above `</head>`.

```
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/css/boot ↵↵
strap.min.css" rel="stylesheet">
```

6. Reload the page in your browser to see what it looks like with Bootstrap applied. Notice that even with just this small bit of HTML, the addition of Bootstrap makes some adjustments:



Solution: getting-started/Solutions/bootstrap-template.html

```
1.   <!DOCTYPE html>
2.   <html>
3.   <head>
4.   <meta charset="UTF-8">
5.   <title>Basic Bootstrap Template</title>
6.   <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
7.   <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/css/bootstrap.min.css"
             rel="stylesheet">
8.   </head>
9.   <body>
10.    <h1>Hello, world!</h1>
11.  </body>
12.  </html>
```

Exercise 2: How to Download Bootstrap

5 to 10 minutes

In this exercise, you'll download Bootstrap and modify the basic Bootstrap template that you created in the last exercise

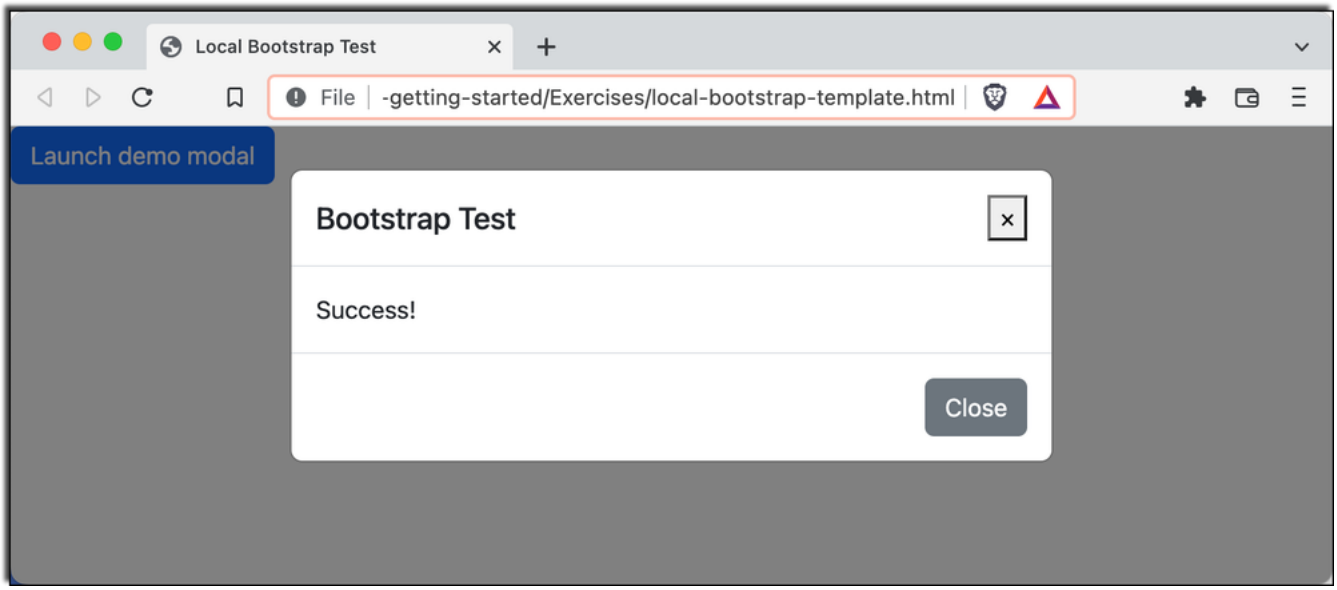
1. Go to `https://getbootstrap.com/docs/5.2/getting-started/download/` and click the **Download** button under the **Compiled CSS and JS** heading. A `.zip` file containing the Bootstrap source code and example files will download to your computer.
2. Find the downloaded file in your `Downloads` folder or whichever folder you downloaded to. The file name will be something similar to `bootstrap-5.2.1-dist.zip`.
3. Extract the downloaded zip file into your `getting-started/Exercises` directory and rename it `bootstrap`. Alternatively, you can extract the zip file in the folder where you downloaded it, rename it to `bootstrap`, and then move the new `bootstrap` folder to the `getting-started/Exercises` folder.
4. Open `getting-started/Exercises/local-bootstrap-template.html` in your editor.
5. Insert the following code just before the closing `</head>` tag to include the Bootstrap CSS from the latest downloaded version:

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

6. Insert the following code near the end of the document, just above the closing `</body>` tag, to include the Bootstrap JavaScript:

```
<script src="bootstrap/js/bootstrap.bundle.min.js"></script>
```

7. Preview the page in your browser. If clicking the button opens a modal window containing the message "Success!", congratulations! If a modal window doesn't open or if the button on your preview doesn't look like the one in the following screenshot, check your code and make sure that your paths to the Bootstrap CSS and JavaScript are correct:



Solution: getting-started/Solutions/local-bootstrap-template.html

```
1.   <!DOCTYPE html>
2.   <html>
3.   <head>
4.   <meta charset="UTF-8">
5.   <title>Local Bootstrap Test</title>
6.   <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-
             fit=no">
7.   <link href="bootstrap/css/bootstrap.min.css" rel="stylesheet">
8.   </head>
9.   <body>
     -------Lines 10 through 33 Omitted-------
34.    <script src="bootstrap/js/bootstrap.bundle.min.js"></script>
35.  </body>
36.  </html>
```


1.2. Applying Styles and Functions with Class

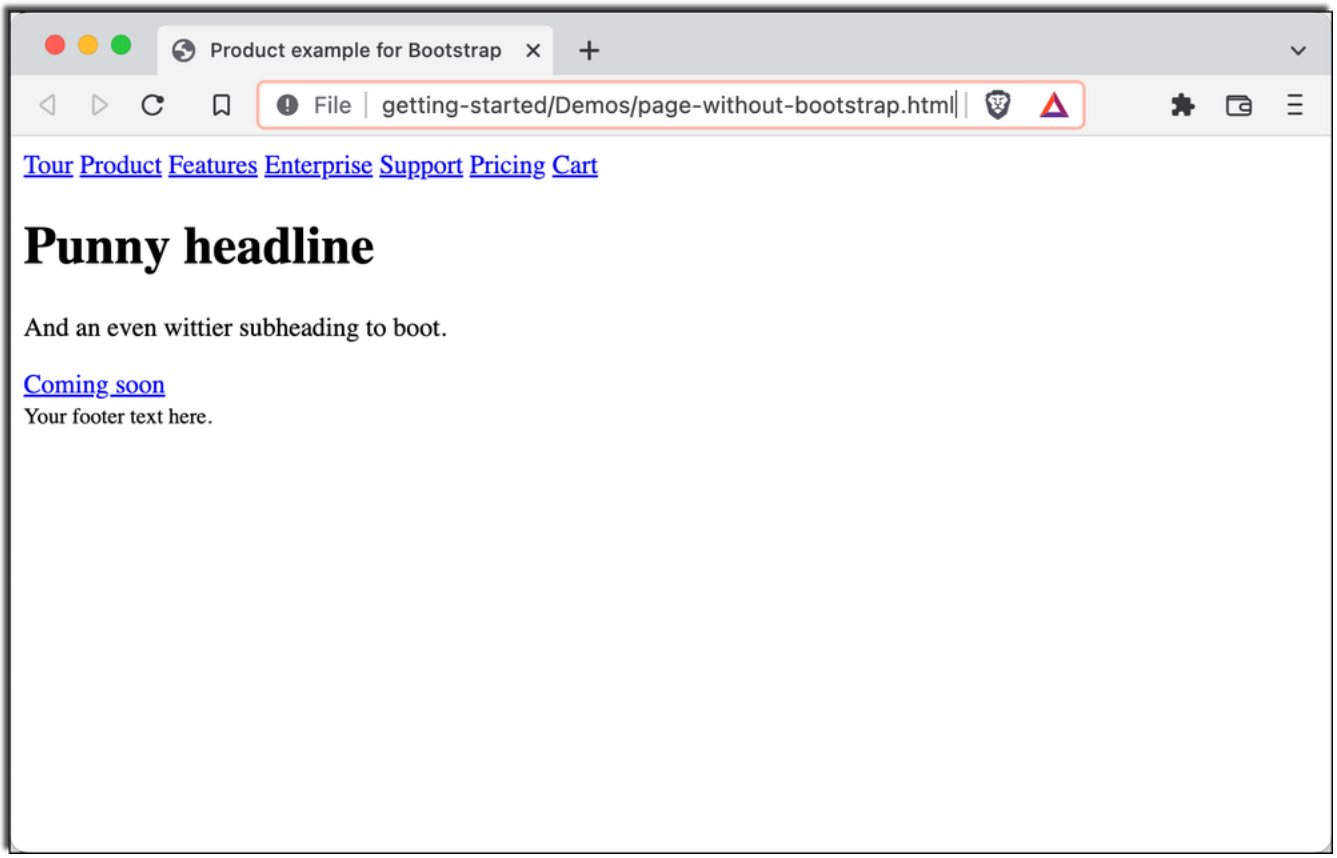
The easiest way to get started with Bootstrap is by applying its pre-defined styles and functionality to an existing web page. Most of Bootstrap's functionality can be used just by adding values to the `class` attributes of your HTML elements.

Without Bootstrap, the following HTML page will display as just text, links, and line breaks:

Demo 1.1: getting-started/Demos/page-without-bootstrap.html

```
1.  <!DOCTYPE html>
2.  <html lang="en">
3.  <head>
4.  <meta charset="UTF-8">
5.  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-
            fit=no">
6.  <title>Product example for Bootstrap</title>
7.  </head>
8.  <body>
9.  <nav>
10.   <div>
11.     <a href="#">Tour</a>
12.     <a href="#">Product</a>
13.     <a href="#">Features</a>
14.     <a href="#">Enterprise</a>
15.     <a href="#">Support</a>
16.     <a href="#">Pricing</a>
17.     <a href="#">Cart</a>
18.   </div>
19. </nav>
20. <div>
21.   <div>
22.     <h1>Punny headline</h1>
23.     <p>And an even wittier subheading to boot.</p>
24.     <a href="#">Coming soon</a>
25.   </div>
26.   <div></div>
27.   <div></div>
28. </div>
29. <footer>
30.   <div>
31.     <div>
32.       <small>Your footer text here.</small>
33.     </div>
34.   </div>
35. </footer>
36. </body>
37. </html>
```


The preceding code will render the following:



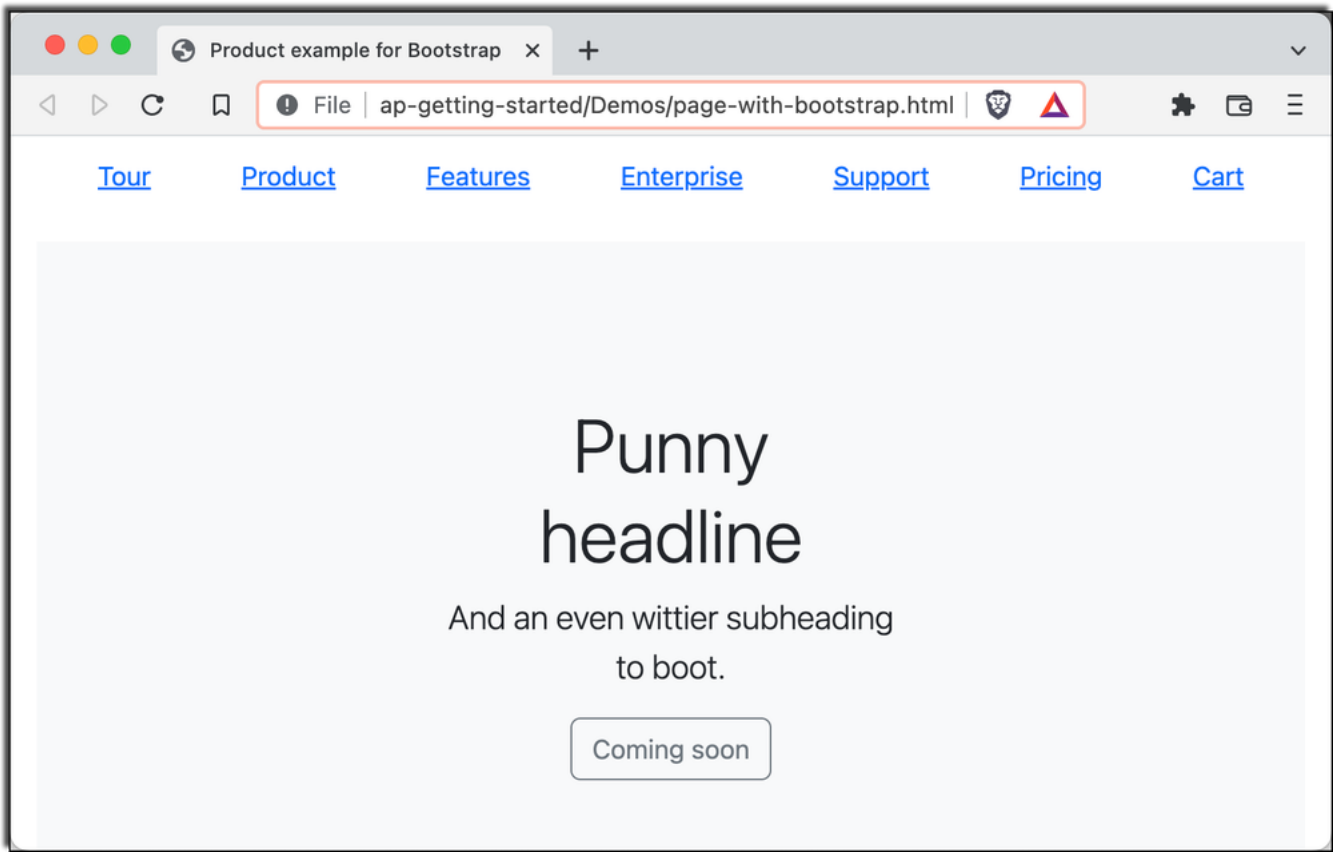
In the following revised example, we've added the Bootstrap CSS link tag, the Bootstrap JavaScript tag, and some Bootstrap classes.

Demo 1.2: getting-started/Demos/page-with-bootstrap.html

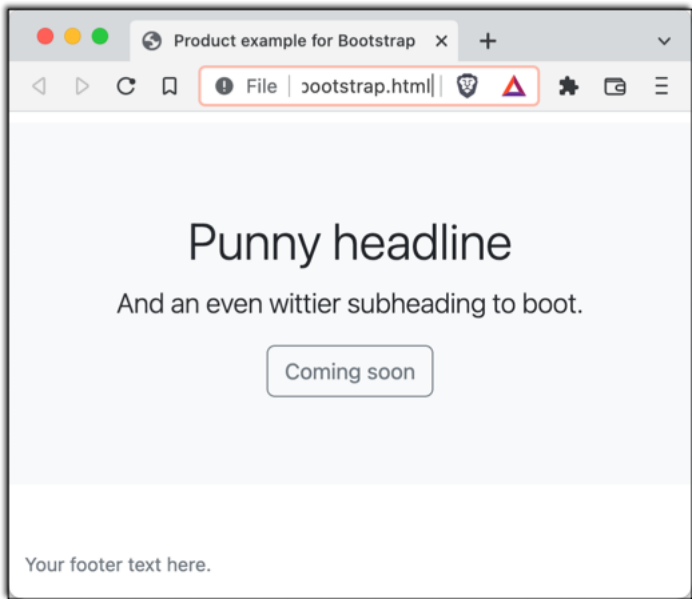
```
1.  <!DOCTYPE html>
2.  <html lang="en">
3.  <head>
4.  <meta charset="UTF-8">
5.  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-
        fit=no">
6.  <title>Product example for Bootstrap</title>
7.  <!-- Bootstrap core CSS -->
8.  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/css/boot ↵↵
        strap.min.css" rel="stylesheet" />
9.  </head>
10. <body>
11. <nav class="navbar site-header sticky-top py-1">
12.   <div class="container d-flex flex-column flex-md-row
13.         justify-content-between">
14.     <a class="navbar-link py-2 d-none d-md-inline-block nav-item"
          href="#">Tour</a>
15.     <a class="navbar-link py-2 d-none d-md-inline-block" href="#">Product</a>
16.     <a class="navbar-link py-2 d-none d-md-inline-block" href="#">Features</a>
17.     <a class="navbar-link py-2 d-none d-md-inline-block" href="#">Enterprise</a>
18.     <a class="navbar-link py-2 d-none d-md-inline-block" href="#">Support</a>
19.     <a class="navbar-link py-2 d-none d-md-inline-block" href="#">Pricing</a>
20.     <a class="navbar-link py-2 d-none d-md-inline-block" href="#">Cart</a>
21.   </div>
22. </nav>
23. <div class="position-relative overflow-hidden p-3 p-md-5 m-md-3
24.   text-center bg-light">
25.   <div class="col-md-5 p-lg-5 mx-auto my-5">
26.     <h1 class="display-4 font-weight-normal">Punny headline</h1>
27.     <p class="lead font-weight-normal">
28.       And an even wittier subheading to boot.
29.     </p>
30.     <a class="btn btn-outline-secondary" href="#">Coming soon</a>
31.   </div>
32.   <div class="product-device shadow-sm d-none d-md-block"></div>
33.   <div class="product-device product-device-2 shadow-sm
34.     d-none d-md-block"></div>
35. </div>
36. <footer class="container py-5">
37.   <div class="row">
38.     <div class="col-12 col-md">
39.       <small class="d-block mb-3 text-muted">
40.         Your footer text here.</small>
41.     </div>
42.   </div>
```

```
43.   </footer>
44.   <!-- Bootstrap core JavaScript
45.   ================================================== -->
46.   <!-- Placed at the end of the document so the pages load faster -->
47.   <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/js/bootstrap.bun ↵↵
          dle.min.js"></script>
48.   </body>
49.   </html>
```

The Bootstrap additions will change the page to render as follows:



And notice how it automatically adjusts when viewed in a smaller window:



In the following lessons, you'll learn to use Bootstrap's classes to add style and functionality to websites without writing JavaScript or CSS code.

Conclusion

In this lesson, you have learned what Bootstrap is, how to include the Bootstrap CSS and JavaScript into an HTML page, and how Bootstrap lets you transform a plain HTML page into a responsive and mobile-first web page just by adding values to `class` attributes.

LESSON 2
Bootstrap Layout

Topics Covered

☑ The viewport width vs. the device width.

☑ Breakpoints.

☑ Responsive layouts.

☑ Containers, rows, and columns.

Introduction

The most widely used function of Bootstrap is to easily and quickly create responsive web layouts. In this lesson, you'll learn the fundamentals of Bootstrap's 12-column grid and how to use these fundamentals to design web pages that work anywhere.

2.1. What Is the Viewport?

One of the things that makes the web so amazing is that there are so many different computing devices that can access it. In addition to desktop and laptop computers, we also have smart phones, tablets, TVs, watches, video game consoles, eReaders, and more.

Device width refers to the width (in pixels) of a device's screen. The device width usually refers to the resolution of the device. For example, if your screen has a resolution of 1440 x 900 pixels, the device width is 1440 pixels.

The device width is not always the same as the resolution, however. Many mobile devices use more than one device pixel to display each dot on the screen. For example, iPhone retina screens use two or three physical pixels to display each CSS pixel. So, while the actual screen may be 640 pixels wide, it actually has a device width of 320 pixels.

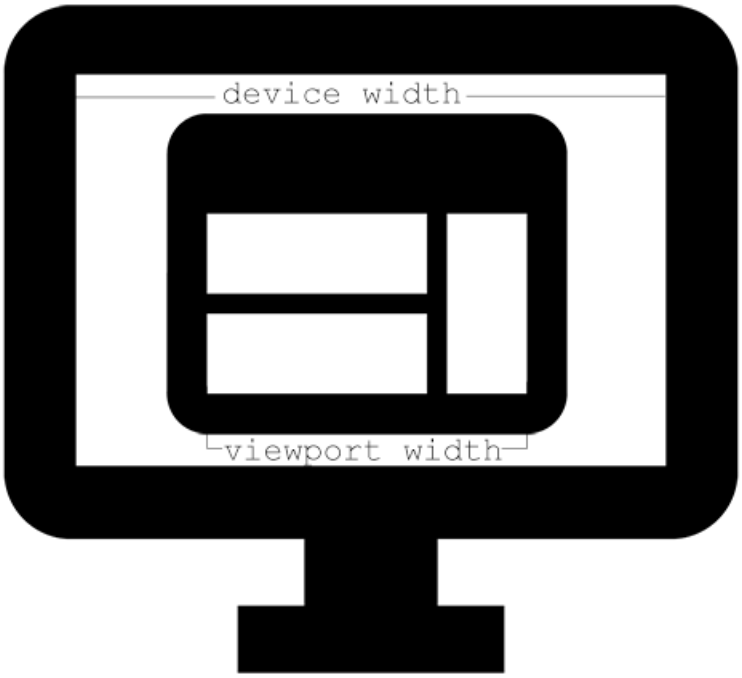
Knowing the width of a device is important for responsive design, but even more important is to know how wide the web browser is. We call the space in which a web page appears the viewport. When you increase the size of your web browser, the viewport width increases. You can think of the viewport as being like a window through which you view a web page. As you scroll down a web page, you change

what part of the page is visible through the viewport. In some cases, web pages can be wider than the viewport, which will require the user to scroll horizontally to make hidden parts of the web page visible.

Responsive web pages adjust the width of a web page to fit the viewport. As the viewport gets wider or narrower, the web page also gets wider or narrower.

On desktop and laptop computers, the viewport width can be smaller than the device width. On mobile devices, the device width and the viewport width are usually the same, since the browser takes up the full width of the screen.

Bootstrap uses the viewport width to adjust the layout of content in a web page. It does this using *breakpoints*.



2.2. Understanding Breakpoints

What looks great on one device might not work at all on a smaller one. The most important factor to consider when laying out web pages is the width of the viewport. In large viewports, a four-column layout often looks and works great. The same four-column grid would be difficult or impossible to use

on a mobile phone. One solution to this problem is to create a separate version of the site for mobile devices. But, mobile devices come in all sizes.

A web designer often has no control or no idea what any one user will access web pages with. The one thing that you can be sure of is that there's no way to customize a web page for every size device that someone might use to view the website.

The solution is to decide on several common ranges of widths and make layouts that work well within each of those ranges. The pixel width that marks the line between two different layouts is called a "breakpoint."

❖ 2.2.1. Bootstrap's Breakpoints

Bootstrap has six breakpoints, which allow web designers to accommodate device sizes ranging from smart phones to widescreen desktop monitors. These five breakpoints are:

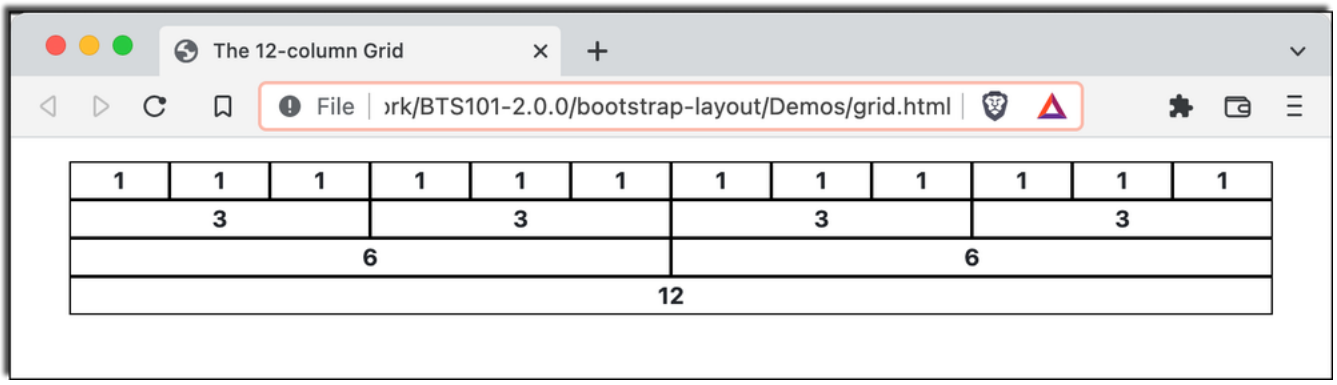
1. **Extra-small (xs)** – less than 576 pixels. An example of an *extra-small* device is a phone in portrait mode.
2. **Small (sm)** – 576 pixels and up. An example of a *small* device is a phone in landscape mode.
3. **Medium (md)** – 768 pixels and up. An example of a *medium* device is a tablet computer.
4. **Large (lg)** – 992 pixels and up. An example of a *large* device is a desktop computer.
5. **Extra-large (xl)** – 1200 pixels and up. An example of an *extra-large* device is a wide-screen monitor on a desktop computer.
6. **Extra-extra-large (xxl)** – 1400 pixels and up. An example of an *extra-extra-large* device is a wide-screen monitor or TV.

By specifying layouts for different breakpoints, you can create up to six different layouts for your website. Typically, however, web designers will only specify two or three layouts. Because each breakpoint applies by default to larger devices, if you only specify a layout for medium devices, that layout will be used for medium, large, extra-large, and extra-extra-large devices.

❖ 2.2.2. Introducing Bootstrap's Grid System

Bootstrap divides a web page into twelve equally sized columns.[1] By dividing your content among these columns in different configurations, you can create an infinite number of layouts.

1. The HTML file used for this screenshot is at `layout/Demos/grid.html`.



The three components of Bootstrap's grid system that make layouts possible are:

1. containers
2. rows
3. columns

2.3. Using Responsive Classes

By default, the content in a Bootstrap page spans all twelve Bootstrap columns and is optimized for extra-small devices. When you want to create a multi-column layout for larger devices, you can do so using Bootstrap's responsive classes.

❖ 2.3.1. Containers

The first step in creating a multi-column layout is to create a container. There are two kinds of containers: *fixed* and *fluid*. A fixed container will be a fixed size, based on the closest breakpoint below the current viewport width. For example, if your viewport width is 1000-pixels wide, falling between 992 and 1,200 pixels (the large and extra-large breakpoints), the fixed container will be the smaller of the two: 992 pixels. A fluid container will increase its size to fill the viewport. So, in the same 1000-pixel viewport, a fluid container will be 1000-pixels wide. Fixed containers are created using the `container` class. Fluid containers are created using the `container-fluid` class.

Here's an example of a fixed container:

```
<div class="container"></div>
```

And here's an example of a fluid container:

```
<div class="container-fluid"></div>
```

❖ 2.3.2. Rows and Columns

Inside of a container, you can create horizontal divisions by using the `row` class.

```
<div class="container">
  <div class="row"></div>
</div>
```

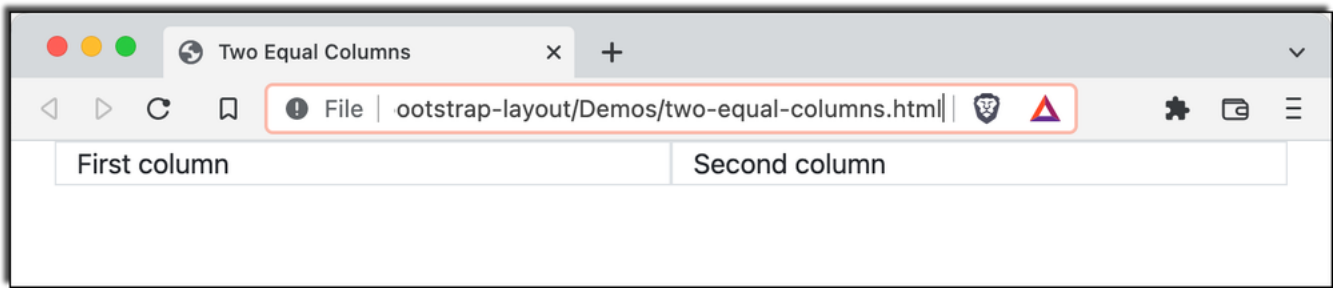
A Bootstrap layout can have as many rows as necessary.

Inside your rows, you can create vertical divisions by using the responsive column classes. Responsive column classes start with `col-`, followed by the abbreviation of the breakpoint at which the class should apply (for example, `col-xs`, `col-sm`, `col-md`, `col-lg`, `col-xl`, and `col-xxl`) and then optionally followed by the number of columns that the content should span. For example, to create a layout that will display in two columns at viewport widths larger than the `md` breakpoint, you can use the following code:

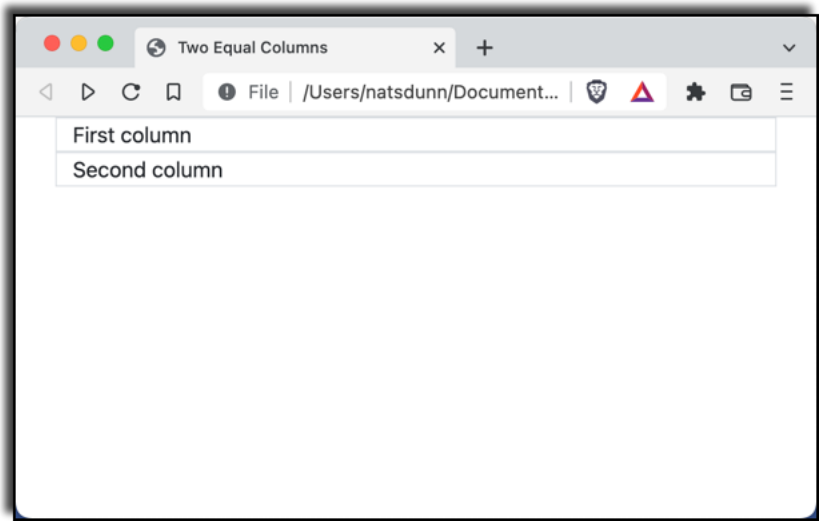
Demo 2.1: layout/Demos/two-equal-columns.html

```
     -------Lines 1 through 10 Omitted-------
11.    <div class="container">
12.      <div class="row">
13.        <div class="col-md border">
14.          First column
15.        </div>
16.        <div class="col-md border">
17.          Second column
18.        </div>
19.      </div>
20.    </div>
     -------Lines 21 through 26 Omitted-------
```

The preceding code creates two equal-width columns in browsers with a viewport width of over 768 pixels (borders added for clarity):



At widths under 768 pixels, the content of the two columns will stack on top of each other (borders added for clarity):

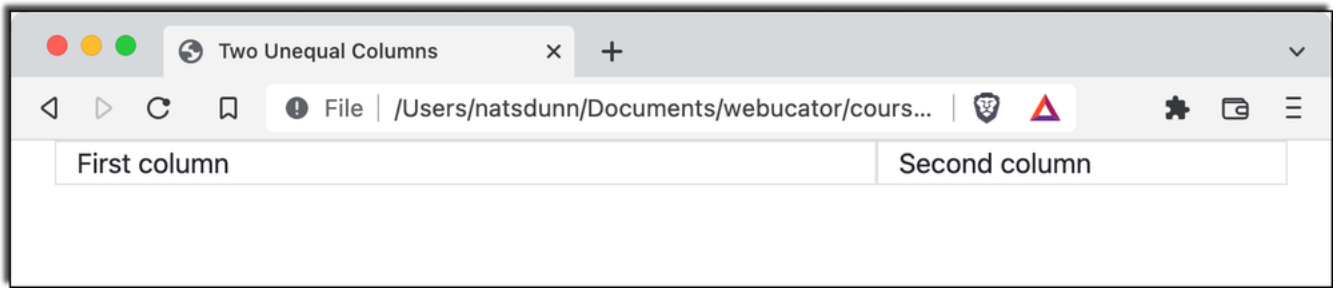


If you want your columns to be different widths, you can specify the number of the twelve Bootstrap columns that a layout columns should span. For example, to create a layout with the first column taking up two thirds of the horizontal space and the second column taking up one third, you can use the following code:

Demo 2.2: layout/Demos/two-unequal-columns.html

```
      -------Lines 1 through 10 Omitted-------
11.     <div class="container">
12.       <div class="row">
13.         <div class="col-md-8 border">
14.           First column
15.         </div>
16.         <div class="col-md-4 border">
17.           Second column
18.         </div>
19.       </div>
20.     </div>
      -------Lines 21 through 26 Omitted-------
```


This will display as follows on a viewport wider than 768 pixels (borders added for clarity):

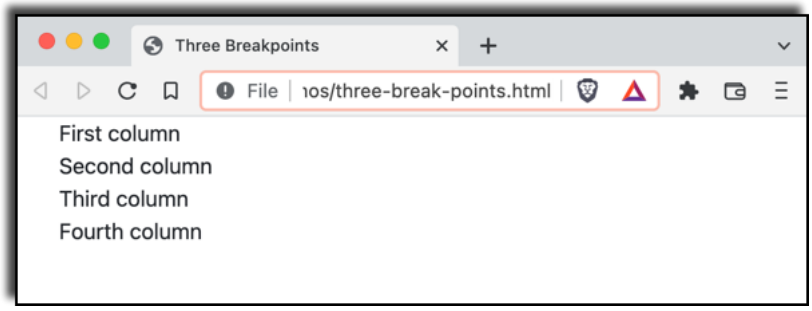


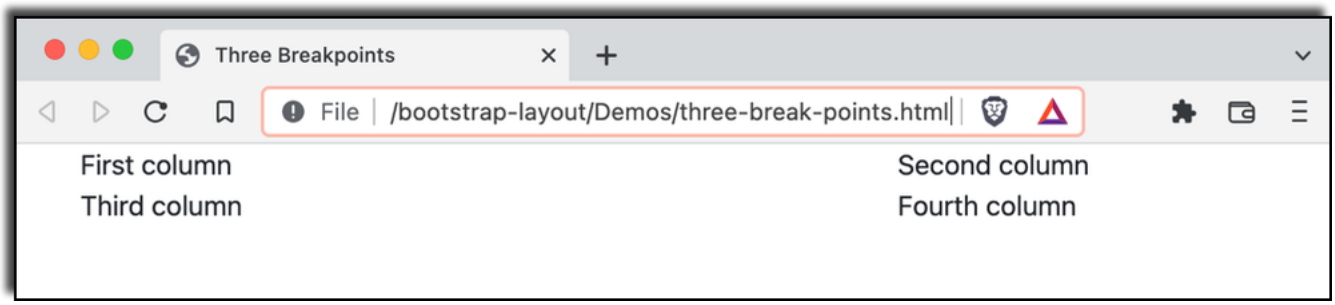
If you want to specify layouts for more than two breakpoints, you can add more responsive classes to your code by just separating them with spaces, as in the following example:

Demo 2.3: layout/Demos/three-break-points.html

```
      -------Lines 1 through 10 Omitted-------
11.     <div class="row">
12.       <div class="col-md-8 col-lg-6">
13.         First column
14.       </div>
15.       <div class="col-md-4 col-lg-2">
16.         Second column
17.       </div>
18.       <div class="col-md-8 col-lg-2">
19.         Third column
20.       </div>
21.       <div class="col-md-4 col-lg-2">
22.         Fourth column
23.       </div>
24.     </div>
25.   </div>
      -------Lines 26 through 31 Omitted-------
```

When the number of columns spanned in a row total more than twelve, as in the case of the `md` layout in the preceding code, Bootstrap will automatically create a new row. In the above example, the columns will display in one column for `xs` and `sm` viewports, two rows for the `md` breakpoint, and in a single row for the `lg` breakpoint, as shown below:






Three Breakpoints
File
/work/BTS101-2.0.0/bootstrap-layout/Demos/three-break-points.html
First column
Second column
Third column
Fourth column

Exercise 3: Making Grids

15 to 25 minutes

In this exercise, you will create a responsive Bootstrap grid layout. The `body` of the starting code looks like this:

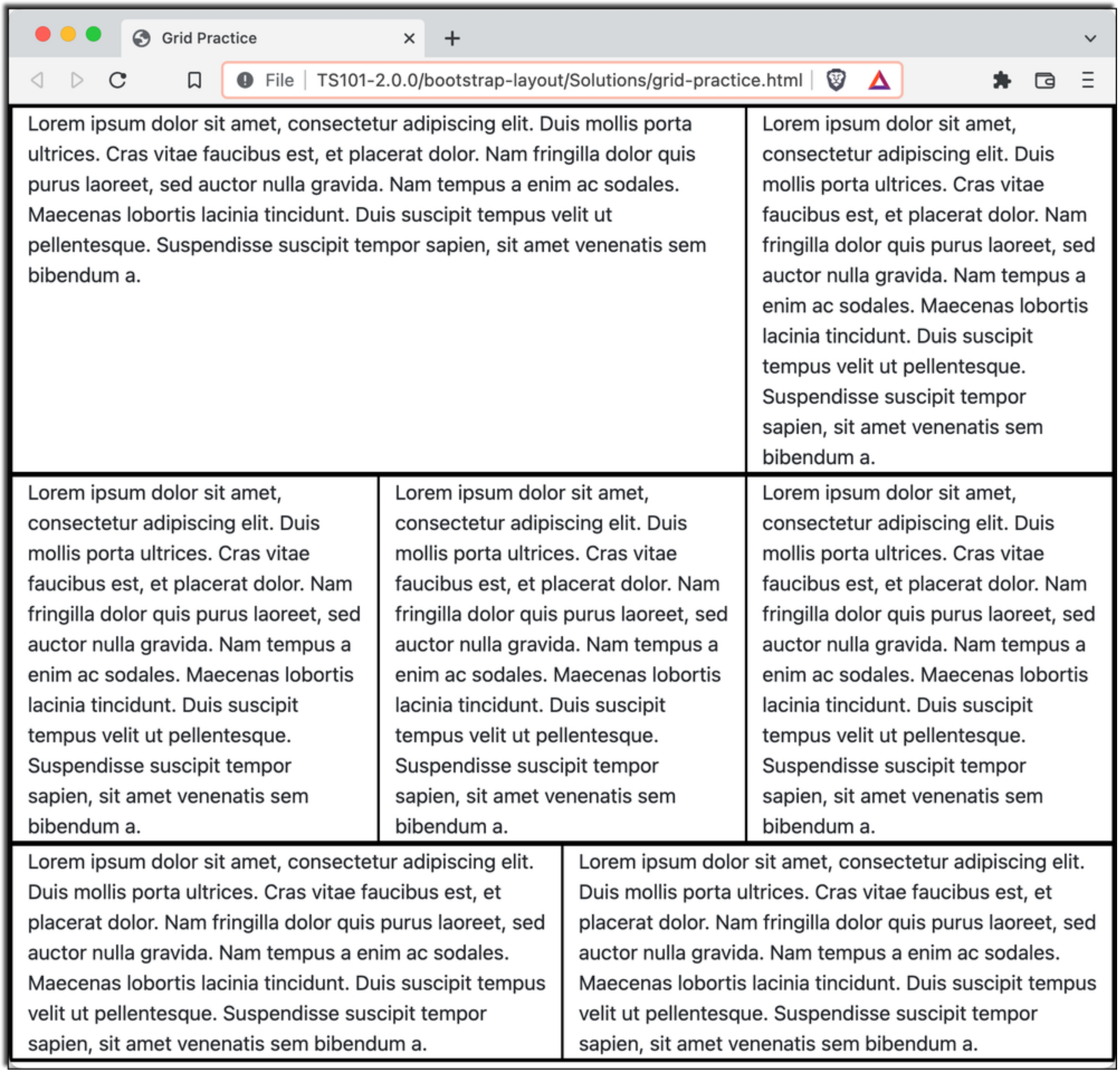
Exercise Code 3.1: layout/Exercises/grid-practice.html

```
     -------Lines 1 through 13 Omitted-------
14.    <div>
15.      Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis
     -------Lines 16 through 21 Omitted-------
22.    </div>
23.    <div>
24.      Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis
     -------Lines 25 through 30 Omitted-------
31.    </div>
32.    <div>
33.      Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis
     -------Lines 34 through 39 Omitted-------
40.    </div>
41.    <div>
42.      Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis
     -------Lines 43 through 48 Omitted-------
49.    </div>
50.    <div>
51.      Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis
     -------Lines 52 through 57 Omitted-------
58.    </div>
59.    <div>
60.      Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis
     -------Lines 61 through 66 Omitted-------
67.    </div>
68.    <div>
69.      Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis
     -------Lines 70 through 75 Omitted-------
76.    </div>
     -------Lines 77 through 78 Omitted-------
```

For this exercise, you will need to add structure (i.e., more `div` elements) and add classes to the `<div>` tags.

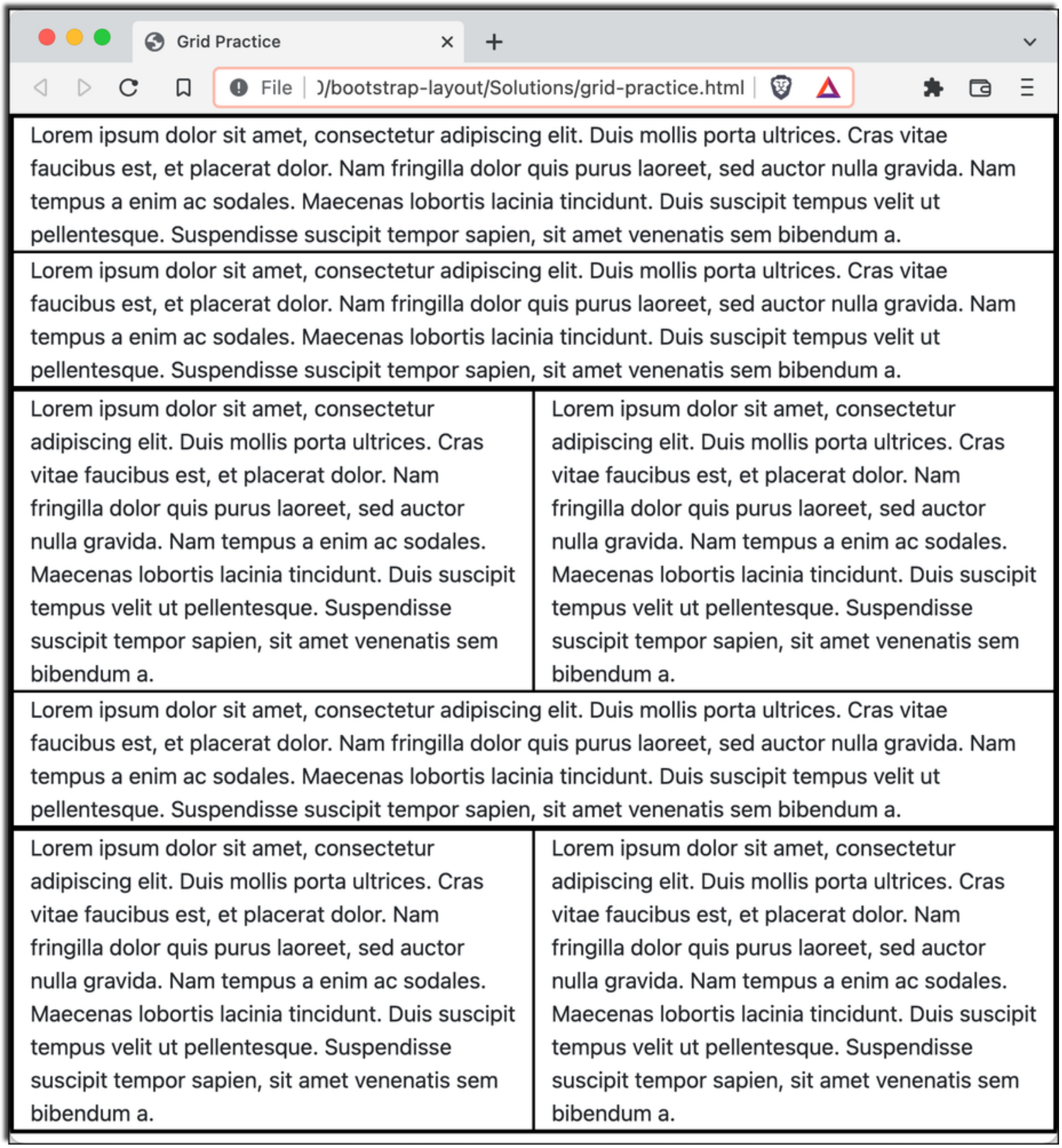
Leave the `style` block in the `head`. We use that simply to create borders, so it's easier to see the rows and columns. Later, you'll learn the Bootstrap way of adding borders.

1. Open the file named `grid-practice.html` inside the `layout/Exercises` folder.
2. Put a link to the Bootstrap CSS in the `<head>` of the page. You can use either the copy of Bootstrap that you downloaded in the previous lesson or the CDN link.
3. Make a responsive grid layout that matches the following layout when viewed in a medium or larger window:



*The fake paragraphs of text were generated using `https://www.lipsum.com/`.

4. Make the same page look like the following when resized to a "small" width:


Grid Practice
File
)/bootstrap-layout/Solutions/grid-practice.html
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis mollis porta ultrices. Cras vitae faucibus est, et placerat dolor. Nam fringilla dolor quis purus laoreet, sed auctor nulla gravida. Nam tempus a enim ac sodales. Maecenas lobortis lacinia tincidunt. Duis suscipit tempus velit ut pellentesque. Suspendisse suscipit tempor sapien, sit amet venenatis sem bibendum a.
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis mollis porta ultrices. Cras vitae faucibus est, et placerat dolor. Nam fringilla dolor quis purus laoreet, sed auctor nulla gravida. Nam tempus a enim ac sodales. Maecenas lobortis lacinia tincidunt. Duis suscipit tempus velit ut pellentesque. Suspendisse suscipit tempor sapien, sit amet venenatis sem bibendum a.
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis mollis porta ultrices. Cras vitae faucibus est, et placerat dolor. Nam fringilla dolor quis purus laoreet, sed auctor nulla gravida. Nam tempus a enim ac sodales. Maecenas lobortis lacinia tincidunt. Duis suscipit tempus velit ut pellentesque. Suspendisse suscipit tempor sapien, sit amet venenatis sem bibendum a.
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis mollis porta ultrices. Cras vitae faucibus est, et placerat dolor. Nam fringilla dolor quis purus laoreet, sed auctor nulla gravida. Nam tempus a enim ac sodales. Maecenas lobortis lacinia tincidunt. Duis suscipit tempus velit ut pellentesque. Suspendisse suscipit tempor sapien, sit amet venenatis sem bibendum a.
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis mollis porta ultrices. Cras vitae faucibus est, et placerat dolor. Nam fringilla dolor quis purus laoreet, sed auctor nulla gravida. Nam tempus a enim ac sodales. Maecenas lobortis lacinia tincidunt. Duis suscipit tempus velit ut pellentesque. Suspendisse suscipit tempor sapien, sit amet venenatis sem bibendum a.
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis mollis porta ultrices. Cras vitae faucibus est, et placerat dolor. Nam fringilla dolor quis purus laoreet, sed auctor nulla gravida. Nam tempus a enim ac sodales. Maecenas lobortis lacinia tincidunt. Duis suscipit tempus velit ut pellentesque. Suspendisse suscipit tempor sapien, sit amet venenatis sem bibendum a.
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis mollis porta ultrices. Cras vitae faucibus est, et placerat dolor. Nam fringilla dolor quis purus laoreet, sed auctor nulla gravida. Nam tempus a enim ac sodales. Maecenas lobortis lacinia tincidunt. Duis suscipit tempus velit ut pellentesque. Suspendisse suscipit tempor sapien, sit amet venenatis sem bibendum a.
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis mollis porta ultrices. Cras vitae faucibus est, et placerat dolor. Nam fringilla dolor quis purus laoreet, sed auctor nulla gravida. Nam tempus a enim ac sodales. Maecenas lobortis lacinia tincidunt. Duis suscipit tempus velit ut pellentesque. Suspendisse suscipit tempor sapien, sit amet venenatis sem bibendum a.

Solution: layout/Solutions/grid-practice.html

```
     -------Lines 1 through 15 Omitted-------
16.    <div class="container-fluid">
17.      <div class="row">
18.        <div class="col-md-8">
19.          Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis
     -------Lines 20 through 25 Omitted-------
26.        </div>
27.        <div class="col-md-4">
28.          Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis
     -------Lines 29 through 35 Omitted-------
36.      </div>
37.      <div class="row">
38.        <div class="col-sm-6 col-md">
39.          Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis
     -------Lines 40 through 46 Omitted-------
47.        <div class="col-sm-6 col-md">
48.          Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis
     -------Lines 49 through 55 Omitted-------
56.        <div class="col-md">
57.          Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis
     -------Lines 58 through 64 Omitted-------
65.      </div>
66.      <div class="row">
67.        <div class="col-sm col-md">
68.          Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis
     -------Lines 69 through 75 Omitted-------
76.        <div class="col-sm col-md">
77.          Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis
     -------Lines 78 through 84 Omitted-------
85.      </div>
86.    </div>
     -------Lines 87 through 93 Omitted-------
```


Conclusion

In this lesson, you have learned about the browser viewport and how it is different from the device width. You also learned how to use Bootstrap's responsive classes to create layouts.

LESSON 3

Creating Responsive Navigation

Topics Covered

☑ The base `nav` component.

☑ Styling the `nav` component.

☑ The `navbar` component.

Introduction

Bootstrap contains many different components that combine CSS and JavaScript to allow you to access powerful responsive functionality and effects just by adding classes to elements. In this lesson, you'll learn how to use two of the most common Bootstrap components: `nav` and `navbar`.

3.1. Using the `nav` Component

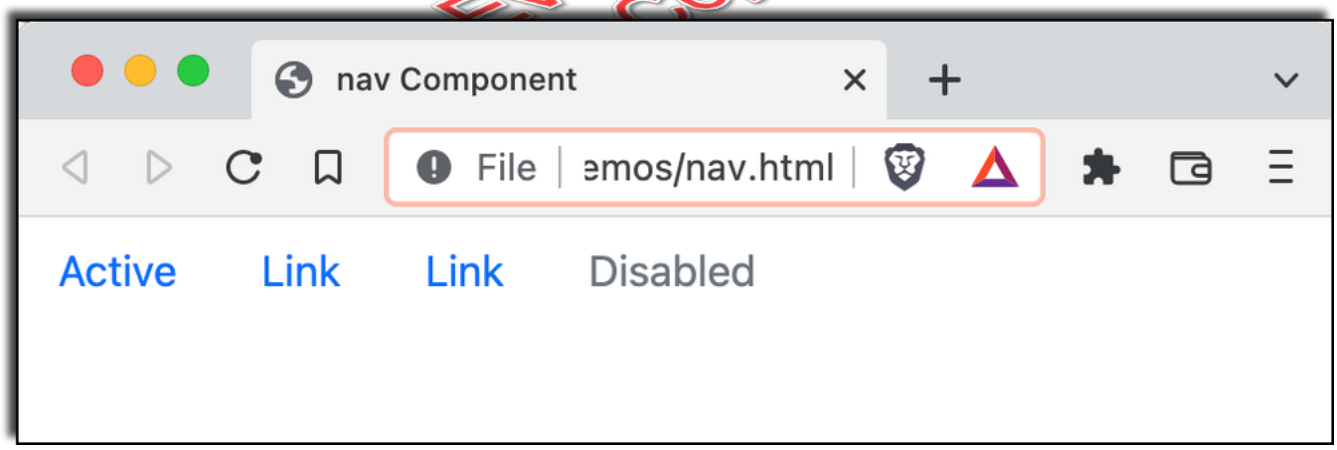
Any website with more than one page will have some sort of navigation. Bootstrap's `nav` component makes it easy for you to create horizontal or vertical navigation links for your website and to manage their styles and alignment.

Here's an example that shows how to use the `nav` component:

Demo 3.1: responsive-navigation/Demos/nav.html

```
      -------Lines 1 through 10 Omitted-------
11.     <ul class="nav">
12.       <li class="nav-item">
13.         <a class="nav-link active" href="#">Active</a>
14.       </li>
15.       <li class="nav-item">
16.         <a class="nav-link" href="#">Link</a>
17.       </li>
18.       <li class="nav-item">
19.         <a class="nav-link" href="#">Link</a>
20.       </li>
21.       <li class="nav-item">
22.         <a class="nav-link disabled" href="#">Disabled</a>
23.       </li>
24.     </ul>
      -------Lines 25 through 30 Omitted-------
```

This code will render the following:



To create navigation with the `nav` component, add the `nav` class to an element and add the `nav-item` class to one or more children of that element. It's a common practice to use an HTML list for navigation, as above, although it's not required. You could also use a `div` or `nav` element with links inside it, as shown below:

Demo 3.2: responsive-navigation/Demos/nav-nav.html

```
      -------Lines 1 through 10 Omitted-------
11.     <nav class="nav">
12.       <a class="nav-item nav-link active" href="#">Active</a>
13.       <a class="nav-item nav-link" href="#">Link</a>
14.       <a class="nav-item nav-link" href="#">Link</a>
15.       <a class="nav-item nav-link disabled" href="#">Disabled</a>
16.     </nav>
      -------Lines 17 through 23 Omitted-------
```

Notice that when you use an HTML list for navigation, the `nav-item` class should be applied to the `li` element, but, when you don't use an HTML list, the `nav-item` class is applied directly to the link.

The `nav` component overrides the styles of the links it contains to add larger click areas, space between the links, and color coding for the different states of a link. By using overrides and modifiers, you can further customize the appearance of a navigation list created using `nav`.

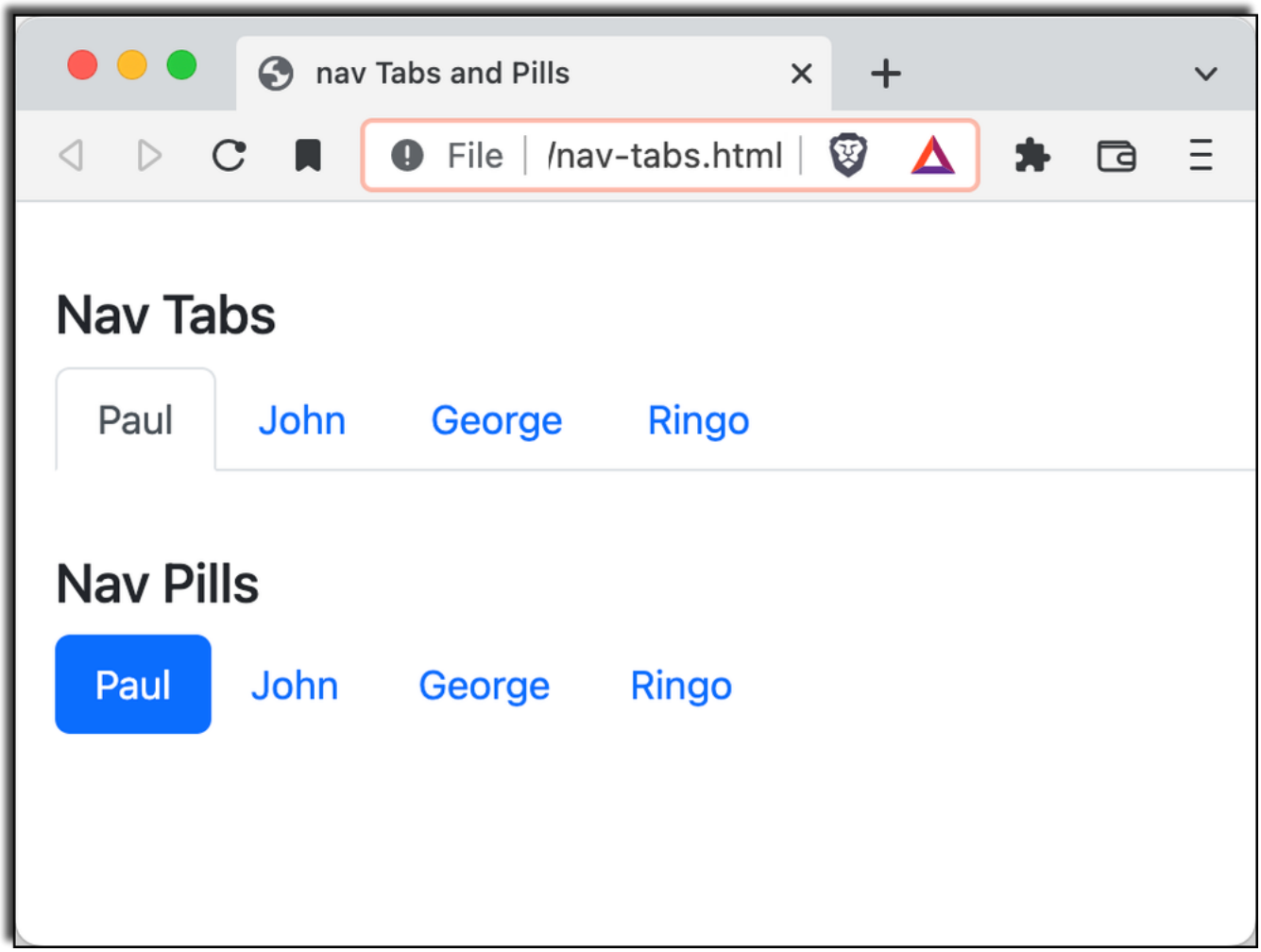
❖ 3.1.1. `nav` Modifiers

Modifier classes allow you to customize the appearance of your `nav` by adding additional classes after the `nav` classes, separated by a space. You can add as many of these as you need to each element to create the effect that you need. Examples of modifier classes include: `nav-tabs` and `nav-pills`, both of which are shown in the code below:

Demo 3.3: responsive-navigation/Demos/nav-tabs-and-pills.html

```
      -------Lines 1 through 19 Omitted-------
20.     <h4>Nav Tabs</h4>
21.     <ul class="nav nav-tabs">
22.       <li class="nav-item">
23.         <a class="nav-item nav-link active" href="#">Paul</a>
24.       </li>
25.       <li class="nav-item">
26.         <a class="nav-item nav-link" href="#">John</a>
27.       </li>
28.       <li class="nav-item">
29.         <a class="nav-item nav-link" href="#">George</a>
30.       </li>
31.       <li class="nav-item">
32.         <a class="nav-item nav-link" href="#">Ringo</a>
33.       </li>
34.     </ul>
35.
36.     <h4>Nav Pills</h4>
37.     <ul class="nav nav-pills">
38.       <li class="nav-item">
39.         <a class="nav-item nav-link active" href="#">Paul</a>
40.       </li>
41.       <li class="nav-item">
42.         <a class="nav-item nav-link" href="#">John</a>
43.       </li>
44.       <li class="nav-item">
45.         <a class="nav-item nav-link" href="#">George</a>
46.       </li>
47.       <li class="nav-item">
48.         <a class="nav-item nav-link" href="#">Ringo</a>
49.       </li>
50.     </ul>
      -------Lines 51 through 57 Omitted-------
```


These modifier classes result in more stylish navigation bars:



The `active` class is used to make one of the tabs or pills active when the page is loaded.

❖ 3.1.2. HTML `data-` Attributes

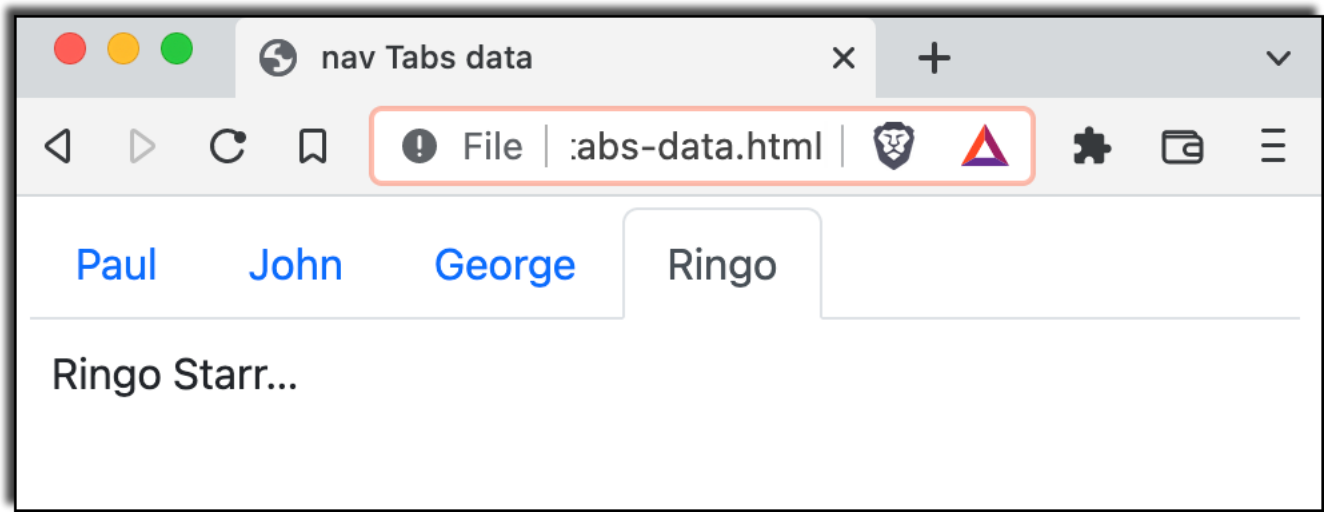
In HTML, attributes that start with `data-` are used to store custom data that can be used by JavaScript. Bootstrap makes use of `data-bs-` attributes to allow you to provide configuration data for components. For example, the `nav` component has a `data-bs-toggle` attribute that tells the Bootstrap JavaScript to toggle highlight the active link in a `nav` component. You can set `data-bs-toggle` to `dropdown`, `tab`, or `pill`, depending on the type of navigation link you want to create.

The following code illustrates the use of the `data-bs-toggle` attribute within a `nav-tabs` component to create a tabbed navigation component:

Demo 3.4: responsive-navigation/Demos/nav-tabs-data.html

```
      -------Lines 1 through 10 Omitted-------
11.     <div class="container">
12.       <ul class="nav nav-tabs" id="my-tabbed-nav-tabs">
13.         <li class="nav-item">
14.           <button class="nav-item nav-link active" id="paul-tab"
15.             data-bs-toggle="tab" data-bs-target="#paul-tab-pane">Paul</button>
16.         </li>
17.         <li class="nav-item"><button class="nav-item nav-link" id="john-tab"
18.             data-bs-toggle="tab" data-bs-target="#john-tab-pane">John</button>
19.         </li>
20.         <li class="nav-item"><button class="nav-item nav-link" id="george-tab"
21.            data-bs-toggle="tab" data-bs-target="#george-tab-pane">George</button>
22.         </li>
23.         <li class="nav-item"><button class="nav-item nav-link" id="ringo-tab"
24.             data-bs-toggle="tab" data-bs-target="#ringo-tab-pane">Ringo</button>
25.         </li>
26.       </ul>
27.       <div class="tab-content" id="myTabContent">
28.         <div class="tab-pane fade show active" id="paul-tab-pane"
29.           role="tabpanel" aria-labelledby="paul-tab" tabindex="0">
30.           <p>Paul McCartney…</p>
31.         </div>
32.         <div class="tab-pane fade" id="john-tab-pane"
33.           role="tabpanel" aria-labelledby="john-tab" tabindex="0">
34.           <p>John Lennon…</p>
35.         </div>
36.         <div class="tab-pane fade" id="george-tab-pane"
37.           role="tabpanel" aria-labelledby="george-tab" tabindex="0">
38.           <p>George Harrison…</p>
39.         </div>
40.         <div class="tab-pane fade" id="ringo-tab-pane"
41.           role="tabpanel" aria-labelledby="ringo-tab" tabindex="0">
42.           <p>Ringo Starr…</p>
43.         </div>
44.       </div>
45.     </div>
      -------Lines 46 through 51 Omitted-------
```


There is a lot going on here, which we will explain momentarily, but first notice, open the file and click the **Ringo** tab. You will "Paul McCartney…" fade out and "Ringo Starr…" fade in. And you will also notice that the **Ringo** tab becomes active:



Things to notice:

1. The `data-bs-toggle` attribute is used with `nav` tabs and pills when the navigation leads to content on the same page. Usually, it is revealing hidden content as in the preceding example. While you could continue to use links (`<a>` tags) for this, we are using `<button>` tags as clicking the tab does not take the user to a new location.
2. The `data-bs-toggle` attribute is sufficient for changing the style of the active tab, but Bootstrap needs to know what to do when a button is clicked. We use the `data-bs-target` to tell Bootstrap what element on the page to show when a button is clicked. For example, `data-bs-target="#paul-tab-pane"` will show the element with the `id` of `paul-tab-pane`.
3. The `show` class in the "paul-tab-pane" `div` is used to make that content visible when the page loads.
4. The `fade` class is used to make the content fade in and out when switching tabs.
5. The `aria-labelled-by` attribute is used for accessibility purposes. It identifies the element that is used to label the given content. It is necessary because the HTML markup itself does not make this relationship clear. For example, from the HTML alone, it is impossible to tell that the content inside of the "paul-tab-pane" `div` is labelled by the "paul-tab" `button`. Setting `aria-labelled-by` to "paul-tab" makes that relationship clear for screen readers.
6. The `role` attribute is discussed next.

❖ 3.1.3. The `role` Attribute

In HTML, elements can often play different "roles" within a page. This is particularly true when you're using Bootstrap. A list of links in Bootstrap could be tabbed navigation, or button navigation (called

"pill" navigation) or dropdown navigation. The way to indicate to Bootstrap which way to display a list is by adding new values to the `class` attribute. A `nav` component will change to a tabbed `nav` with just the addition of the `nav-tabs` attribute. However, for users who use screen readers and for computers that are parsing the page, this change of behavior and style doesn't actually change what the component *is*.

To change the role of an HTML element so that it can be better understood by screen readers and machine parsers, you should use the `role` attribute.

The following code shows how to add `role` attributes to a tabbed navigation component:


```
<div class="nav nav-tabs" id="my-tabbed-nav" role="tablist">
  <a class="nav-item nav-link" id="nav-home"
    data-bs-toggle="tab" href="home.html" role="tab">Home</a>
  <a class="nav-item nav-link" id="nav-about"
    data-bs-toggle="tab" href="about.html" role="tab">About Us</a>
</div>
```

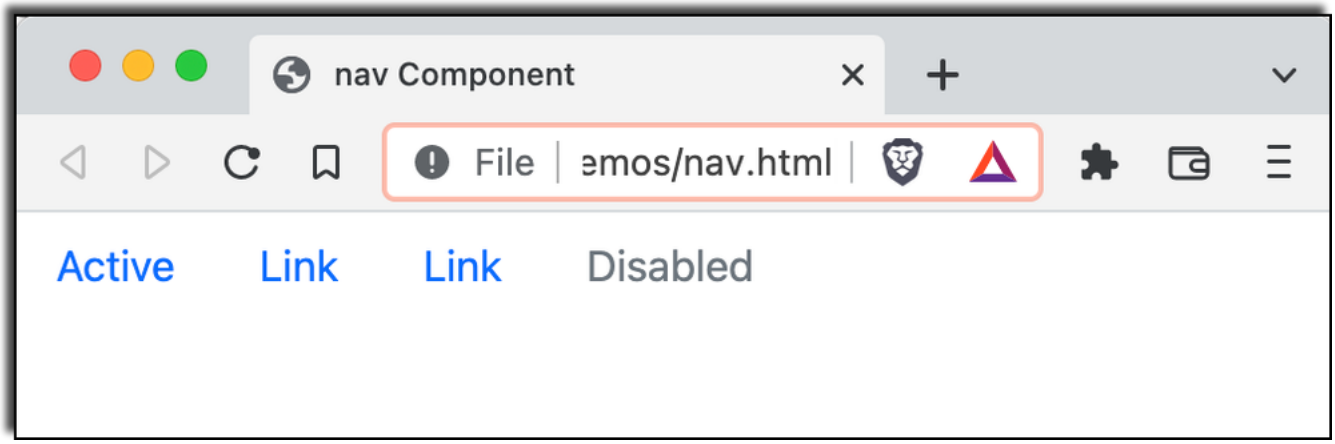
To learn more about the `role` attribute, see `https://developer.mozilla.org/en-US/docs/Web/Accessibility/ARIA/Roles`.

Exercise 4: Working with nav Modifier Classes

15 to 25 minutes

In this exercise, you will learn about some of the ways you can modify the appearance of a `nav` with modifier classes.

1. Open `responsive-navigation/Exercises/nav-modifier-classes.html`.
2. Preview this page in your browser. You should see the same links that are shown in the basic `nav` image you saw earlier in this lesson:



3. Remove the `active` class from the first link and reload the page. Notice that nothing happens. The `active` class doesn't cause any style changes to the base `nav` class.
4. Remove the `disabled` class and reload the page to see the result.
5. Change the base `nav` to tabbed navigation by adding the `nav-tabs` class to the `ul` element. The `ul` should now have two classes, separated by a space, like this:

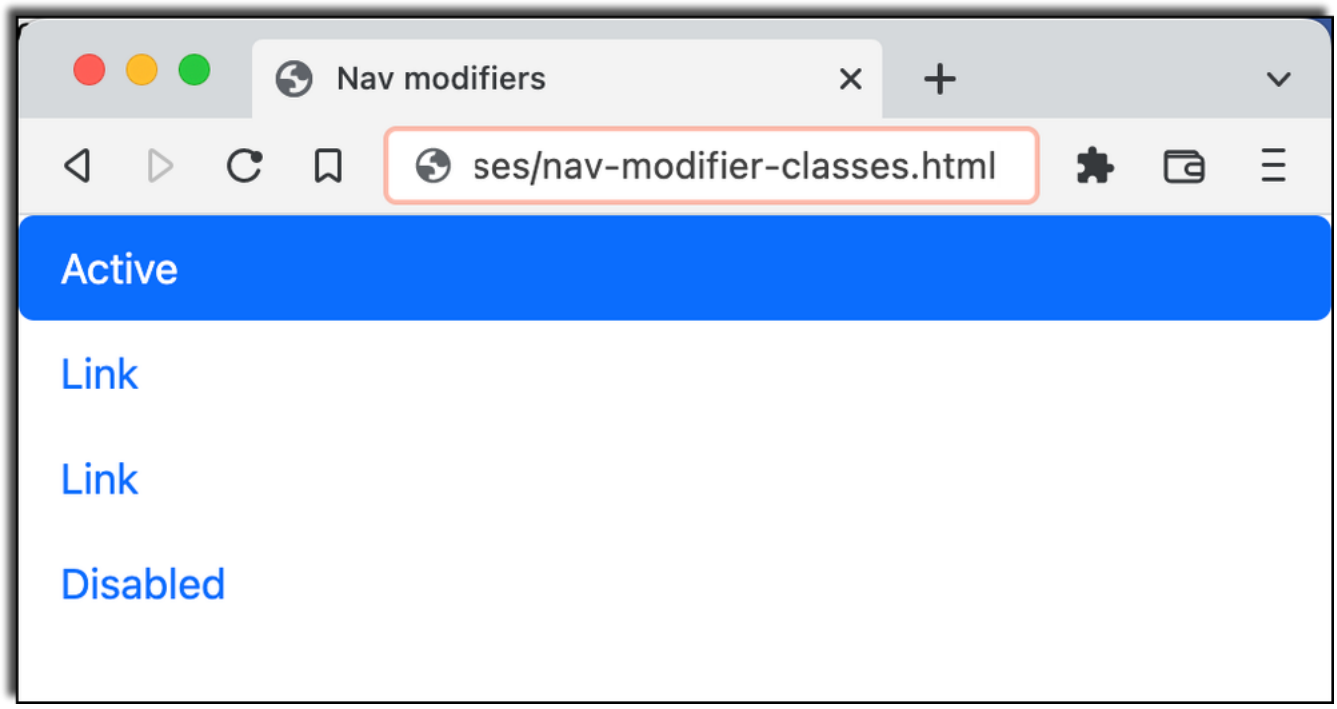
```
<ul class="nav nav-tabs">
```

6. Preview your page in a browser to see the tabbed navigation. Hover over the links to see them change subtly.
7. Add the `active` class back onto the first link. Notice that the active tab appears to be in front of the others, as it would in a stack of tabbed file folders.

8. Change the `nav-tabs` class to `nav-pills` and view the results.
9. Turn the `nav` component into a vertical `nav` component by adding the `flex-column` class to the same root `nav` element (the `ul` element that has the `nav` class):

```
<ul class="nav nav-pills flex-column">
```

The page should now look like this:



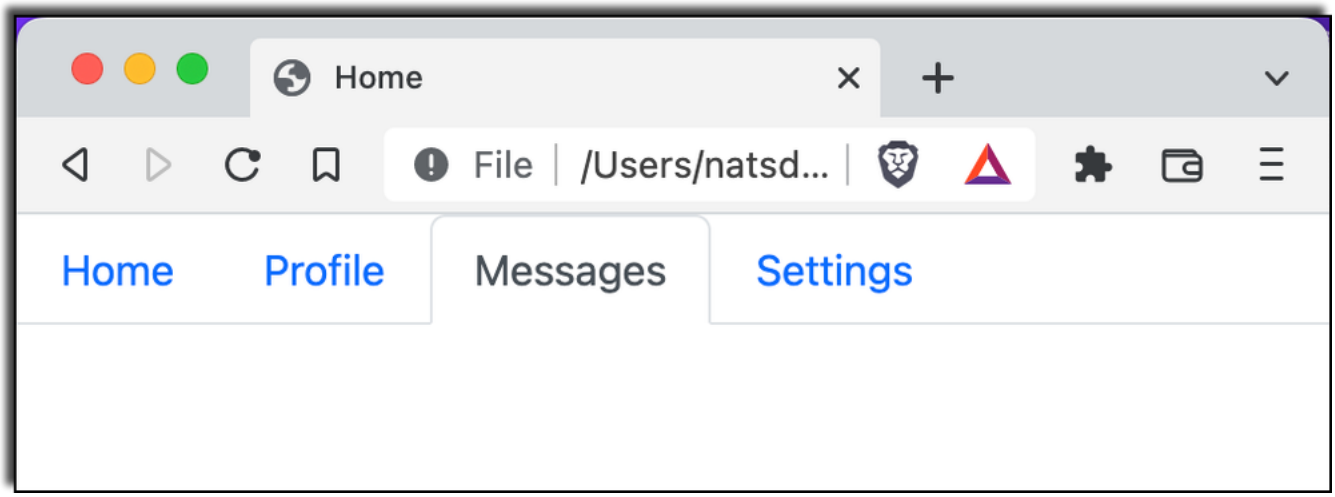
Visit `https://getbootstrap.com/docs` and navigate to **Components** and then to **Navs**. Read about the other available modifiers for the base nav component.

Challenge

Change the navigation you just created into a tabbed or pill navigation that changes the active element when you click different items in the navigation. You may find the JavaScript behavior documentation[2] useful.

The finished page should appear as follows:

2. https://getbootstrap.com/docs/5.2/components/navs-tabs/#javascript-behavior


Home
File | /Users/natsd...
Home
Profile
Messages
Settings

Solution: responsive-navigation/Solutions/nav-modifier-classes.html

```
     -------Lines 1 through 10 Omitted-------
11.    <ul class="nav nav-pills flex-column">
12.      <li class="nav-item">
13.        <a class="nav-link active" href="#">Active</a>
14.      </li>
15.      <li class="nav-item">
16.        <a class="nav-link" href="#">Link</a>
17.      </li>
18.      <li class="nav-item">
19.        <a class="nav-link" href="#">Link</a>
20.      </li>
21.      <li class="nav-item">
22.        <a class="nav-link" href="#">Disabled</a>
23.      </li>
24.    </ul>
     -------Lines 25 through 31 Omitted-------
```

Challenge Solution: responsive-navigation/Solutions/nav-challenge.html


```
     -------Lines 1 through 10 Omitted-------
11.    <ul class="nav nav-tabs" id="my-tab" role="tablist">
12.      <li class="nav-item">
13.        <a class="nav-link active" id="home-tab"
14.          data-bs-toggle="tab" href="#home" role="tab">Home</a>
15.      </li>
16.      <li class="nav-item">
17.        <a class="nav-link" id="profile-tab"
18.          data-bs-toggle="tab" href="#profile" role="tab">Profile</a>
19.      </li>
20.      <li class="nav-item">
21.        <a class="nav-link" id="messages-tab"
22.          data-bs-toggle="tab" href="#messages" role="tab">Messages</a>
23.      </li>
24.      <li class="nav-item">
25.        <a class="nav-link" id="settings-tab"
26.          data-bs-toggle="tab" href="#settings" role="tab">Settings</a>
27.      </li>
28.    </ul>
     -------Lines 29 through 34 Omitted-------
```

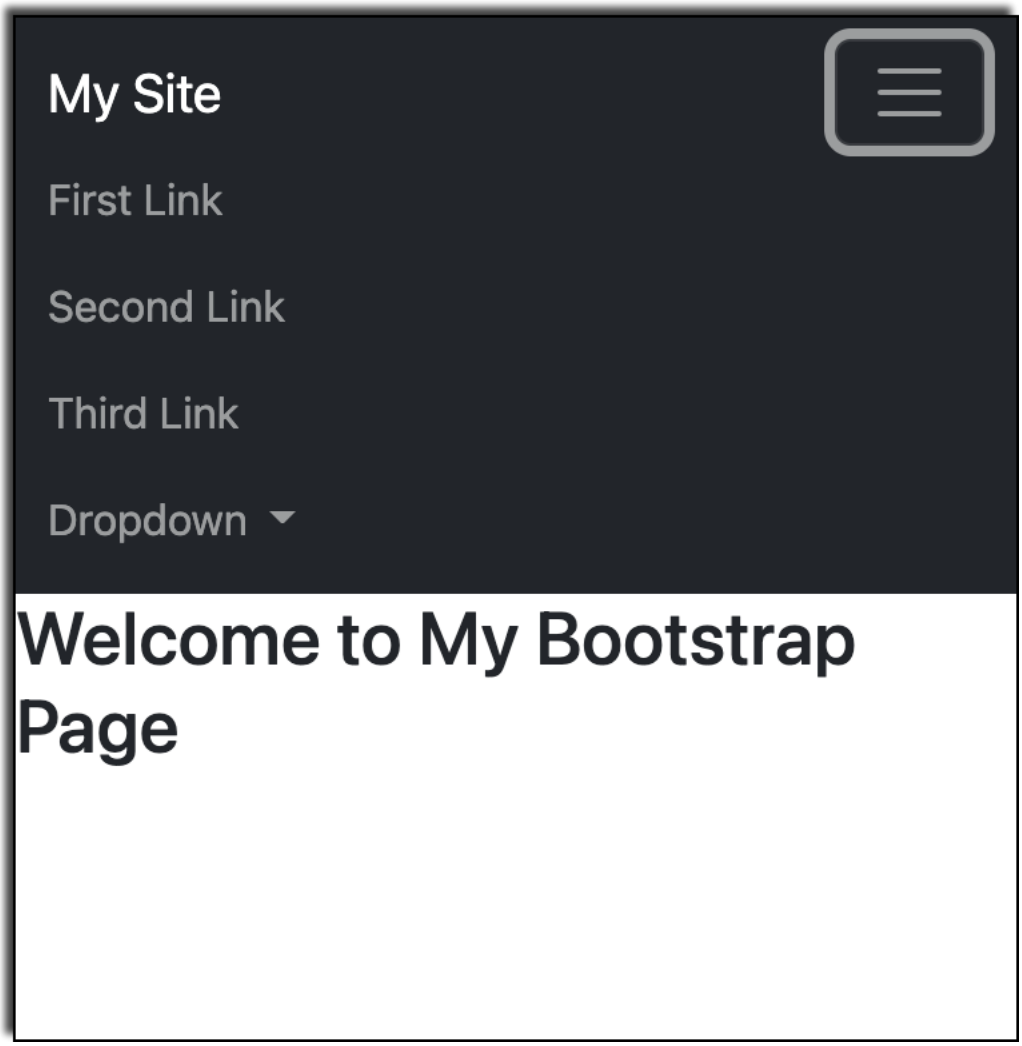
3.2. Using the `navbar` Component

The `navbar` component creates a responsive navigation header. To see what the `navbar` component is capable of, visit the `navbar` examples page on the Bootstrap website, at `https://getbootstrap.com/docs/5.2/examples/navbars/`. If you're viewing the page on a desktop or laptop browser you'll see a series of horizontal navigation headers.

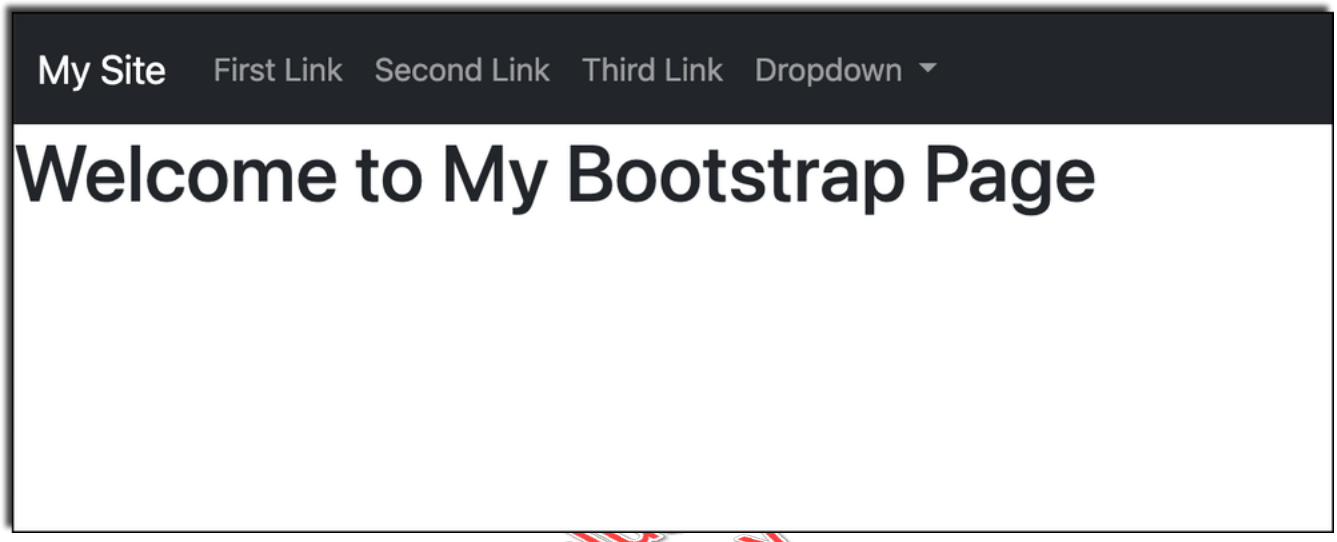
Change the size of the viewport to see how the different navigation headers expand and collapse at different sizes.

Navigation headers include links that you can apply modifier classes to, like you did with the `nav` component. They can also include logos, forms, and dropdowns. What really makes the `navbar` component useful is its ability to display as a mobile-friendly navigation dropdown on small devices, and to expand for larger devices. Also, headers created using the `navbar` component are hidden by default when printing.

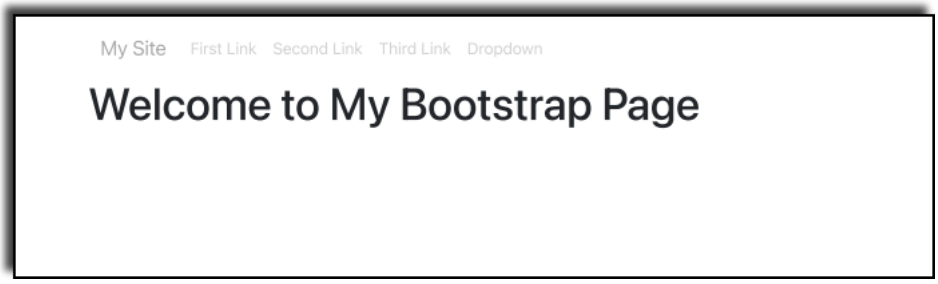
The following three images show how the same `navbar` (`responsive-navigation/Demos/responsive-navbar.html`) looks on a mobile device (with the menu open), in a desktop browser, and when printed:



Mobile



Desktop



Print

In the following exercise, you'll create this basic responsive navigation header using `navbar`.

Exercise 5: Create a Responsive Navigation Header

15 to 25 minutes

In this exercise, you'll use the Bootstrap `navbar` component to create an expanding (and collapsing) navigation header.

1. Open `responsive-navigation/Exercises/responsive-navbar.html`.
2. Add the `navbar` class to the `nav` element.
3. Add the `navbar-expand-lg` class to the `nav` element. There should now be two different class values in the `nav` element's `class` attribute, like this:

```
<nav class="navbar navbar-expand-lg">
```


4. Style the `navbar` by adding the `navbar-dark` and `bg-dark` classes to the `nav` element.
5. Create a `div` element inside the `nav` element and give it a class of `container-fluid`. This will be the container for your navigation.
6. Create a link just inside the `nav` component and assign it the `navbar-brand` class. This class will position the linked text or image to the left of the navigation, which is traditionally the place where website "branding" (such as a logo, for example) is placed.

```
<nav class="navbar navbar-expand-lg">
  <a class="navbar-brand" href="#">My Site</a>
</nav>
```

7. To create the button for the collapsed navigation, also known as the **navbar toggler**, create a `button` element with a `navbar-toggler` class, a `data-bs-toggle` attribute, and a `data-bs-target` attribute.

```
<nav class="navbar navbar-expand-lg">
  <a class="navbar-brand" href="#">My Site</a>
  <button class="navbar-toggler"
    type="button"
    data-bs-toggle="collapse"
    data-bs-target="#navbar-content">
    <span class="navbar-toggler-icon"></span>
  </button>
</nav>
```

8. Create a `div` element below the **navbar toggler** and give it the `collapse` and `navbar-collapse` classes. Give it an `id` attribute with a value of `navbar-content`. This is the element that will hold the navigation links that will be collapsed for mobile devices and expanded for large screens.

9. Inside the new `div`, make an unordered list containing several list items containing links. Use `#` for the values of the `href` attributes. In a real site, these would contain links to other pages or other anchors within the same document, but for demonstration purposes, we'll just link all of the navigation items to nowhere.


```
<ul>
  <li>
    <a href="#">First Link</a>
  </li>
  <li>
    <a href="#">Second Link</a>
  </li>
  <li>
    <a href="#">Third Link</a>
  </li>
</ul>
```

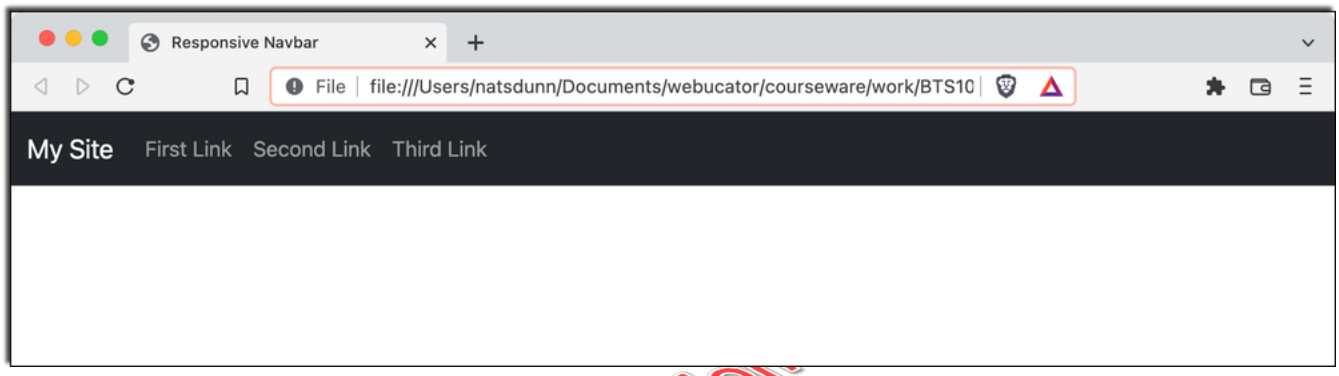
10. Apply the `navbar-nav` class to the `ul` element. The `navbar-nav` class works the same as the `nav` class, but inside of a `navbar`.

11. Add the `nav-item` class to each of the `li` elements.

12. Add the `nav-link` class to each of the `a` elements.

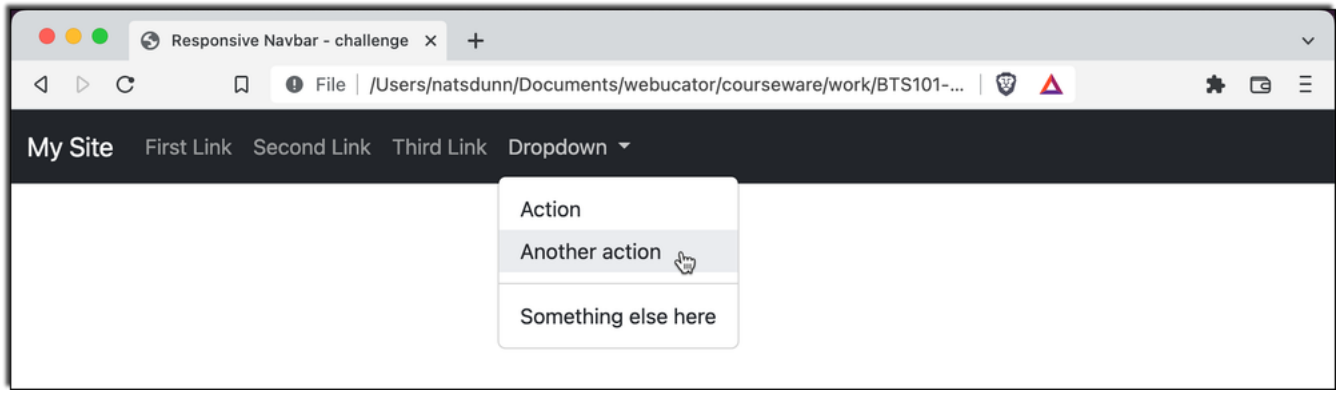
13. Save the page and open it in a browser. Drag the right edge of the browser to resize it. If you did everything correctly, the navigation with switch between expanded and collapsed as you resize between larger and smaller than the `lg` breakpoint.

The finished page should appear as follows:



Challenge

For a challenge, visit the `Bootstrap documentation (https://getbootstrap.com/docs/5.2/components/navs-tabs/#using-dropdowns)` and figure out how to add a dropdown list to your navigation header:



Solution: responsive-navigation/Solutions/responsive-navbar.html

```
      -------Lines 1 through 10 Omitted-------
11.     <nav class="navbar navbar-dark bg-dark navbar-expand-lg">
12.     <div class="container-fluid">
13.       <a class="navbar-brand" href="#">My Site</a>
14.       <button class="navbar-toggler" type="button"
15.         data-bs-toggle="collapse" data-bs-target="#navbar-content">
16.         <span class="navbar-toggler-icon"></span>
17.       </button>
18.       <div class="collapse navbar-collapse" id="navbar-content">
19.         <ul class="navbar-nav">
20.           <li class="nav-item">
21.             <a class="nav-link" href="#">First Link</a>
22.           </li>
23.           <li class="nav-item">
24.             <a class="nav-link" href="#">Second Link</a>
25.           </li>
26.           <li class="nav-item">
27.             <a class="nav-link" href="#">Third Link</a>
28.           </li>
29.         </ul>
30.       </div>
31.     </div>
32.     </nav>
      -------Lines 33 through 39 Omitted-------
```

Solution: responsive-navigation/Solutions/responsive-navbar-challenge.html

```
     -------Lines 1 through 10 Omitted-------
11.    <nav class="navbar navbar-dark bg-dark navbar-expand-lg">
12.    <div class="container-fluid">
13.      <a class="navbar-brand" href="#">My Site</a>
14.      <button class="navbar-toggler" type="button"
15.        data-bs-toggle="collapse" data-bs-target="#navbar-content">
16.        <span class="navbar-toggler-icon"></span>
17.      </button>
18.      <div class="collapse navbar-collapse" id="navbar-content">
19.        <ul class="navbar-nav">
20.          <li class="nav-item">
21.            <a class="nav-link" href="#">First Link</a>
22.          </li>
23.          <li class="nav-item">
24.            <a class="nav-link" href="#">Second Link</a>
25.          </li>
26.          <li class="nav-item">
27.            <a class="nav-link" href="#">Third Link</a>
28.          </li>
29.          <li class="nav-item dropdown">
30.            <a class="nav-link dropdown-toggle" href="#"  role="button"
31.              id="navbar-dropdown" data-bs-toggle="dropdown">
32.              Dropdown
33.            </a>
34.            <div class="dropdown-menu">
35.              <a class="dropdown-item" href="#">Action</a>
36.              <a class="dropdown-item" href="#">Another action</a>
37.              <div class="dropdown-divider"></div>
38.              <a class="dropdown-item" href="#">Something else here</a>
39.            </div>
40.          </li>
41.        </ul>
42.      </div>
43.      </div>
44.    </nav>
     -------Lines 45 through 51 Omitted-------
```

Conclusion

In this lesson, you have learned how to use Bootstrap's `nav` and `navbar` components to create responsive navigation and navigation headers.

LESSON 4
Bootstrap Typography

Topics Covered

☑ The Reboot stylesheet.

☑ `rem` and `em`.

☑ Bootstrap's block and inline typography classes.

☑ Styling text.

Introduction

Typography is the visual component of the written word. It relates to the size, color, shape, and spacing of letters, words, sentences, headings, paragraphs and more. Anyone who has done much web development knows that the way type appears in browsers may change for each user based on a large number of factors, including:

1. The size of the user's browser and screen.
2. The fonts installed on the user's computer.
3. The browser used to view the web page.
4. The CSS styles supported by the browser and how they were applied to the page.
5. Various settings on the user's computer.

Bootstrap simplifies typography on the web by creating a consistent and attractive baseline that developers can build upon or modify. In this lesson, you'll learn what kinds of typography changes Bootstrap makes by default and how to use the classes Bootstrap provides to create the typographical effect that you want.

4.1. How Bootstrap Updates Browser Defaults

Many of the typography benefits you get from using Bootstrap happen automatically without you having to think about them at all. Once you include the Bootstrap CSS into a web page, it just magically

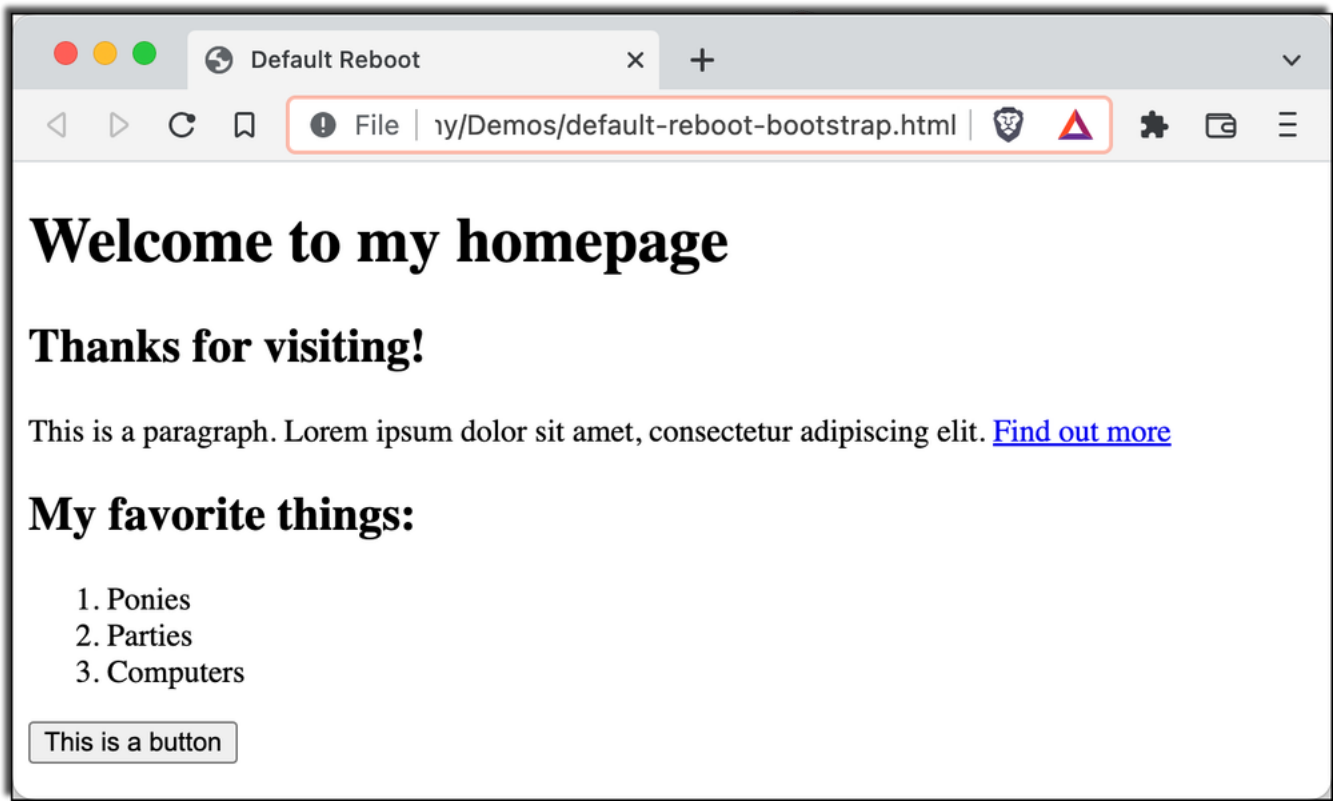
looks better. This is because of how Bootstrap updates the styles that automatically get applied to HTML elements by web browsers, called the browser default styles.

❖ 4.1.1. Introducing Reboot

Reboot is a CSS reset. A CSS reset is a stylesheet that developers can include before any other stylesheets to override browser default styles and make web pages look the same on any browser. The term reset was first used to refer to the **reset.css** file created by the original CSS guru, Eric Meyer. Over the years, improvements have been made to Meyers' CSS reset. The most popular of these, and the one that Bootstrap included prior to version 4.0, is **normalize.css**.

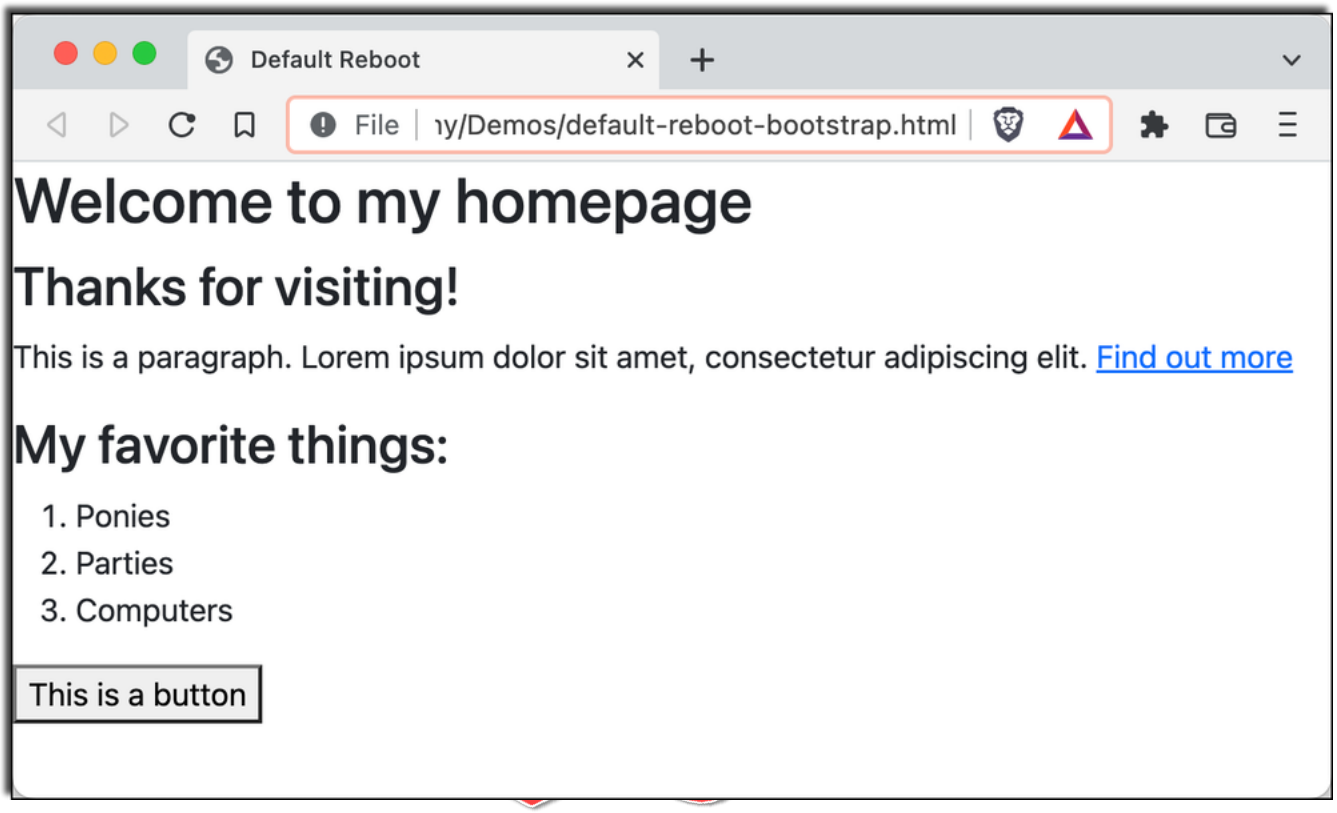
With version 4.0 of Bootstrap, the CSS reset stylesheet that Bootstrap uses was changed to **Reboot.css**. **Reboot.css** is based on **normalize.css**, but it includes some improvements to set a more attractive baseline than previous CSS resets.

To see what a difference the CSS reset makes to a plain HTML file, look at the following plain HTML file[3], without any CSS applied, and then at the same file with `Reboot.css` applied:



Without Reboot.css

3. The HTML file used for these screenshots is at `typography/Demos/default-reboot-bootstrap.html`.



With Reboot.css

The HTML used to produce the preceding screenshots doesn't have any special styles applied. All of the differences between how the page displays with no additional CSS added and how it displays with Reboot or Bootstrap applied are due to changes that Reboot makes to the default element styles.

Rather than starting with the rather boring and unattractive default browser styles, Reboot lets you start styling your page from a more modern-looking default style. Some of the changes that Reboot makes to the default styles are:

- Some margins have been set to 0. This change is made more for consistency than appearance. Designers using Bootstrap know that they have to be explicit about the size of the margin.
- The `font-family`, `line-height`, and `text-align` styles have been given default values.
- The `background-color` has been set to white (`#fff`).
- Some default styles use rems rather than ems.

For additional changes with screenshots, see `https://getbootstrap.com/docs/5.2/content/reboot/`.

4.2. Understanding rem and em

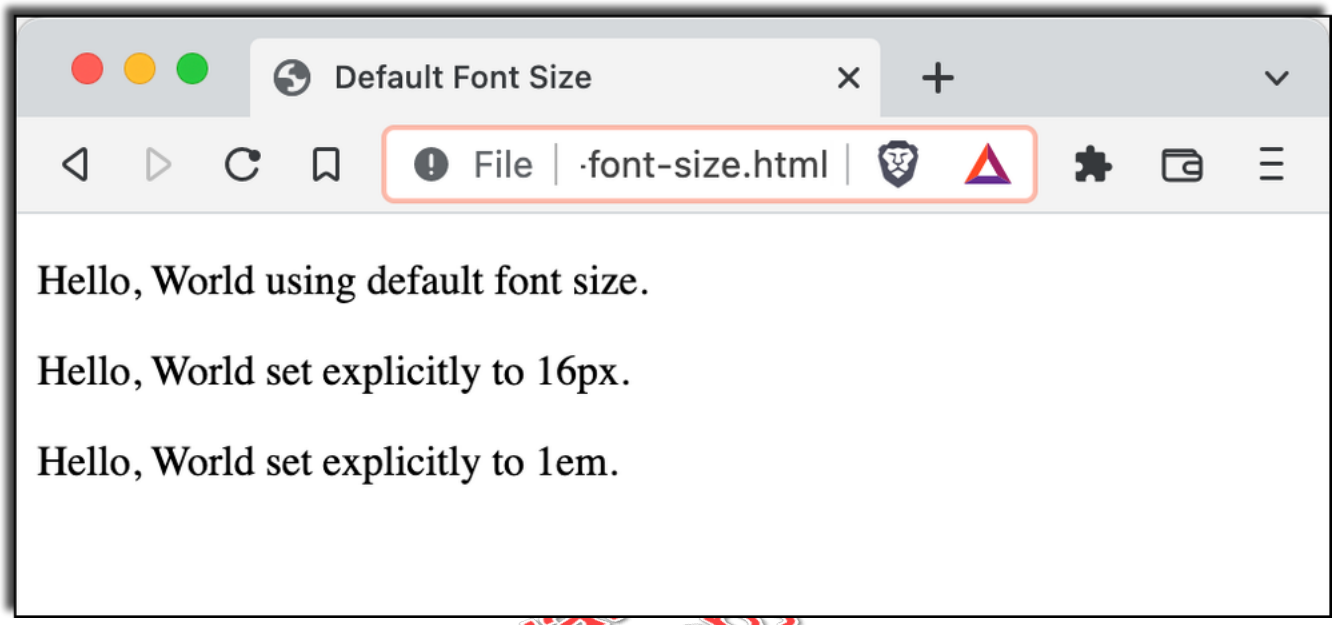
If you've been around web development for a while, or if you have experience with print publishing, you've likely heard of the `em` unit. It's a unit of measurement that's relative to the font size of its direct or nearest parent. Browsers usually have a default font size of `16px`. So, unless that's changed, `1em` would be `16px`, `2em` would be `32px`, and `1.5em` would be `24px`.

The following demo will show that the default font size in an HTML document is `16px`:

Demo 4.1: typography/Demos/default-font-size.html


```
1.   <!DOCTYPE html>
2.   <html lang="en">
3.   <head>
4.   <meta charset="UTF-8">
5.   <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-
           fit=no">
6.   <title>Default Font Size</title>
7.   </head>
8.   <body>
9.     <p>Hello, World using default font size.</p>
10.    <p style="font-size:16px;">Hello, World set explicitly to 16px.</p>
11.    <p style="font-size:1em;">Hello, World set explicitly to 1em.</p>
12.  </body>
13.  </html>
```

Notice that the font size is the same in all three paragraphs:



❖ 4.2.1. em

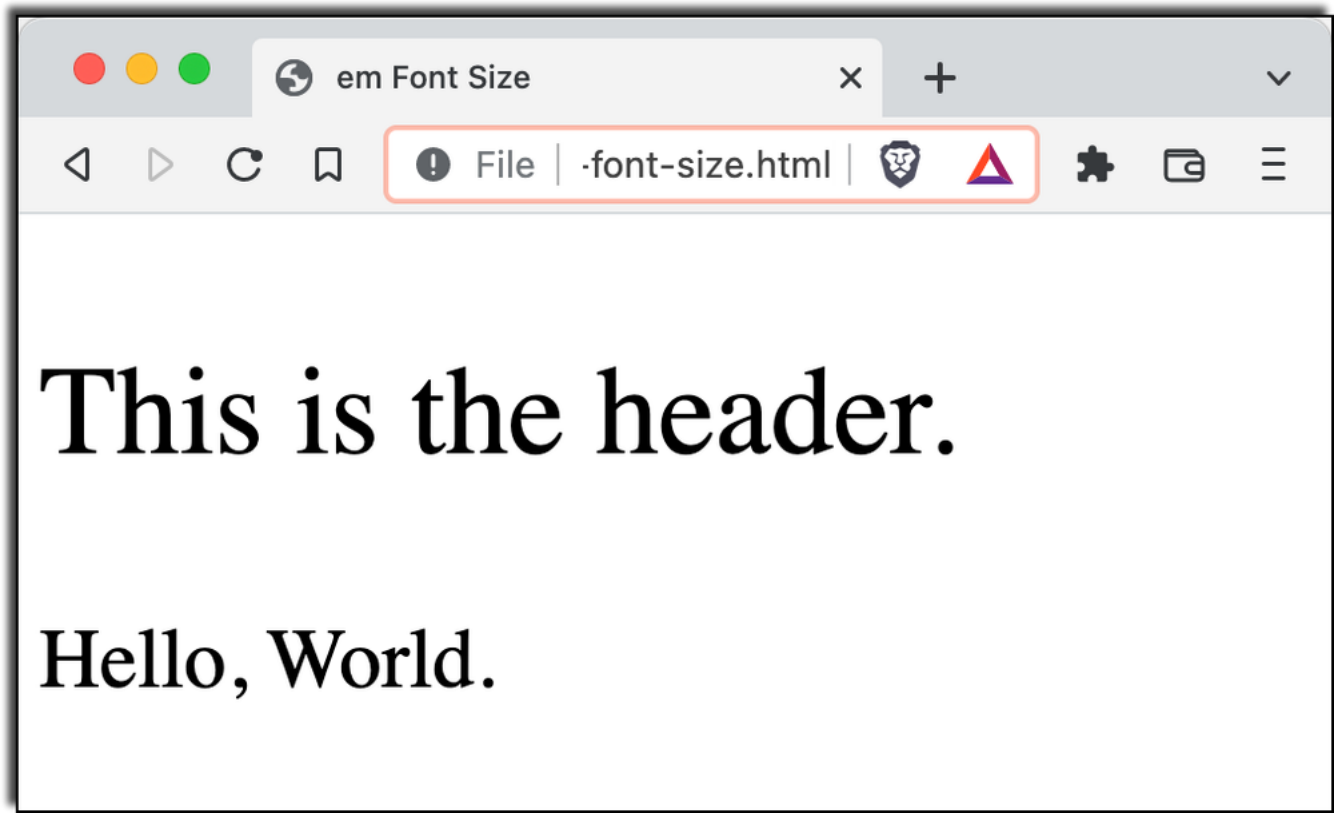
As we just saw, setting the `font-size` of the `p` element to `1em` doesn't change the font size. When you set the font size using `em`, you are setting it relative to its parent's font size. So, the text of a child element with a font size of `2em` will be twice as big as its parent element's text. If the child element has its own child element with a font size set explicitly to `2em`, that child element's text will be twice as big as its parent's text and four times as big as its grandparent's text.

The following example includes a `header` tag around the paragraph text. The `font-size` of the `header` element is set to `1.5em` and the font-size of the `p` element is set to `2em`. The result is that the `p` element inside the `header` will have a font size of `48px` (`16px x 1.5 x 2`), while the `p` element outside of the `header` will have a font size of `32px` (`16px x 2`):

Demo 4.2: typography/Demos/em-font-size.html

```
1.   <!DOCTYPE html>
2.   <html lang="en">
3.   <head>
4.   <meta charset="UTF-8">
5.   <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-
             fit=no">
6.   <title>em Font Size</title>
7.   <style>
8.     header {
9.       font-size: 1.5em;
10.    }
11.
12.    p {
13.      font-size: 2em;
14.    }
15.  </style>
16.  </head>
17.  <body>
18.    <header>
19.      <p>This is the header.</p>
20.    </header>
21.    <p>Hello, World.</p>
22.  </body>
23.  </html>
```


The preceding code will render the following:



❖ 4.2.2. rem

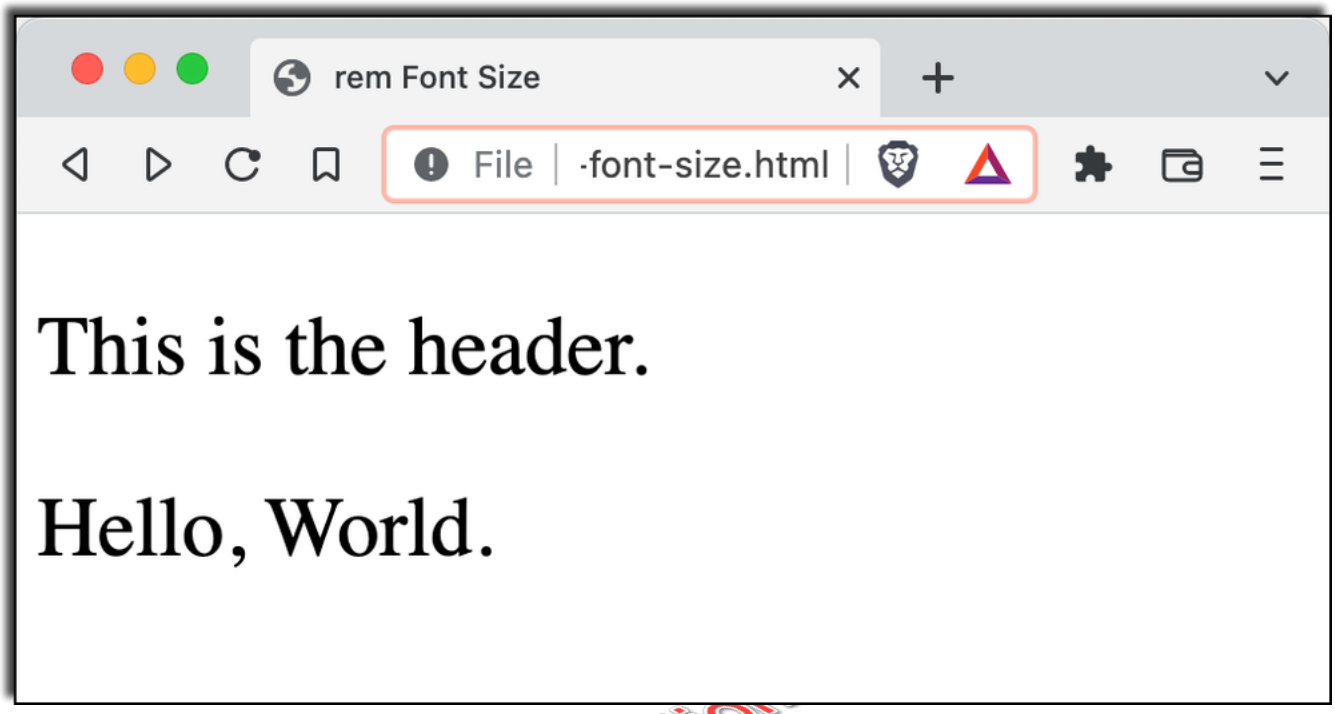
Whereas `em` resizes fonts based on the parent element's font size, `rem` resizes fonts relative to the font size of the `html` element. In the following example, the `em` units have been changed to `rem` units. The result is that both the text inside the `header` and the text outside of the header will be the same size, because both are paragraphs, which are specified as `2rem`, or twice the size of the text of the `html` element, which is still `16px`.

The following demo shows how `rem` adjust font size relative to the `font-size` property of the `html` element:

Demo 4.3: typography/Demos/rem-font-size.html

```
1.   <!DOCTYPE html>
2.   <html lang="en">
3.   <head>
4.   <meta charset="UTF-8">
5.   <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-
             fit=no">
6.   <title>rem Font Size</title>
7.   <style>
8.     header {
9.       font-size: 1.5rem;
10.    }
11.
12.    p {
13.      font-size: 2rem;
14.    }
15.  </style>
16.  </head>
17.  <body>
18.    <header>
19.      <p>This is the header.</p>
20.    </header>
21.    <p>Hello, World.</p>
22.  </body>
23.  </html>
```


The preceding code will render the following:



❖ 4.2.3. Headings and Paragraphs

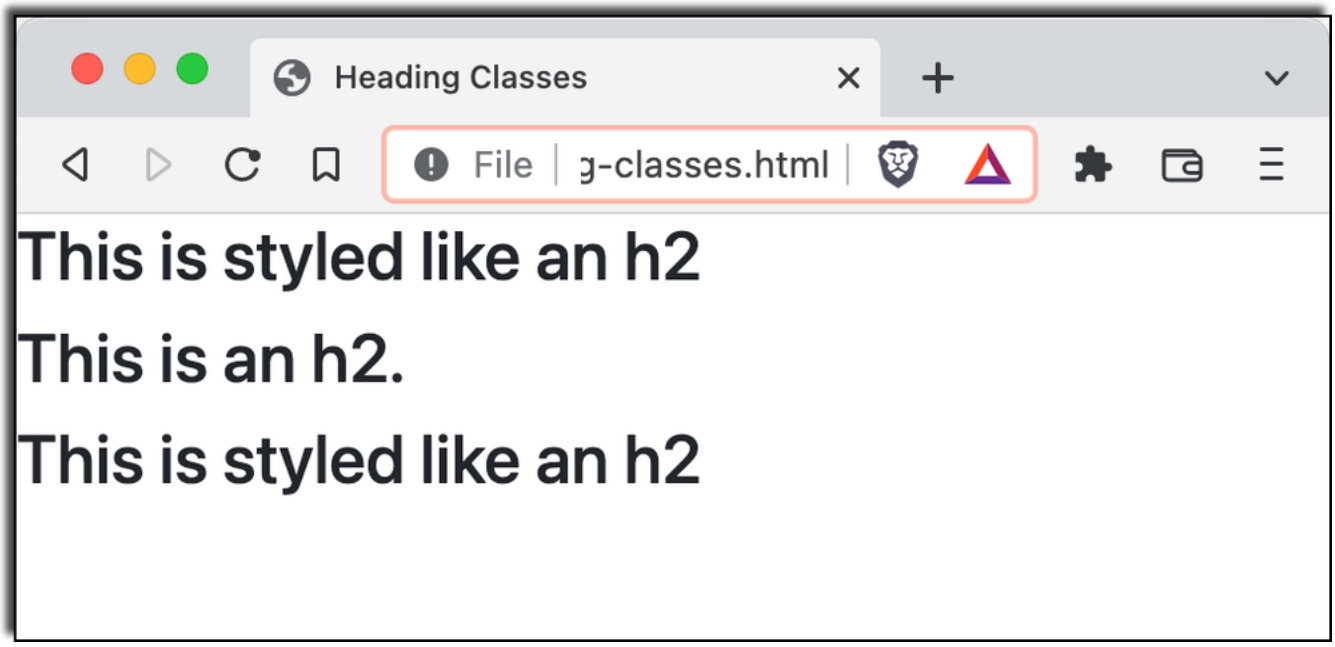
Heading elements (`h1` - `h6`) and paragraphs have their top-margins removed in Bootstrap. Headings have a `.5rem` bottom margin, and paragraphs have a `1rem` bottom margin. Bootstrap also provides *heading* classes (e.g., `class="h2"`) that can be applied to any element.

The following code renders text inside a `header` element and text inside a `footer` element using the `h2` class. It also contains an `h2` element. Notice that all three blocks of text render with the same margins and font size:

Demo 4.4: typography/Demos/heading-classes.html

```
     -------Lines 1 through 10 Omitted-------
11.    <header class="h2">This is styled like an h2</header>
12.    <h2>This is an h2.</h2>
13.    <footer class="h2">This is styled like an h2</footer>
     -------Lines 14 through 15 Omitted-------
```

The preceding code will render the following:



Display Headings

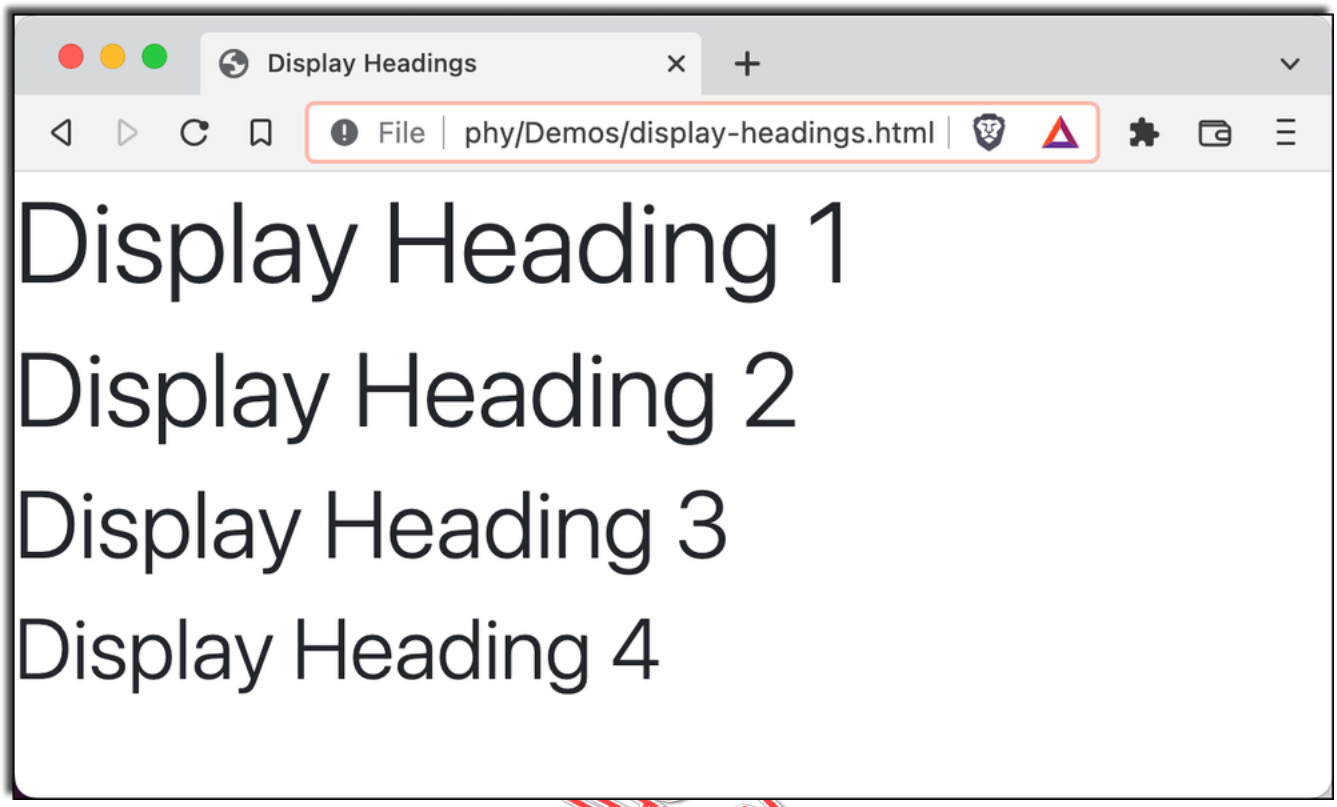
When you need a heading to really stand out, you can use **Display headings**, which are larger versions of headings. The classes for using display headings are `display-1`, `display-2`, `display-3`, and `display-4`.

The following demo shows what each of the display headings looks like:

Demo 4.5: typography/Demos/display-headings.html

```
     -------Lines 1 through 9 Omitted-------
10.  <body>
11.    <h1 class="display-1">Display Heading 1</h1>
12.    <h1 class="display-2">Display Heading 2</h1>
13.    <h1 class="display-3">Display Heading 3</h1>
14.    <h1 class="display-4">Display Heading 4</h1>
15.  </body>
16.  </html>
```

The preceding code will render the following:



Lead Paragraphs

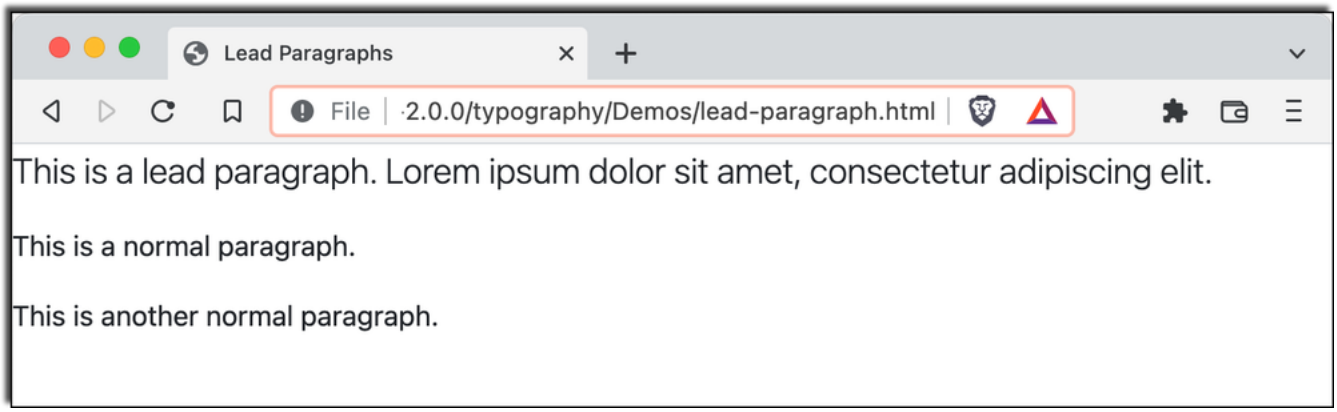
If you want to make a paragraph stand out from the others, you can use the `lead` class.

The following demo shows the effect of the `lead` class:

Demo 4.6: typography/Demos/lead-paragraph.html

```
     -------Lines 1 through 9 Omitted-------
10.  <body>
11.    <p class="lead">
12.      This is a lead paragraph.
13.      Lorem ipsum dolor sit amet, consectetur adipiscing elit.
14.    </p>
15.    <p>This is a normal paragraph.</p>
16.    <p>This is another normal paragraph.</p>
17.  </body>
18.  </html>
```

The preceding code will render the following:



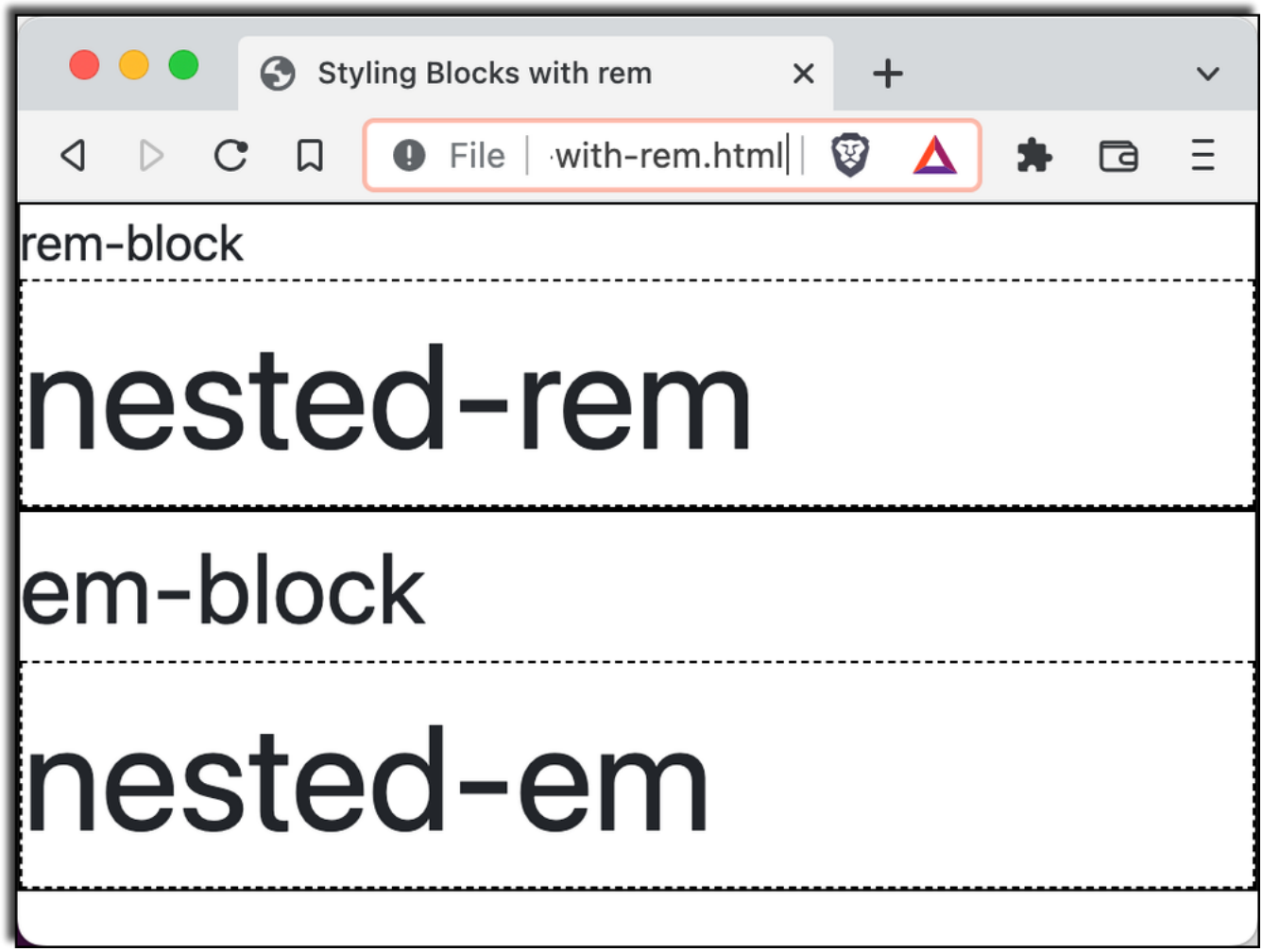
Exercise 6: Styling Blocks with rem

5 to 15 minutes

In this exercise, you will style text inside blocks using `em` and `rem`.

1. Open the file named `styling-blocks-with-rem.html` inside the `typography/Exercises` folder.
2. Preview the file in your browser.
3. Modify the styles for the `html` element to set the `font-size` to `10px` and view the results in your browser
4. Change the `font-size` for the `container` to `2em`.
5. Change the `font-size` for the `em-block` to `2em`.
6. Change the `font-size` for the `rem-block` to `2rem`.
7. Change the `font-size` for the `nested-em` block to `1.5em`.
8. Using `rem`, make the `font-size` of the `nested-rem` block the same as the `nested-em-block`.

The resulting page should look like this:



Solution: typography/Solutions/styling-blocks-with-rem.html

```
      -------Lines 1 through 8 Omitted-------
9.    <style>
10.     html {
11.       font-size: 10px;
12.     }
13.
14.     #container {
15.       font-size: 2em; /* 20px */
16.     }
17.
18.     #em-block {
19.       border: 1px solid black;
20.       font-size: 2em; /* 40px */
21.     }
22.
23.     #rem-block {
24.       border: 1px solid black;
25.       font-size: 2rem; /* 20px */
26.     }
27.
28.     #nested-em {
29.       border: 1px dashed black;
30.       font-size: 1.5em; /* 60px */
31.     }
32.
33.     #nested-rem {
34.       border: 1px dashed black;
35.       font-size: 6rem;
36.     }
37.   </style>
      -------Lines 38 through 59 Omitted-------
```


4.3. Styling Text Inside Blocks

Sometimes, rather than applying typographical styles to types of blocks or classes of blocks, you just need to make an adjustment to a particular word, or to the content of a single block. Bootstrap's tools for these local text styling situations are inline elements and text utilities.

❖ 4.3.1. Inline Elements

Inline HTML elements are those elements that style text inside of a block. Except for the different default styles (such as the default `font-face` set by the Reboot stylesheet, inline elements are styled and have the same purposes in Bootstrap as in normal HTML. For example, the `strong` element causes text to be bold, and the `em` element causes text to be italicized.

❖ 4.3.2. Text Utilities

Text utilities are pre-built classes in Bootstrap that make modifying frequently used style properties of text as easy as adding words to an element's `class` attribute.

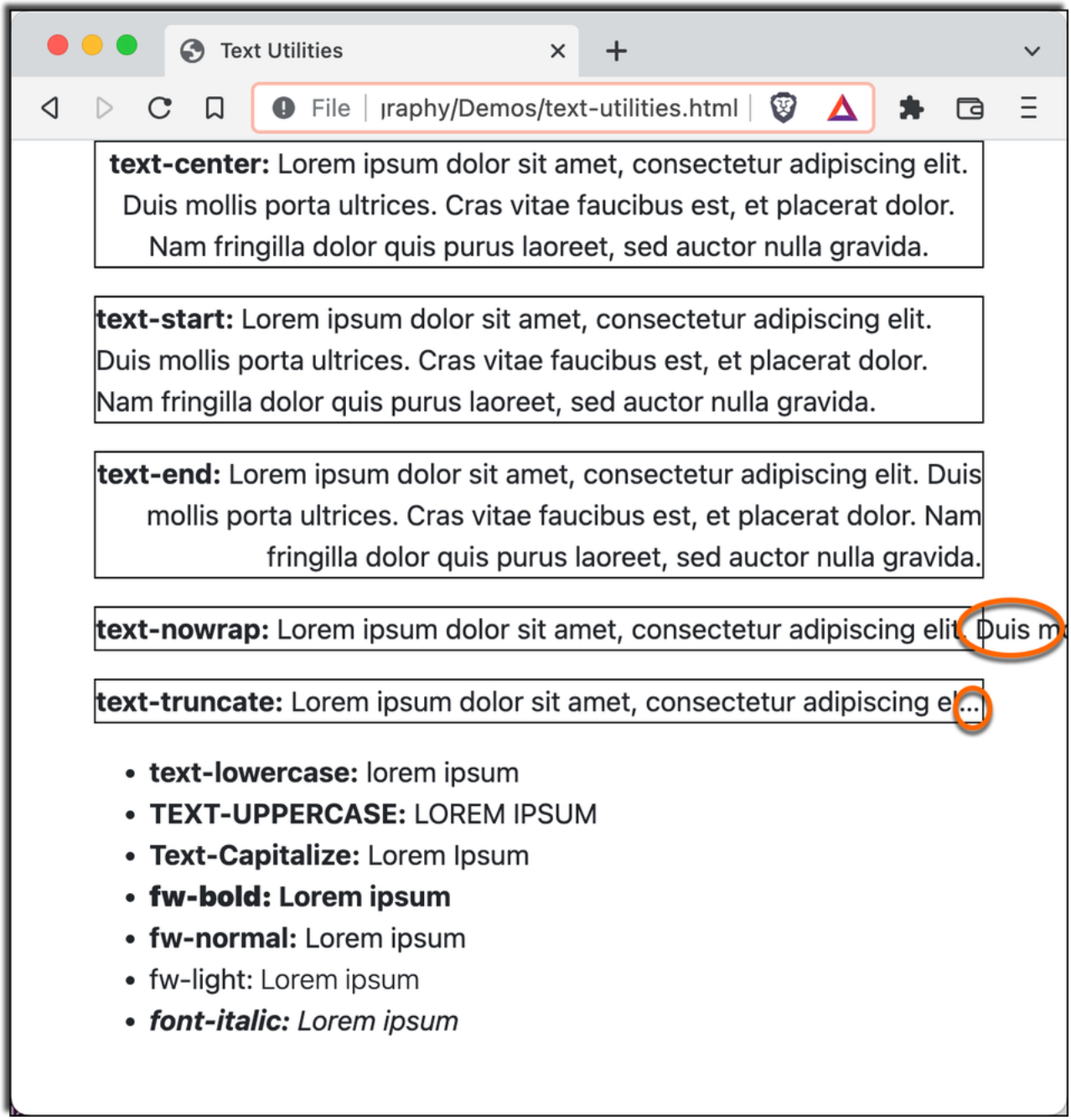
The following demo illustrates the use of several of the text utilities:

Demo 4.7: typography/Demos/text-utilities.html

```
      -------Lines 1 through 16 Omitted-------
17.       <p class="text-center"><strong>text-center:</strong>
18.         Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis
19.         mollis porta ultrices. Cras vitae faucibus est, et placerat
20.         dolor. Nam fringilla dolor quis purus laoreet, sed auctor
21.         nulla gravida.</p>
22.       <p class="text-start"><strong>text-start:</strong>
23.         Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis
24.         mollis porta ultrices. Cras vitae faucibus est, et placerat
25.         dolor. Nam fringilla dolor quis purus laoreet, sed auctor
26.         nulla gravida.</p>
27.       <p class="text-end"><strong>text-end:</strong>
28.         Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis
29.         mollis porta ultrices. Cras vitae faucibus est, et placerat
30.         dolor. Nam fringilla dolor quis purus laoreet, sed auctor
31.         nulla gravida.</p>
32.       <p class="text-nowrap"><strong>text-nowrap:</strong>
33.         Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis
34.         mollis porta ultrices. Cras vitae faucibus est, et placerat
35.         dolor. Nam fringilla dolor quis purus laoreet, sed auctor
36.         nulla gravida.</p>
37.       <p class="text-truncate"><strong>text-truncate:</strong>
38.         Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis
39.         mollis porta ultrices. Cras vitae faucibus est, et placerat
40.         dolor. Nam fringilla dolor quis purus laoreet, sed auctor
41.         nulla gravida.</p>
42.       <ul>
43.         <li class="text-lowercase">
44.           <strong>text-lowercase:</strong> Lorem ipsum
45.         </li>
46.         <li class="text-uppercase">
47.           <strong>text-uppercase:</strong> Lorem ipsum
48.         </li>
49.         <li class="text-capitalize">
50.           <strong>text-capitalize:</strong> Lorem ipsum
51.         </li>
52.         <!-- fw- is the prefix for font-weight -->
53.         <li class="fw-bold">
54.           <strong>fw-bold:</strong> Lorem ipsum
55.         </li>
56.         <li class="fw-normal">
57.           <strong>fw-normal:</strong> Lorem ipsum
58.         </li>
59.         <li class="fw-light">
```

```
60.           <strong>fw-light:</strong> Lorem ipsum
61.         </li>
62.         <!-- fst- is the prefix for font-style -->
63.         <li class="fst-italic">
64.           <strong>font-italic:</strong> Lorem ipsum
65.         </li>
66.       </ul>
      -------Lines 67 through 69 Omitted-------
```


The preceding code will render the following:



These classes are intuitively named. We've circled a couple of interesting things:

1. The `text-nowrap` class will allow contained text to continue out of its container.
2. The `text-truncate` class will truncate the text and add an ellipsis so that it stays within the container without wrapping.

3. `fw-` is the prefix for `font-weight`.
4. `fst-` is the prefix for `font-style`.

For documentation on text utilities, including information on making text utility classes responsive, see `https://getbootstrap.com/docs/5.2/utilities/text/`.

Conclusion

In this lesson, you have learned:

- How Bootstrap resets browser default styles using Reboot.
- How Bootstrap modifies the default styles of elements to create consistent and attractive new defaults.
- The difference between `em` and `rem`.
- How to use text utilities to apply common styles to text.

LESSON 5
Tables

Topics Covered

☑ Responsive tables.

☑ Tables with and without borders.

☑ Styling table headers.

☑ Styling table cells and rows.

Introduction

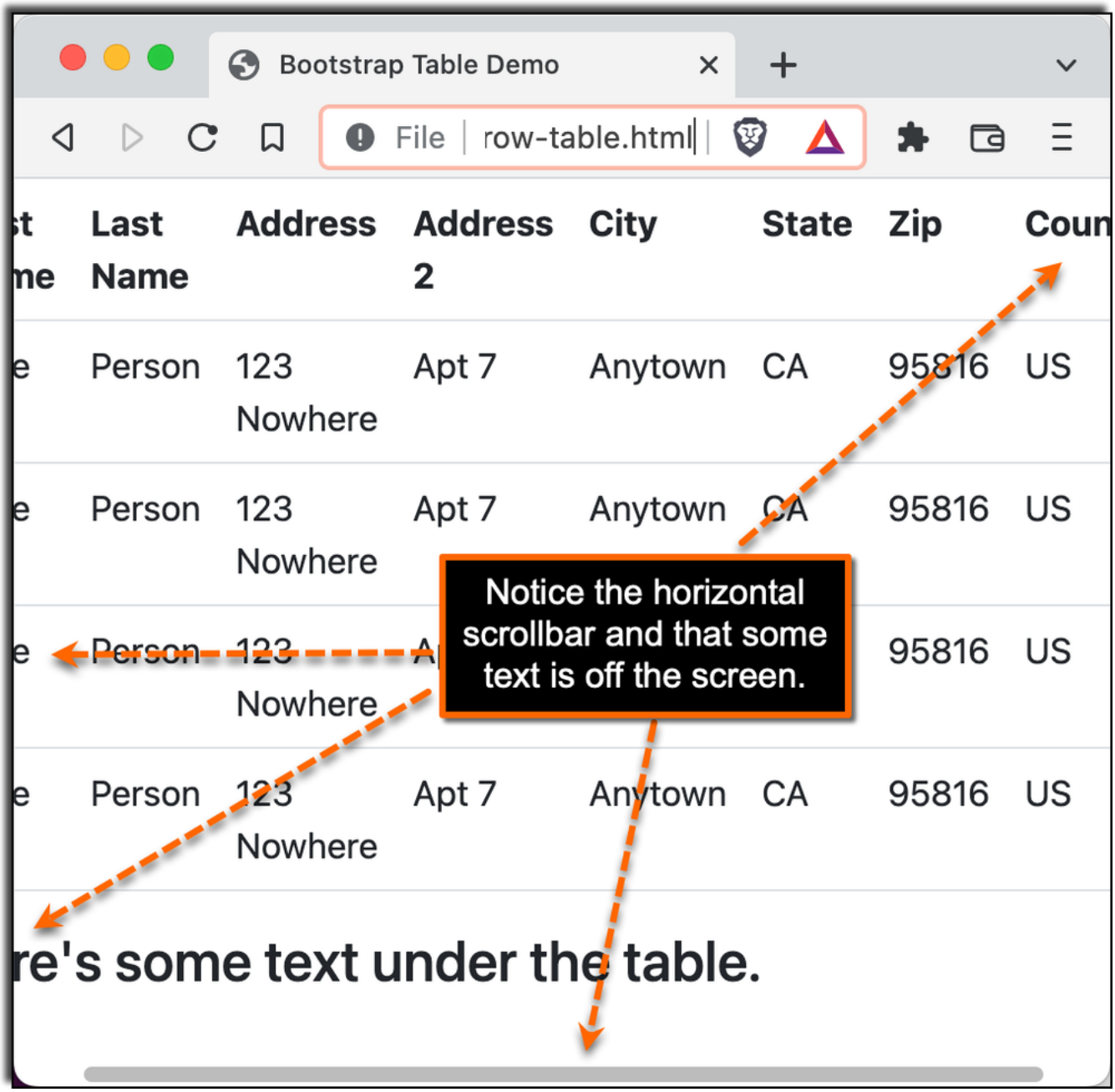
HTML tables can be used to display data in rows and columns. By default, Bootstrap doesn't apply special styles to the HTML `table` element. The reason for this is to avoid clashing with third-party components such as calendars and date pickers that use tables. To apply Bootstrap's table styles to a `table` element, apply the `table` class to it (e.g., `<table class="table">`).

Once you've applied the `table` class, you'll have access to all of Bootstrap's special table styles and behaviors, which we'll explore in this lesson.

5.1. Responsive Tables

By default, when you apply the `table` class to a `table` element, it will cause the table to expand to 100% of the width of its container. As the viewport width gets smaller, the width of the cells in the table will be reduced so that the table fits horizontally. Bootstrap doesn't reduce the size of the text inside the table, so there's only so small that table can get before the user is forced to scroll the browser window horizontally, as shown in the following example:[4]

4. The HTML file used for the narrow table screenshot is at `tables/Demos/narrow-table.html`.



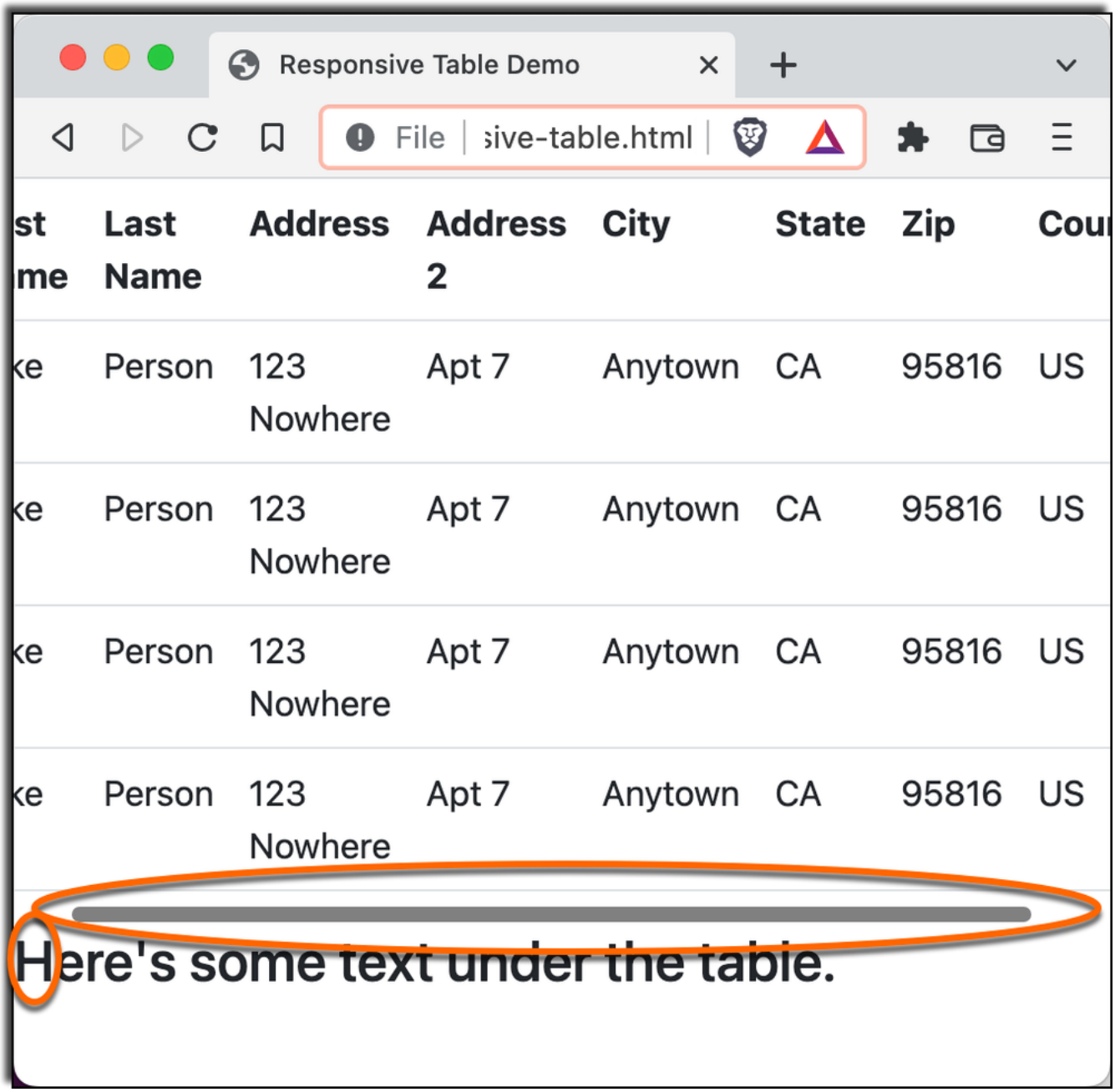
Notice the horizontal scrollbar and that some text is off the screen.

To make wide tables more usable, you can wrap them with the `table-responsive` class to make only the table scroll horizontally, as shown in the following code:

Demo 5.1: tables/Demos/narrow-responsive-table.html

```
     -------Lines 1 through 10 Omitted-------
11.    <div class="table-responsive">
12.    <table class="table">
13.      <tr>
14.        <th>First Name</th>
15.        <th>Last Name</th>
16.        <th>Address</th>
17.        <th>Address 2</th>
18.        <th>City</th>
19.        <th>State</th>
20.        <th>Zip</th>
21.        <th>Country</th>
22.      </tr>
     -------Lines 23 through 52 Omitted-------
53.      <tr>
54.        <td>Fake</td>
55.        <td>Person</td>
56.        <td>123 Nowhere</td>
57.        <td>Apt 7</td>
58.        <td>Anytown</td>
59.        <td>CA</td>
60.        <td>95816</td>
61.        <td>US</td>
62.      </tr>
63.    </table>
     -------Lines 64 through 71 Omitted-------
```


Compare the following image of a responsive table to the previous image:



Notice that the horizontal scrollbar only scrolls the table and that the text below the table does not scroll.

❖ 5.1.1. Breakpoint-specific Responsive Tables

If you only want tables to scroll up to certain breakpoints, you can use breakpoint-specific responsive table classes. For example, if your table works well without scrolling at sizes above `md`, you can use the `table-responsive-md` class on the table's container to make the table only scroll at sizes below `md`.

You can create breakpoint-specific responsive tables by adding any one of the breakpoint class abbreviations (`-sm`, `-md`, `-lg`, `-xl`, `-xxl`) to `table-responsive`.

5.2. Overall Table Styles

You can change the overall style of a table in Bootstrap using the following classes applied to the `table` element:

- `table-dark`: Styles the table with light colored text on a dark background.
- `table-striped`: Styles even and odd-numbered rows differently (also known as zebra-striping).
- `table-hover`: Enables a hover state on rows within a table.
- `table-sm`: Makes tables more compact by reducing cell padding by 50%.
- `table-bordered`: Adds a border to all sides of the table and its cells, rather than the default behavior of Bootstrap tables, which is to add a border between rows.
- `table-borderless`: Removes all borders from a table and its cells.

Note that some of these classes can be combined. For example, you can create a borderless, striped, compact table by combining the `table-borderless`, `table-striped`, and `table-sm` classes.

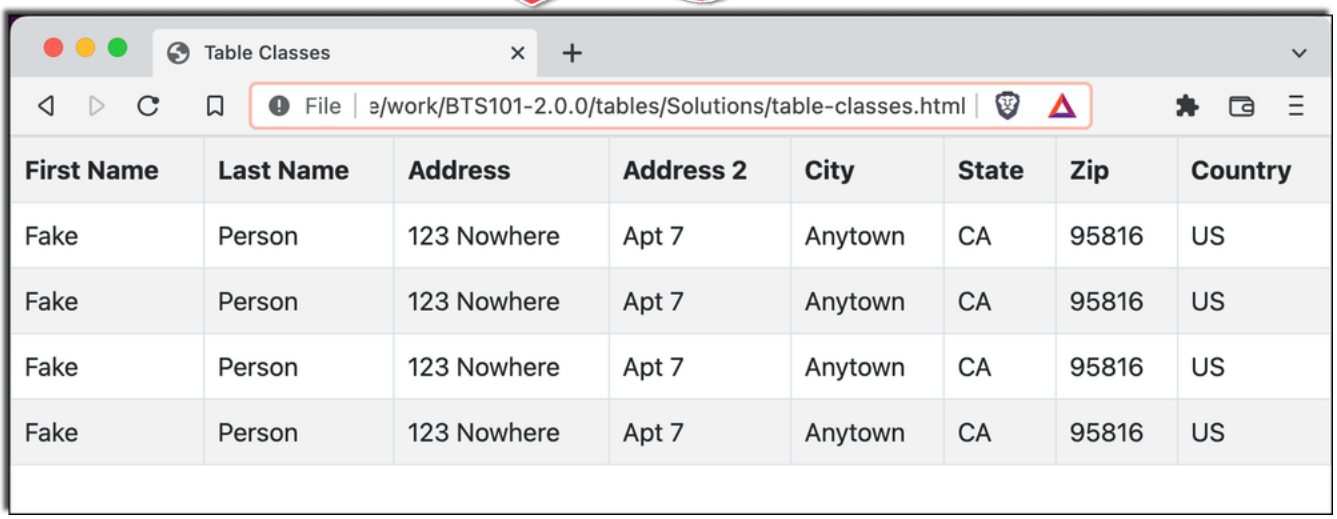
Exercise 7: Styling Tables

15 to 25 minutes

In this exercise, you will experiment with the `table` classes to create different effects on a table.

1. Open the file name `table-classes.html` inside the `tables/Exercises` folder.
2. Open the file in your web browser to preview it, and refresh your browser after each of the changes to verify that you've completed the step correctly.
3. Add the `table` class to style it as a Bootstrap table.
4. Add borders to the table and its cells.
5. Add zebra striping to the table (e.g., every other row is shaded differently).
6. Add a hover state to the rows in the table.
7. Invert the default colors of the table.

The resulting table should look like this:



First Name	Last Name	Address	Address 2	City	State	Zip	Country
Fake	Person	123 Nowhere	Apt 7	Anytown	CA	95816	US
Fake	Person	123 Nowhere	Apt 7	Anytown	CA	95816	US
Fake	Person	123 Nowhere	Apt 7	Anytown	CA	95816	US
Fake	Person	123 Nowhere	Apt 7	Anytown	CA	95816	US

And when you hover over a row, the background color should change slightly.

Solution: tables/Solutions/table-classes.html

```
      -------Lines 1 through 10 Omitted-------
11.   <table class="table table-bordered table-striped table-hover">
      -------Lines 12 through 68 Omitted-------
```

5.3. Table Headers

Headers created using the HTML `thead` element can be styled using the `table-light` or `table-dark` classes to distinguish them visually from the table content.

❖ 5.3.1. The `scope` Attribute

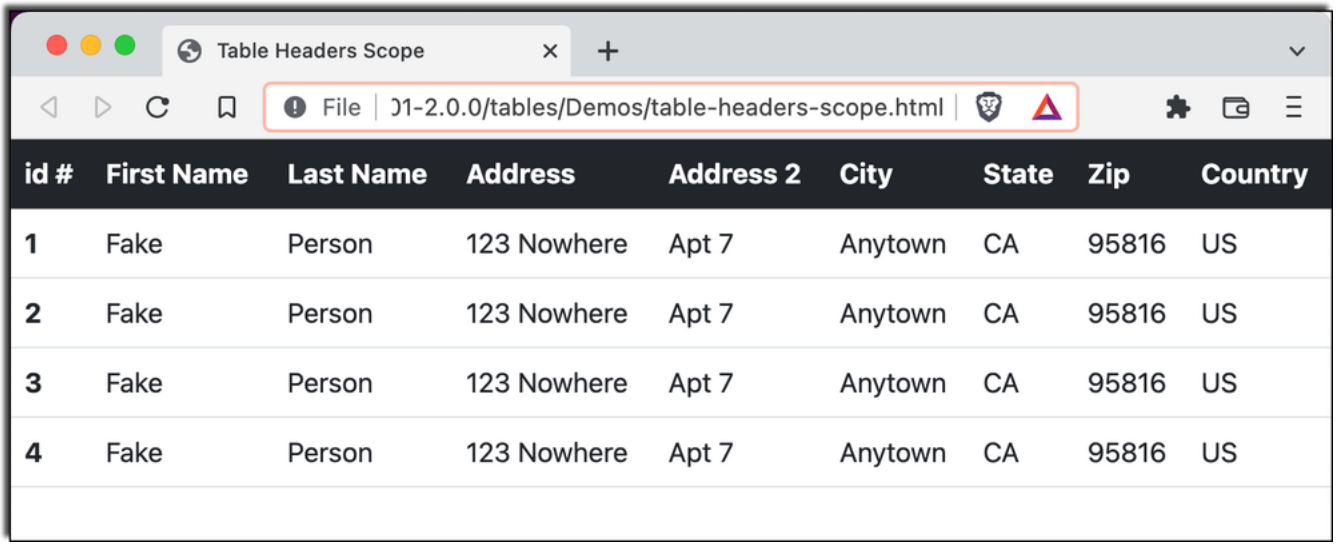
The `scope` attribute allows you to specify whether a table header relates to a column or row. Although it doesn't affect the visual display of a table, it does provide key information for screen readers which allows visually impaired users to read the entire column or row at once, for example.

The following demo shows the use of `table-dark` and the `scope` attribute.

Demo 5.2: tables/Demos/table-headers-scope.html

```
     -------Lines 1 through 10 Omitted-------
11.  <table class="table">
12.    <thead class="table-dark">
13.      <tr>
14.        <th scope="col">id #</th>
15.        <th scope="col">First Name</th>
16.        <th scope="col">Last Name</th>
17.        <th scope="col">Address</th>
18.        <th scope="col">Address 2</th>
19.        <th scope="col">City</th>
20.        <th scope="col">State</th>
21.        <th scope="col">Zip</th>
22.        <th scope="col">Country</th>
23.      </tr>
24.    </thead>
25.    <tbody>
26.      <tr>
27.        <th scope="row">1</th>
28.        <td>Fake</td>
29.        <td>Person</td>
30.        <td>123 Nowhere</td>
31.        <td>Apt 7</td>
32.        <td>Anytown</td>
33.        <td>CA</td>
34.        <td>95816</td>
35.        <td>US</td>
36.      </tr>
     -------Lines 37 through 69 Omitted-------
70.    </tbody>
71.  </table>
     -------Lines 72 through 77 Omitted-------
```


The preceding code will render the following:



id #	First Name	Last Name	Address	Address 2	City	State	Zip	Country
1	Fake	Person	123 Nowhere	Apt 7	Anytown	CA	95816	US
2	Fake	Person	123 Nowhere	Apt 7	Anytown	CA	95816	US
3	Fake	Person	123 Nowhere	Apt 7	Anytown	CA	95816	US
4	Fake	Person	123 Nowhere	Apt 7	Anytown	CA	95816	US

5.4. Contextual Classes

Bootstrap has a built-in system of colors that are used to convey meaning, called **contextual** classes. Using contextual classes, you can style text, backgrounds, and tables using keywords that describe the purpose of the styling. For example, the `danger` keyword causes something to be styled in red, and the `success` keyword causes the element it's applied to to be colored green.

Contextual keywords are used by combining them with the type of data or style to change. For example, to style a warning message in yellow, you could use the `text-warning` class.

Individual table rows or columns can be styled with different background colors using contextual keywords prefaced with `table-`. For example, you can use `table-success`, `table-danger`, `table-info`, or `table-danger`. Contextual classes can be applied to `th`, `tr`, or `td` elements. To see a complete list of contextual classes that can be applied to tables, visit `https://getbootstrap.com/docs/5.2/content/tables/#variants`.

Conclusion

In this lesson, you have learned:

- How to create breakpoint-specific responsive tables.
- How to apply Bootstrap's table classes.

- How to style table headers and use the `scope` attribute.
- How to use contextual classes.

LESSON 6
Bootstrap Forms

Topics Covered

☑ Input types.

☑ Form layout.

☑ Sizing form elements.

☑ The `form-control` class.

Introduction

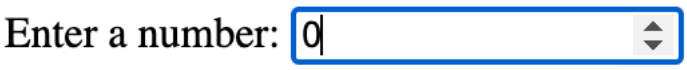
Forms are the most important part of most interactive web applications, since they're the primary way to gather user input. However, HTML forms are also the cause of some of the biggest inconsistencies between how browsers render web pages. Bootstrap solves many of the problems with HTML forms with attractive and functional default styles and classes that can be applied to forms and input elements. In this lesson, you'll learn to use Bootstrap's form styles, layout options, and components.

6.1. Browser Input Inconsistencies

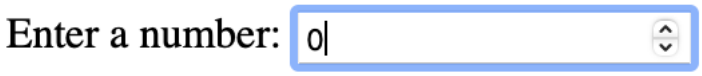
The `type` attribute of the `input` element tells web browsers what type of data you expect the user to enter into a form field. It can also be used by the browser to customize the input to make entering this type of data easier. For example, the following input element has a type of `number`:

```
<input type="number" value="0">
```

Without Bootstrap, this element will be rendered slightly differently in different browsers and on different operating systems. For example, here's how it looks in Chrome on MacOS:



Here's how the same number input renders in Safari on MacOS:



The difference is subtle, but notice that the font family of the input number used by Chrome is different from that used by Safari.

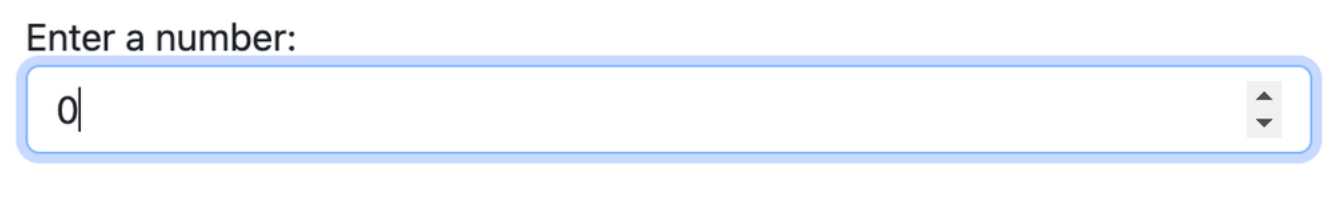
Bootstrap allows you to erase these differences using customizable form styles and components that render the same across browsers and operating systems.

❖ 6.1.1. Form Controls

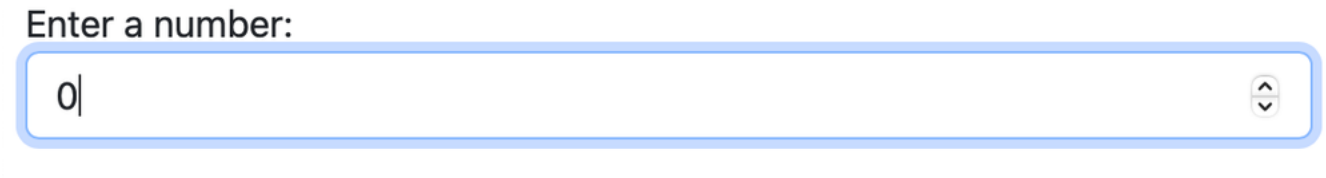
To apply Bootstrap's styles to form inputs, use the `form-control` class. Here's an example of the previous number input with the `form-control` class applied.

```
<input type="number" value="0" class="form-control">
```

With `form-control` applied, the input is styled much more consistently between the two browsers. Here's the styled input in Chrome:



And here it is in Safari:



Notice that both use the same font family.

❖ 6.1.2. Input Types

The `type` attribute of the `input` element supports a wide variety of different types of data, including:

- `button`
- `check box`
- `color`
- `date`
- `datetime-local`
- `email`
- `file`
- `hidden`
- `image`
- `month`
- `number`
- `password`
- `radio`
- `range`
- `reset`
- `search`
- `submit`
- `tel`
- `text`
- `time`
- `url`
- `week`

Not all browsers understand all input types or treat them the same. If you use the input type value of `color`, some browsers will cause a color picker to display, while others will treat this input type the same as `text`. Bootstrap's `form-control` can help with smoothing the differences between how browsers treat different input types in most cases, but you will need to use additional classes for some input types. For example, to use the Bootstrap color picker with an input type of `color` you'll need to use the `form-control-color` class in addition to `form-control`.

6.2. Form Layout

Bootstrap applies `display:block` and `width:100%` to almost all form controls. As a result, form controls in Bootstrap stack horizontally by default. This works well for small screens, where there isn't the horizontal space for more than one form control.

❖ 6.2.1. Form Groups

You can create basic organization inside your form, even when it's simply a vertical stack of controls, by using form groups. Form groups cause Bootstrap to add margins to the bottom of the groups to create some separation between different areas of your form.

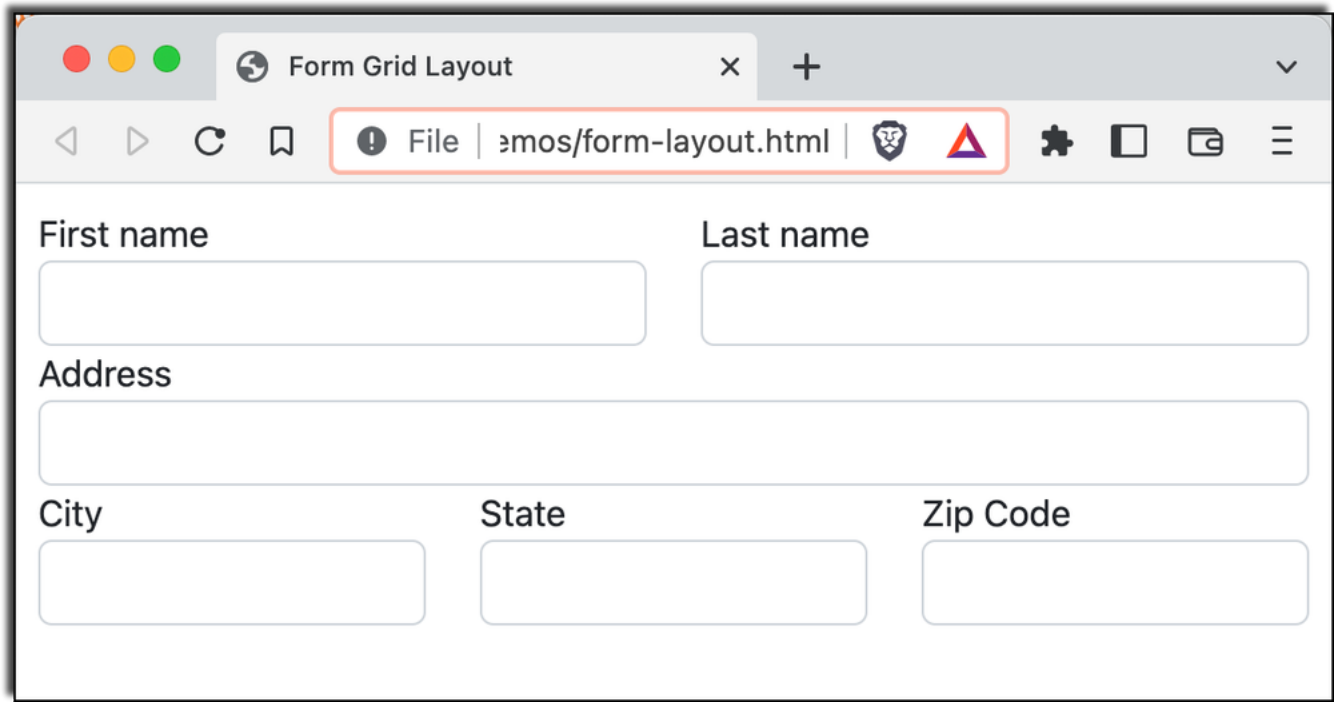
❖ 6.2.2. Grid Layout

On larger devices, however, you may want to change the default layout of Bootstrap forms. You can do this using the same tools that you learned about in the Bootstrap Layout lesson (see page 17), namely by using `row` and `col` classes. In the following example, the `col` class is used without the breakpoint or number of columns to simply indicate that the `div` it is applied to should be a new column.

Demo 6.1: forms/Demos/form-layout.html

```
     -------Lines 1 through 15 Omitted-------
16.  <form>
17.    <div class="row">
18.      <div class="col">
19.        <label for="first-name">First name</label>
20.        <input id="first-name" type="text" class="form-control">
21.      </div>
22.      <div class="col">
23.        <label for="last-name">Last name</label>
24.        <input id="last-name" type="text" class="form-control">
25.      </div>
26.    </div>
27.    <div class="row">
28.      <div class="col">
29.        <label for="address1">Address</label>
30.        <input id="address1" type="text" class="form-control">
31.      </div>
32.    </div>
33.    <div class="row">
34.      <div class="col">
35.        <label for="city">City</label>
36.        <input id="city" type="text" class="form-control">
37.      </div>
38.      <div class="col">
39.        <label for="state">State</label>
40.        <input id="state" type="text" class="form-control">
41.      </div>
42.      <div class="col">
43.        <label for="zip">Zip Code</label>
44.        <input id="zip" type="text" class="form-control">
45.      </div>
46.    </div>
47.  </form>
     -------Lines 48 through 53 Omitted-------
```


The preceding code will render the following:



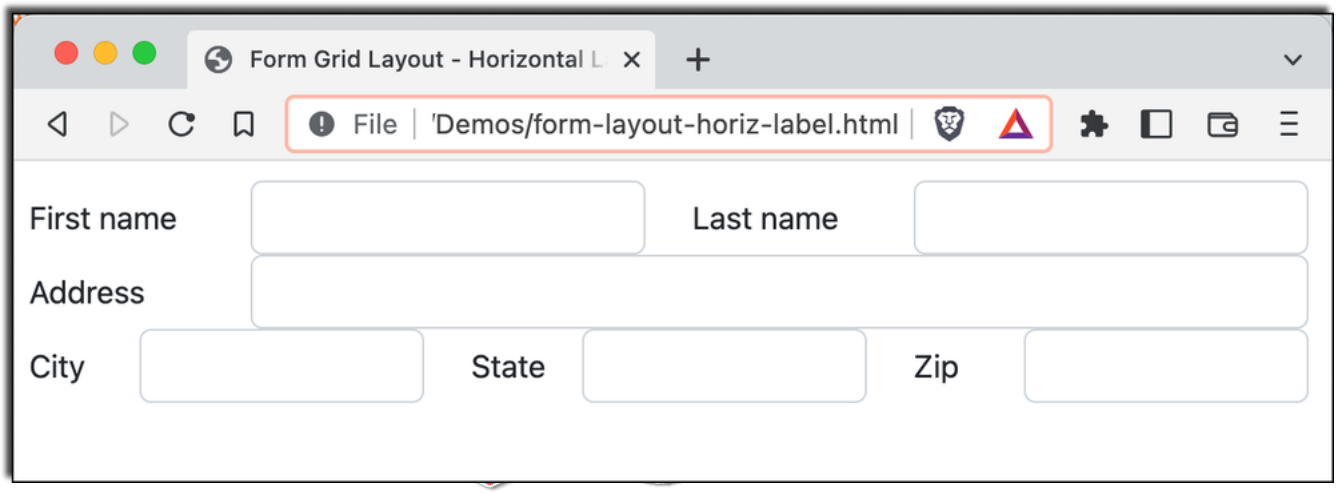
❖ 6.2.3. Horizontal Forms

You can create horizontal forms or align form labels with form inputs using the `col-*-*` classes. When you align form labels horizontally with inputs, add the `col-form-label` class to the label so that it will be vertically centered with the input, as in the following demo:

Demo 6.2: forms/Demos/form-layout-horiz-label.html

```
     -------Lines 1 through 16 Omitted-------
17.  <form>
18.    <div class="row">
19.      <label for="first-name" class="col-sm-2 col-form-label">
20.        First name
21.      </label>
22.      <div class="col-sm-4">
23.        <input id="first-name" type="text" class="form-control">
24.      </div>
25.      <label for="last-name" class="col-sm-2 col-form-label">
26.        Last name
27.      </label>
28.      <div class="col-sm-4">
29.        <input id="last-name" type="text" class="form-control">
30.      </div>
31.    </div>
32.    <div class="row">
33.      <label for="address1" class="col-sm-2 col-form-label">
34.        Address
35.      </label>
36.      <div class="col-sm-10">
37.        <input id="address1" type="text" class="form-control">
38.      </div>
39.    </div>
40.    <div class="row">
41.      <label for="city" class="col-sm-1 col-form-label">City</label>
42.      <div class="col-sm-3">
43.        <input id="city" type="text" class="form-control">
44.      </div>
45.      <label for="state" class="col-sm-1 col-form-label">State</label>
46.      <div class="col-sm-3">
47.        <input id="state" type="text" class="form-control">
48.      </div>
49.      <label for="zip" class="col-sm-1 col-form-label">Zip</label>
50.      <div class="col-sm-3">
51.        <input id="zip" type="text" class="form-control">
52.      </div>
53.    </div>
54.  </form>
     -------Lines 55 through 60 Omitted-------
```

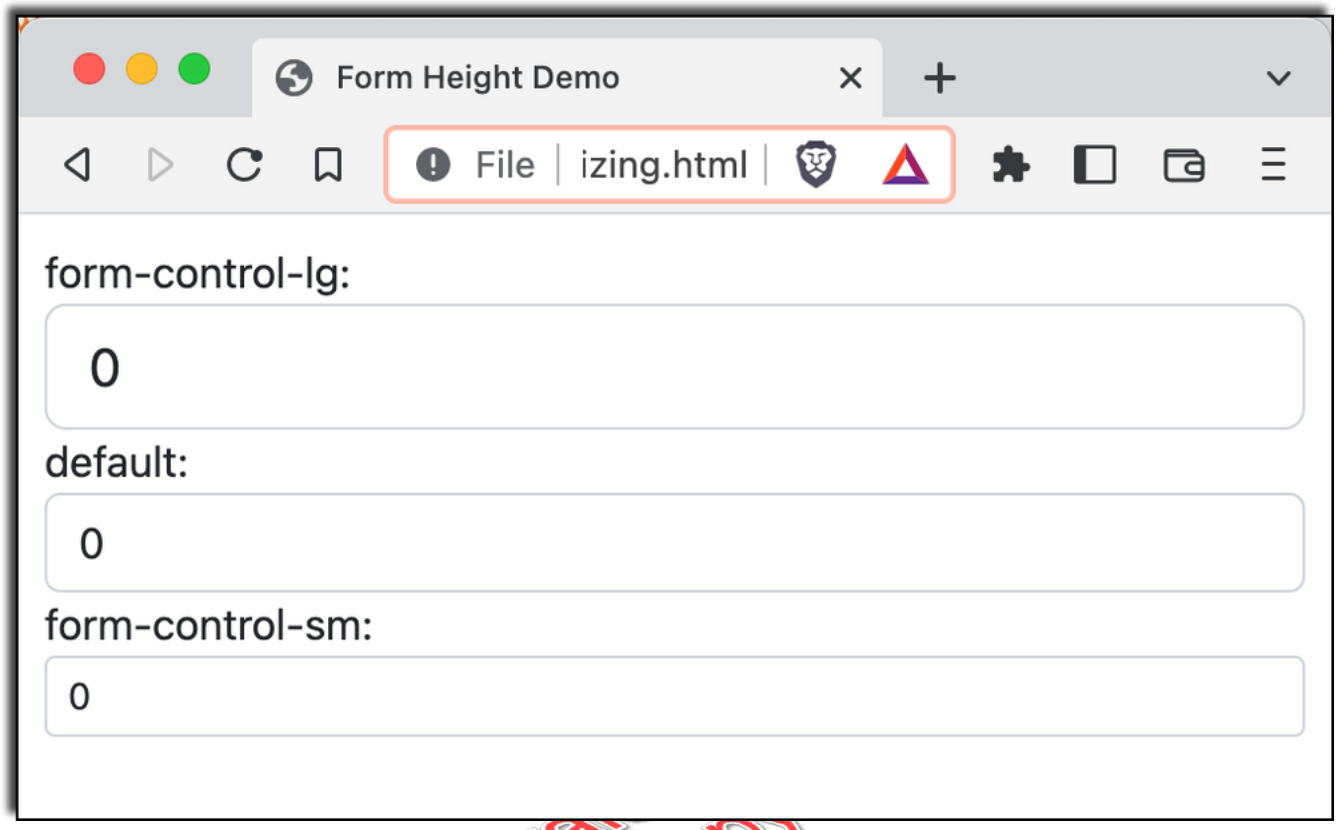

The preceding code will render the following:



Note that you may need to add padding between rows when you align labels and inputs in this way.

❖ 6.2.4. Sizing

Whereas form control widths are controlled by the size of their containers, form control heights can be controlled using two sizing classes: `form-control-lg` and `form-control-sm`. The `lg` height is larger than the default, and the `sm` height is smaller, as shown in the following screenshot:



❖ 6.2.5. Margins

In CSS, you add margins using the following properties:

- `margin-top`
- `margin-right`
- `margin-bottom`
- `margin-left`

You can also use the shorthand property `margin`.

Bootstrap includes classes for adding margin. These classes all begin with a prefix starting with the letter `m` and end with a number between `0` and `5` (or the word `auto`).

The prefixes are:

- `m-` – margin added to all four sides.
- `mt-` – margin added to the top.

- `mb-` – margin added to the bottom.
- `ms-` – margin added to the start, which is the left for scripts read from left to right.
- `me-` – margin added to the end, which is the right for scripts read from left to right.
- `mx-` – margin added to the left and right.
- `my-` – margin added to the top and bottom.

By default, the margin sizes are based on a fraction of `1rem`. Some examples:

- `m-0` – removes all margins.
- `m-1` – adds `.25rem` margin to all four sides.
- `mt-2` – adds `.5rem` margin to the top.
- `me-3` – adds `1rem` margin to the end.
- `mx-4` – adds `1.5rem` margin to the left and right.
- `my-5` – adds `3rem` margin to the top and bottom.

You will learn more about margins and other spacing utilities in the Bootstrap Utilities lesson (see page 143).

Exercise 8: Styling a Form

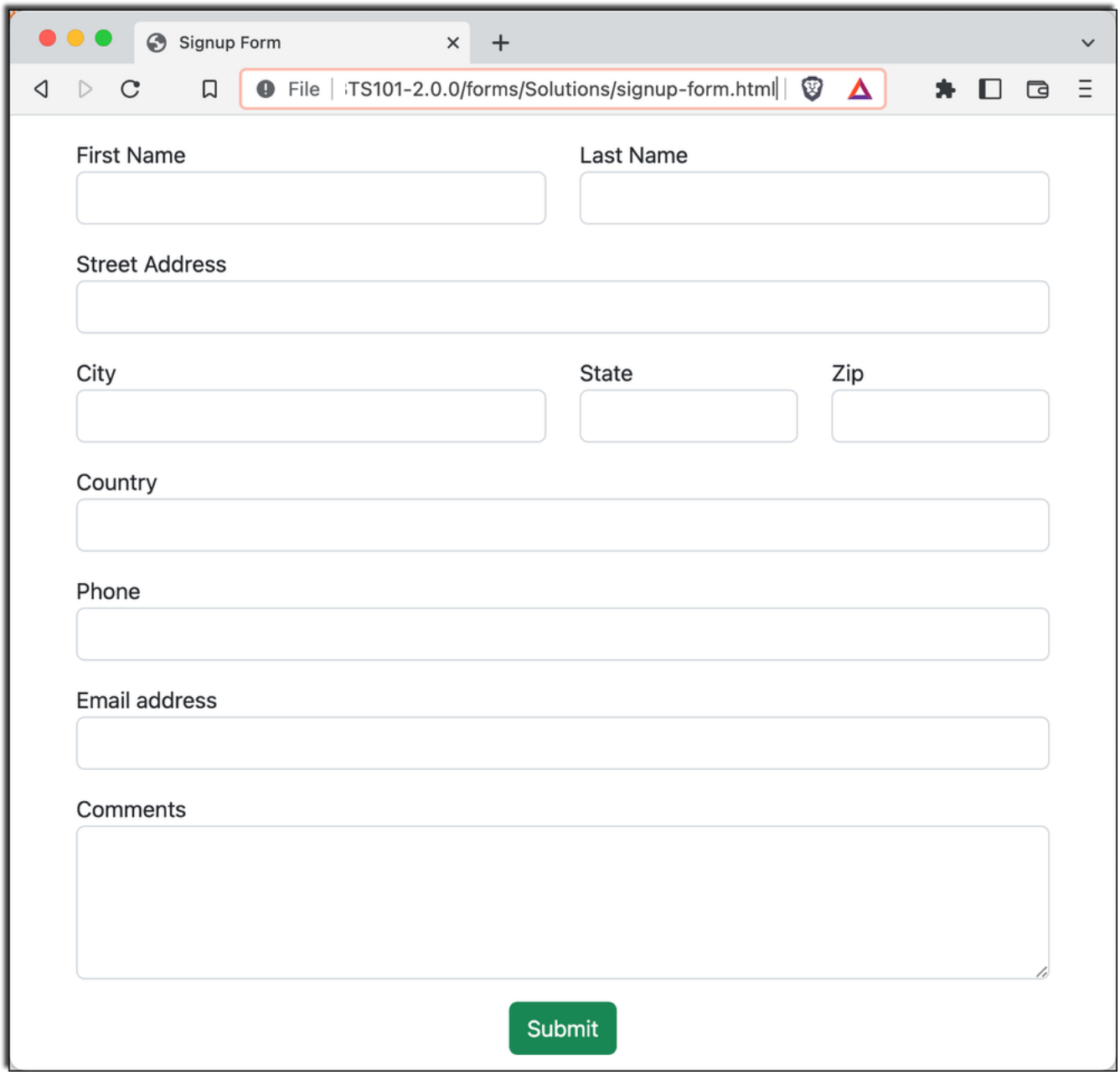
15 to 25 minutes

In this exercise, you will take a plain HTML form and style it with Bootstrap.

1. Open the file named `signup-form.html` in the `forms/Exercises` folder. The form currently contains the following fields and a submit button:

   A. first-name
   B. last-name
   C. street-address
   D. city
   E. state
   F. postal-code
   G. country
   H. phone
   I. email
   J. comments

2. Preview the form in your browser to see how it looks by default. You will use Bootstrap to make it look like this:



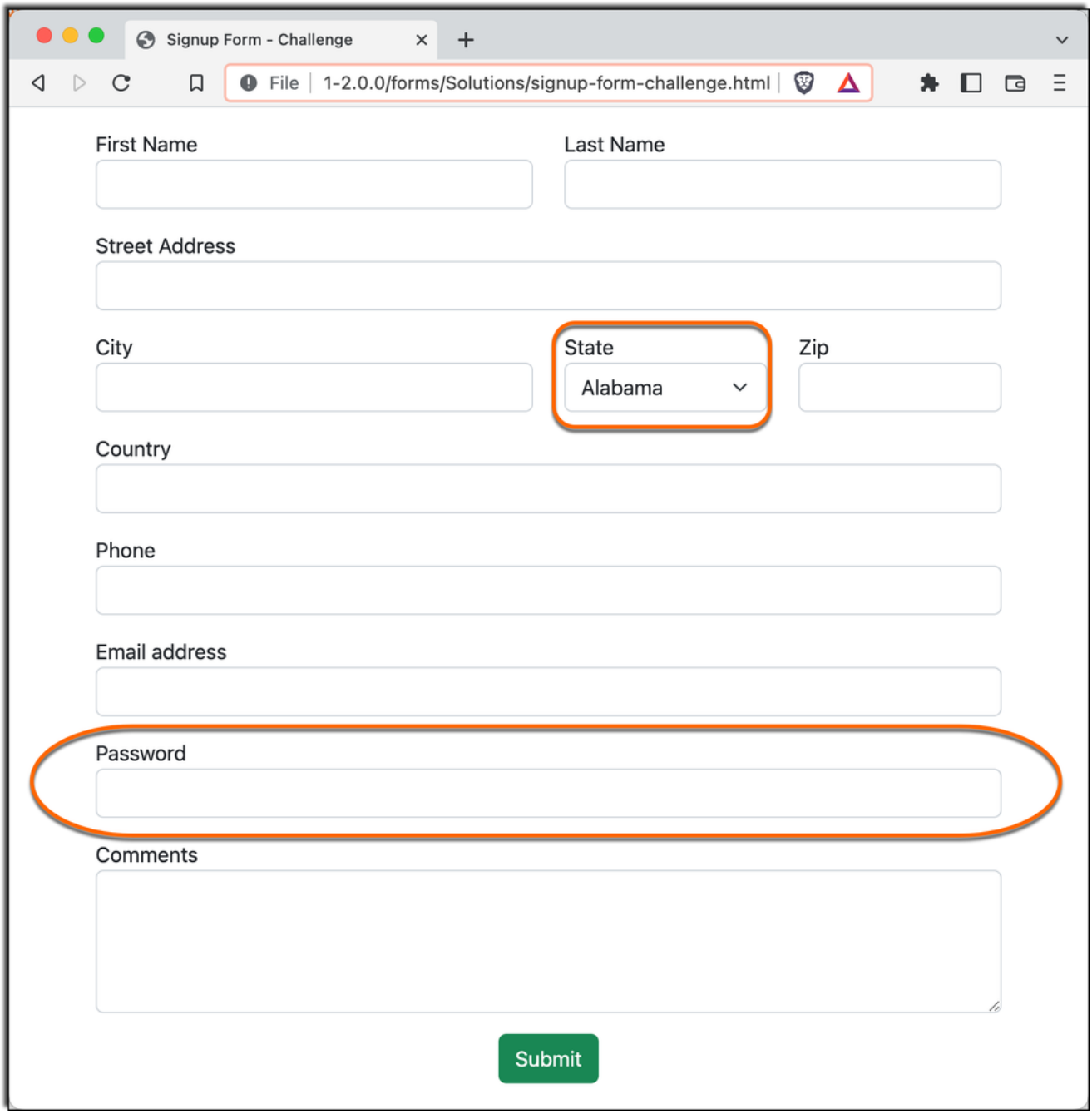
3. Add the Bootstrap CSS and JavaScript to the file.
4. Add the `form-control` class to each of the `input` elements and to the `textarea` element.
5. Add a label to each form control, for example:

```
<label for="first-name">First Name</label>
<input type="text" class="form-control" name="first-name" id="first-name">
```

6. Use grid classes to make First Name and Last Name be on the same line by adding a container with the `row` class around them and adding `col-md-6` to each of their container elements.
7. Make City, State, and Postal Code be on the same line by adding `col-md-6`, `col-md-3`, and `col-md-3`, respectively, to their container elements.
8. Surround each of the remaining form elements with their own row container.
9. Add `col-md-12` to all other form groups so the text area spans all twelve columns.
10. Add the `btn` and `btn-success` classes to the `input` element with a `type` of `submit` to create a green button. You will learn more about buttons in the Bootstrap Components lesson (see page 119).
11. Add the `text-center` class to the button's container element to center your button.
12. Add the `mt-3` class to each element holding a "row" to add vertical spacing to the top of each row.

Challenge

Make the state field a dropdown, and add a password field to the form as shown below:



To add the dropdown, you will use the `form-select` class in place of the `form-control` class:

```
<select class="form-select" id="state">
```

Solution: forms/Solutions/signup-form.html

```
    -------Lines 1 through 10 Omitted-------
11.  <div class="container">
12.    <form>
13.      <div class="mt-3 row">
14.        <div class="col-md-6">
15.          <label for="first-name">First Name</label>
16.          <input type="text" class="form-control" id="first-name">
17.        </div>
18.        <div class="col-md-6">
19.          <label for="last-name">Last Name</label>
20.          <input type="text" class="form-control" id="last-name">
21.        </div>
22.      </div>
23.      <div class="mt-3 row">
24.        <div class="col-md-12">
25.          <label for="street-address">Street Address</label>
26.          <input type="text" class="form-control" id="street-address">
27.        </div>
28.      </div>
29.      <div class="mt-3 row">
30.        <div class="col-md-6">
31.          <label for="city">City</label>
32.          <input type="text" class="form-control" id="city">
33.        </div>
34.        <div class="col-md-3">
35.          <label for="state">State</label>
36.          <input type="text" class="form-control" id="state">
37.        </div>
38.        <div class="col-md-3">
39.          <label for="postal-code">Zip</label>
40.          <input type="text" class="form-control" id="postal-code">
41.        </div>
42.      </div>
43.      <div class="mt-3 row">
44.        <div class="col-md-12">
45.          <label for="country">Country</label>
46.          <input type="text" class="form-control" id="country">
47.        </div>
48.      </div>
49.      <div class="mt-3 row">
50.        <div class="col-md-12">
51.          <label for="phone">Phone</label>
52.          <input type="tel" class="form-control" id="phone">
53.        </div>
```

```
54.       </div>
55.       <div class="mt-3 row">
56.         <div class="col-md-12">
57.           <label for="email">Email address</label>
58.           <input type="email" class="form-control" id="email">
59.         </div>
60.       </div>
61.       <div class="mt-3 row">
62.         <div class="col-md-12">
63.           <label for="comments">Comments</label>
64.           <textarea id="comments" rows="4"
65.             class="form-control"></textarea>
66.         </div>
67.       </div>
68.       <div class="mt-3 row">
69.         <div class="col-md-12 text-center">
70.           <input type="submit" class="btn btn-success" value="Submit">
71.         </div>
72.       </div>
73.     </form>
74.   </div>
      -------Lines 75 through 80 Omitted-------
```

Challenge Solution: forms/Solutions/signup-form-challenge.html

```
      -------Lines 1 through 28 Omitted-------
29.       <div class="mt-3 row">
30.         <div class="col-md-6">
31.           <label for="city">City</label>
32.           <input type="text" class="form-control" id="city">
33.         </div>
34.         <div class="col-md-3">
35.           <label for="state">State</label>
36.           <select class="form-select" id="state">
37.             <option value="AL">Alabama</option>
38.             <option value="AK">Alaska</option>
39.             <option value="AZ">Arizona</option>
40.             <option value="AR">Arkansas</option>
41.             <option value="CA">California</option>
42.             <option value="CO">Colorado</option>
43.             <option value="CT">Connecticut</option>
44.             <option value="DE">Delaware</option>
45.             <option value="DC">District Of Columbia</option>
46.             <option value="FL">Florida</option>
47.             <option value="GA">Georgia</option>
48.             <option value="HI">Hawaii</option>
49.             <option value="ID">Idaho</option>
50.             <option value="IL">Illinois</option>
51.             <option value="IN">Indiana</option>
52.             <option value="IA">Iowa</option>
53.             <option value="KS">Kansas</option>
54.             <option value="KY">Kentucky</option>
55.             <option value="LA">Louisiana</option>
56.             <option value="ME">Maine</option>
57.             <option value="MD">Maryland</option>
58.             <option value="MA">Massachusetts</option>
59.             <option value="MI">Michigan</option>
60.             <option value="MN">Minnesota</option>
61.             <option value="MS">Mississippi</option>
62.             <option value="MO">Missouri</option>
63.             <option value="MT">Montana</option>
64.             <option value="NE">Nebraska</option>
65.             <option value="NV">Nevada</option>
66.             <option value="NH">New Hampshire</option>
67.             <option value="NJ">New Jersey</option>
68.             <option value="NM">New Mexico</option>
69.             <option value="NY">New York</option>
70.             <option value="NC">North Carolina</option>
71.             <option value="ND">North Dakota</option>
```

```
72.             <option value="OH">Ohio</option>
73.             <option value="OK">Oklahoma</option>
74.             <option value="OR">Oregon</option>
75.             <option value="PA">Pennsylvania</option>
76.             <option value="RI">Rhode Island</option>
77.             <option value="SC">South Carolina</option>
78.             <option value="SD">South Dakota</option>
79.             <option value="TN">Tennessee</option>
80.             <option value="TX">Texas</option>
81.             <option value="UT">Utah</option>
82.             <option value="VT">Vermont</option>
83.             <option value="VA">Virginia</option>
84.             <option value="WA">Washington</option>
85.             <option value="WV">West Virginia</option>
86.             <option value="WI">Wisconsin</option>
87.             <option value="WY">Wyoming</option>
88.           </select>
89.         </div>
90.         <div class="col-md-3">
91.           <label for="postal-code">Zip</label>
92.           <input type="text" class="form-control" id="postal-code">
93.         </div>
94.       </div>
      -------Lines 95 through 112 Omitted-------
113.     <div class="mt-3 row">
114.       <div class="col-md-12">
115.         <label for="password">Password</label>
116.         <input type="password" class="form-control" id="password">
117.       </div>
118.     </div>
      -------Lines 119 through 137 Omitted-------
```


Conclusion

In this lesson, you have learned:

- To resize form elements horizontally and vertically.
- To lay out form elements using Bootstrap.
- To assign appropriate input types to form controls.

LESSON 7
Images

Topics Covered

☑ Responsive images.

☑ Aligning images.

☑ Figures.

Introduction

Images are a vital part of many websites, but they can also create problems for mobile and responsive websites if they aren't handled correctly. In this lesson, you'll learn about Bootstrap's classes that you can add to image elements to make them responsive. You will also learn how to apply some other frequently-used image classes.

7.1. Making Images Responsive

Responsive images, like responsive web pages in general, are images that adjust to fit different viewports sizes. The two most common ways to make an image responsive are:

1. Apply CSS styles that scale the image.
2. Use the `picture` element to specify multiple source files for an image.

❖ 7.1.1. Responsive CSS

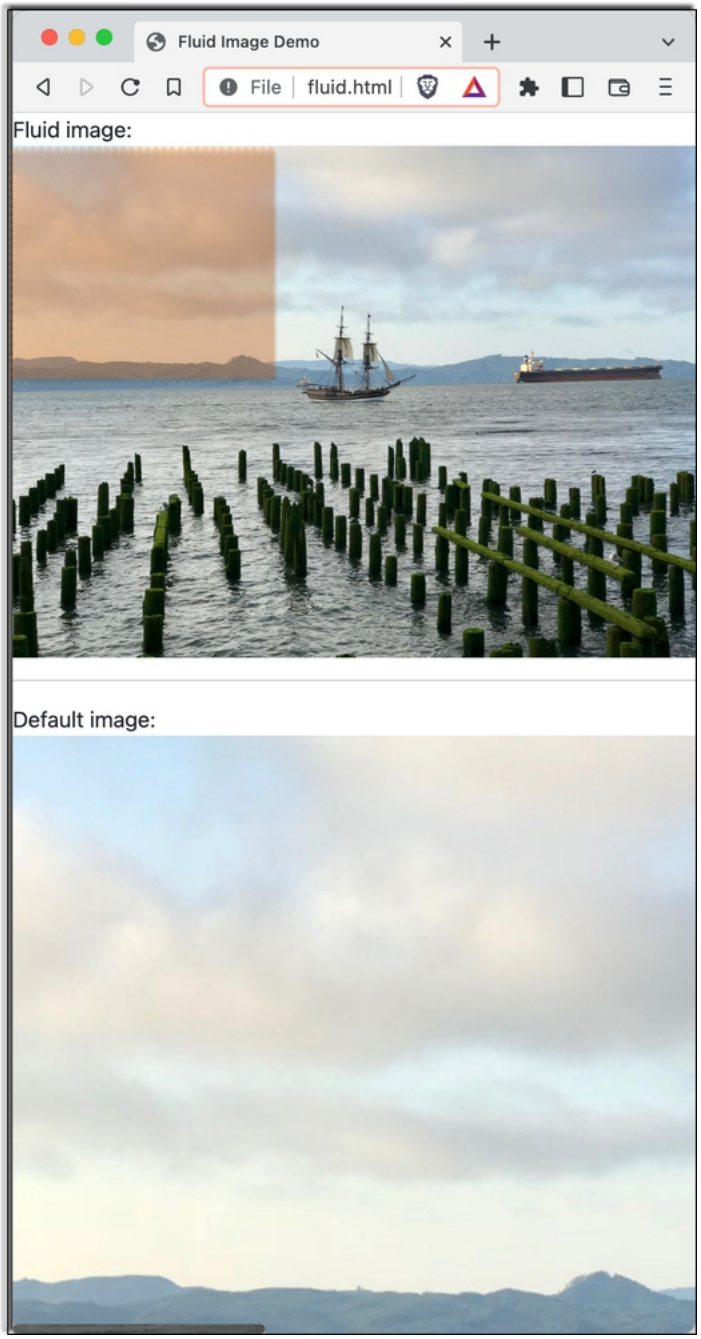
To make images responsive using CSS, set the `max-width` property of the image to `100%` and the `height` property to `auto`. The effect of these changes is to make an image scale to the full width of its container while maintaining its aspect ratio. Bootstrap wraps up both of these changes into the `img-fluid` class.

In the following code, the first image has the `img-fluid` class applied and the second doesn't:

Demo 7.1: images/Demos/img-fluid.html

```
      -------Lines 1 through 10 Omitted-------
11.   Fluid image:
12.   <img src="images/sample-image.jpg" alt="sample fluid image" class="img-fluid">
13.   <hr>
14.   Default image:
15.   <img src="images/sample-image.jpg" alt="sample non-fluid image">
      -------Lines 16 through 21 Omitted-------
```


The preceding code will render the following:



Notice that the second image (without the `img-fluid` class) extends beyond the viewport. To make this clearer, we have highlighted the portion of the top image that is visible in the bottom image.

❖ 7.1.2. Using the Picture Element

The HTML `picture` element allows you to specify multiple images for a single `img` element, along with widths at which each image should be used. Although Bootstrap doesn't make any changes to the behavior of the `picture` element, you can combine use of the `picture` element with the `img-fluid` class to make images scale fluidly between the breakpoints you define inside your `picture` element, as shown in the following example:

Demo 7.2: images/Demos/picture-element.html

```
     -------Lines 1 through 10 Omitted-------
11.  <picture>
12.    <source media="(min-width: 700px)" srcset="images/lg-wht-flower.jpg">
13.    <source media="(min-width: 600px)" srcset="images/md-blu-flower.jpg">
14.    <img src="images/sm-red-flower.jpg" alt="Flowers" class="img-fluid">
15.  </picture>
     -------Lines 16 through 22 Omitted-------
```


Open this file in a browser and make the browser window as narrow as you can. If your browser window is narrower than 600px, it will show the `sm-red-flower.jpg` in the `img` tag. Slowly expand the width of the viewport and notice that the image scales until it gets to 700px wide, at which point it changes to the second source image (`md-blu-flower.jpg`). Continue to expand the viewport to see it change again to to the largest image (`lg-wht-flower.jpg`).

7.2. Aligning Images

Bootstrap's text alignment classes and float classes can be used to align images.

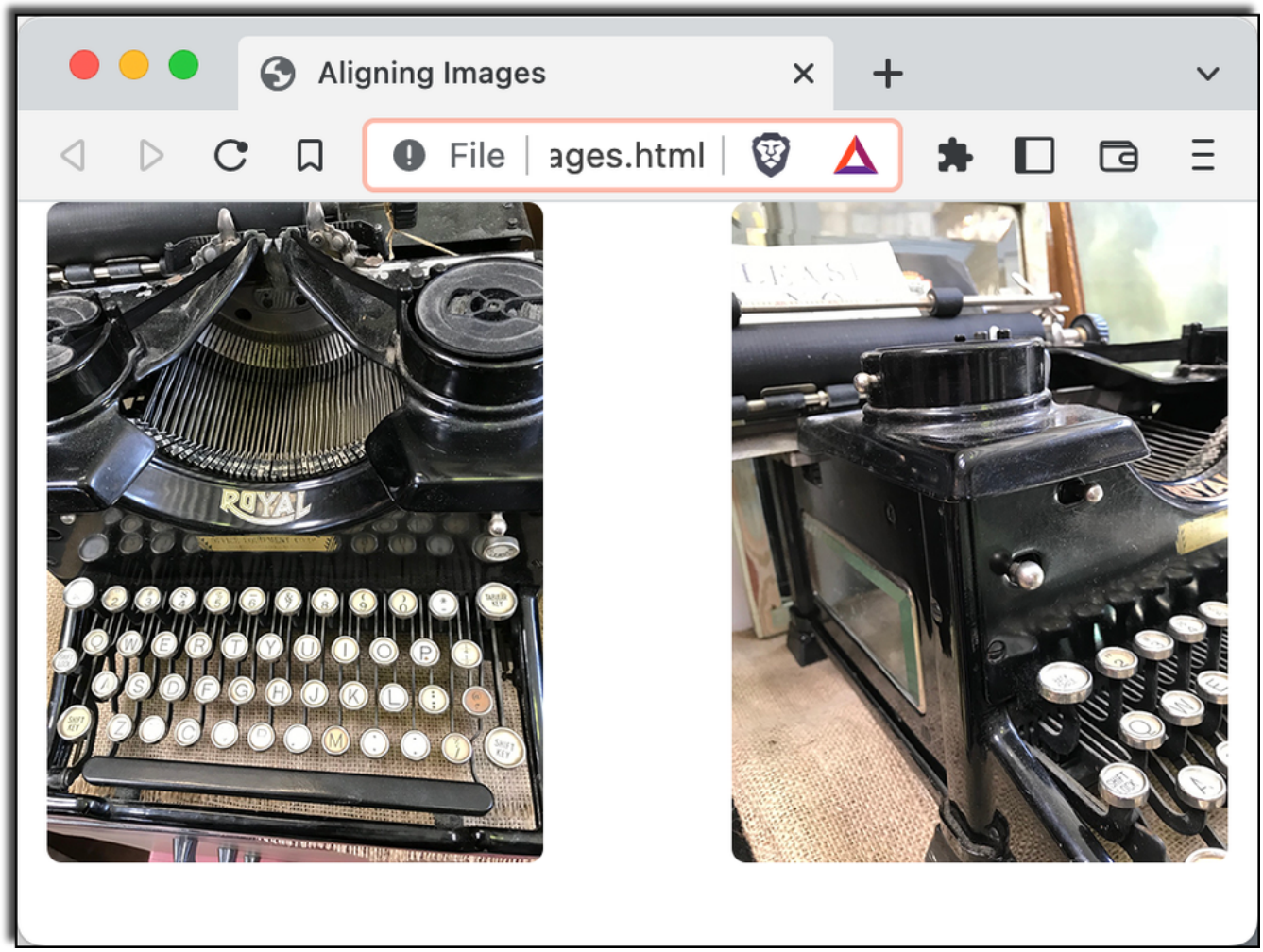
❖ 7.2.1. Float Classes

Bootstrap has three float classes: `float-start`, `float-end`, and `float-none`. When applied to images, they can be used to align images horizontally, as shown in this example:

Demo 7.3: images/Demos/align-images.html

```
     -------Lines 1 through 15 Omitted-------
16.  <div class="container">
17.    <div class="row">
18.      <div class="col-sm-12">
19.      <img src="images/typewriter-1.jpg" alt="typewriter #1" class="rounded float-
               start">
20.      <img src="images/typewriter-2.jpg" alt="typewriter #2" class="rounded float-
               end">
21.    </div>
22.  </div>
23.  </div>
     -------Lines 24 through 29 Omitted-------
```


The preceding code will render the following:



Note that, in addition to floating the images, we have applied the `rounded` class to apply rounded corners to the images.

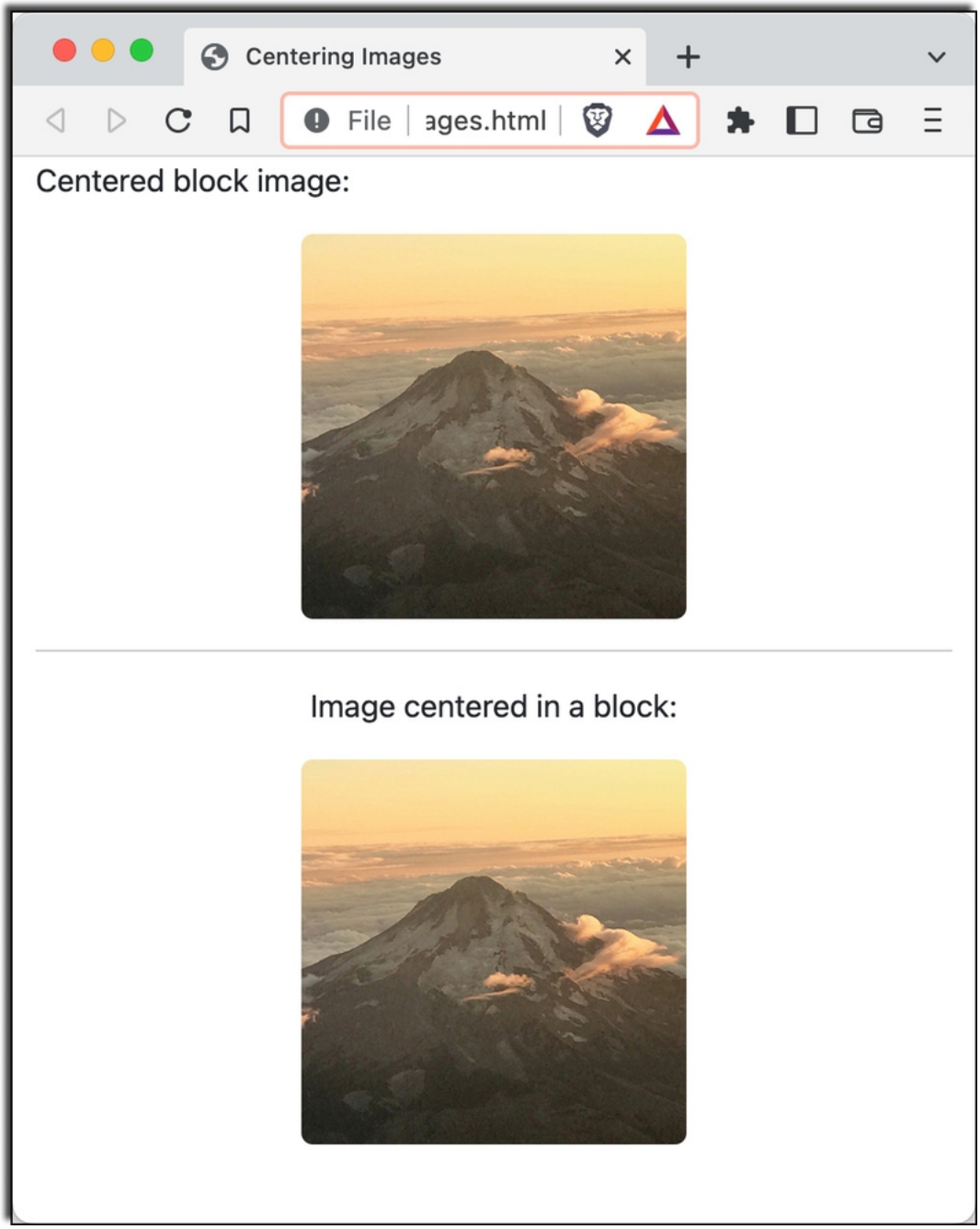
❖ 7.2.2. Centering Images

You can center images by either wrapping them in an element with the `text-center` class applied or by changing the `img` into a block element (by applying the `d-block` class) and then applying automatically-sized left and right margins to it (by applying the `mx-auto` class) as shown in the following demo:

Demo 7.4: images/Demos/centering-images.html

```
     -------Lines 1 through 15 Omitted-------
16.  <div class="container">
17.    <p>Centered block image:</p>
18.    <img src="images/mt-hood.jpg" alt="mt hood" class="rounded mx-auto d-block">
19.    <hr>
20.    <div class="text-center">
21.      <p>Image centered in a block:</p>
22.      <img src="images/mt-hood.jpg" alt="mt hood" class="rounded">
23.    </div>
24.  </div>
     -------Lines 25 through 30 Omitted-------
```


The preceding code will render the following:



Utility Classes

The `mx-auto` and `d-block` classes used in the previous demo are examples of Bootstrap utility classes. These are shorthand ways to modify style properties through classes.

The `mx-auto` class specifies that the margins (`m`) on the x-axis (`x`) should be set to auto (`auto`). You can read more about the `m-` utility classes at `https://getbootstrap.com/docs/5.2/utilities/spacing/`.

The `d-block` class is from the display utilities, which gives you a quick way to set the `display` property of an element. You can read more about the `d-` utility classes at `https://getbootstrap.com/docs/5.2/utilities/display/`.

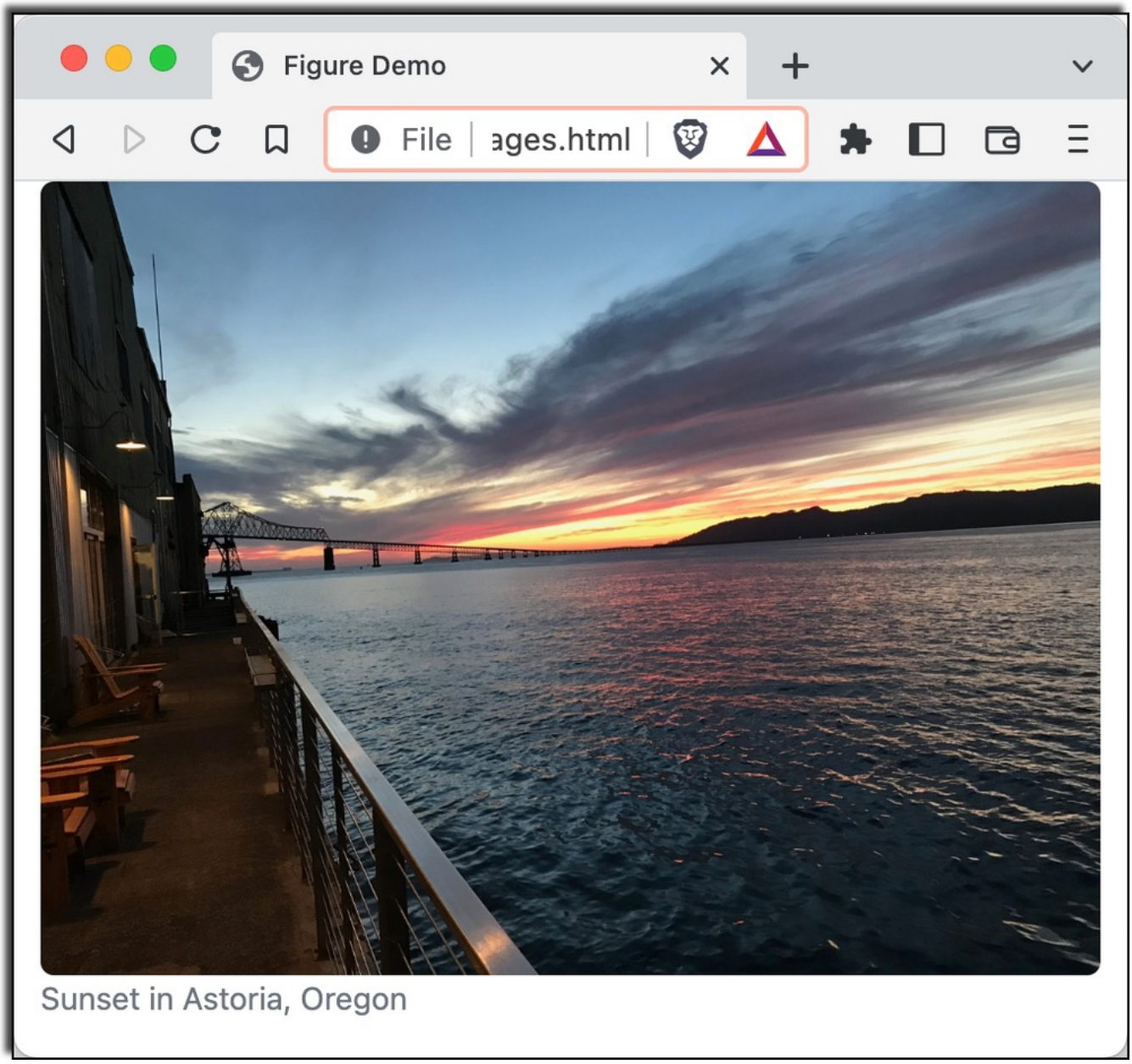
7.3. Figures

The HTML `figure` element is useful for displaying content along with a caption. It's frequently used to display captioned images or video clips. Bootstrap's `figure` and `figure-caption` classes apply attractive default styles to figures.

Demo 7.5: images/Demos/figure-images.html

```
     -------Lines 1 through 10 Omitted-------
11.  <div class="container">
12.    <div class="row">
13.      <div class="col-sm-12">
14.        <figure class="figure">
15.          <img src="images/sunset.jpg" alt="sunset in Astoria" class="rounded img-
               fluid">
16.          <figcaption class="figure-caption">Sunset in Astoria, Oregon</figcaption>
17.        </figure>
18.      </div>
19.    </div>
20.  </div>
     -------Lines 21 through 26 Omitted-------
```

The preceding code will render the following:



Exercise 9: Create a Page with Images

15 to 25 minutes

This is an open-ended exercise. Use what you've learned in this lesson (and elsewhere in the course) to create a new responsive web page containing images.

1. Create a new HTML page from scratch in the `images/Exercises` folder.
2. Add Bootstrap to the page.
3. Add several images styled with Bootstrap classes. You can use your own images, use the ones in the `images/Exercises/images` folder, or download some from the web.

Conclusion

In this lesson, you have learned:

- To use `img-fluid` and the `picture` element to create responsive images.
- To align images using helper and text alignment classes.
- To style figures with Bootstrap classes.

LESSON 8
Bootstrap Components

Topics Covered

☑ Buttons.

☑ Carousels.

☑ Alerts.

☑ Toggling content visibility.

☑ Modal windows.

☑ Other Bootstrap components.

Introduction

Components in Bootstrap are reusable functionality that can be customized and controlled using JavaScript and CSS. You've already seen and worked with several components, including `Nav`, `Navbar`, and `Dropdowns`. In this lesson, you'll learn about several other components.

8.1. Styling Buttons

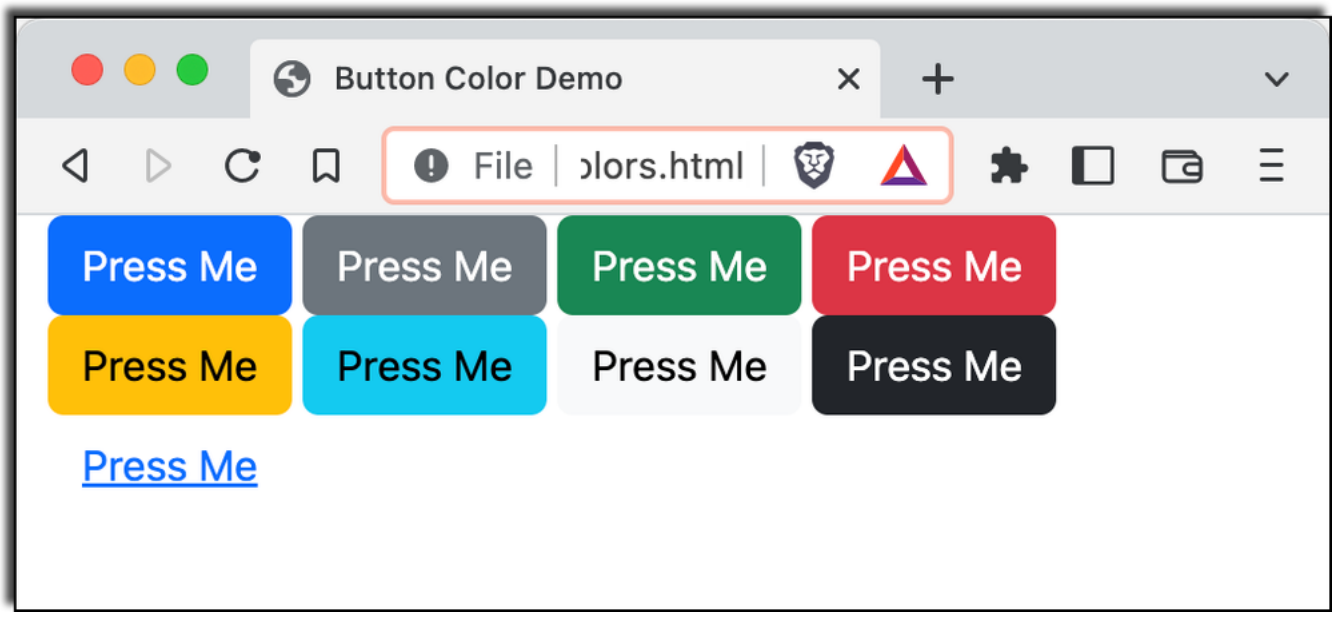
Bootstrap's button classes allow you to style plain HTML buttons and links in multiple sizes, colors, and states. To create a Bootstrap button, apply the `btn` class to a `button`, an `input` (with a type value of `button`, `submit`, or `reset`), or an `a` element.

Applying the `btn` class will style an element as a rounded button, but the button color and button text color will be unset. To specify the button color, use a contextual button class in addition to the `btn` class, as shown in the following demo:

Demo 8.1: components/Demos/button-colors.html

```
      -------Lines 1 through 10 Omitted-------
11.   <div class="container">
12.       <button class="btn btn-primary">Press Me</button>
13.       <button class="btn btn-secondary">Press Me</button>
14.       <button class="btn btn-success">Press Me</button>
15.       <button class="btn btn-danger">Press Me</button>
16.       <button class="btn btn-warning">Press Me</button>
17.       <button class="btn btn-info">Press Me</button>
18.       <button class="btn btn-light">Press Me</button>
19.       <button class="btn btn-dark">Press Me</button>
20.       <button class="btn btn-link">Press Me</button>
21.   </div>
      -------Lines 22 through 27 Omitted-------
```

The preceding code will render the following:



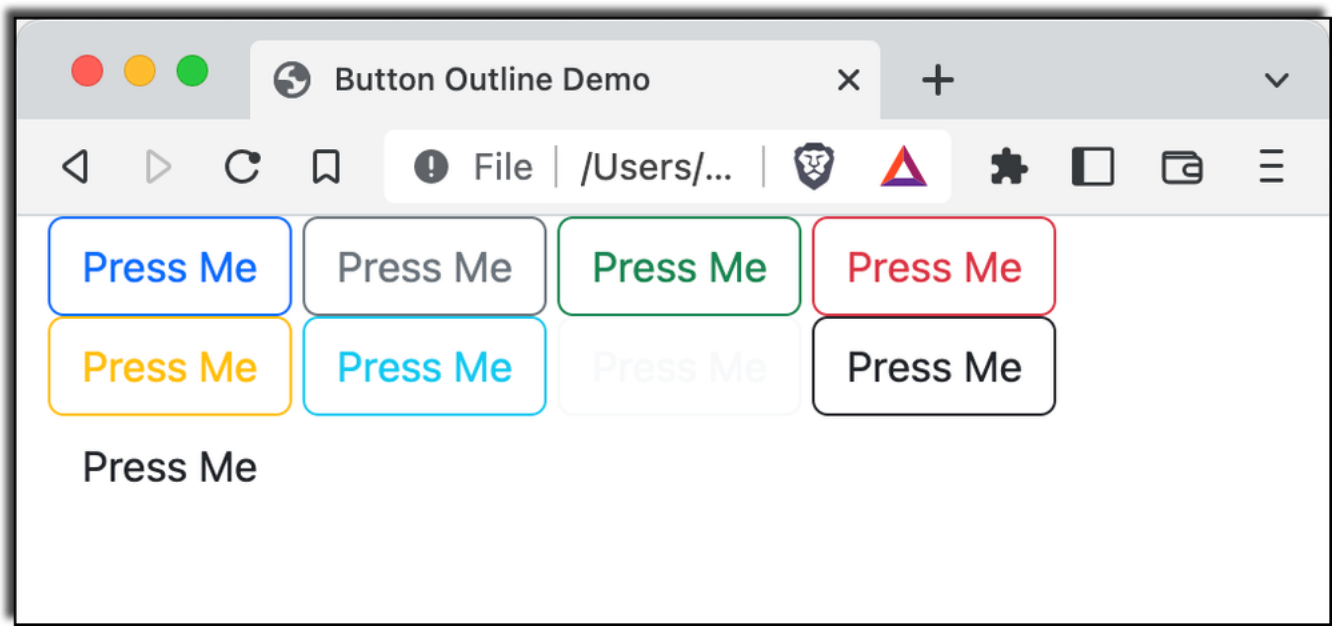
❖ 8.1.1. Outline Buttons

To create a button with just an outline, rather than a background color, use `btn-outline-*` contextual classes instead of `btn-*` classes, as shown in this demo:

Demo 8.2: components/Demos/button-outline.html

```
      -------Lines 1 through 10 Omitted-------
11.   <div class="container">
12.       <button class="btn btn-outline-primary">Press Me</button>
13.       <button class="btn btn-outline-secondary">Press Me</button>
14.       <button class="btn btn-outline-success">Press Me</button>
15.       <button class="btn btn-outline-danger">Press Me</button>
16.       <button class="btn btn-outline-warning">Press Me</button>
17.       <button class="btn btn-outline-info">Press Me</button>
18.       <button class="btn btn-outline-light">Press Me</button>
19.       <button class="btn btn-outline-dark">Press Me</button>
20.       <button class="btn btn-outline-link">Press Me</button>
21.   </div>
      -------Lines 22 through 27 Omitted-------
```

The preceding code will render the following:



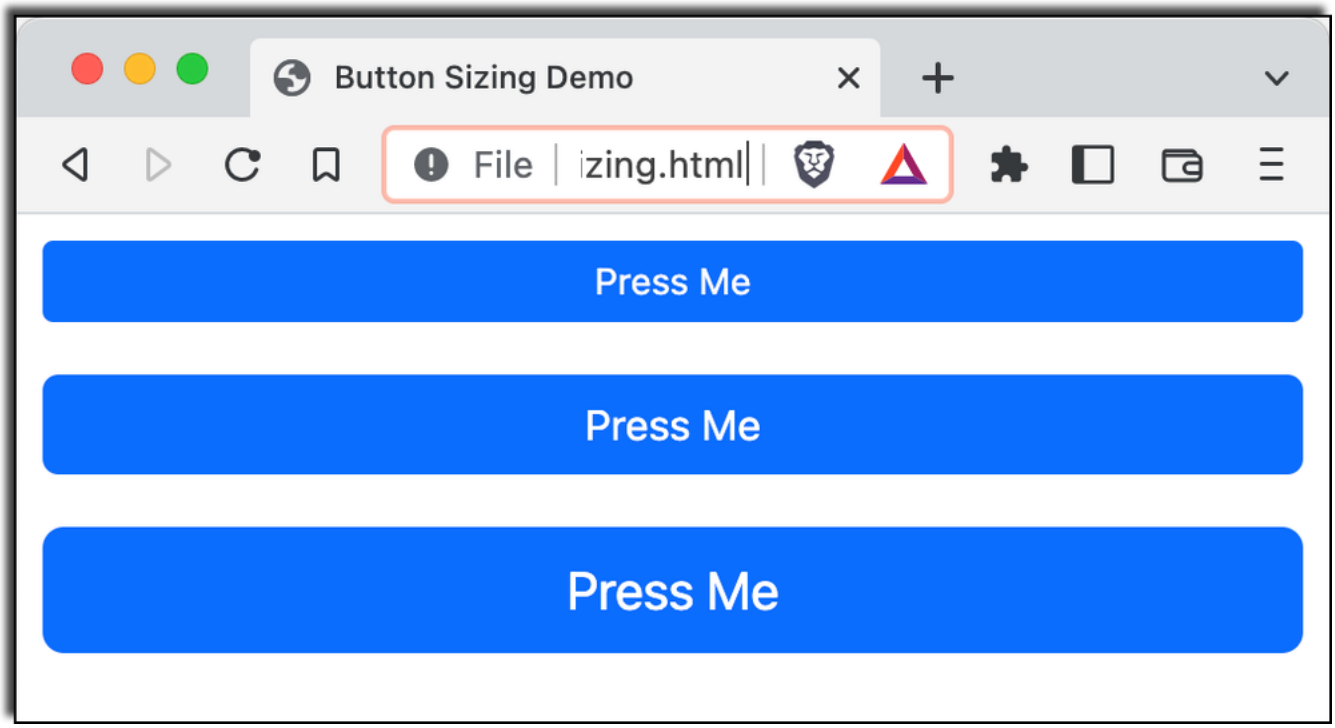
❖ 8.1.2. Button Sizes

As with form controls, buttons can also have sizing classes (`-sm`. and `-lg`) applied to them:

Demo 8.3: components/Demos/button-sizing.html

```
      -------Lines 1 through 15 Omitted-------
16.   <div class="container">
17.     <div class="row">
18.       <button class="btn btn-primary btn-sm">Press Me</button>
19.     </div>
20.     <div class="row">
21.       <button class="btn btn-primary">Press Me</button>
22.     </div>
23.     <div class="row">
24.       <button class="btn btn-primary btn-lg">Press Me</button>
25.     </div>
26.   </div>
      -------Lines 27 through 32 Omitted-------
```

The preceding code will render the following:



Extending Bootstrap Classes

Open `components/Demos/button-sizing.html` in your editor and you'll see that we added this rule to add padding to the `row` class:

```
<style>
  .row {
    padding: 10px;
  }
</style>
```

You can use this method to extend Bootstrap classes. You can also override properties of a Bootstrap class, but to do so, you must use a selector that is at least as specific as the selector used in the Bootstrap code.

8.2. Carousel

Carousels are a popular (and, some would argue, overused) technique to automatically rotate through content on a web page. In the following exercise, you'll use Bootstrap's Carousel component to create a Carousel containing images and labels.

Exercise 10: Make a Carousel

15 to 25 minutes

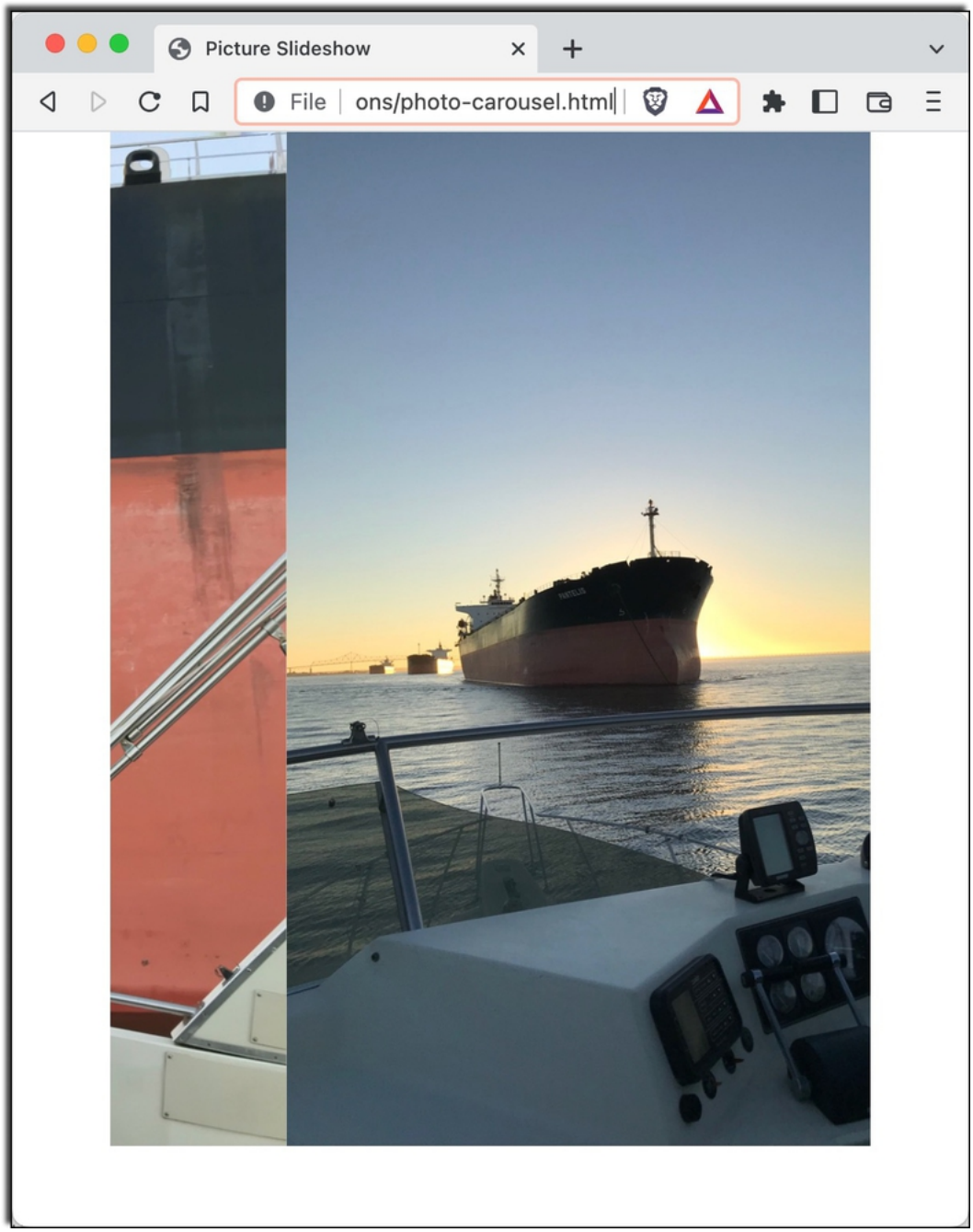
In this exercise, you will build a slideshow of images by using Bootstrap's Carousel component.

1. Open `photo-carousel.html` from the `components/Exercises` folder.
2. Create a `div` inside the **container** `div` and give it the `carousel` and `slide` classes.
3. Add a `data-bs-ride` attribute with a value of `carousel` and a unique `id` attribute to the carousel `div`. Your main **carousel** `div` should now look like this:

```
<div id="my-photo-slideshow" class="carousel slide" data-bs-ride="carousel">

</div>
```


4. Inside the main **carousel** `div`, create a `div` with a class value of `carousel-inner`.
5. Inside the **carousel-inner** `div`, create three `div`s with class values of `carousel-item`.
6. Give the first `carousel-item` an additional class of `active` to make it show up when the slideshow first loads. Each of the `carousel-item` elements will be a separate slide in the carousel. The `data-bs-ride` attribute in the `carousel` component is what will tell the Bootstrap JavaScript to transition between them. The next step is to put images into the `carousel-item` elements.
7. Add an image to each of the `carousel-item` elements. You can use your own images or the images provided in the `components/Exercises/images` folder.
8. Add the `d-block` class to each image to set the CSS `display` property to `block`, and add the `w-100` class to each image to set the CSS `width` property to `100%` (of the containing `div`).
9. Open `photo-carousel.html` in your browser to see how it works. The page should initially load the first image and then transition through the images, starting at the beginning once it has reached the last image. The screenshot below shows it mid-transition between two slides:

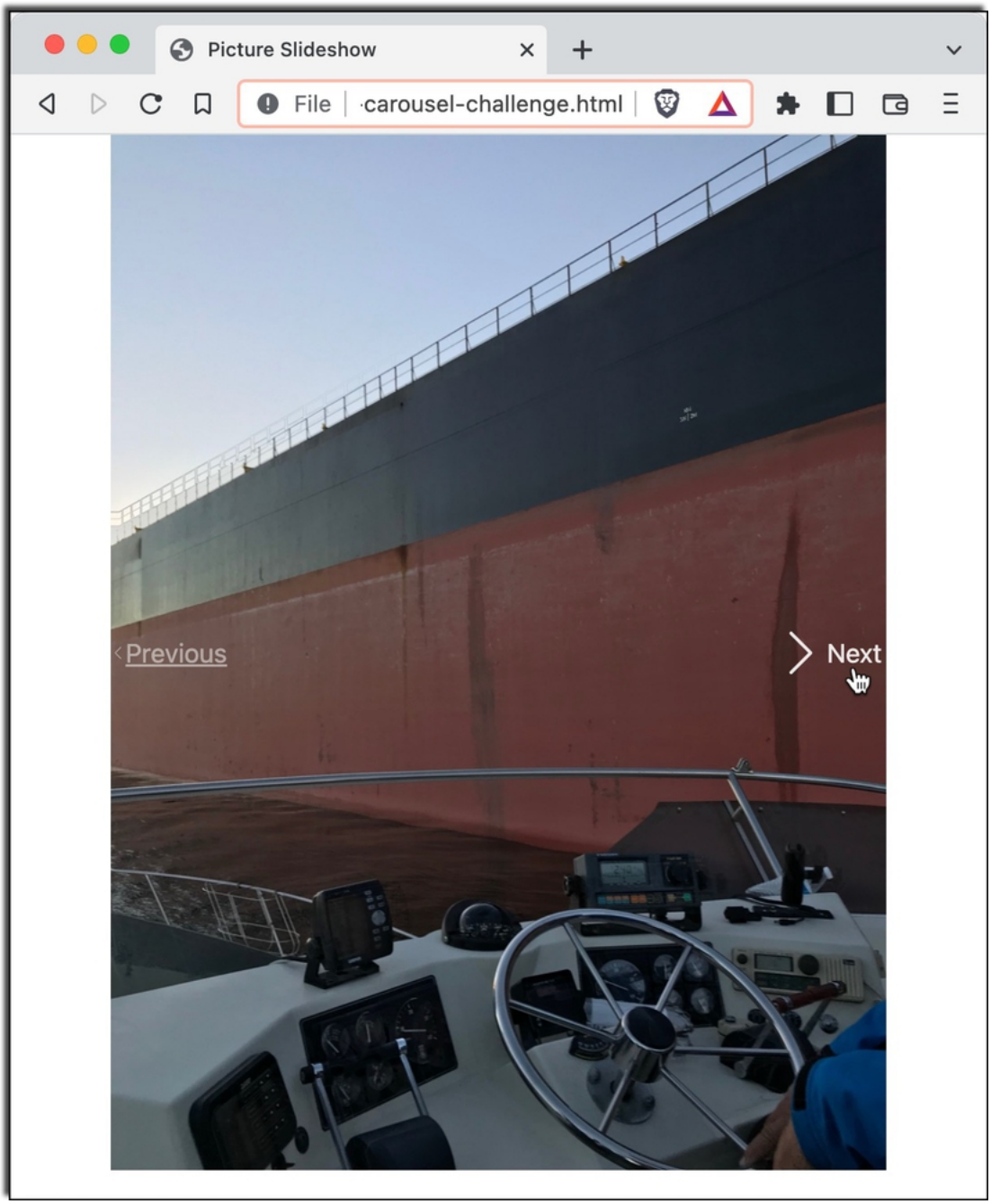


Challenge

Visit the Carousel documentation on the Bootstrap site at `https://getbootstrap.com/docs/5.2/components/carousel/`. Follow the examples and instructions in the Bootstrap documentation to add the following to your carousel:

- Controls
- Captions
- Crossfade effect

The resulting page should look something like this:



Clicking the **Previous** and **Next** controls should switch slides.

Solution: components/Solutions/photo-carousel.html

```
     -------Lines 1 through 15 Omitted-------
16.  <div class="container">
17.    <div id="my-photo-slideshow" class="carousel slide"
18.      data-bs-ride="carousel">
19.      <div class="carousel-inner">
20.        <div class="carousel-item active">
21.         <img class="d-block w-100" src="images/picture1.jpg" alt="First Picture">
22.        </div>
23.        <div class="carousel-item">
24.         <img class="d-block w-100" src="images/picture2.jpg" alt="Second Picture">
25.        </div>
26.        <div class="carousel-item">
27.         <img class="d-block w-100" src="images/picture3.jpg" alt="Third Picture">
28.        </div>
29.      </div>
30.    </div>
31.  </div>
     -------Lines 32 through 37 Omitted-------
```

Challenge Solution: components/Solutions/photo-carousel-challenge.html

```
     -------Lines 1 through 15 Omitted-------
16.  <div class="container">
17.    <div id="my-photo-slideshow" class="carousel slide carousel-fade"
18.      data-bs-ride="carousel">
19.      <div class="carousel-inner">
20.        <div class="carousel-item active">
21.         <img class="d-block w-100" src="images/picture1.jpg" alt="First Picture">
22.          <div class="carousel-caption d-none d-md-block">
23.            <h5>Approaching the ship</h5>
24.          </div>
25.        </div>
26.        <div class="carousel-item">
27.         <img class="d-block w-100" src="images/picture2.jpg" alt="Second Picture">
28.          <div class="carousel-caption d-none d-md-block">
29.            <h5>Getting closer...</h5>
30.          </div>
31.        </div>
32.        <div class="carousel-item">
33.         <img class="d-block w-100" src="images/picture3.jpg" alt="Third Picture">
34.          <div class="carousel-caption d-none d-md-block">
35.            <h5>Here we are!</h5>
36.          </div>
37.        </div>
38.      </div>
39.      <a class="carousel-control-prev" href="#my-photo-slideshow"
40.        role="button" data-bs-slide="prev">
41.        <span class="carousel-control-prev-icon"
42.          aria-hidden="true"></span>
43.        <span class="sr-only">Previous</span>
44.      </a>
45.      <a class="carousel-control-next" href="#my-photo-slideshow"
46.        role="button" data-bs-slide="next">
47.        <span class="carousel-control-next-icon"
48.          aria-hidden="true"></span>
49.        <span class="sr-only">Next</span>
50.      </a>
51.    </div>
52.  </div>
     -------Lines 53 through 58 Omitted-------
```

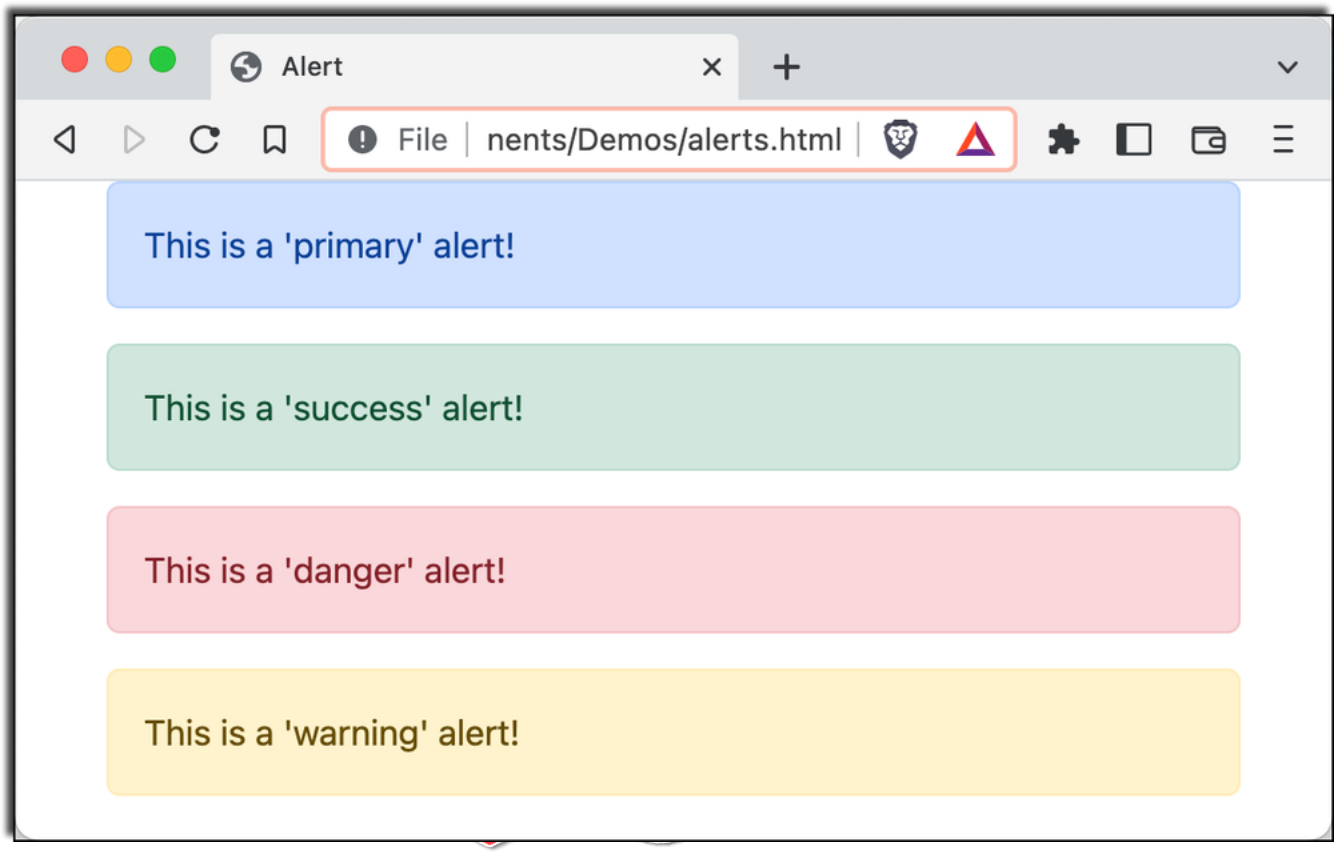
8.3. Alerts

Bootstrap's Alert component creates a box with a contextual color background, rounded corners, and a message. Alerts are useful for giving users feedback in response to actions. For example, if a user clicks a **Save** button, you could display an alert with the `alert-success` class, which will display with a light blue background and green text by default. Alerts can be created with eight different contextual classes. A few of them are shown in the following demo:

Demo 8.4: components/Demos/alerts.html

```
     -------Lines 1 through 11 Omitted-------
12.    <div class="alert alert-primary" role="alert">
13.      This is a 'primary' alert!
14.    </div>
15.    <div class="alert alert-success" role="alert">
16.      This is a 'success' alert!
17.    </div>
18.    <div class="alert alert-danger" role="alert">
19.      This is a 'danger' alert!
20.    </div>
21.    <div class="alert alert-warning" role="alert">
22.      This is a 'warning' alert!
23.    </div>
     -------Lines 24 through 30 Omitted-------
```


The preceding code will render the following:



For documentation on alerts see `https://getbootstrap.com/docs/5.2/components/alerts/`.

8.4. Collapse

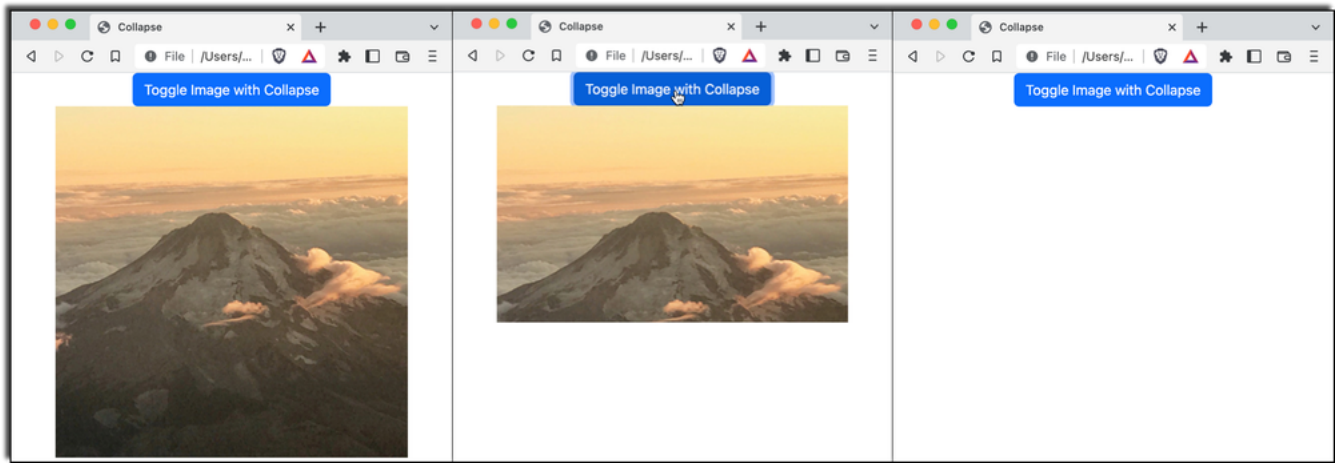
The purpose of the Collapse component is to show and hide content.

To use the Collapse component, assign the `collapse` class to a container, along with an `id` attribute. Then, create an element, such as a link or button, that will be used to trigger the collapse. The element that will trigger the collapse needs a `data-bs-toggle` attribute with a value of `collapse` and a `data-bs-target` attribute with a value equal to the `id` of the container you want to hide and show. When you click the triggering element, the collapsible content will animate between showing and hiding. Adding the `show` class to the collapsible content makes it visible by default. View the following demo to see an example:

Demo 8.5: components/Demos/collapse.html

```
      -------Lines 1 through 11 Omitted-------
12.     <button class="btn btn-primary" type="button"
13.       data-bs-toggle="collapse" data-bs-target="#collapse-example"
14.       aria-expanded="false" aria-controls="collapse-example">
15.       Toggle Image with Collapse
16.     </button>
17.     <div class="collapse show" id="collapse-example">
18.       <img src="images/mt-hood.jpg" alt="MT Hood">
19.     </div>
      -------Lines 20 through 26 Omitted-------
```

The screenshots below show the image before, during, and after collapsing:



For documentation on the Collapse component see `https://getbootstrap.com/docs/5.2/components/collapse/`.

8.5. Modal

Modals are popup windows that may or may not contain interactive elements such as buttons, forms, tooltips, and more. Bootstrap's Modal component is one of the more complex of Bootstrap's components. That's not to say that it's difficult to use, but that it has a lot of functionality. Like the other components, however, you can access the functionality and change the look and behavior of modals just by assigning attributes to HTML elements.

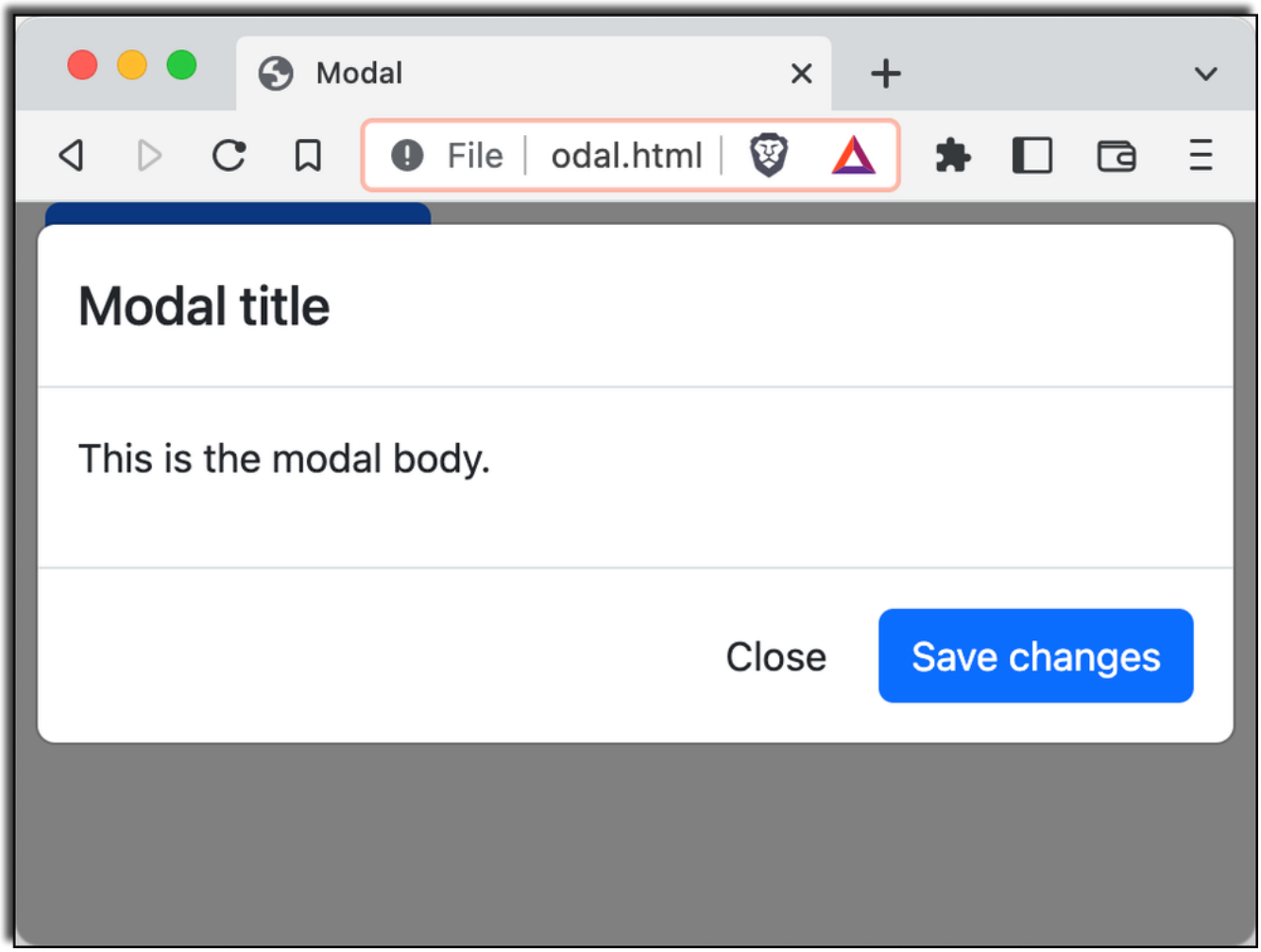
To create a modal, use the `modal` class on a container. Inside a modal container, you can use additional containers to specify different types of content. For example, the `modal-dialog` class creates a dialog window that can have buttons for functionality such as closing the modal and saving. Once you've created a modal, it will only show up when it's activated by another element, such as a button.

To create an element that activates a modal, add a `data-bs-toggle` attribute with a value of `modal` and a `data-bs-target` attribute with a value equal to the `id` of the modal you want to activate, as shown in the following demo:

Demo 8.6: components/Demos/modal.html

```
     -------Lines 1 through 10 Omitted-------
11.  <div class="container">
12.    <button type="button" class="btn btn-primary" data-bs-toggle="modal" data-bs-
             target="#test-modal">
13.      Launch the modal
14.    </button>
15.    <div class="modal" id="test-modal">
16.      <div class="modal-dialog">
17.        <div class="modal-content">
18.          <div class="modal-header">
19.            <h4 class="modal-title">Modal title</h4>
20.          </div>
21.          <div class="modal-body">
22.            <p>This is the modal body.</p>
23.          </div>
24.          <div class="modal-footer">
25.            <button type="button" class="btn btn-default" data-bs-dis ↵↵
              miss="modal">Close</button>
26.            <button type="button" class="btn btn-primary">Save changes</button>
27.          </div>
28.        </div>
29.      </div>
30.    </div>
31.  </div>
     -------Lines 32 through 37 Omitted-------
```


The preceding code will render the following:



Notice that the **Close** button includes the attribute `data-bs-dismiss="modal"`; whereas, the **Save changes** button has no equivalent attribute. When you click the **Close** button, the modal is closed, but nothing happens when you click the **Save changes** button. You would have to write custom JavaScript to catch those click events and call a callback function to add the behavior you want.

For documentation on the Modal component see `https://getbootstrap.com/docs/5.2/components/modal/`.

8.6. Pagination

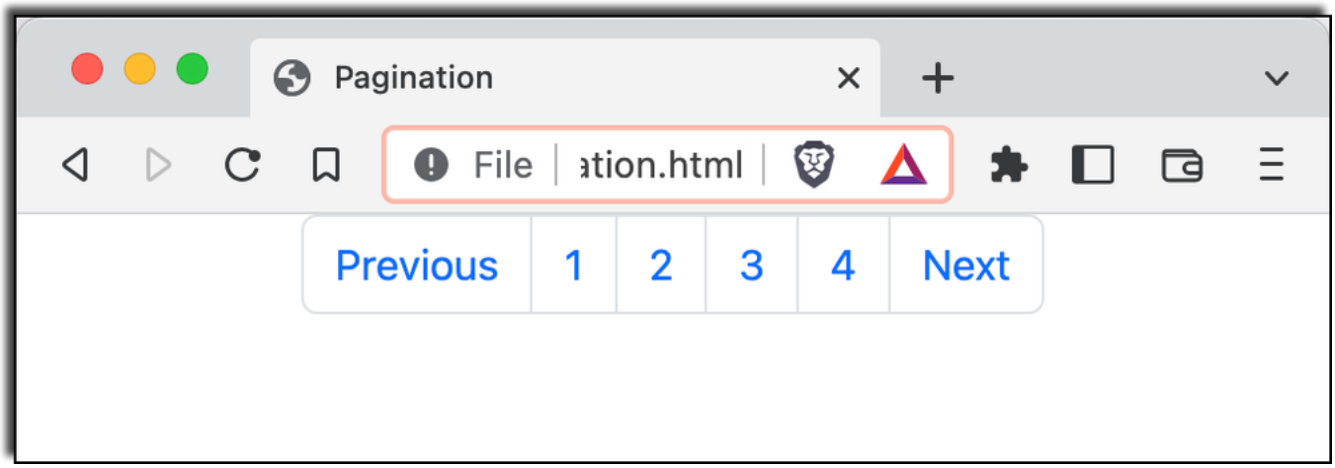
The Pagination component takes a list of links and turns it into a responsive navigation component that can be used for navigating between multiple related pages, such as a list of products or pages in an article.

To create pagination, use the `pagination` class on a list element, add the `page-item` class to the list items inside the list element, and add the `page-link` class to the links inside the list items, as shown in the following demo:

Demo 8.7: components/Demos/pagination.html

```
      -------Lines 1 through 10 Omitted-------
11.   <div class="container">
12.     <ul class="pagination justify-content-center">
13.       <li class="page-item"><a class="page-link" href="#">Previous</a></li>
14.       <li class="page-item"><a class="page-link" href="#">1</a></li>
15.       <li class="page-item"><a class="page-link" href="#">2</a></li>
16.       <li class="page-item"><a class="page-link" href="#">3</a></li>
17.       <li class="page-item"><a class="page-link" href="#">4</a></li>
18.       <li class="page-item"><a class="page-link" href="#">Next</a></li>
19.     </ul>
20.   </div>
      -------Lines 21 through 26 Omitted-------
```


The preceding code will render the following:



We have added the `justify-content-center` class to the `ul` element to center the pagination control in its container.

Note that the Pagination component is a design/layout component. To have the links result in paginating through content/pages, you would need to write custom JavaScript and/or server-side code.

For documentation on the Pagination component see `https://getbootstrap.com/docs/5.2/components/pagination/`.

8.7. Card

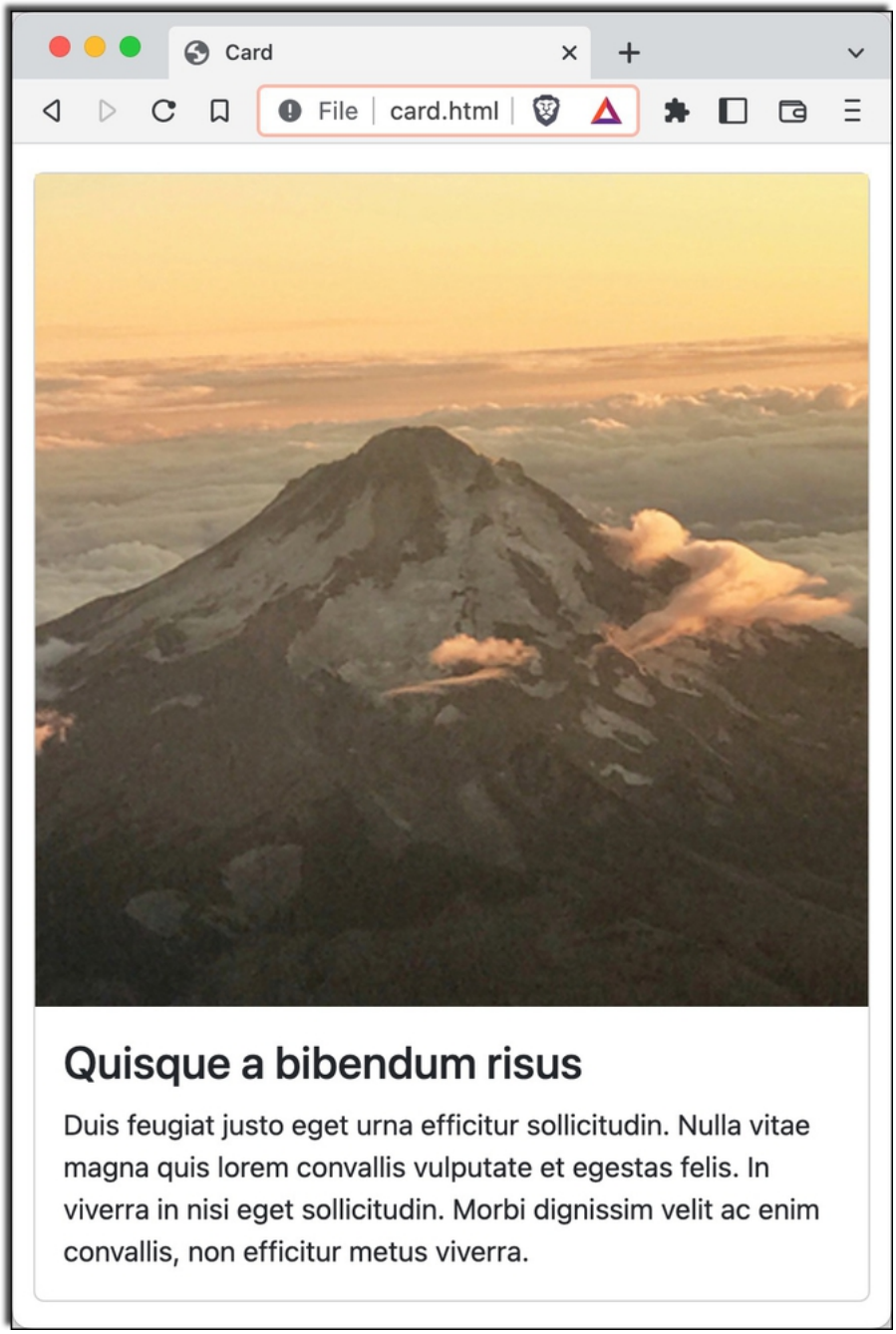
Bootstrap's Card component lets you create containers for content. Cards can have headers, footers, contextual background colors, and many different kinds of content. Cards can be positioned and formatted using Bootstrap's utilities.

To create a card, apply a class value of `card` to a container element (most often a `div` element), and then use the different card content types to put images, text, lists, links, and other content inside the card component. Here's an example of a card containing an image with a title and text:

Demo 8.8: components/Demos/card.html

```
-------Lines 1 through 11 Omitted-------
12.     <div class="card">
13.       <img src="images/mt-hood.jpg" alt="Mt. Hood">
14.       <div class="card-body">
15.         <h2 class="card-title">Quisque a bibendum risus</h2>
16.         <p class="card-text">
17.           Duis feugiat justo eget urna efficitur sollicitudin.
18.           Nulla vitae magna quis lorem convallis vulputate et
19.           egestas felis. In viverra in nisi eget sollicitudin.
20.           Morbi dignissim velit ac enim convallis, non efficitur
21.           metus viverra.
22.         </p>
23.       </div>
24.     </div>
-------Lines 25 through 31 Omitted-------
```

The preceding code will render the following:



For documentation on the Card component see `https://getbootstrap.com/docs/5.2/components/card/`.

8.8. Tooltip

Tooltips are the messages that appear over a button or a link to give more information about that particular thing. They're often used to describe the function of a button, or tool – hence, the name.

Tooltips can be created without the assistance of Bootstrap by simply adding a `title` attribute to a link. However, browsers render tooltips differently, and HTML doesn't give you any ability to customize default tooltips or position them.

Bootstrap's Tooltip component relies on an external JavaScript library called "popper" to do its job, and tooltips are disabled in Bootstrap by default. The result of these two things is that you'll need to do a little configuration before you can use the Tooltip component.

The first thing to do is to make sure that you're including `popper.js`. The easiest way is to just make sure the version of the Bootstrap JavaScript file that you're including into your file is the one that includes `popper.js`. You can find the latest version of Bootstrap at `https://getbootstrap.com/docs/5.2/getting-started/download/`. The best file to include is `bootstrap.bundle.min.js`.

Once you've replaced the `bootstrap.js` link with the `bootstrap.bundle.js` link, you'll need to write a piece of code after the other `script` tags that will enable tooltips. Here's what that `script` element looks like:

```
<script>
  const tooltipTriggerList = document.querySelectorAll('[data-bs-toggle="tooltip"]');
  const tooltipList = [...tooltipTriggerList].map((tooltipTriggerEl) => new boot ↵↵
strap.Tooltip(tooltipTriggerEl));
</script>
```

With those two things done, you're ready to start using Tooltips.

Tooltips and Accessibility

Although it's possible to add tooltips to many different types of elements, it's recommended and best for accessibility if you only add them to HTML elements that are interactive, such as links, inputs, and buttons.

To create a tooltip:

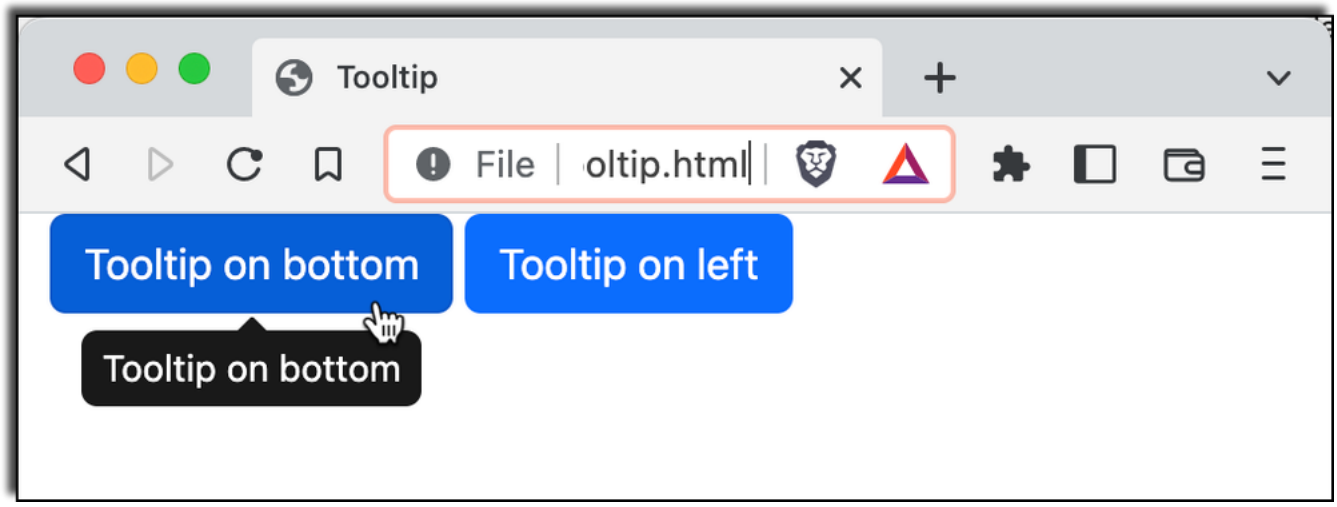
1. Add a `data-bs-toggle` attribute to the element the tip should appear on and give it a value of `tooltip`.
2. Give the element a `title` attribute containing the text of the tooltip.

By default, tooltips will appear after the element that triggers them. If you want to manually position a tooltip, you can do that by using the `data-bs-placement` attribute with a value of `top`, `right`, `bottom`, or `left` as shown in the following demo:

Demo 8.9: components/Demos/tooltip.html

```
     -------Lines 1 through 10 Omitted-------
11.  <div class="container">
12.    <button type="button" data-bs-toggle="tooltip" data-bs-placement="bottom" ti ↵↵
              tle="Tooltip on bottom"
13.      class="btn btn-primary">
14.      Tooltip on bottom
15.    </button>
16.    <button type="button" data-bs-toggle="tooltip" data-bs-placement="left" ti ↵↵
              tle="Tooltip on left"
17.      class="btn btn-primary">
18.      Tooltip on left
19.    </button>
20.  </div>
21.  <!-- Bootstrap core JavaScript
22.  ================================================== -->
23.  <!-- Placed at the end of the document so the pages load faster -->
24.  <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/js/bootstrap.bun ↵↵
              dle.min.js"></script>
25.  <script>
26.    const tooltipTriggerList = document.querySelectorAll('[data-bs-tog ↵↵
              gle="tooltip"]');
27.    const tooltipList = [...tooltipTriggerList].map((tooltipTriggerEl) => new
              bootstrap.Tooltip(tooltipTriggerEl));
28.  </script>
     -------Lines 29 through 30 Omitted-------
```


The preceding code will render the following:



For documentation on tooltips see `https://getbootstrap.com/docs/5.2/components/tooltips/`.

8.9. Popover

Bootstrap's Popover component creates informational windows that hover above the other content on your page and can be triggered by clicking another element. Popovers are basically larger versions of tooltips. As such, they have the same configuration requirements as Tooltips: you have the include the `popper.js` library (which is part of the `bootstrap.bundle.min.js` file), and you have to specifically enable Popovers. Here's a `script` element and script to insert at the end of your document's `body` element to enable popovers:

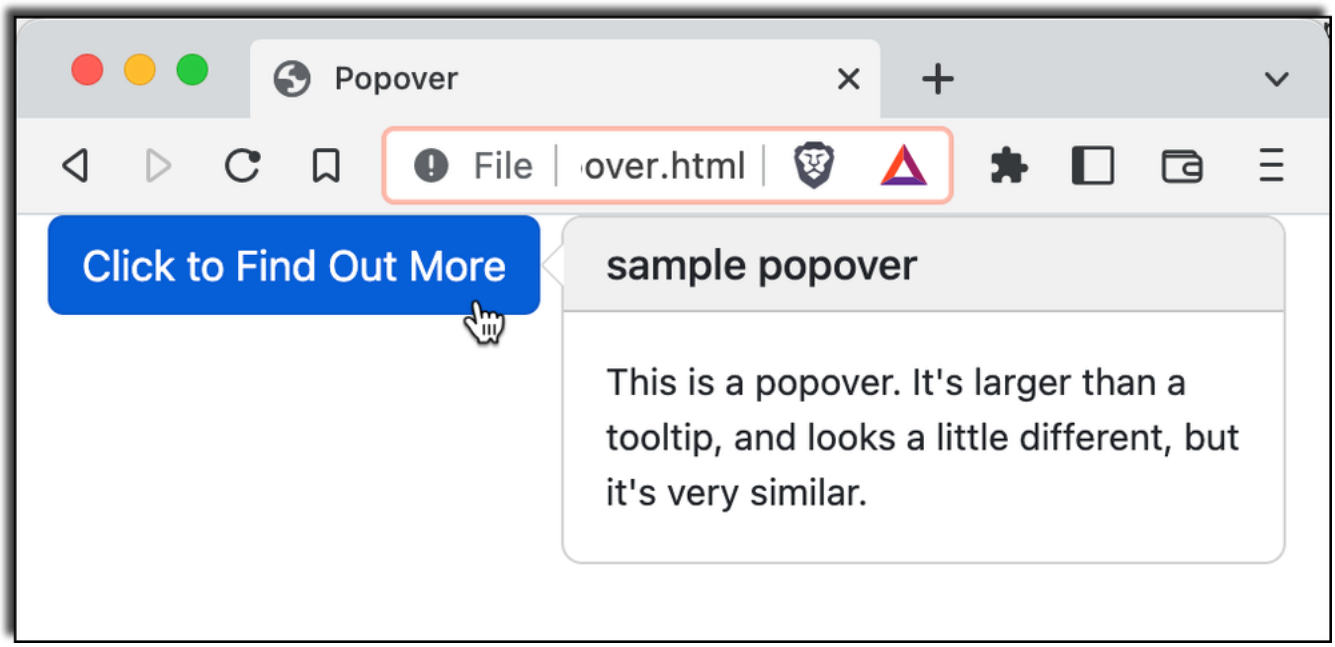
```
<script>
  const popoverTriggerList = document.querySelectorAll('[data-bs-toggle="popover"]');
  const popoverList = [...popoverTriggerList].map(popoverTriggerEl => new boot ↵↵
strap.Popover(popoverTriggerEl));
</script>
```

To create a popover, use a `data-bs-toggle` attribute with a value of `popover`, a `title` attribute that will appear as the popover's title, and a `data-bs-content` attribute with a value of whatever you want to appear on the popover. Popovers may also be positioned using the `data-bs-placement` attribute. Here's an example of a simple popover:

Demo 8.10: components/Demos/popover.html

```
      -------Lines 1 through 10 Omitted-------
11.   <div class="container">
12.     <button type="button" class="btn btn-primary"
13.       data-bs-toggle="popover" title="sample popover"
14.       data-bs-content="This is a popover. It's larger than a tooltip, and
15.                        looks a little different, but it's very similar.">
16.       Click to Find Out More
17.     </button>
18.   </div>
19.   <!-- Bootstrap core JavaScript
20.   ================================================== -->
21.   <!-- Placed at the end of the document so the pages load faster -->
22.   <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/js/bootstrap.bun ↵↵
            dle.min.js"></script>
23.   <script>
24.     const popoverTriggerList = document.querySelectorAll('[data-bs-tog ↵↵
            gle="popover"]');
25.     const popoverList = [...popoverTriggerList].map(popoverTriggerEl => new boot ↵↵
            strap.Popover(popoverTriggerEl));
26.   </script>
      -------Lines 27 through 28 Omitted-------
```


The preceding code will render the following:



For documentation on popovers see `https://getbootstrap.com/docs/5.2/components/popovers/`.

Conclusion

In this lesson, you have learned:

- How to use button styles.
- How to cycle through elements with the carousel.
- How to make and trigger modal windows.
- How to format content using the `Card` component.
- How to create tool tips and popovers.

To learn more about these and other Bootstrap components, visit `https://getbootstrap.com/docs/5.2/components`.

LESSON 9
Bootstrap Utilities

Topics Covered

☑ Styling elements with utilities.

☑ Applying borders with Bootstrap.

☑ Positioning elements.

☑ Shadows.

Introduction

Utilities are helper classes that you can use to quickly style elements without writing any CSS code. Like many of the classes you've seen in this course, you apply utility classes by adding one or more classes to elements. In this lesson, you'll learn about some of the most frequently-used utilities.

9.1. Borders

The Bootstrap border utilities simplify the process of applying borders to single elements by providing a large assortment of variations on the basic `border` class. If you just use the `border` class, Bootstrap will apply a simple 1-pixel gray border to the element. You can add borders to specific sides of an element by adding one of the four directions (`-top`, `-end`, `-bottom`, `-start`) to the basic `border` class (e.g., `border-top`).

❖ 9.1.1. Border Colors

You can change the color of the border by applying an additional `border-` class with one of the contextual classes specified. For example, the following element would have a 1px red border by default:

```
<div class="border border-danger"></div>
```

❖ 9.1.2. Border Width

Bootstrap has five border width classes built in. To change the border width, apply a `border-1`, `border-2`, `border-3`, `border-4`, or `border-5` class. Higher numbers represent thicker borders. You can remove a border from an element or from a certain side of an element, by using a `-0` at the end of a border class:

```
<div class="border border-bottom-0"></div>
```

❖ 9.1.3. Rounded Corners

As we saw earlier in the course, you can give an element rounded corners using the `rounded` class. You can also round specific corners using `rounded-top`, `rounded-end`, `rounded-bottom`, and `rounded-start`. In addition, you can make an element circular using `rounded-circle` or you can make a border that's flat on the top and bottom and curved on the sides by using `rounded-pill`:

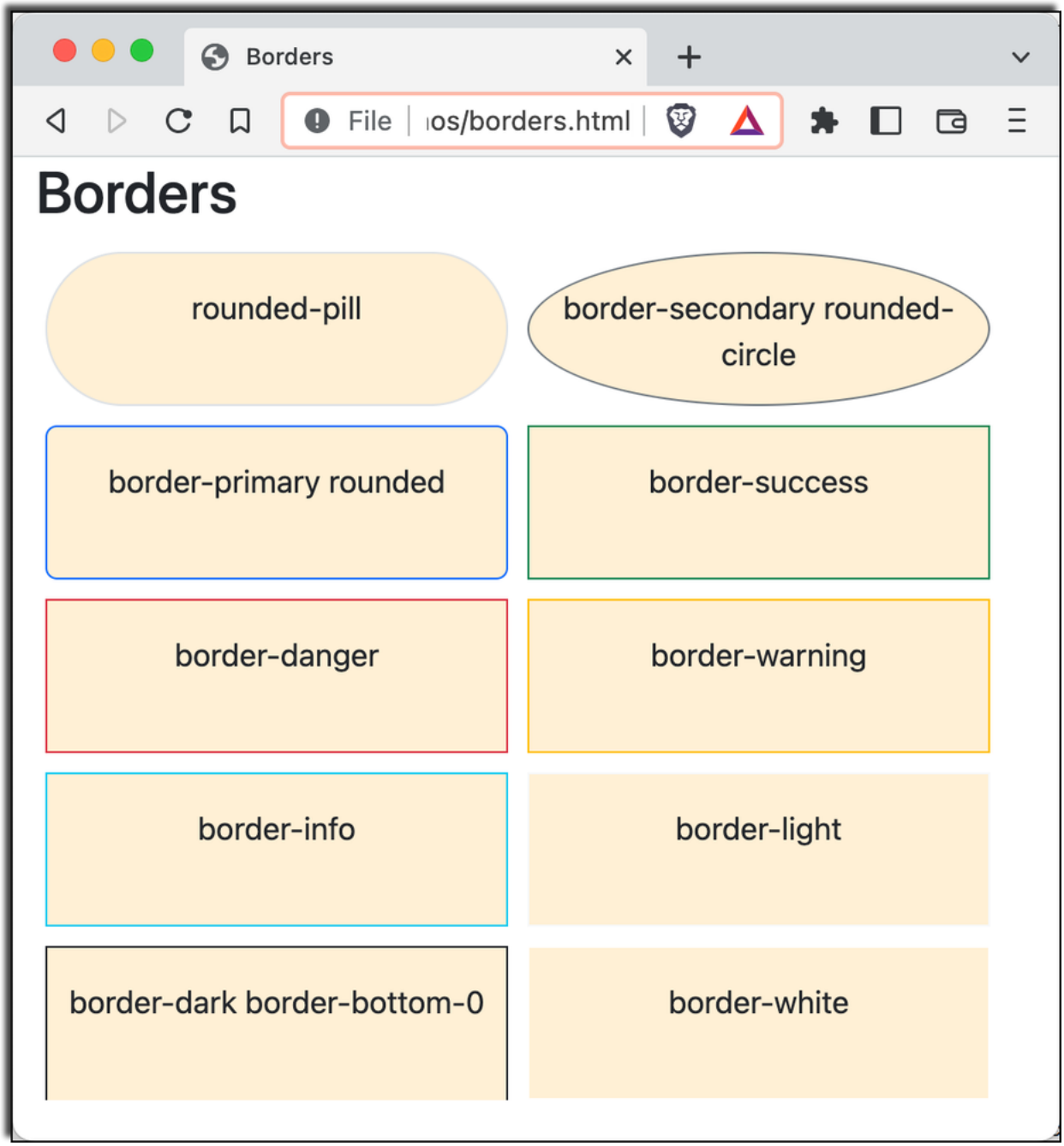
```
<div class="border rounded-circle"></div>
```


The sample below uses Bootstrap's border utilities:

Demo 9.1: utilities/Demos/borders.html

```
      -------Lines 1 through 21 Omitted-------
22.   <div class="container">
23.     <h1>Borders</h1>
24.     <div class="border rounded-pill">rounded-pill</div>
25.     <div class="border border-secondary rounded-circle">border-secondary rounded-
            circle</div>
26.     <div class="border border-primary rounded">border-primary rounded</div>
27.     <div class="border border-success">border-success</div>
28.     <div class="border border-danger">border-danger</div>
29.     <div class="border border-warning">border-warning</div>
30.     <div class="border border-info">border-info</div>
31.     <div class="border border-light">border-light</div>
32.     <div class="border border-dark border-bottom-0">border-dark border-bottom-
            0</div>
33.     <div class="border border-white">border-white</div>
34.   </div>
      -------Lines 35 through 40 Omitted-------
```

The preceding code will render the following:



Note that we have added formatting to all of the `div.border` elements so that it's easier to see how the borders appear.

For documentation on borders see `https://getbootstrap.com/docs/5.2/utilities/borders/`.

9.2. Position

The position utilities include several common positioning styles that you may want to apply to elements. The first group of positioning classes simply changes the type of positioning being used by an element (the value of its CSS `position` property). These classes are:

- `position-static`
- `position-relative`
- `position-absolute`
- `position-fixed`
- `position-sticky`

The other type of positioning classes apply more complex styles to elements to position them. This group includes:

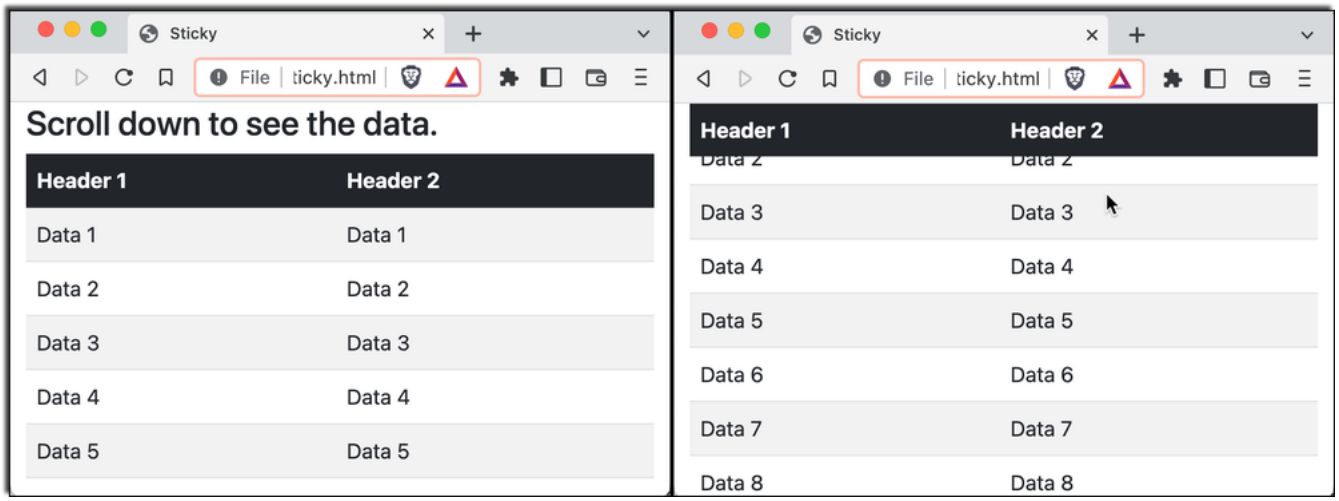
- `fixed-top`
- `fixed-bottom`
- `sticky-top`

The `sticky` property is a hybrid of relative and fixed positioning. It will be treated as relatively positioned until it scrolls to the top of the window, at which point it will stick in place. This is helpful for header rows. Note that, when you use this with tables, the `sticky-top` class should be applied to the `th` elements as shown in the following demo:

Demo 9.2: utilities/Demos/sticky.html

```
      -------Lines 1 through 11 Omitted-------
12.     <h2>Scroll down to see the data.</h2>
13.     <table class="table table-striped">
14.       <thead class="table-dark">
15.         <tr>
16.           <th class="sticky-top">Header 1</th>
17.           <th class="sticky-top">Header 2</th>
18.         </tr>
19.       </thead>
20.       <tbody>
21.         <tr><td>Data 1</td><td>Data 1</td></tr>
22.         <tr><td>Data 2</td><td>Data 2</td></tr>
23.         <tr><td>Data 3</td><td>Data 3</td></tr>
24.         <tr><td>Data 4</td><td>Data 4</td></tr>
25.         <tr><td>Data 5</td><td>Data 5</td></tr>
26.         <tr><td>Data 6</td><td>Data 6</td></tr>
      -------Lines 27 through 56 Omitted-------
```


As you scroll down the page, the sticky table header will stay at the top of the viewport, as shown in this image (the right side is after scrolling):



For documentation on position see `https://getbootstrap.com/docs/5.2/utilities/position/`.

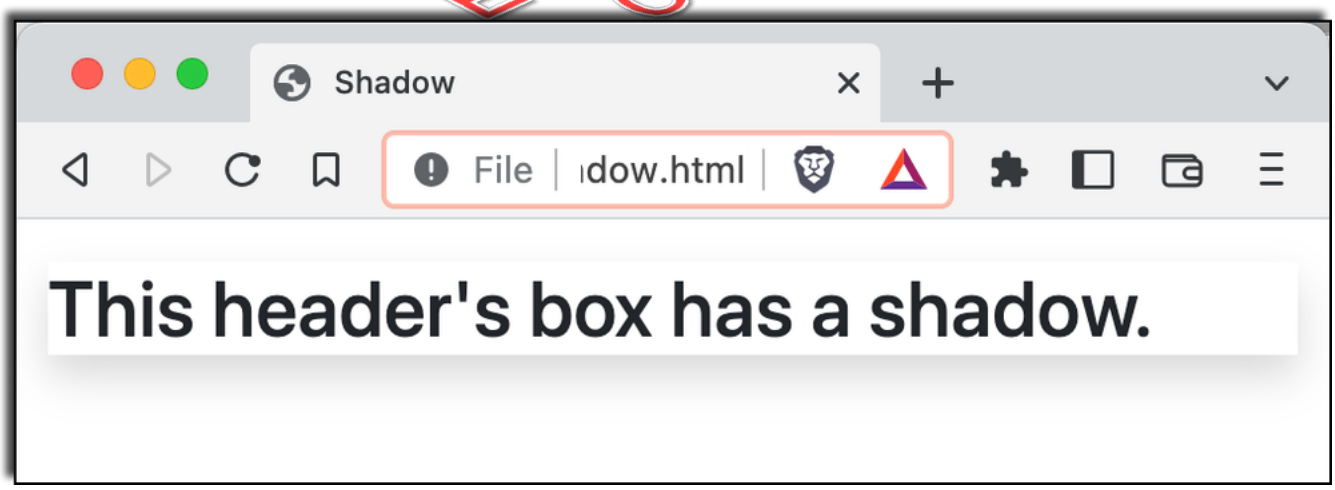
9.3. Shadows

The shadow utility creates box shadows on elements. By itself, the `shadow` class will apply a box shadow (sometimes also known as a "drop shadow") using the CSS `box-shadow` property. The default shadow will have a horizontal offset of `0`, a vertical offset of `.5rem`, a blur radius of `1rem`, and be black with an opacity of `.15`. So, the effect will be a light grey shadow slightly below the element it's applied to. If you want a larger shadow, you can use the `shadow-lg` class. If you want a smaller shadow, use the `shadow-sm` class. The following demo shows a paragraph with a normal shadow applied:

Demo 9.3: utilities/Demos/shadow.html

```
     -------Lines 1 through 10 Omitted-------
11.  <div class="container mt-3">
12.    <h1 class="shadow">This header's box has a shadow.</h1>
13.  </div>
     -------Lines 14 through 19 Omitted-------
```


The preceding code will render the following:



For documentation on shadows see `https://getbootstrap.com/docs/5.2/utilities/shadows/`.

9.4. Spacing Utilities

Bootstrap has several built-in spacing utilities that you can use to adjust the margins and padding around an element. To adjust the padding, use the `p-` classes, and to adjust margins, use the `m-` classes. Both of these utilities can have one of the viewport sizes and an amount of spacing ranging from `0` to `5`. Default pacing values for both `p-` and `m-` are as follows:

- `0` - no space
- `1` - `.25rem`
- `2` - `.5rem`
- `3` - `1rem`
- `4` - `1.5rem`
- `5` - `3rem`

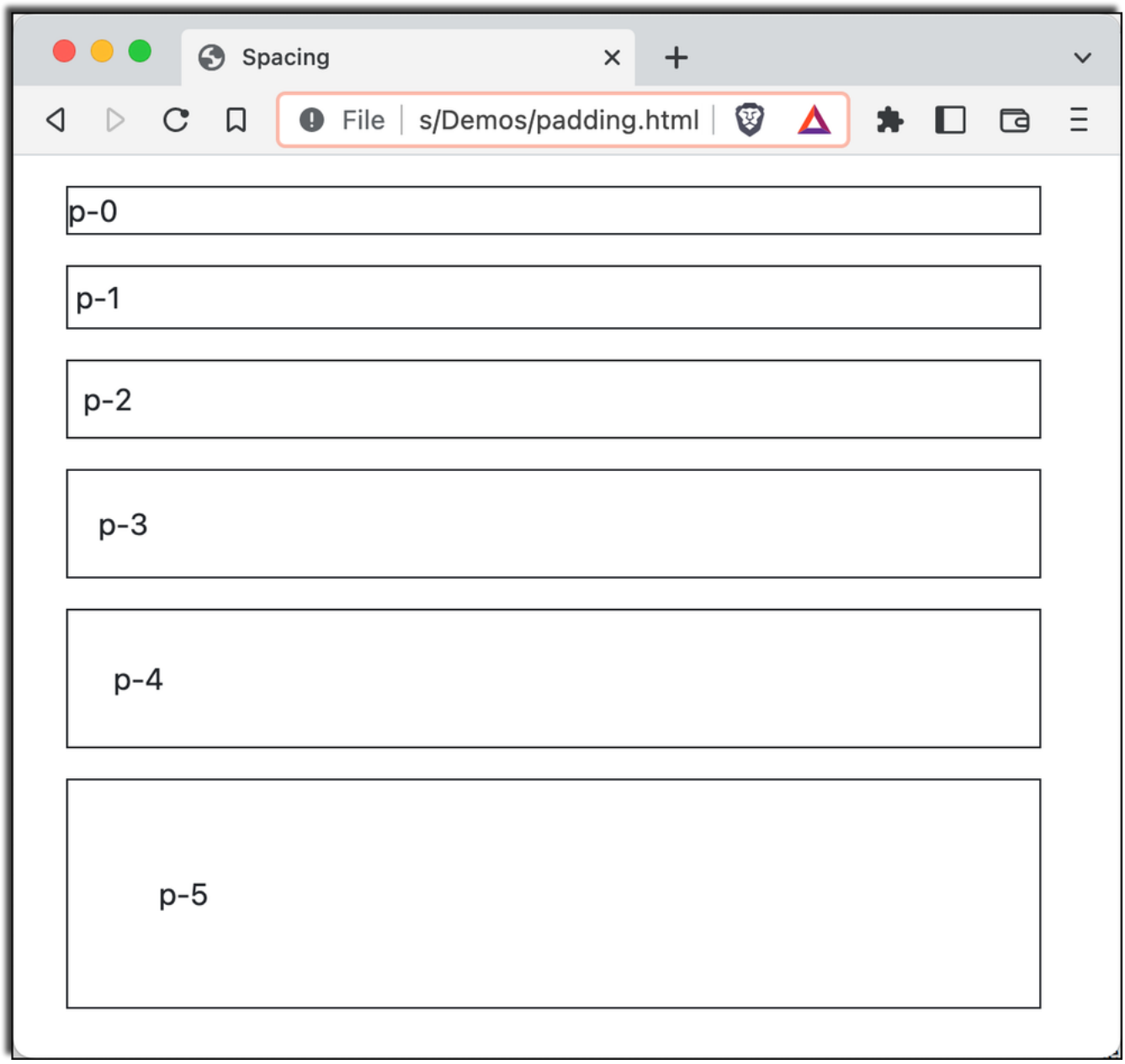
For example, to give an element a top margin of `3rem` at viewport sizes of `sm` and above, use the `m-sm-5` class.

The following demo shows paragraph elements with each of the six padding sizes applied:

Demo 9.4: utilities/Demos/padding.html

```
     -------Lines 1 through 10 Omitted-------
11.  <div class="container m-3">
12.    <p class="border p-0">p-0</p>
13.    <p class="border p-1">p-1</p>
14.    <p class="border p-2">p-2</p>
15.    <p class="border p-3">p-3</p>
16.    <p class="border p-4">p-4</p>
17.    <p class="border p-5">p-5</p>
18.  </div>
     -------Lines 19 through 24 Omitted-------
```

The preceding code will render the following:



For documentation on spacing see `https://getbootstrap.com/docs/5.2/utilities/spacing/`.

9.5. Miscellaneous Utilities, Helpers, and Components

Bootstrap has several other utilities, helpers, and components that help with specific CSS styling. In this section, we'll briefly cover some of the more useful ones.

❖ 9.5.1. Clearfix Helper

All elements and content that come after a floated element (one that has `float:left` or `float:right` applied to it) will flow around the floated element until an element has been properly cleared. This has caused web developers headaches for years. The many different solutions to the problem came to be known as "clearfixes." Bootstrap contains a `clearfix` helper class that implements current best practices for clearing floats. To use it, just apply the `clearfix` class to the first element you want to clear (i.e., not float to the left or right) after a floated element.

For documentation on clearfix see `https://getbootstrap.com/docs/5.2/helpers/clearfix/`.

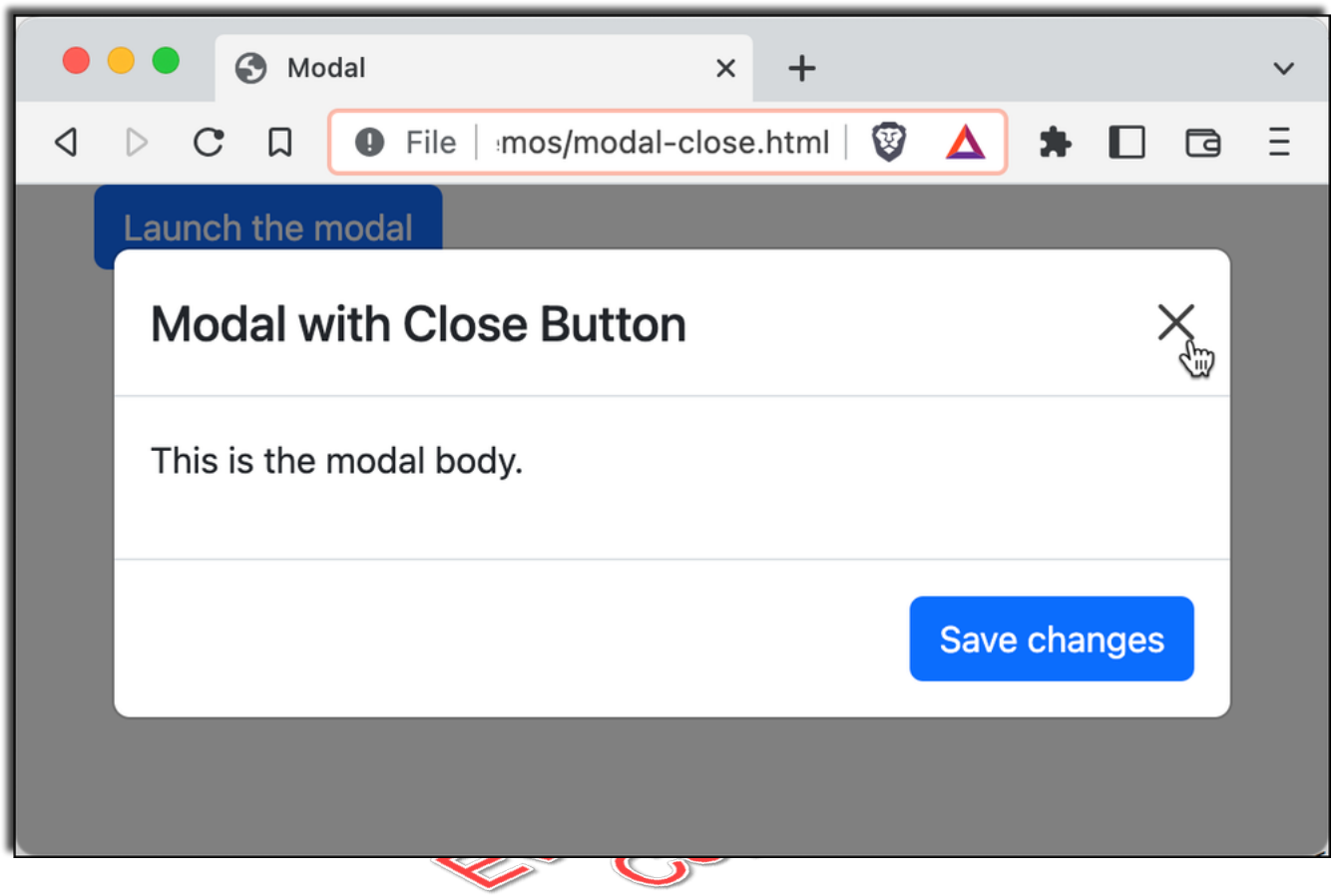
❖ 9.5.2. Close Button Component

The close button component styles a button element as a grey "X" that you can use as a close button for modals and alerts. The class to apply to a button to use the close button component is just `btn-close`.

Demo 9.5: utilities/Demos/modal-close.html

```
     -------Lines 1 through 15 Omitted-------
16.      <div class="modal-dialog">
17.        <div class="modal-content">
18.          <div class="modal-header">
19.            <h4 class="modal-title">Modal with Close Button</h4>
20.            <button type="button" class="btn-close" data-bs-dismiss="modal">
21.              <span aria-hidden="true">&times;</span>
22.            </button>
23.          </div>
24.          <div class="modal-body">
25.            <p>This is the modal body.</p>
26.          </div>
27.          <div class="modal-footer">
28.            <button type="button" class="btn btn-primary">
29.              Save changes
30.            </button>
31.          </div>
32.        </div>
33.      </div>
34.    </div>
     -------Lines 35 through 41 Omitted-------
```


The preceding code will render the following:



Note that you need to include `data-bs-dismiss="modal"` in the button for it to work.

For documentation on the close button component see `https://getbootstrap.com/docs/5.2/components/close-button/`.

❖ 9.5.3. Ratio Helper

The ratio helper classes style embedded objects, such as iframe elements, embed elements, video elements, and object elements to make them responsive while maintain their width-to-height ratio (also known as *aspect ratio*). To use ratio, apply the `ratio` class to a `div` element surrounding the embedded object. You can also use optional aspect ratio classes to cause the embedded object to maintain its aspect ratio as it scales up and down. The aspect ratio classes are:

- ratio-1x1
- ratio-4x3
- ratio-16x9
- ratio-21x9

The following demo shows how to embed a video and make it responsive with the `ratio` class:

Demo 9.6: utilities/Demos/ratio.html

```
     -------Lines 1 through 10 Omitted-------
11.  <div class="container ratio ratio-16x9">
12.    <iframe src="https://www.youtube.com/embed/KjKd3Ba8RWw"></iframe>
13.  </div>
     -------Lines 14 through 19 Omitted-------
```

To see how it works, open the file in your browser and slowly resize the browser window. As you grow and shrink the browser, the size of the video will adjust.

For documentation on the ratio helper see `https://getbootstrap.com/docs/5.2/helpers/ratio/`.

❖ 9.5.4. Visually Hidden Helper

If you want an element on your page to be hidden to every browser that uses the page except for screen readers, you can use the `visually-hidden` class. If you want to "show" the element again to screen readers when it has focus, you can also add the `visually-hidden-focusable` class.

For documentation on the visually hidden helper see `https://getbootstrap.com/docs/5.2/helpers/visually-hidden/`.

❖ 9.5.5. Visibility Utility

The visibility utilities simply set whether an element is visible (by using the `visible` class) or invisible (by using the `invisible` class). One important thing to know is that these classes set values of the CSS `visibility` property, not the `display` property. So, setting an element to `invisible` only hides it, it doesn't remove it from your web page or affect the layout of the page.

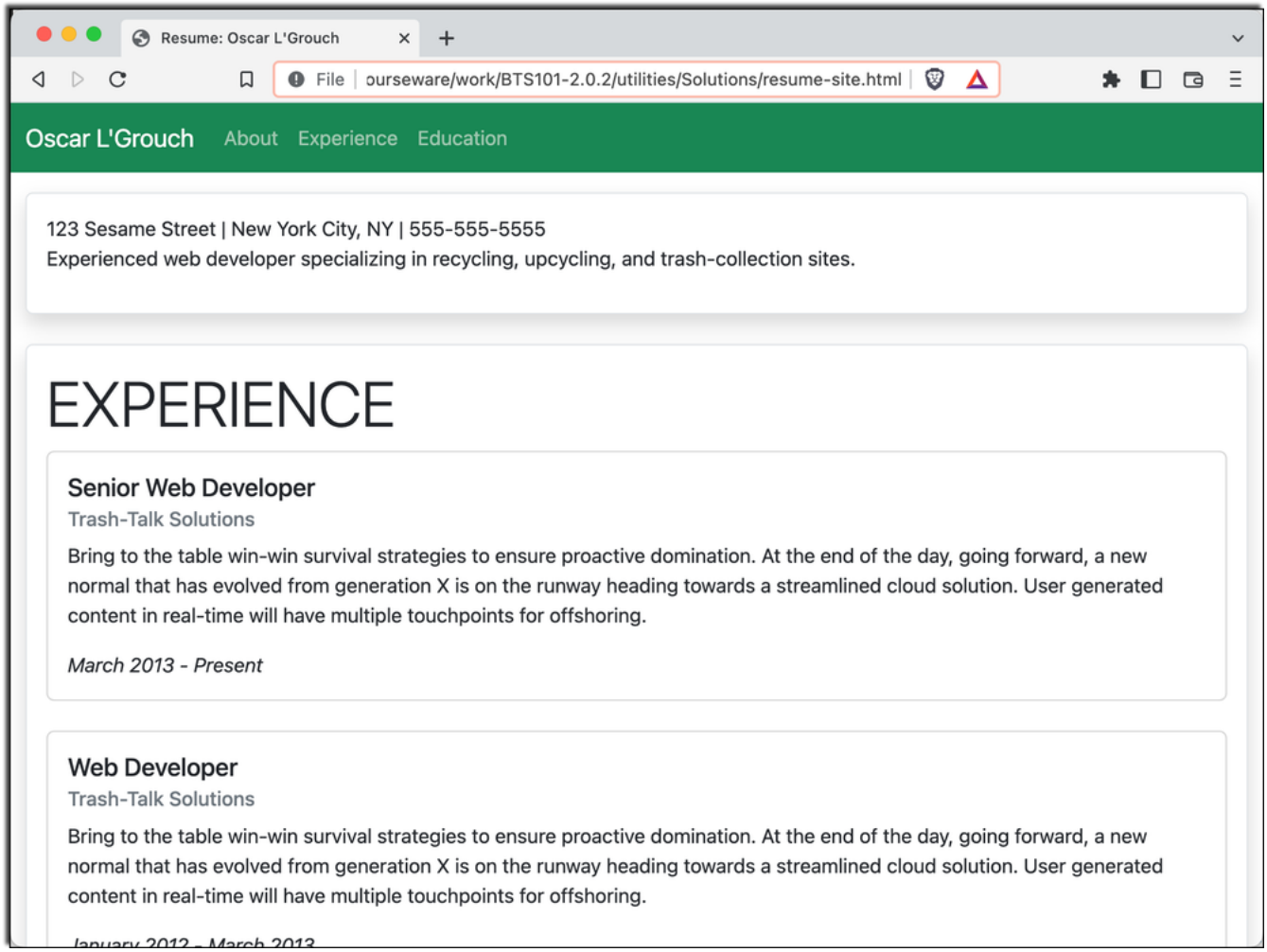
For documentation on the visibility utility see `https://getbootstrap.com/docs/5.2/utilities/visibility/`.

Exercise 11: Build a Single-page Website, Part 1

45 to 60 minutes

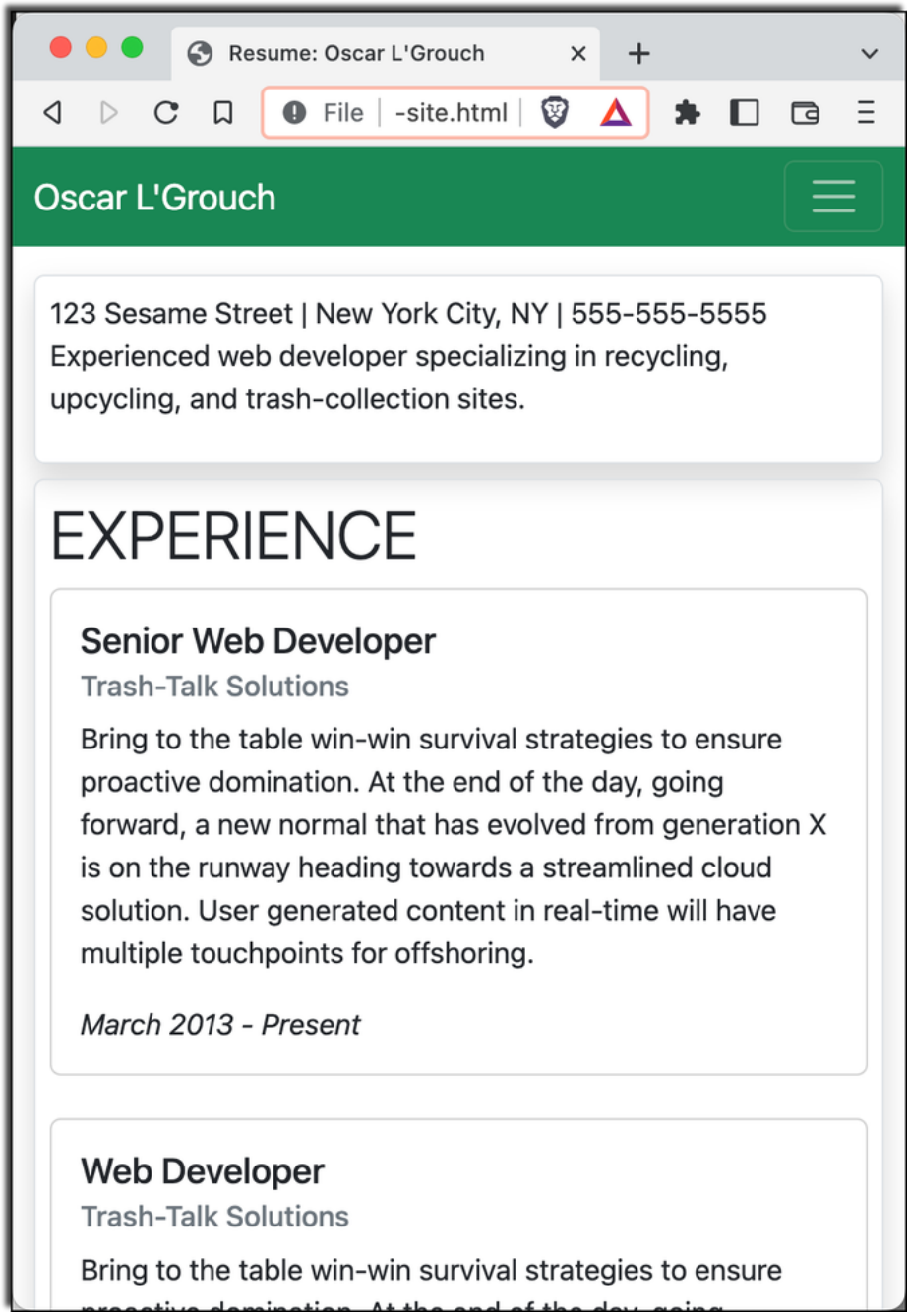
In this exercise, you will start creating a single-page resume site using many of the components, utilities, and classes you've learned about in this course.

1. Open `resume-site.html` from the `utilities/Exercises` folder.
2. Open the file in a web browser. This resume currently has just plain HTML markup with no Bootstrap classes applied.
3. Style the resume using Bootstrap classes to match the spacing, typography, and layout of the following image (when viewed in a desktop browser):



Note that the navigation bar at the top is missing from the starting HTML. You will have to add it.

4. Add or change the Bootstrap classes to make the resume match the following layout when viewed on a mobile device (or "extra small" viewport):



Notice how the navigation bar is collapsed in mobile view. You will need to add the toggler icon and add classes so that the menu is collapsed by default except on large (and larger) viewports.

Solution: utilities/Solutions/resume-site.html

```
      -------Lines 1 through 10 Omitted-------
11.   <nav class="navbar navbar-expand-lg navbar-dark bg-success sticky-top">
12.     <div class="container-fluid">
13.       <a class="navbar-brand" href="#page-top">Oscar L'Grouch</a>
14.       <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-
              bs-target="#navbar-nav"
15.         aria-controls="navbar-nav" aria-expanded="false" aria-label="Toggle navi ↵↵
              gation">
16.         <span class="navbar-toggler-icon"></span>
17.       </button>
18.       <div class="collapse navbar-collapse" id="navbar-nav">
19.         <ul class="navbar-nav">
20.           <li class="nav-item">
21.             <a class="nav-link active" aria-current="page" href="#about">About</a>
22.           </li>
23.           <li class="nav-item">
24.             <a class="nav-link" href="#experience">Experience</a>
25.           </li>
26.           <li class="nav-item">
27.             <a class="nav-link" href="#education">Education</a>
28.           </li>
29.         </ul>
30.       </div>
31.     </div>
32.   </nav>
33.   <div id="page-top" class="container-fluid mt-3 mb-5">
34.     <section id="about" class="p-2 p-lg-3 mb-2 mb-lg-4 border rounded shadow">
35.       <div>123 Sesame Street | New York City, NY | 555-555-5555</div>
36.       <p>Experienced web developer specializing in recycling, upcycling, and trash-
              collection sites.</p>
37.     </section>
38.     <section id="experience" class="p-2 p-lg-3 mb-2 mb-lg-4 border rounded shadow">
39.       <h2 class="text-uppercase display-4">Experience</h2>
40.       <div class="card mb-4">
41.         <div class="card-body">
42.           <h5 class="card-title">Senior Web Developer</h5>
43.           <h6 class="card-subtitle mb-2 text-muted">Trash-Talk Solutions</h6>
44.           <p class="card-text">Bring to the table win-win survival strategies to
                ensure proactive domination. At the end
45.             of the day, going forward, a new normal that has evolved from generation
                  X is on the runway heading towards a
46.             streamlined cloud solution. User generated content in real-time will
                have multiple touchpoints for offshoring.</p>
47.           <div class="fst-italic">March 2013 - Present</div>
48.         </div>
```

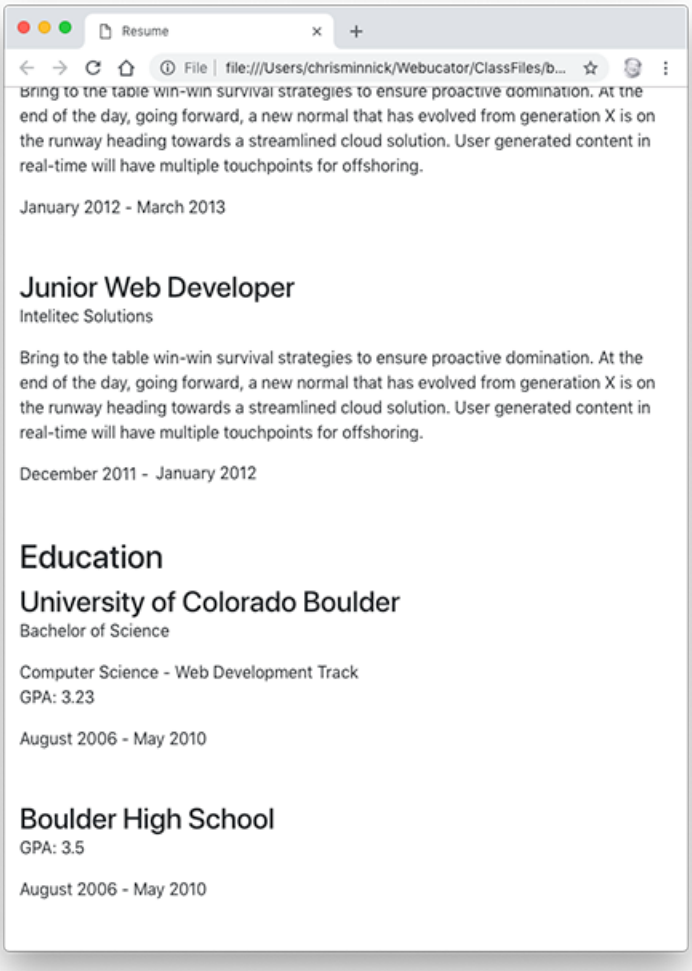
```
      </div>
      <div class="card mb-4">
        <div class="card-body">
          <h5 class="card-title">Web Developer</h5>
          <h6 class="card-subtitle mb-2 text-muted">Trash-Talk Solutions</h6>
          <p class="card-text">Bring to the table win-win survival strategies to
           ensure proactive domination. At the end
          of the day, going forward, a new normal that has evolved from generation
            X is on the runway heading towards a
           streamlined cloud solution. User generated content in real-time will
          have multiple touchpoints for offshoring.</p>
          <div class="fst-italic">January 2012 - March 2013</div>
        </div>
      </div>
      <div class="card mb-4">
        <div class="card-body">
          <h5 class="card-title">Junior Web Developer</h5>
          <h6 class="card-subtitle mb-2 text-muted">Trash-Talk Solutions</h6>
          <p class="card-text">Bring to the table win-win survival strategies to
           ensure proactive domination. At the end
          of the day, going forward, a new normal that has evolved from generation
            X is on the runway heading towards a
           streamlined cloud solution. User generated content in real-time will
          have multiple touchpoints for offshoring.</p>
          <div class="fst-italic">January 2012 - March 2013</div>
        </div>
      </div>
    </section>
    <section id="education" class="p-2 p-lg-3 mb-2 mb-lg-4 border rounded shadow">
      <h2 class="text-uppercase display-4">Education</h2>
      <div class="card mb-4">
        <div class="card-body">
          <h5 class="card-title">University of Colorado Boulder</h5>
          <ul class="card-text">
            <li>Bachelor of Science</li>
            <li>Computer Science - Web Development Track</li>
            <li>GPA: 3.23</li>
          </ul>
          <div class="fst-italic">August 2006 - May 2010</div>
        </div>
      </div>
      <div class="card mb-4">
        <div class="card-body">
          <h5 class="card-title">Boulder High School</h5>
          <ul class="card-text">
            <li>GPA: 3.5</li>
```

```
89.           </ul>
90.           <div class="fst-italic">August 2002 - May 2006</div>
91.         </div>
92.       </div>
93.     </section>
94.   </div>
95.
      -------Lines 96 through 101 Omitted-------
```

Code Explanation

Here are the steps to create the final resume for this exercise:

1. Inside the about section, give the `h1` a `display-1` class to make it stand out.
2. Give the `p` containing your "objective" a class of `lead`.
3. Set the margins of the `hrs` between the sections to 0 with the `m-0` class.
4. Use the `mb-5` class to add a 3rem bottom margin to each of the three `div`s containing the experience items.
5. Improve the spacing inside each experience item by applying `mb-0` to the `h3` containing the job title, and `mb-3` to the company name subheading.
6. Using the margin settings that you implemented in the experience section, fix the margins in the education section.
7. Add .5rem of padding to each of the three section elements, using the `p-2` class.
8. In large viewports, it would look better to have more padding around the page and around each of the resume sections, but on small viewports we need to take advantage of all the space we have. To increase the padding to 3rem for large screens use a responsive padding utility class, `p-lg-5`, on each of the `section` elements. At this point, the page should look like the following on large screens:



9. Make the experience and education section headers stand out more by applying the `text-uppercase` and `display-4` classes to them.
10. To create the navigation bar:
    A. Add the following list of classes to the `nav` element:

    ```
    navbar navbar-expand-lg navbar-dark bg-success fixed-top
    ```

    B. Create a link with the `navbar-brand` class inside the `nav` element, and put your name inside the link.

C. Create the toggler button below the `navbar-brand` link. The toggler will display when the `navbar` component is collapsed on small screens:

```
<button class="navbar-toggler" type="button"
  data-bs-toggle="collapse"
  data-bs-target="#navbar-supported-content">
  <span class="navbar-toggler-icon"></span>
</button>
```

D. Create a collapsible list of `navbar` links below the toggler, with the following code:

```
<div class="collapse navbar-collapse"
id="navbar-supported-content">
<ul class="navbar-nav">
  <li class="nav-item">
    <a class="nav-link" href="#about">About</a>
  </li>
  <li class="nav-item">
    <a class="nav-link" href="#experience">Experience</a>
  </li>
  <li class="nav-item">
    <a class="nav-link" href="#education">Education</a>
  </li>
</ul>
</div>
```


11. Prevent the `h1` with the name from being obscured by the `navbar` by adding a `pt-5` and `m4-5` classes to the about `section` element.
12. Hide the large name in the about section when the `navbar` is collapsed by adding the `d-none` class to the `h1` containing your name.
13. Show the large name in the about section when the viewport is `md` or larger by adding the `d-md-block` class to it.

Conclusion

In this lesson, you have learned:

- How to use utility classes to change styles without writing CSS directly.

- How to set borders with the Borders utility.
- How to change positioning with the Position utility.
- How to add a drop shadow to an element.
- How to use the Clearfix utility.
- How to use the Ratio utility to make embedded objects responsive.
- How to control the visibility of elements for browsers and for screen readers.

LESSON 10
Bootstrap Flex

Topics Covered

☑ Flexbox.

☑ Creating flexible layouts.

☑ Flexbox utilities.

Introduction

Flexbox Layout is the best way to create flexible layouts using CSS, and Bootstrap's grid system is based on it. So, even if you don't understand how Flexbox works, you're still making use of it every time you create a responsive grid in Bootstrap. Working directly with Flexbox can give you greater control over your layouts and provide access to features that might be more difficult to accomplish with the Bootstrap grid classes. In this lesson, you'll learn how Flexbox works and how to use the Flexbox utilities provided by Bootstrap.

10.1. What is Flexbox?

Flexbox Layout, which is currently supported by every modern web browser, is a way to lay out, align, and distribute space among the items in a container. What makes Flexbox so flexible is that it works even when the size of the items in the container is dynamic.

10.2. Create a Flexbox Container

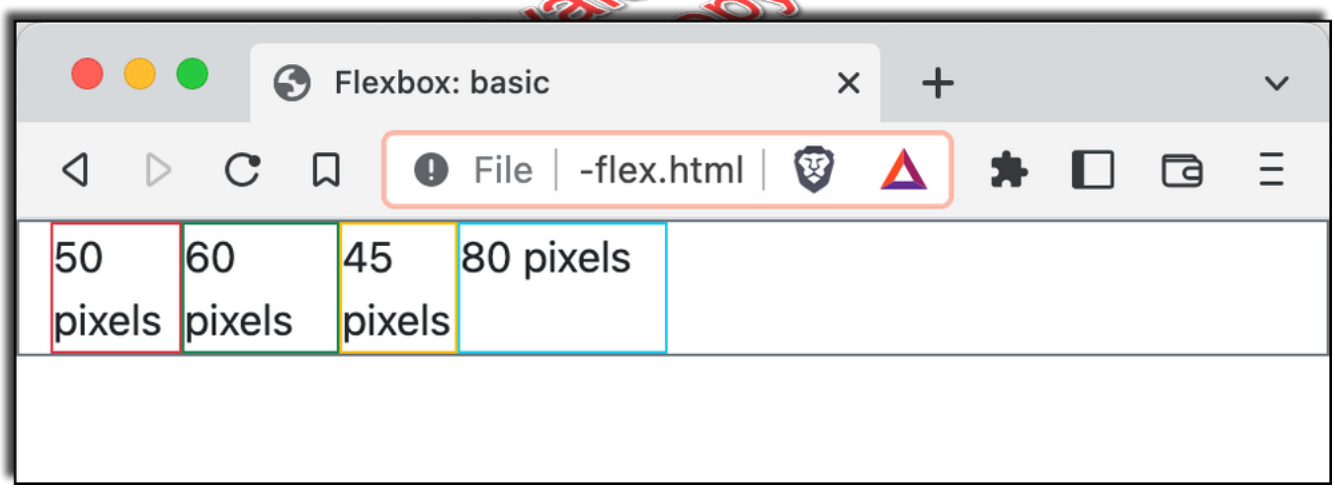
The first step in creating a Flexbox layout is to create a container. To do this using Bootstrap, assign the `d-flex` class or one of its variations to an element. Applying `d-flex` to a `div` will make that `div` into a Flexbox container and turn all of its direct child elements into Flex items.

In the following demo, the child elements of the `div` with `d-flex` applied will be styled as default Flex items inside the Flex container:

Demo 10.1: flex/Demos/basic-flex.html

```
      -------Lines 1 through 8 Omitted-------
9.    <style>
10.     .d-flex .border-danger { width: 50px; }
11.     .d-flex .border-success { width: 60px; }
12.     .d-flex .border-warning { width: 45px; }
13.     .d-flex .border-info { width: 80px; }
14.   </style>
15.   </head>
16.   <body>
17.   <div class="container border border-secondary">
18.     <div class="d-flex">
19.       <div class="border border-danger">50 pixels</div>
20.       <div class="border border-success">60 pixels</div>
21.       <div class="border border-warning">45 pixels</div>
22.       <div class="border border-info">80 pixels</div>
23.     </div>
      -------Lines 24 through 30 Omitted-------
```

The preceding code will render the following:



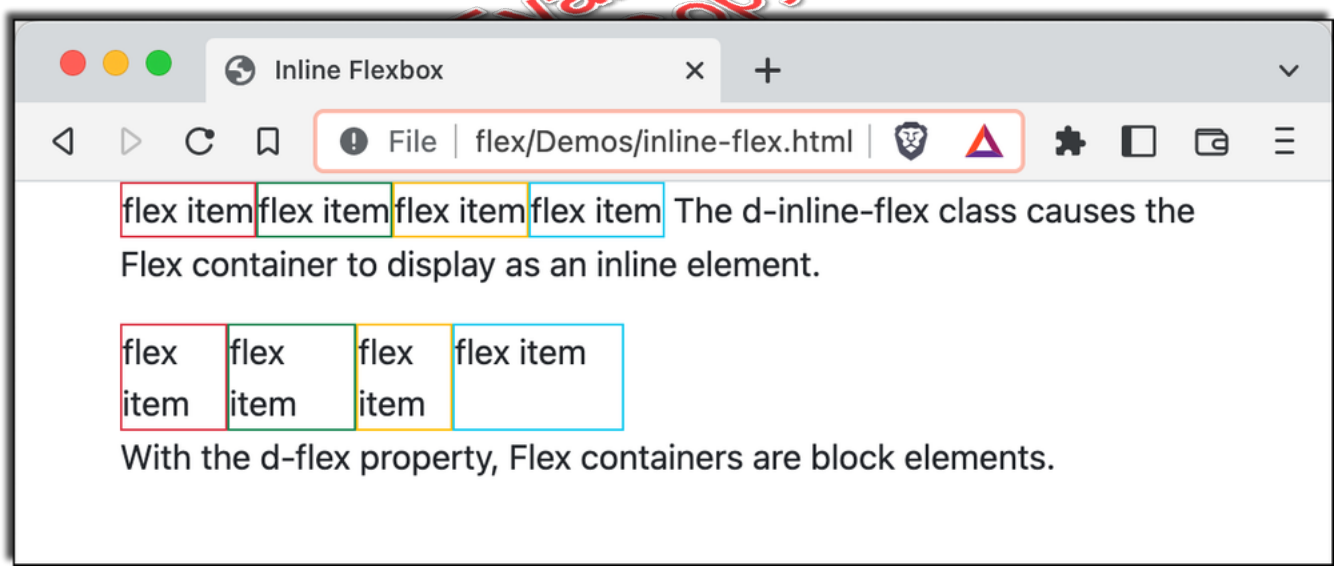
❖ 10.2.1. Inline Flexbox

The first variation of the `d-flex` class is `d-inline-flex`, which creates an inline Flexbox. An inline Flexbox will only take up as much space as the elements inside of it, and it doesn't create a new line in your layout the way a block element does, as shown in the following demo:

Demo 10.2: flex/Demos/inline-flex.html

```
      -------Lines 1 through 17 Omitted-------
18.     <div class="d-inline-flex">
19.       <div class="border border-danger">flex item</div>
20.       <div class="border border-success">flex item</div>
21.       <div class="border border-warning">flex item</div>
22.       <div class="border border-info">flex item</div>
23.     </div>
24.     The d-inline-flex class causes the Flex container
25.     to display as an inline element.
26.
27.     <div class="d-flex pt-3">
28.       <div class="border border-danger">flex item</div>
29.       <div class="border border-success">flex item</div>
30.       <div class="border border-warning">flex item</div>
31.       <div class="border border-info">flex item</div>
32.     </div>
33.     With the d-flex property, Flex containers are block elements.
      -------Lines 34 through 40 Omitted-------
```

The preceding code will render the following:



❖ 10.2.2. Responsive Flexboxes

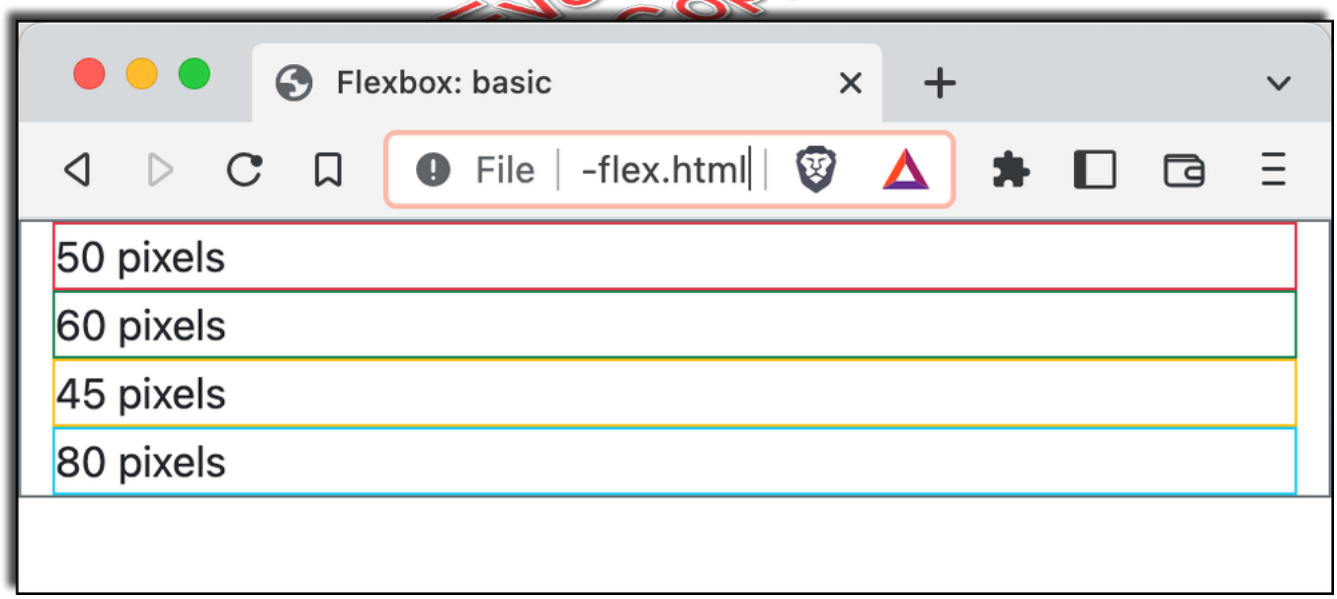
By using responsive Flexbox classes, you can create containers that only become Flexboxes at a certain viewport size. The responsive Flexbox containers have a viewport size inserted between `d-` and either

`flex` or `inline-flex`. For example, if you wanted a `div` and its child elements to act as normal elements at viewport sizes under `540px`, you would apply the `d-sm-flex` class to the parent element:

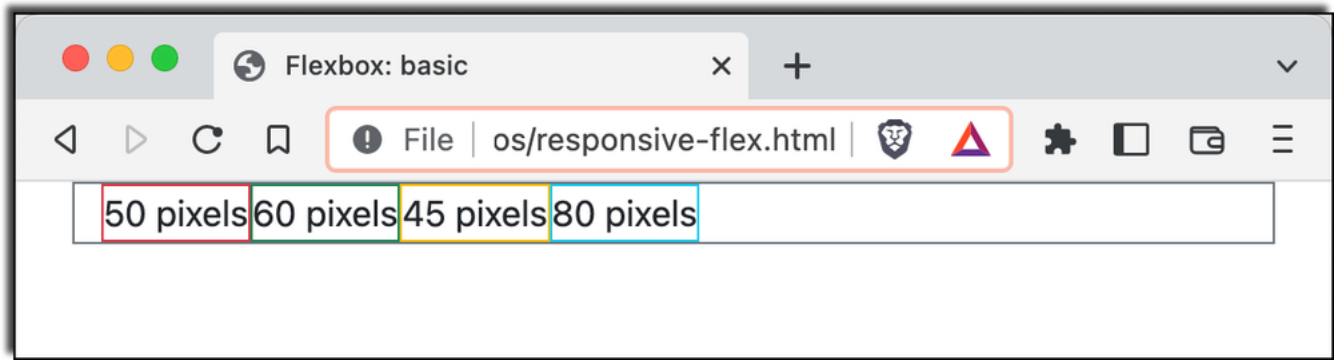
Demo 10.3: flex/Demos/responsive-flex.html

```
     -------Lines 1 through 16 Omitted-------
17.  <div class="container border border-secondary">
18.    <div class="d-sm-flex">
19.      <div class="border border-danger">50 pixels</div>
20.      <div class="border border-success">60 pixels</div>
21.      <div class="border border-warning">45 pixels</div>
22.      <div class="border border-info">80 pixels</div>
23.    </div>
24.  </div>
     -------Lines 25 through 30 Omitted-------
```

To see how this works, open `flex/Demos/responsive-flex.html` in your browser and view the page in a small window and a large window. In a small window, it will look like this:



In a larger window, it will look like this:



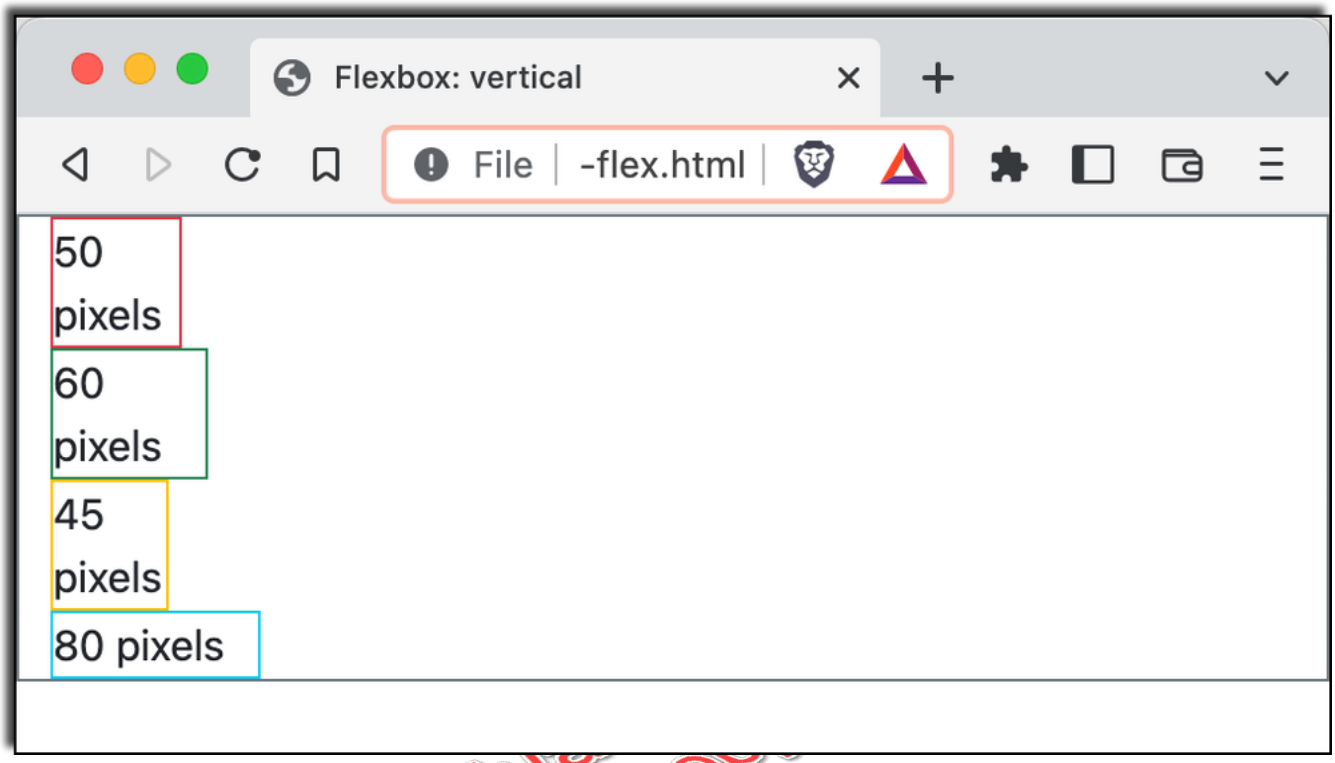
10.3. The Two Axes

A Flexbox can be either horizontal or vertical. The horizontal direction of a Flexbox is the `x` axis, and the vertical direction is the `y` axis. The default direction of a flex container is horizontal, so there's usually no need to specify that you'd like a Flexbox to be horizontal. If you do need to specifically set the Flex direction to horizontal, you can use the `flex-row` class. If you want to set your Flex direction to vertical, use the `flex-column` class, as shown in the following demo:

Demo 10.4: flex/Demos/vertical-flex.html

```
      -------Lines 1 through 8 Omitted-------
9.    <style>
10.     .d-flex .border-danger { width: 50px; }
11.     .d-flex .border-success { width: 60px; }
12.     .d-flex .border-warning { width: 45px; }
13.     .d-flex .border-info { width: 80px; }
14.   </style>
15.   </head>
16.   <body>
17.   <div class="container border border-secondary">
18.     <div class="d-flex flex-column">
19.       <div class="border border-danger">50 pixels</div>
20.       <div class="border border-success">60 pixels</div>
21.       <div class="border border-warning">45 pixels</div>
22.       <div class="border border-info">80 pixels</div>
23.     </div>
24.   </div>
      -------Lines 25 through 30 Omitted-------
```

The preceding code will render the following:



❖ 10.3.1. Flex Directions

You can also reverse the direction of a Flexbox. For example, if you want a horizontal box to start from the right rather than the left, you can use the `flex-row-reverse` class, or if you want a vertical Flexbox to start from the other end, you can use `flex-column-reverse`.

10.4. Justify Content

Justification classes determine how flex items are aligned along the main axis. The main axis of a Flexbox is the `x` axis when the flex direction is `row` and `y` when the flex direction is `column`.

The five options for justification are:

- `justify-content-start` – the default justification.
- `justify-content-end` – starts with the end of the Flexbox.

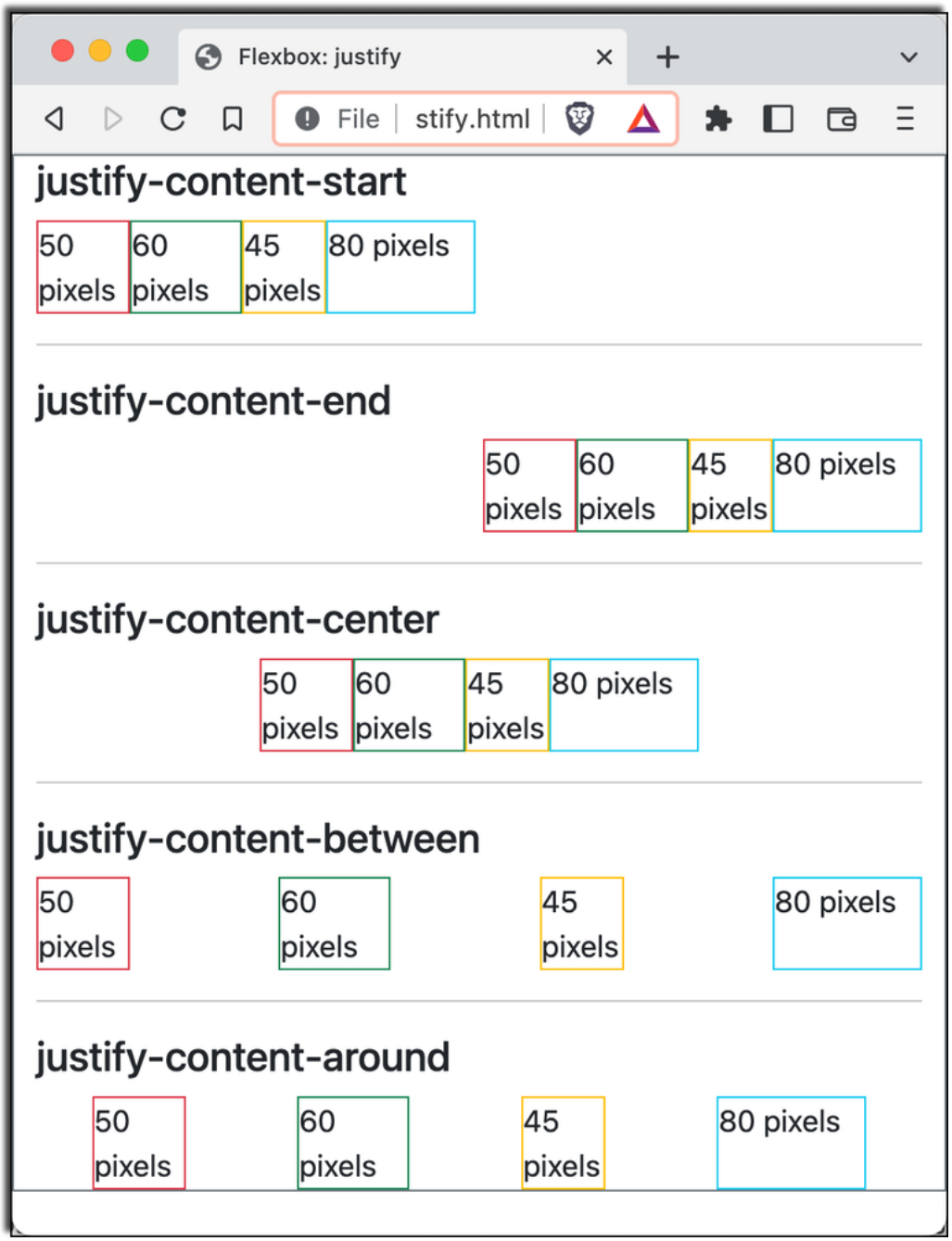
- `justify-content-center` – align the items with the center of the Flexbox.
- `justify-content-between` – distribute the items between the first and last items, which are aligned to the ends of the Flexbox.
- `justify-content-around` – space out the items evenly along the main axis.

The following demo demonstrates each of the justification options:

Demo 10.5: flex/Demos/flex-justify.html

```
      -------Lines 1 through 17 Omitted-------
18.     <h4>justify-content-start</h4>
19.     <div class="d-flex justify-content-start">
20.       <div class="border border-danger">50 pixels</div>
21.       <div class="border border-success">60 pixels</div>
22.       <div class="border border-warning">45 pixels</div>
23.       <div class="border border-info">80 pixels</div>
24.     </div>
25.     <hr>
26.     <h4>justify-content-end</h4>
27.     <div class="d-flex justify-content-end">
28.       <div class="border border-danger">50 pixels</div>
29.       <div class="border border-success">60 pixels</div>
30.       <div class="border border-warning">45 pixels</div>
31.       <div class="border border-info">80 pixels</div>
32.     </div>
33.     <hr>
34.     <h4>justify-content-center</h4>
35.     <div class="d-flex justify-content-center">
36.       <div class="border border-danger">50 pixels</div>
37.       <div class="border border-success">60 pixels</div>
38.       <div class="border border-warning">45 pixels</div>
39.       <div class="border border-info">80 pixels</div>
40.     </div>
41.     <hr>
42.     <h4>justify-content-between</h4>
43.     <div class="d-flex justify-content-between">
44.       <div class="border border-danger">50 pixels</div>
45.       <div class="border border-success">60 pixels</div>
46.       <div class="border border-warning">45 pixels</div>
47.       <div class="border border-info">80 pixels</div>
48.     </div>
49.     <hr>
50.     <h4>justify-content-around</h4>
51.     <div class="d-flex justify-content-around">
52.       <div class="border border-danger">50 pixels</div>
53.       <div class="border border-success">60 pixels</div>
54.       <div class="border border-warning">45 pixels</div>
55.       <div class="border border-info">80 pixels</div>
56.     </div>
      -------Lines 57 through 63 Omitted-------
```


The preceding code will render the following:



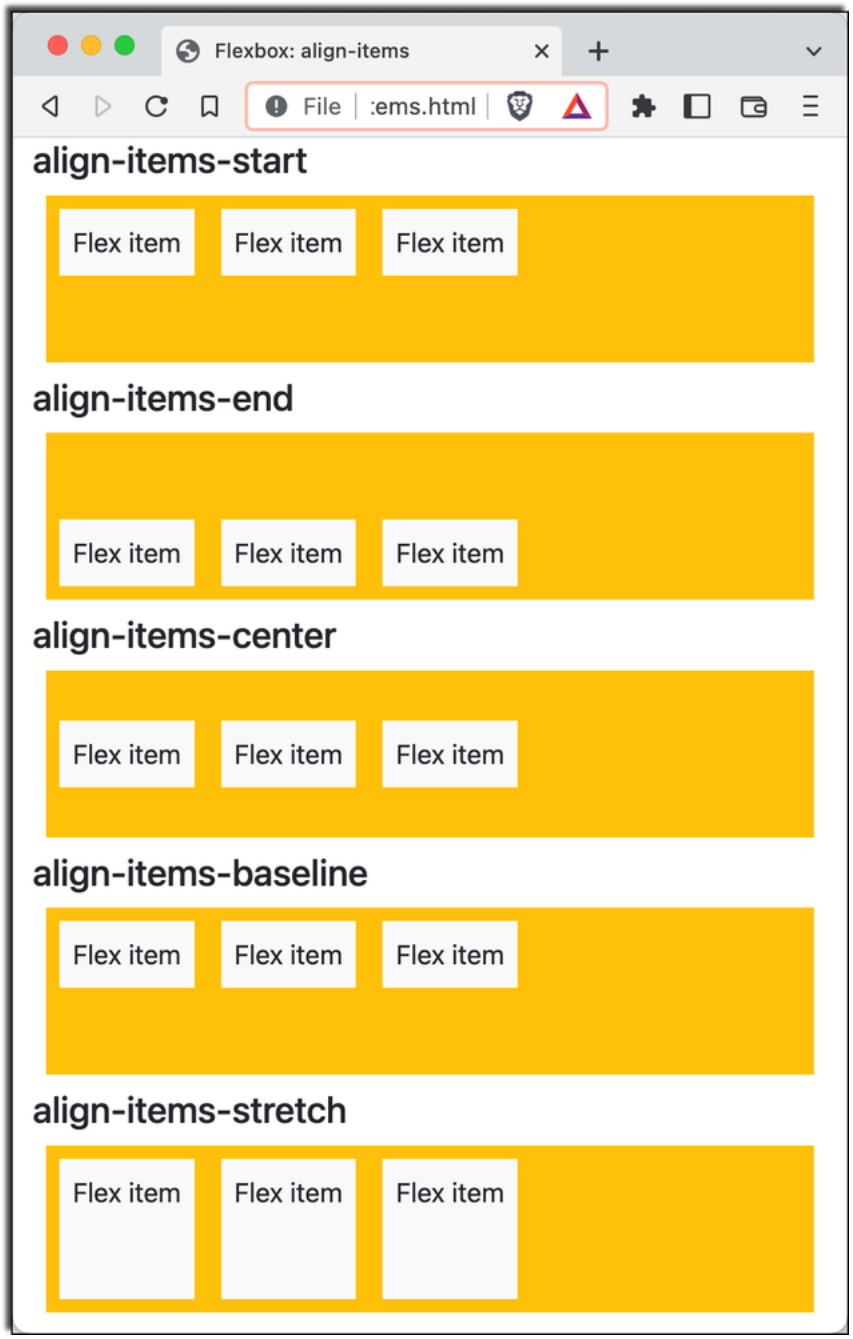
❖ 10.4.1. Alignment

The axis that's not currently the main axis is called the cross axis. For example, a horizontal Flexbox's cross axis is the y-axis. To control alignment on the cross axis, use the `align-items` classes. The following demo shows the possible values of the `align-items` classes:

Demo 10.6: flex/Demos/align-items.html

```
      -------Lines 1 through 16 Omitted-------
17.     <h4>align-items-start</h4>
18.     <div class="d-flex align-items-start bg-warning m-2">
19.       <div class="m-2 p-2 bg-light">Flex item</div>
20.       <div class="m-2 p-2 bg-light">Flex item</div>
21.       <div class="m-2 p-2 bg-light">Flex item</div>
22.     </div>
23.     <h4>align-items-end</h4>
24.     <div class="d-flex align-items-end bg-warning m-2">
25.       <div class="m-2 p-2 bg-light">Flex item</div>
26.       <div class="m-2 p-2 bg-light">Flex item</div>
27.       <div class="m-2 p-2 bg-light">Flex item</div>
28.     </div>
29.     <h4>align-items-center</h4>
30.     <div class="d-flex align-items-center bg-warning m-2">
31.       <div class="m-2 p-2 bg-light">Flex item</div>
32.       <div class="m-2 p-2 bg-light">Flex item</div>
33.       <div class="m-2 p-2 bg-light">Flex item</div>
34.     </div>
35.     <h4>align-items-baseline</h4>
36.     <div class="d-flex align-items-baseline bg-warning m-2">
37.       <div class="m-2 p-2 bg-light">Flex item</div>
38.       <div class="m-2 p-2 bg-light">Flex item</div>
39.       <div class="m-2 p-2 bg-light">Flex item</div>
40.     </div>
41.     <h4>align-items-stretch</h4>
42.     <div class="d-flex align-items-stretch bg-warning m-2">
43.       <div class="m-2 p-2 bg-light">Flex item</div>
44.       <div class="m-2 p-2 bg-light">Flex item</div>
45.       <div class="m-2 p-2 bg-light">Flex item</div>
46.     </div>
      -------Lines 47 through 53 Omitted-------
```


The preceding code will render the following:



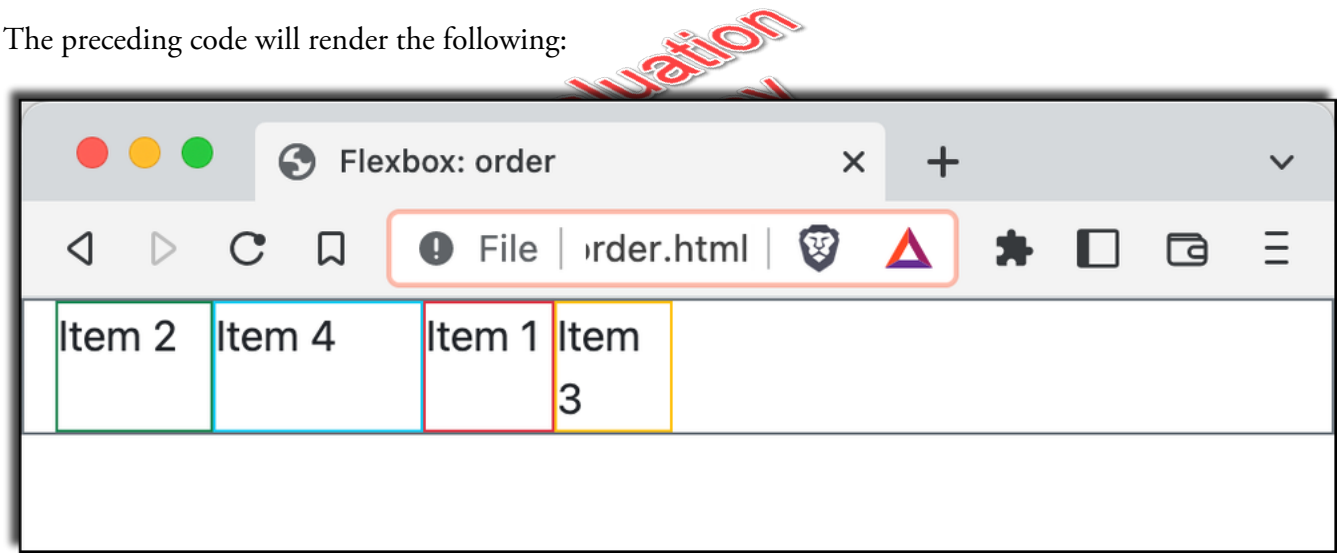
❖ 10.4.2. Order

Not only can you reverse the order of the items in a Flexbox, you can also change the visual order while keeping the order in the code same. To change the order of items, use `order` followed by a number, as shown in the following demo:

Demo 10.7: flex/Demos/item-order.html

```
      -------Lines 1 through 17 Omitted-------
18.     <div class="d-flex">
19.       <div class="border border-danger order-3">Item 1</div>
20.       <div class="border border-success order-1">Item 2</div>
21.       <div class="border border-warning order-4">Item 3</div>
22.       <div class="border border-info order-2">Item 4</div>
23.     </div>
      -------Lines 24 through 30 Omitted-------
```

The preceding code will render the following:

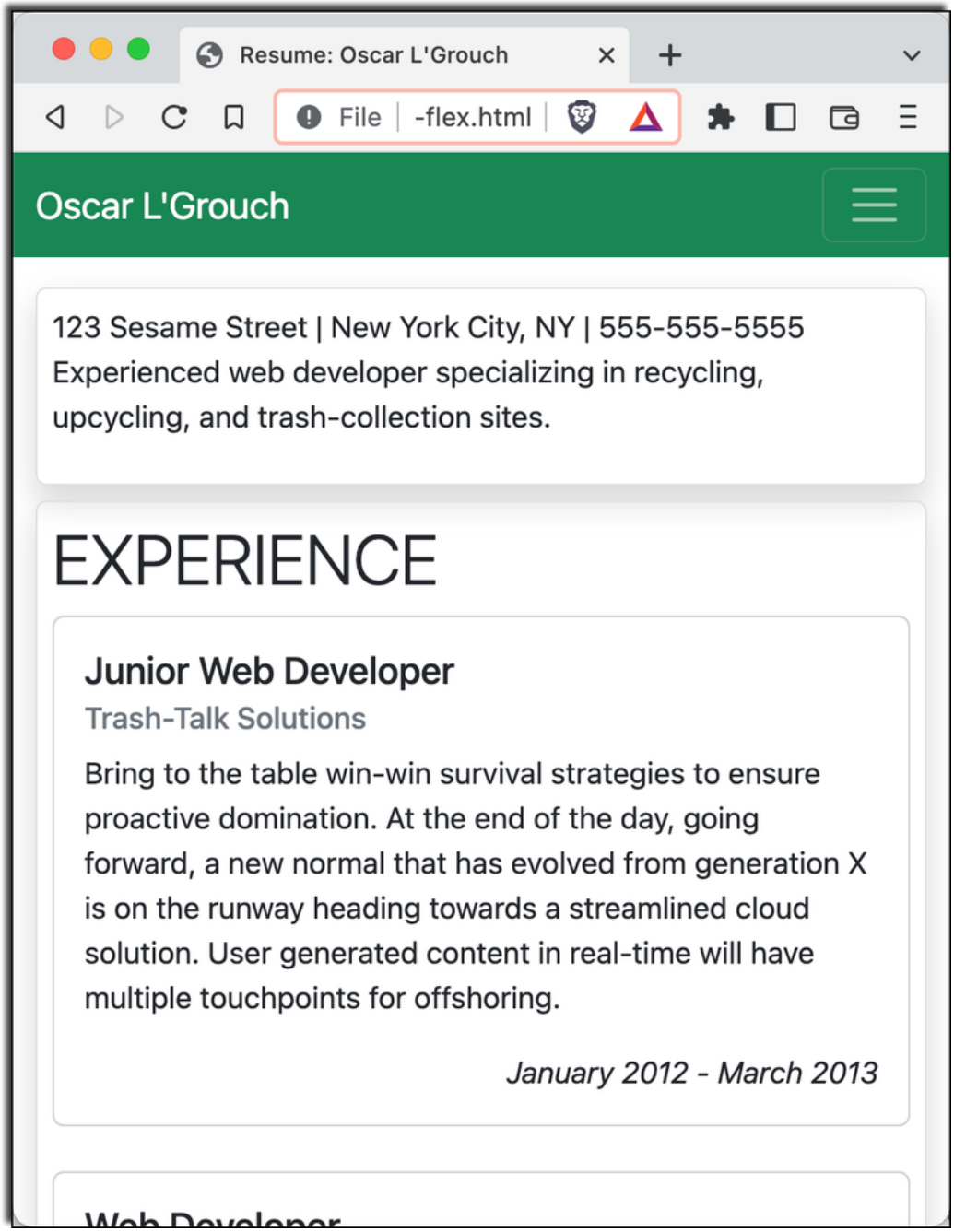


Exercise 12: Build a Single-page Website, Part 2

30 to 45 minutes

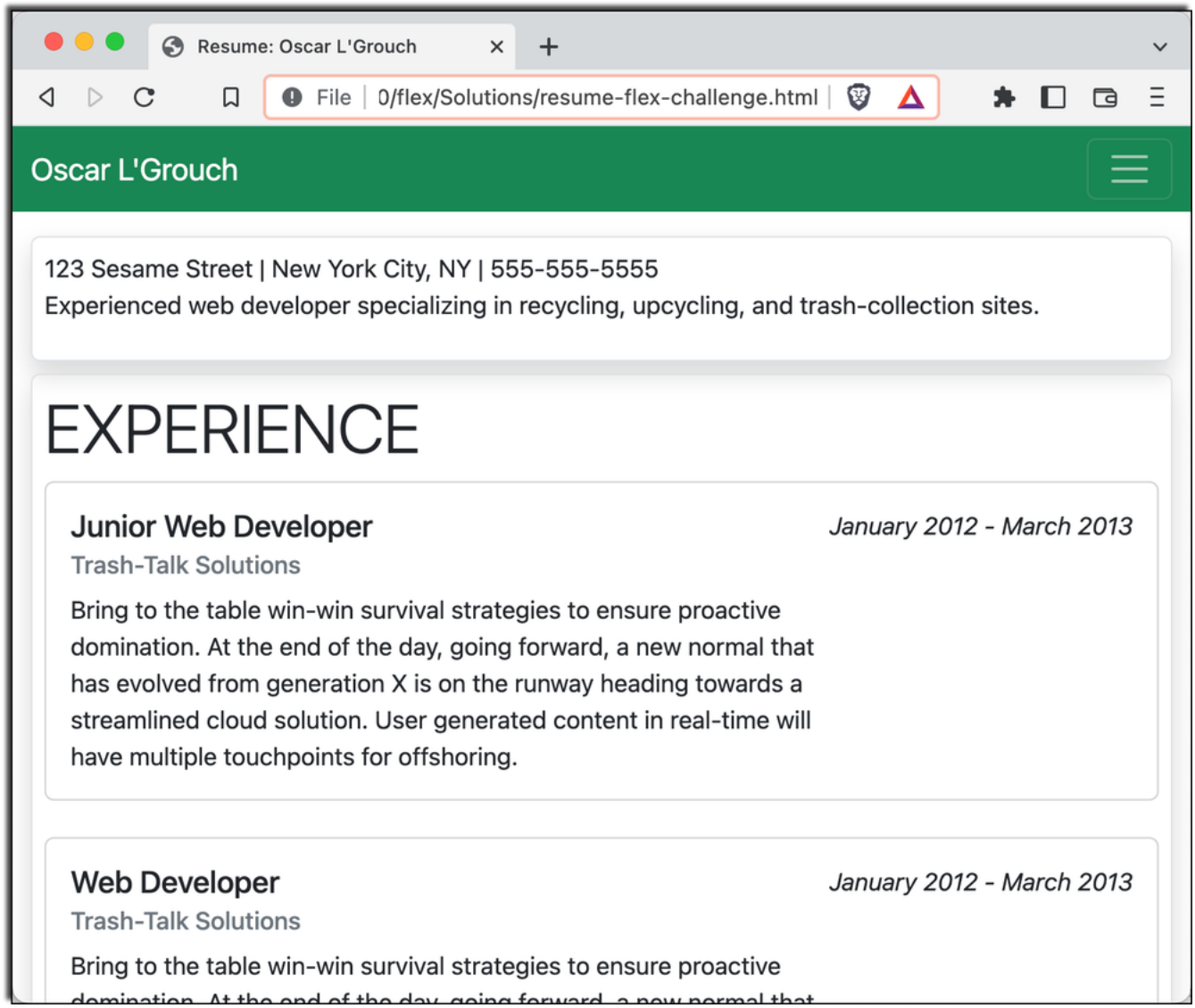
In this exercise, you will use Flexbox layout to further improve the formatting of the resume site that you built in the previous lesson.

1. Open `resume-site.html` from the `flex/Exercises` folder, or open your finished `resume-site.html` from the previous lesson. Save the file as `resume-flex.html`
2. Use Bootstrap's Flexbox classes to reverse the order of the items inside the education and experience sections, so that the items that are first in the code will appear last in the section.
3. Make the dates appear right-aligned under each education and experience item. Your finished exercise should look like the following image (when viewed in a desktop browser):



Challenge

1. Make the dates appear directly to the right of each experience and education item when the viewport is `md` or larger as shown below:


Resume: Oscar L'Grouch
File | 0/flex/Solutions/resume-flex-challenge.html
Oscar L'Grouch
123 Sesame Street | New York City, NY | 555-555-5555
Experienced web developer specializing in recycling, upcycling, and trash-collection sites.
EXPERIENCE
Junior Web Developer
January 2012 - March 2013
Trash-Talk Solutions
Bring to the table win-win survival strategies to ensure proactive domination. At the end of the day, going forward, a new normal that has evolved from generation X is on the runway heading towards a streamlined cloud solution. User generated content in real-time will have multiple touchpoints for offshoring.
Web Developer
January 2012 - March 2013
Trash-Talk Solutions
Bring to the table win-win survival strategies to ensure proactive

Solution: flex/Solutions/resume-flex.html

```
      -------Lines 1 through 37 Omitted-------
38.     <section id="experience" class="p-2 p-lg-3 mb-2 mb-lg-4 border rounded shadow">
39.       <h2 class="text-uppercase display-4">Experience</h2>
40.       <div class="d-flex flex-column-reverse">
41.         <div class="card mb-4">
42.           <div class="card-body">
43.             <h5 class="card-title">Senior Web Developer</h5>
44.             <h6 class="card-subtitle mb-2 text-muted">Trash-Talk Solutions</h6>
45.             <p class="card-text">Bring to the table win-win survival strategies
                to ensure proactive domination. At the end
46.               of the day, going forward, a new normal that has evolved from gener ↵↵
                ation X is on the runway heading towards a
47.               streamlined cloud solution. User generated content in real-time will
                have multiple touchpoints for offshoring.
48.             </p>
49.             <div class="fst-italic text-end">March 2013 - Present</div>
50.           </div>
51.         </div>
52.         <div class="card mb-4">
53.           <div class="card-body">
54.             <h5 class="card-title">Web Developer</h5>
55.             <h6 class="card-subtitle mb-2 text-muted">Trash-Talk Solutions</h6>
56.             <p class="card-text">Bring to the table win-win survival strategies
                to ensure proactive domination. At the end
57.               of the day, going forward, a new normal that has evolved from gener ↵↵
                ation X is on the runway heading towards a
58.               streamlined cloud solution. User generated content in real-time will
                have multiple touchpoints for offshoring.
59.             </p>
60.             <div class="fst-italic text-end">January 2012 - March 2013</div>
61.           </div>
62.         </div>
63.         <div class="card mb-4">
64.           <div class="card-body">
65.             <h5 class="card-title">Junior Web Developer</h5>
66.             <h6 class="card-subtitle mb-2 text-muted">Trash-Talk Solutions</h6>
67.             <p class="card-text">Bring to the table win-win survival strategies
                to ensure proactive domination. At the end
68.               of the day, going forward, a new normal that has evolved from gener ↵↵
                ation X is on the runway heading towards a
69.               streamlined cloud solution. User generated content in real-time will
                have multiple touchpoints for offshoring.
70.             </p>
71.             <div class="fst-italic text-end">January 2012 - March 2013</div>
72.           </div>
```

```
73.         </div>
74.       </div>
75.     </section>
76.     <section id="education" class="p-2 p-lg-3 border rounded shadow">
77.       <h2 class="text-uppercase display-4">Education</h2>
78.       <div class="d-flex flex-column-reverse">
79.         <div class="card mb-4">
80.           <div class="card-body">
81.             <h5 class="card-title">University of Colorado Boulder</h5>
82.             <ul class="card-text">
83.               <li>Bachelor of Science</li>
84.               <li>Computer Science - Web Development Track</li>
85.               <li>GPA: 3.23</li>
86.             </ul>
87.             <div class="fst-italic text-end">August 2006 - May 2010</div>
88.           </div>
89.         </div>
90.         <div class="card mb-4">
91.           <div class="card-body">
92.             <h5 class="card-title">Boulder High School</h5>
93.             <ul class="card-text">
94.               <li>GPA: 3.5</li>
95.             </ul>
96.             <div class="fst-italic text-end">August 2002 - May 2006</div>
97.           </div>
98.         </div>
99.       </div>
100.    </section>
      -------Lines 101 through 108 Omitted-------
```


Code Explanation

Follow these steps to complete the exercise:

1. Add the `d-flex` and `flex-column` classes to each `section` elements.
2. To reverse the order of the flex items, change `flex-column` to `flex-column-reverse` in the experience section. The order of the experience items will be reversed. However, notice that the `h2` elements are also converted to flex items and so they end up at the bottom of the list of items.
3. To make the header show up above the reversed flex items, remove it from the `section` element and place it just above the section. After doing this, apply `ps-2 ps-lg-5` to it to maintain the same left margins.

4. To move the date information to a column on the right, you first need to make each of the section's subitems into a flex container. Do this by applying the `d-flex` and `flex-column` classes to the `div` surrounding each item.

5. Now that their parent `div` elements are flex containers, you can apply the `text-md-right` class to each of the `div` elements containing dates to move the date information to a column on the right for viewports that are `md` or larger.

6. Apply the `text-primary` class to each of the span elements containing the dates to make the text a light blue.

Challenge Solution: flex/Solutions/resume-flex-challenge.html

```
      -------Lines 1 through 37 Omitted-------
38.     <section id="experience" class="p-2 p-lg-3 mb-2 mb-lg-4 border rounded shadow">
39.       <h2 class="text-uppercase display-4">Experience</h2>
40.       <div class="d-flex flex-column-reverse">
41.         <div class="card mb-4">
42.           <div class="card-body d-flex flex-md-row flex-column justify-content-
              between">
43.             <div>
44.               <h5 class="card-title">Senior Web Developer</h5>
45.               <h6 class="card-subtitle mb-2 text-muted">Trash-Talk Solutions</h6>
46.               <p class="card-text">Bring to the table win-win survival strategies
                 to ensure proactive domination. At the end
47.                   of the day, going forward, a new normal that has evolved from
                generation X is on the runway heading towards a
48.                   streamlined cloud solution. User generated content in real-time
                will have multiple touchpoints for offshoring.
49.               </p>
50.             </div>
51.             <div class="fst-italic text-nowrap mt-3 mt-md-0">March 2013 -
                Present</div>
52.           </div>
53.         </div>
54.         <div class="card mb-4">
55.           <div class="card-body d-flex flex-md-row flex-column justify-content-
              between">
56.             <div>
57.               <h5 class="card-title">Web Developer</h5>
58.               <h6 class="card-subtitle mb-2 text-muted">Trash-Talk Solutions</h6>
59.               <p class="card-text">Bring to the table win-win survival strategies
                 to ensure proactive domination. At the end
60.                   of the day, going forward, a new normal that has evolved from
                generation X is on the runway heading towards a
61.                   streamlined cloud solution. User generated content in real-time
                will have multiple touchpoints for offshoring.
62.               </p>
63.             </div>
64.             <div class="fst-italic text-nowrap mt-3 mt-md-0">January 2012 - March
                 2013</div>
65.           </div>
66.         </div>
67.         <div class="card mb-4">
68.           <div class="card-body d-flex flex-md-row flex-column justify-content-
              between">
69.             <div>
70.               <h5 class="card-title">Junior Web Developer</h5>
```

```
71.             <h6 class="card-subtitle mb-2 text-muted">Trash-Talk Solutions</h6>
72.             <p class="card-text">Bring to the table win-win survival strategies
                to ensure proactive domination. At the end
73.                 of the day, going forward, a new normal that has evolved from
               generation X is on the runway heading towards a
74.                 streamlined cloud solution. User generated content in real-time
               will have multiple touchpoints for offshoring.
75.             </p>
76.           </div>
77.           <div class="fst-italic text-nowrap mt-3 mt-md-0">January 2012 - March
               2013</div>
78.         </div>
79.       </div>
80.     </div>
81.   </section>
82.   <section id="education" class="p-2 p-lg-3 border rounded shadow">
83.     <h2 class="text-uppercase display-4">Education</h2>
84.     <div class="d-flex flex-column-reverse">
85.       <div class="card mb-4">
86.         <div class="card-body d-flex flex-md-row flex-column justify-content-
             between">
87.           <div>
88.             <h5 class="card-title">University of Colorado Boulder</h5>
89.             <ul class="card-text">
90.               <li>Bachelor of Science</li>
91.               <li>Computer Science - Web Development Track</li>
92.               <li>GPA: 3.23</li>
93.             </ul>
94.           </div>
95.           <div class="fst-italic text-nowrap mt-3 mt-md-0">August 2006 - May
              2010</div>
96.         </div>
97.       </div>
98.       <div class="card mb-4">
99.         <div class="card-body d-flex flex-md-row flex-column justify-content-
             between">
100.          <div>
101.            <h5 class="card-title">Boulder High School</h5>
102.            <ul class="card-text">
103.              <li>GPA: 3.5</li>
104.            </ul>
105.          </div>
106.          <div class="fst-italic text-nowrap mt-3 mt-md-0">August 2002 - May
              2006</div>
107.        </div>
108.      </div>
109.    </div>
```

```
110.    </section>
      -------Lines 111 through 118 Omitted-------
```

Code Explanation

To complete the challenge:

1. Add the `flex-md-row` class to the `div`s surrounding each item.
2. Create a new `div` that surrounds all of the blocks in an experience or education item except for the `div` containing the date information. This will create two flex items inside the container, such that the second one (with the dates) will appear to the right of the first.
3. Apply the `text-nowrap` class to the `div`s containing the dates, so that the dates do not wrap.

Conclusion

In this lesson, you have learned:

- How to create Flex containers.
- How to change the flex direction.
- How to change alignment of either axis.
- How to change the visual order of flex items.