**310** *Section 1 A crash course in HTML and CSS*

Finally, I added the media queries that change the appearance of the page for different screens sizes. For a tablet in portrait view, I simply reduced some of the fonts so the elements fit better on the screen. For a mobile phone in landscape view, though, I made a number of changes to the styles to implement the changes to the appearance of the page that you saw in figure 8-12.

First, I changed the display property of the navigation menu to "none" to hide it, and I changed the display property of the mobile menu to "block" to display it. This property is set to "none" in the styles outside the media queries so it's not displayed on larger screens. I also changed the background-color property of the SlickNav menu using the .slicknav\_menu class. Note that I included the !important rule for this property so it will override the property set in the slicknay.css file.

Next, I changed the styles for the body so it will fill the screen. To do that, I set the width to 100%, I removed the margins, and I removed the border.

To display the header as shown in figure 8-12, I removed the floating from the image so it's displayed above the two headings. I also aligned the text within the header so the image and headings are centered. Then, I reduced the font-sizes for the headings and I adjusted the margins. To change the layout of the footer, I centered and removed the right margin from the paragraph it contains.

*Chapter 8 How to use Responsive Web Design* **311**

**The CSS that controls the page layout Page 2**

/\* changes to the styles for the sidebar \*/ aside {

width: 35.35353%;

/\* 350 990 x 100 \*/

/\* 20 990 x 100 \*/

/\* 20 990 x 100 \*/

/\* 20 16 \*/

/\* base font size \*/

/\* 14 16 \*/

/\* full width of body \*/

/\* 12 16 \*/

/\* 20 990 x 100 \*/

float: left;

padding: 2.0202%;

background-color: #ffebc6; margin: 1.5em 2.0202% 0 0;

}

aside h2 { font-size: 1.25em; }

aside h3 { font-size: lem;

aside li { font-size: .875em; }

/\* changes to the styles for the footer \*/ footer { width: 100%; ... }

footer p {

font-size: .75em;

text-align: right;

margin-right: 2.0202%;

}

/\* hide the mobile menu initially \*/ #mobile\_menu {

display: none;

}

|  |  |
| --- | --- |
| **/\* tablet portrait to standard** 960 \*/  @media only screen and (max-width: 959px) {  #nav\_menu ul li a { font-size: .875em; section hl, article hl { font-size: 1.5em; } section h2, aside h2 { font-size: 1.125em;  ) | /\* 14 .4 16 \*/  /\* 24 16 \*/  /\* 18 16 \*/ |

/\* mobile landscape to tablet portrait \*/

@media only screen and (max-width: 767px) {

#nav\_menu { display: none; }

#mobile\_menu { display: block; }

.slicknav menu { background-color: #facd8a !important; } body {

width: 100%; /\* full width of screen \*/

margin: 0; /\* no margins \*/

border: none; /\* no border \*/

|  |  |
| --- | --- |
| }  header, footer p { text-align: center; } footer p { margin-right: 0; }  header h2 {  font-size: 1.625em;  margin: .4em 0 .25em 0; | /\* 26 16 \*/ |

}

header h3 {

font-size: lem; /\* base font size \*/

margin-left: 0;

}

header img { float: none;

Figure 8-14 The CSS for the web page (part 2 of 3)

**312** *Section 1 A crash course in HTML and CSS*

To display the content for the page in a single column, floating is removed from both the article and the aside. In addition, the margins and padding for these elements are adjusted so an equal amount of space is displayed at the left and right sides of the screen. Finally, the widths of these elements are set so they take up the rest of the screen. In other words, the widths of these elements are 100% minus the widths of any margins and padding for the elements.

The styles for the image within the article are also changed so the image floats to the left of the text. The width of this element is set to 50% of the width of the article, and the right margin is set to 2%. That means that the text for the article will take up the remaining 48%.

The media query for a phone in portrait view further reduces some of the fonts. In addition, it changes the image in the article so it takes up the full width of the article again. To do that, it removes the floating from the image, changes its width to 100%, and removes the right margin.

*Chapter 8 How to use Responsive Web Design* **313**

**The CSS that controls the page layout**

**Page 3**

article {

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 100 | - (4 | /\*  /\*  x | 100 - (2  20 4 990  2.0202) | x 2.0202) for padding  x 100 \*/  for padding and margins | \*/  \*/ |

width: 95.9596%;

float: none; margin-right: 2.0202%; padding-right: 0;

)

article img {

float: left; width: 50%; margin-right: 2%;

}

aside {

width: 91.9192%; /\*

float: none;

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| margin: 0 2.0202% 2.0202% 2.0202%;  }  } | /\* | 20 | 990 x | 100 | \*/ |
| /\* mobile portrait to mobile landscape \*/ |  |  |  |  |  |
| @media only screen and (max-width: 479px) |  |  |  |  |  |
| header h2 { font-size: 1.375em; } | /\* | 22 4. | 16 \*/ |  |  |
| header h3 { font-size: .8125em; | /\* | 13 | 16 \*/ |  |  |
| p { font-size: .875em; | /\* | 14 | 16 \*/ |  |  |
| article hl, aside h2 |  |  |  |  |  |
| font-size: lem; } | /\* | base | font | size | \*/ |
| aside li { font-size: .875em; } | /\* | 14 | 16 \*/ |  |  |
| ) |  |  |  |  |  |
| article img { |  |  |  |  |  |

float: none;

width: 100%;

margin-right: 0;

)

aside h3 { font-size: .9375em; } /\* 15 16 \*/

footer p { font-size: .6875em; /\* 11 16 \*/

)

Figure 8-14 The CSS for the web page (part 3 of 3)

**314** *Section 1 A crash course in HTML and CSS*

**Perspective**

The use of mobile devices has increased dramatically over the past few years. Because of that, it has become important to design websites that are easy to use from these devices. Although that often means more work, this can be a critical aspect of maintaining your presence in the business world.

In this chapter, you learned how to develop websites using Responsive Web Design. With RWD, you use fluid layouts to adjust the width of a page depending on the screen size. You use media queries to adjust the appearance of a page depending on the screen size. And you use scalable images to adjust the size of the images depending on the screen size. With these features, you can develop responsive websites that have the same look-and-feel on desktops, tablets, and smart phones.

**Terms**

Responsive Web Design (RWD) scalable image

responsive design viewport

fluid design scale

fixed layout @media selector

fluid layout breakpoint

media query

**Summary**

* *Responsive Web Design* refers to a technique that's used to create websites that adapt gracefully to any screen size. A *responsive design* includes fluid layouts, media queries, and scalable images.
* To create a web page with a *fluid layout,* you set the widths of the page and its main structural elements to percents so they increase and decrease depending on the width of the screen.
* To convert the fixed width for an element to a fluid width, you divide the width of the element in pixels by the width of its containing element in pixels and then multiply the result by 100 to get a percent.
* When you develop a responsive design, you should specify font sizes in ems or percents. To convert a font size from pixels to ems, you divide the size by 16 since that's the default size for most browsers. To convert the font size to a percent, you multiply the result of the division by 100.
* To create a *scalable image,* you remove the height and width property from the img element for the image, and you set the max-width property to the percent of its containing block you want it to fill.
* If you want to limit the size of an image to its native size, you can set the width property to a percent and then set the max-width property to the native width in pixels.

*Chapter 8 How to use Responsive Web Design* **315**

* A *media query* is defined by a CSS3 *@media selector* that uses conditional expressions to determine when the styles it contains are applied. You use media queries with **RWD** to change the appearance of a page for different screen sizes.
* When you use media queries with **RWD,** you can develop the design for larger devices first and then work your way down to smaller devices. Or, you can develop the design for the smallest device first and then work your way up to larger devices.
* The *viewport* on a mobile device determines the content that's displayed on the screen. When you use media queries, the viewport should be set so the page is displayed at its full size.
* SlickNav is a jQuery plugin that converts a standard navigation menu to a menu that's easier to use on smaller devices.
* An easy way to test a responsive design is to use the developer tools that are provided by most modern browsers. You can also use device emulators and browsers simulators, or you can deploy the website and then test it on various devices or use a web-based tool like ProtoFluid.

|  |
| --- |
| **Exercise 8-1 Convert the Town Hall home page**  **to use a responsive design** |

In this exercise, you'll convert the Town Hall home page that you worked on in exercise 7-1 so it uses a fluid layout, scalable images, and media queries. When you're through, the page should look like this in mobile phone portrait and landscape orientations:



**San Joaquin Valley Town Hall**

***Celebrating our 75th Year***

**MENU**

**Our Mission**

San Joaquin Valley Town Hall is a non-profit organization that is run by an all-volunteer board of directors. Our mission is to bring nationally and internationally renowned, thought-provoking speakers who inform, educate, and entertain our audience! As one or our members told us:

***"Each year 1 give a ticket package to*** *each of our family members. I think of*



**San Joaquin Valley Town
  
Hall**

***Celebrating our 75th Year***

**MENU**

Our Mission

San Joaquin Valley Town Hall is a non-profit organization that is run by an