Confederation
COLLEGE

CP 160 - Web Programming and Design

Lab 8: Responsive Layout

Hands-On Practice 1: Flexbox Image Gallery

You'll configure an image gallery with flexbox properties in this Hands-On Practice. Create a new folder called **ch8flex1**. Copy the starter1.html file into your ch8flex1 folder. Copy the following files from the starters folder into your ch8flex1 folder:

bird1.jpg, bird2.jpg, bird3.jpg, bird4.jpg, bird5.jpg, and bird6.jpg.

1. Launch a text editor and open the **starter1.html** file. Add the following HTML below the opening main tag to create a div assigned to the `gallery` id that contains six images.

```
<div id="gallery">

<img src="bird1.jpg" width="200" height="150" alt="Red Crested
Cardinal">

<img src="bird2.jpg" width="200" height="150" alt="Rose-Breasted
Grosbeak">

<img src="bird3.jpg" width="200" height="150" alt="Gyrfalcon">

<img src="bird4.jpg" width="200" height="150" alt="Rock Wren">

<img src="bird5.jpg" width="200" height="150" alt="Coopers Hawk">

<img src="bird6.jpg" width="200" height="150" alt="Immature Bald
Eagle">

</div>
```

   The div is the flex container. Each img element is a flex item in the flex container. Save the file with the name **index.html**.

2. Edit the index.html file and configure CSS between the style tags in the head section. Configure an id named `gallery`. Set the `display` property to `flex`, `flex-direction` property to `row`, `flex-wrap` to `wrap`, and `justify-content` to `space-around`. The code follows:

```
#gallery { display: flex;
           flex-direction: row;
           flex-wrap: wrap;
           justify-content: space-around; }
```

Save the file and test it in a browser. Your page should look similar to **Figure 1**. Observe that while the browser configured empty space between the flex items on each row (the main axis), there is no empty space in the vertical (cross axis) area between each row element.

3. Next, you'll configure the flex items to have a margin, which will force some empty space between the rows. Recall that a flex item is a child element of the flex container. In our page, each img element is a flex item. Edit the index.html file and code CSS above the closing style tag for the img selector that sets a 1em margin and a box-shadow.

```
img { margin: 1em;
      box-shadow: 10px 10px #777; }
```

Save the file and test in a browser. As you resize your browser smaller and larger, your page should be similar to **Figures 2, 3,** and **4**.

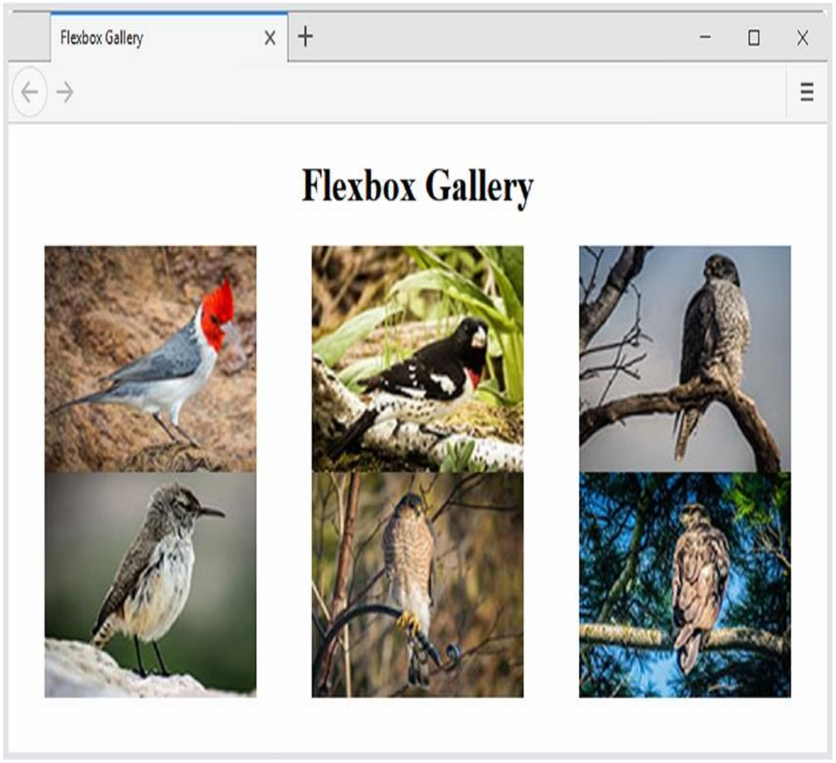


Figure 1 The first version of the gallery.

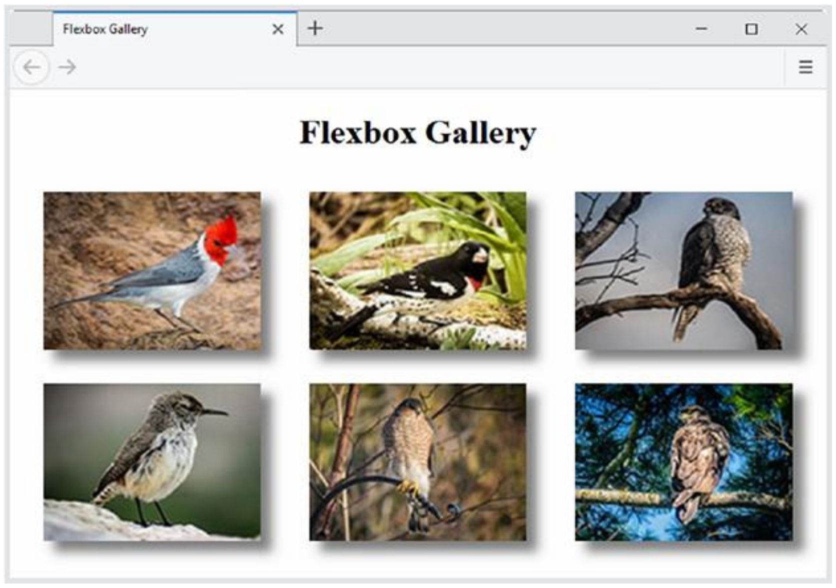


Figure 2 Two rows of flex items.

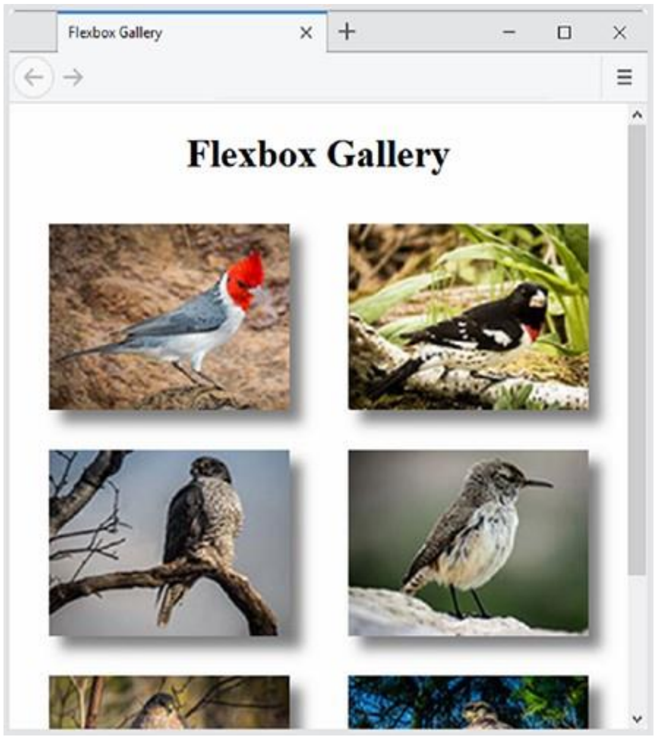


Figure 3 Each row now has two items.

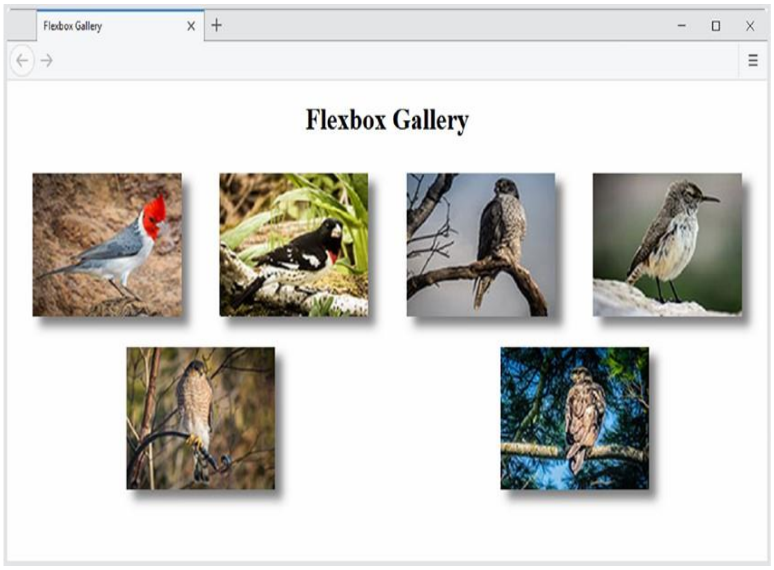


Figure 4 As the browser is resized, more items fit on the first row.

Hands-On Practice 2: Configure the image gallery grid

You'll explore two more ways to configure the image gallery grid displayed in **Figure 5** in this Hands On Practice. Create a new folder called **ch8grid1**. Copy the **starter3**.html file into your ch8grid1 folder. Copy the following files from the starters folder into your ch8grid1 folder: **bird1.jpg, bird2.jpg, bird3.jpg, bird4.jpg, bird5.jpg, and bird6.jpg**.

1. Launch a text editor and open the starter3.html file. Review the HTML and note that it contains a div assigned to the `gallery` id with six img elements for your image gallery. The div is the grid container. Each img element is a grid item since it is a child element of the div. Save the file with the name index.html.
2. Edit the index.html file and configure CSS between the style tags in the head section. Configure an id named `gallery`. Set the `display` property to `grid`. To divide the available browser space into three columns of 200 pixels each, set the `grid-template-columns` property to `repeat(3, 200px)`. To cause the browser to automatically generate rows as needed, set the `gridtemplate-rows` property to `auto`. Configure the base size of the gutters between row and column tracks by setting the `grid-gap` (and `gap`) properties to 2em. The CSS code follows:

```
#gallery { display: grid;
           grid-template-columns: repeat(3, 200px);
           grid-template-rows: auto;
           grid-gap: 2em; gap: 2em; }
```

Save the file and test it in a browser. Your page should look similar to **Figure 5**.

3. Configure the image gallery grid to be responsive and automatically change the number of columns and rows displayed as the browser viewport is resized. Use the `auto-fill` keyword in the `repeat()` function to direct the browser to fill the viewport with as many columns as it can without overflowing. Edit the index.html file and change `repeat(3, 200px)` to `repeat(auto-fill, 200px).`

Save the file and test it in a browser. Your page will look like **Figure 5** when the browser viewport is just large enough to display three images in a row. As you widen the browser viewport, more columns will display in a single row, similar to **Figure 6**. As you narrow the browser viewport, the number of columns will decrease as shown in **Figure 7**.

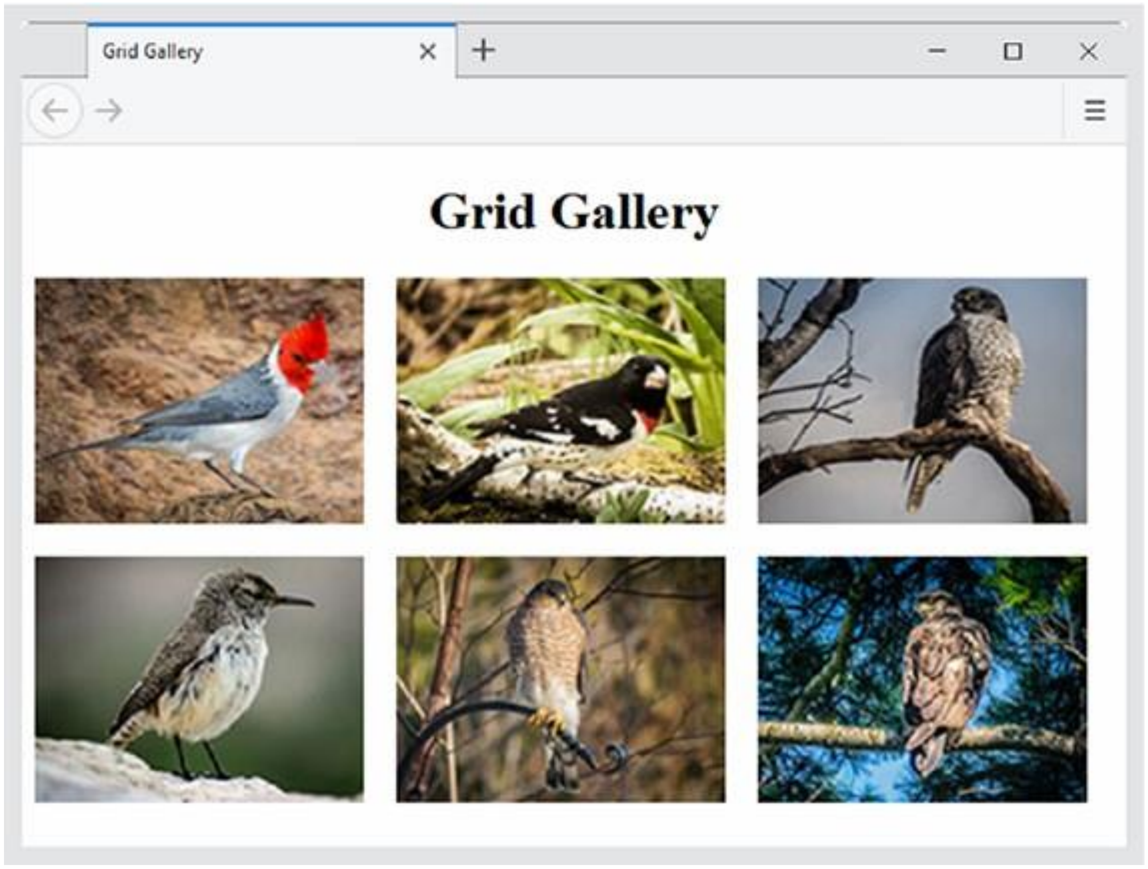


Figure 5 A Basic Grid.

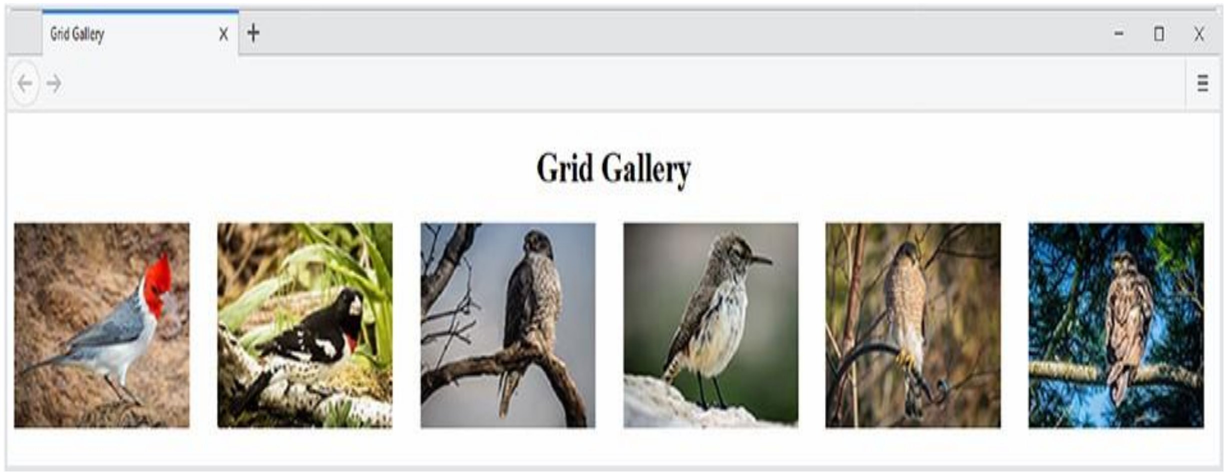


Figure 6 The grid stretches as you widen the browser.

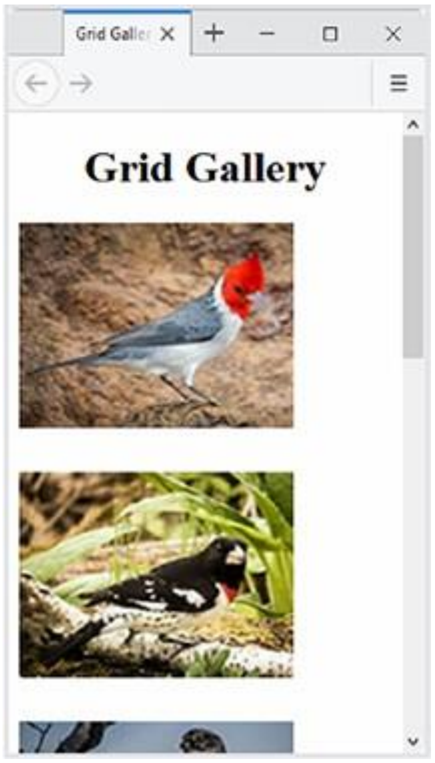


Figure 7 Responsive Grid.

Hands-On Practice 3: Enhance grid layout

You'll use a feature query to progressively enhance an existing web page with grid layout in this Hands-On Practice. The web page displayed in **Figure 8** was configured using float methods from Lecture 7 and is not yet using grid layout. Create a new folder called **ch8grid2**. Copy the **starter4.html** file from the chapter8 folder in the student files into your ch8grid2 folder. Copy the **lighthouse.jpg** and **light.gif** files from the **starters** folder into your ch8grid2 folder.

1. Open your starter4.html file in a browser and the display should be similar to **Figure 8**—this is the display before any code for grid layout is added.
2. Next, launch a text editor and open the starter4.html file. Our grid layout will follow the wireframe in **Figure 9**. You will add an `@supports` rule to the CSS before the ending style tag to check for grid support. You will place code to configure a two-column grid layout. The CSS follows:

```
@supports (display: grid) {
      #wrapper  { display: grid;
                    grid-template-columns:150px 1fr;
                    grid-template-rows: 160px auto auto; }
      header    { grid-row: 1 / 2; gridcolumn: 1 / 3; }
      nav       { grid-row: 2 / 3; gridcolumn: 1 / 2; }
      main      { grid-row: 2 / 3; gridcolumn: 2 / 3; }
      footer    { grid-row: 3 / 4; gridcolumn: 1 / 3; }
}
```

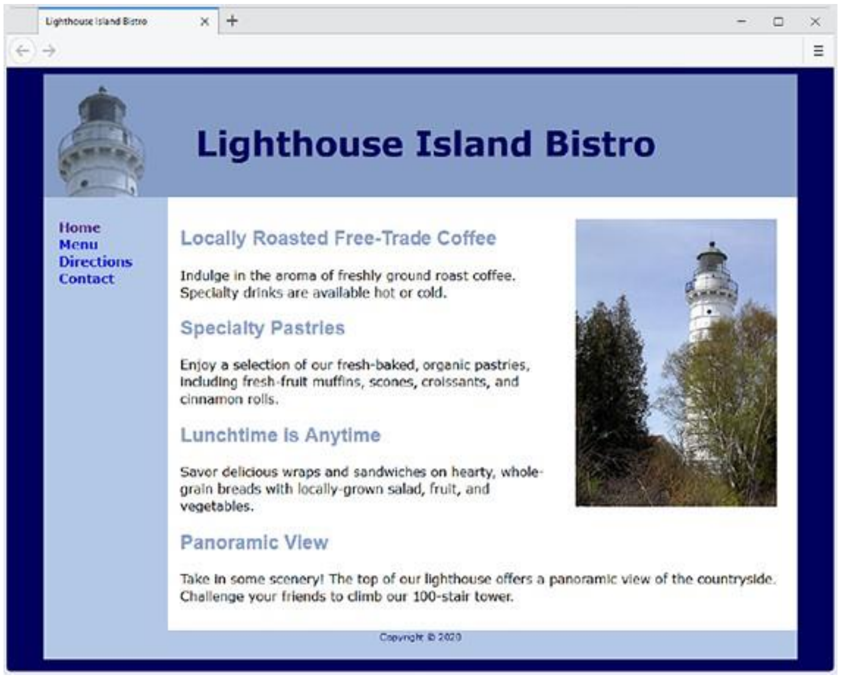
Save your file and test it in a browser that supports grid layout. Your page should look similar to **Figure 10**. Notice that the page looks a bit odd with the main content area beginning too far over to the right.

3. Open your file in a text editor and notice that the main element selector has a 155px left margin set—this is causing the awkward display in **Figure 10**. You need to undo that margin when grid layout is implemented. It's easy to do this by adding a style rule to the `@supports` feature query that eliminates the margin. Add the following style rule to the styles within the `@supports` feature query:

```
main { margin-left: 0; }
```

Save your file. Launch a browser that supports grid layout and test your page. It should look similar to **Figure 11**.

To recap, we applied the principle of progressive enhancement. We began with a web page that was configured with old-fashioned two-column layout using the float property. Next, we configured grid layout within a feature query (which nonsupporting browsers will ignore). Then we looked for any styles that were causing a display issue (the `margin-left` property for the main element in this case) and coded new styles within the feature query to correct the display. The result is a web page that looks good on both supporting and nonsupporting browsers.



Lighthouse Island Bistro

Home
Menu
Directions
Contact

Locally Roasted Free-Trade Coffee

Indulge in the aroma of freshly ground roast coffee. Specialty drinks are available hot or cold.

Specialty Pastries

Enjoy a selection of our fresh-baked, organic pastries, including fresh-fruit muffins, scones, croissants, and cinnamon rolls.

Lunchtime is Anytime

Savor delicious wraps and sandwiches on hearty, whole-grain breads with locally-grown salad, fruit, and vegetables.

Panoramic View

Take in some scenery! The top of our lighthouse offers a panoramic view of the countryside. Challenge your friends to climb our 100-stair tower.

Copyright © 2020

Figure 8 The web page <u>without</u> grid layout.

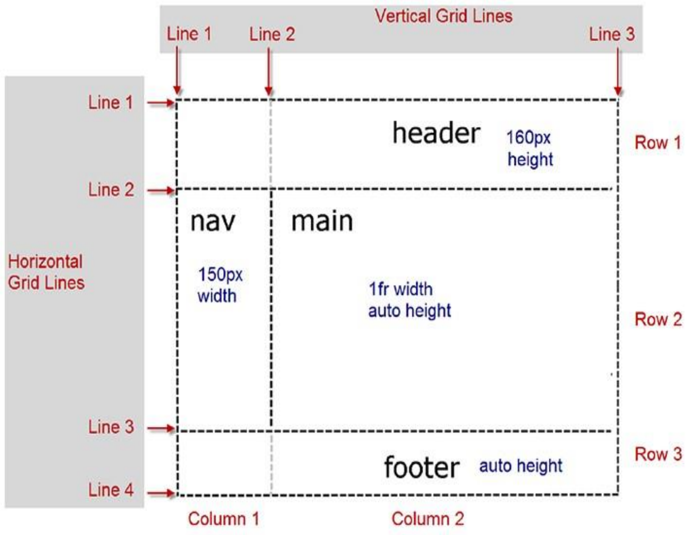
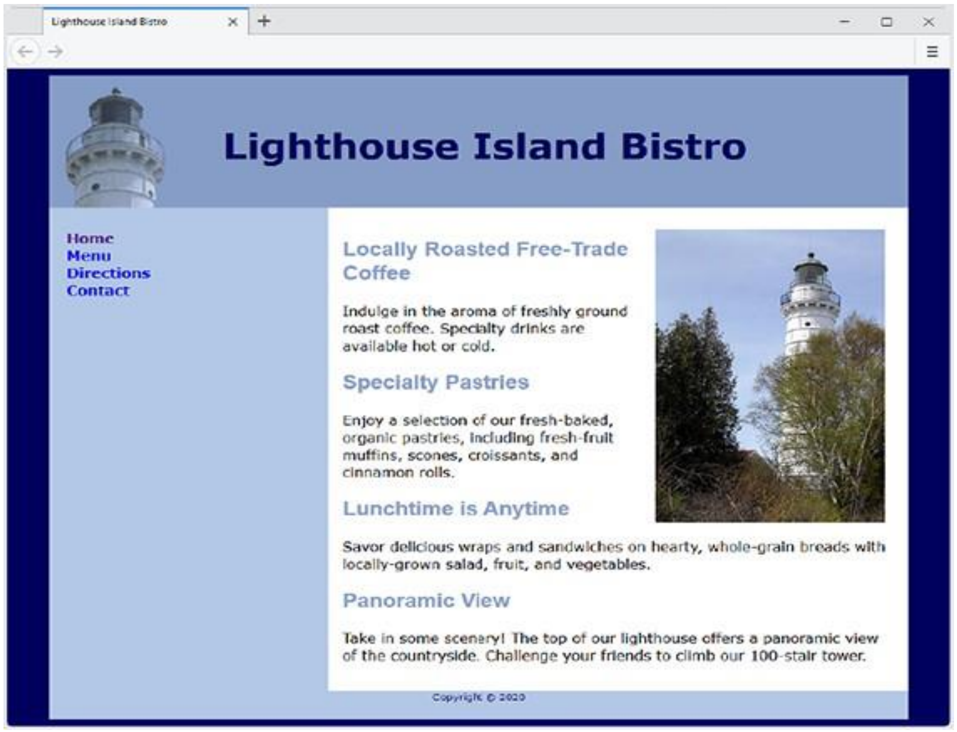


Figure 9 Two-column CSS Grid Layout.



Lighthouse Island Bistro

Home
Menu
Directions
Contact

Locally Roasted Free-Trade Coffee

Indulge in the aroma of freshly ground roast coffee. Specialty drinks are available hot or cold.

Specialty Pastries

Enjoy a selection of our fresh-baked, organic pastries, including fresh-fruit muffins, scones, croissants, and cinnamon rolls.

Lunchtime is Anytime

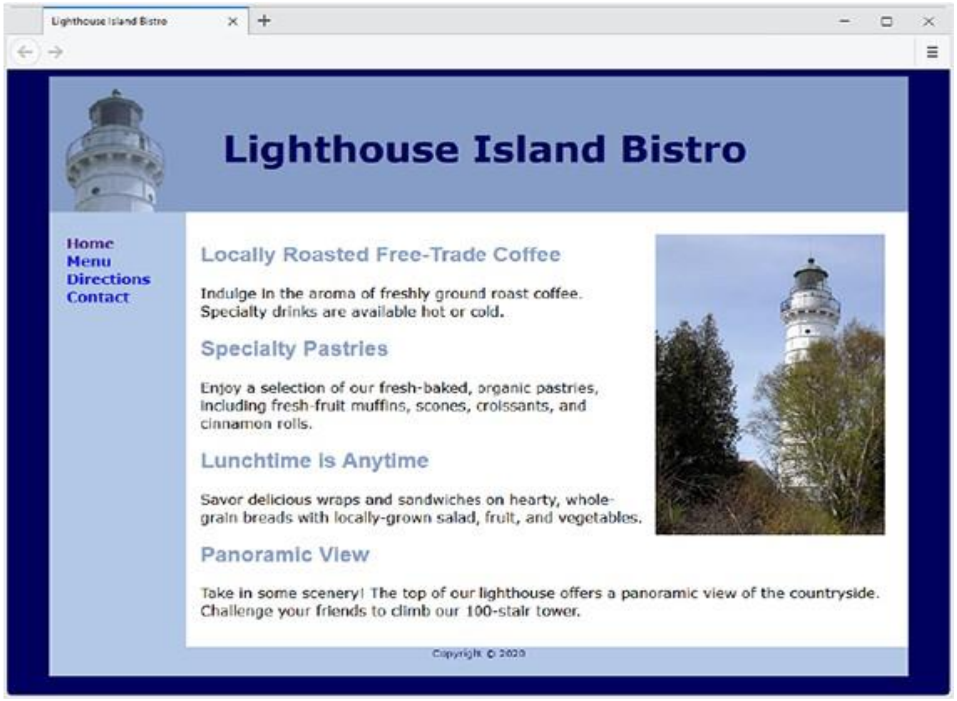
Savor delicious wraps and sandwiches on hearty, whole-grain breads with locally-grown salad, fruit, and vegetables.

Panoramic View

Take in some scenery! The top of our lighthouse offers a panoramic view of the countryside. Challenge your friends to climb our 100-stair tower.

Copyright © 2020

Figure 10 First try at grid layout.



Lighthouse Island Bistro

Home
Menu
Directions
Contact

Locally Roasted Free-Trade Coffee

Indulge in the aroma of freshly ground roast coffee. Specialty drinks are available hot or cold.

Specialty Pastries

Enjoy a selection of our fresh-baked, organic pastries, including fresh-fruit muffins, scones, croissants, and cinnamon rolls.

Lunchtime is Anytime

Savor delicious wraps and sandwiches on hearty, whole-grain breads with locally-grown salad, fruit, and vegetables.

Panoramic View

Take in some scenery! The top of our lighthouse offers a panoramic view of the countryside. Challenge your friends to climb our 100-stair tower.

Copyright © 2020

Figure 11 Successful grid layout.

Hands-On Practice 4: Configure flexible images

In this Hands-On Practice, you'll work with a web page that demonstrates responsive web design. **Figure 12** depicts the default single column page display for small viewports, the two-column page display that triggers when the viewport has a minimum width of 38em units, and the three-column page display that triggers when the viewport has a minimum width of 65em units. You will edit the CSS to configure flexible images.

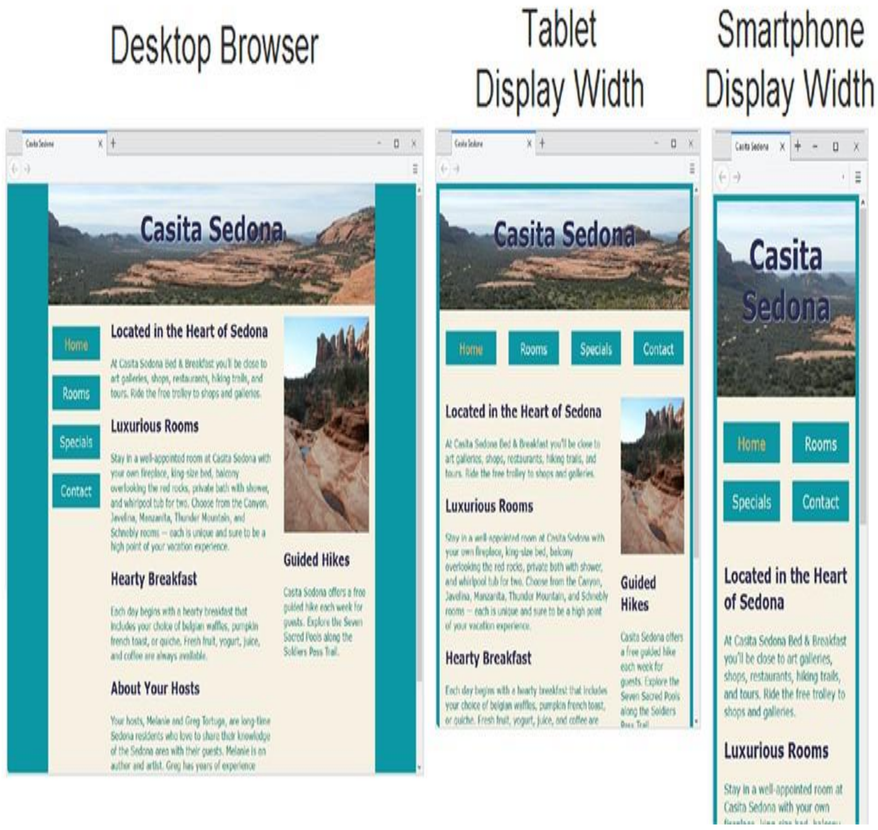


Figure 12 The web page demonstrates responsive web design techniques.

Create a folder named **flexible8**. Copy the **starter8.html** file into the flexible8 folder and rename it index.html. Copy the following images from the starters folder into the flexible8 folder: **header.jpg and pools.jpg**.

Launch a browser and view index.html as shown in **Figure 13**. View the code in a text editor and notice that the `height` and `width` attributes have already been removed from the HTML. View the CSS and notice that the web page uses a grid layout with a flexbox navigation area. Edit the embedded CSS.

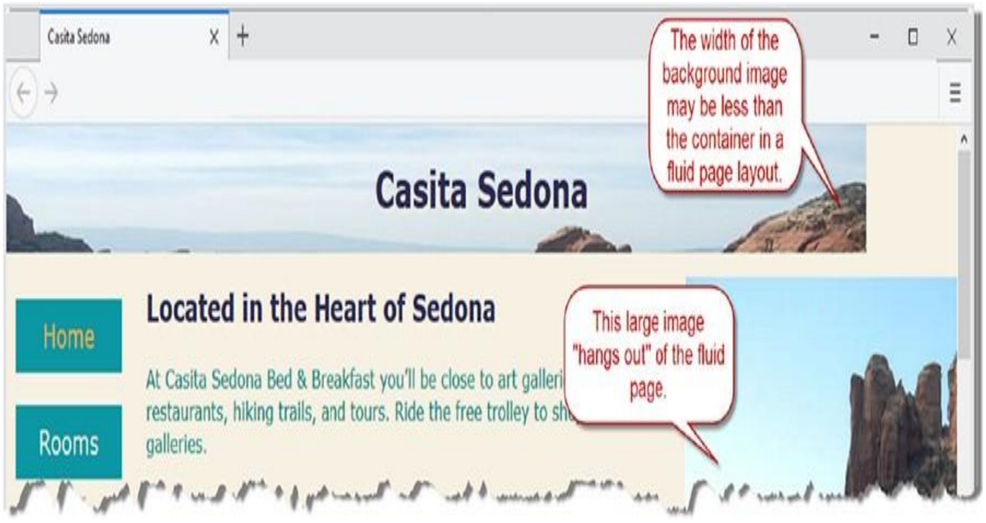

Casita Sedona
Casita Sedona
Located in the Heart of Sedona
At Casita Sedona Bed & Breakfast you'll be close to art galleri restaurants, hiking trails, and tours. Ride the free trolley to sh galleries.
Home
Rooms
The width of the background image may be less than the container in a fluid page layout.
This large image "hangs out" of the fluid page.

Figure 13 The web page before the images are configured to be flexible.

1. Locate the h1 element selector. Add declarations to set the font size to 300%, and bottom padding to 1em. The CSS follows:

```
h1 { text-align: center;
     font-size: 300%;
     padding-bottom: 1em;
     text-shadow: 3px 3px 3px #E9FBFC; }
```

2. Locate the header element selector. Add the `background-size: cover;` declaration to cause the browser to scale the background image to fill the container. The CSS follows:

```
header { background-image: url(header.jpg);
         background-repeat: no-repeat;
         background-size: cover; }
```

3. Add a style rule for the img element selector that sets maximum width to 100% and height to the value auto. The CSS follows:

```
img { max-width: 100%;
      height: auto; }
```

Save the index.html file. Test your index.html file in a desktop browser. As you resize the browser window, you'll see your page respond and look similar to the screen captures in **Figure 12**. The web page demonstrates responsive web design with the following techniques: fluid layout, media queries, and flexible images.

Hands-On Practice 5: Configure responsive image with the picture element (optional in the classroom or as a homework)

In this Hands-On Practice, you will configure responsive images with the picture, source, and img elements as you create the page shown in **Figure 14**.

Create a new folder named **ch8picture**. Copy the large.jpg, medium.jpg, small.jpg, and fallback.jpg files from the starters folder into your **ch8picture** folder. Launch a text editor and open the template file template.html. Save the file as **index.html** in your ch8picture folder.

Modify the file to configure a web page as indicated:

1. Configure the text, Picture Element, within an h1 element and within the title element.
2. Code the following in the body of the web page:

```
<picture>
    <source media="(min-width: 1200px)" srcset="large.jpg">
    <source media="(min-width: 800px)" srcset="medium.jpg">
    <source media="(min-width: 320px)" srcset="small.jpg">
    <img src="fallback.jpg" alt="waterwheel">
</picture>
```

Save your file and test your page.

Notice how a different image is displayed depending on the width of the browser viewport. If the viewport's minimum width is 1200px or greater, the large.jpg image is shown. If the viewport's minimum width is 800px or greater but less than 1200px, the medium.jpg image is displayed. If the viewport's minimum width is 320px greater but less than 800px, the small.jpg image is shown. If none of these criteria are met, the fallback.jpg image should be displayed. As you test, try resizing and refreshing the browser display. You may need to resize the browser, close it, and launch it again to test for display of the different images. Browsers that do not support the new picture element will process the img tag and display the fallback.jpg image.

This Hands-On Practice provided a very basic example of responsive images with the picture element. The picture and element responsive image technique is intended to eliminate multiple image downloads that can occur with CSS flexible image techniques. The browser downloads only the image it chose to display based on the criteria provided.

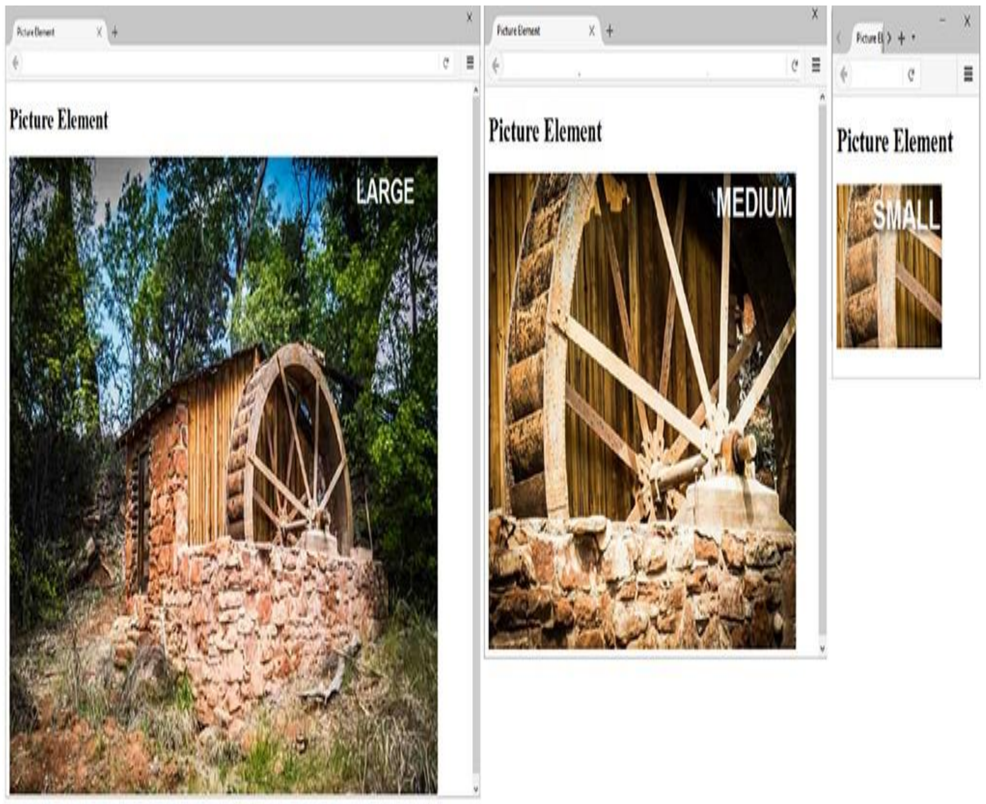


Figure 14 Responsive image with the picture element.

Lab Completion / Submission:

Complete all the lab practices. Take the screenshots of your completed webpages; put them into a single word file and submit it to **Blackboard -> CP160 -> Assessments -> Lab 8 /Assignment 8**; due date: today

After-lab Assignment:

1. Create a one-paragraph conclusion of what you have learned during the lab today.
2. Complete Hands-On Practice **5** if not completed
3. Write the CSS to configure the **nav** element selector as a **flex** container with rows that wrap
4. Write the CSS for a feature query that checks for support of CSS **grid** layout.

Submit to **Blackboard -> CP160 -> Assessments -> Lab 8 /Assignment 8;** due date: 1 week from today.

Project: We keep on learning more web development techniques that can be used in your website project implementation: CSS page layout, responsive layout, form/table (Lecture 9), more media and Interactivity (Lecture 10), javaScript (Lecture 12). We will also learn how to publish/deploy your web site (Lecture 11/Lab 11). Your proposed project can be improved with using these new techniques.