Your First Styling Rule

To use a `class` as a selector in a styling rule, you prefix the class name with a dot (period), as in `.thumbnail-title` . The first styles you are going to add will set the background and foreground colors for the `.thumbnail-title` class.

Open `styles.css` and add your styling rule:

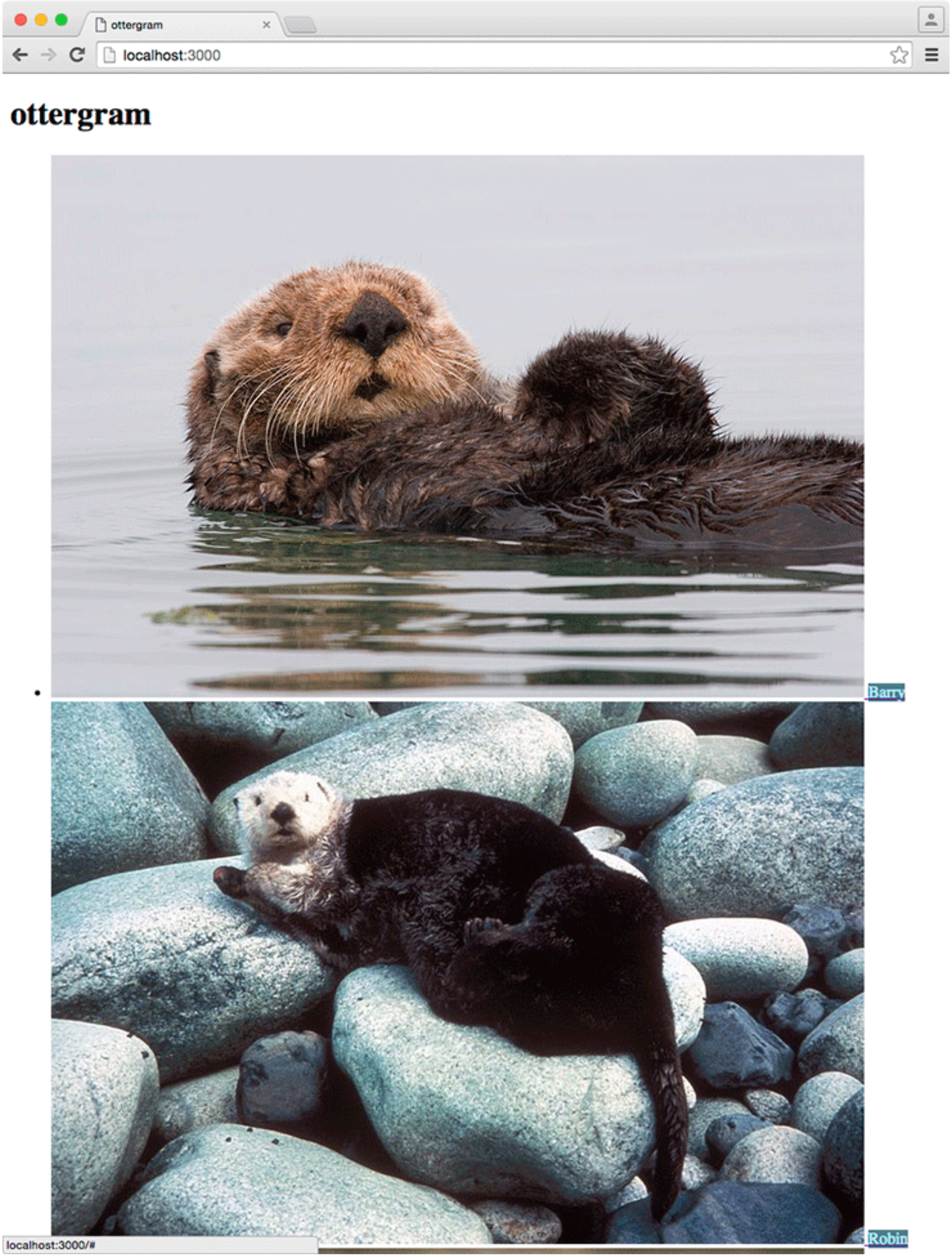
```
.thumbnail-title {
  background: rgb(96, 125, 139);
  color: rgb(202, 238, 255);
}
```

You will learn more about color later in this chapter. For now, just take a look at your changes. Save `styles.css` and make sure `browser-sync` is running. If you need to restart it, the command is:

```
browser-sync start --server --browser "Google Chrome"
                   --files "stylesheets/*.css, *.html"
```

This will open your web page in Chrome (Figure 3.5).

Figure 3.5 A slightly more colorful Ottergram



You can see that you have set the background for the thumbnail titles to a deep gray-blue and the font color to a lighter blue. Nice.

Continue styling the thumbnail titles: Return to `styles.css` and add to your existing styling rule for the `.thumbnail-title` class, as shown:

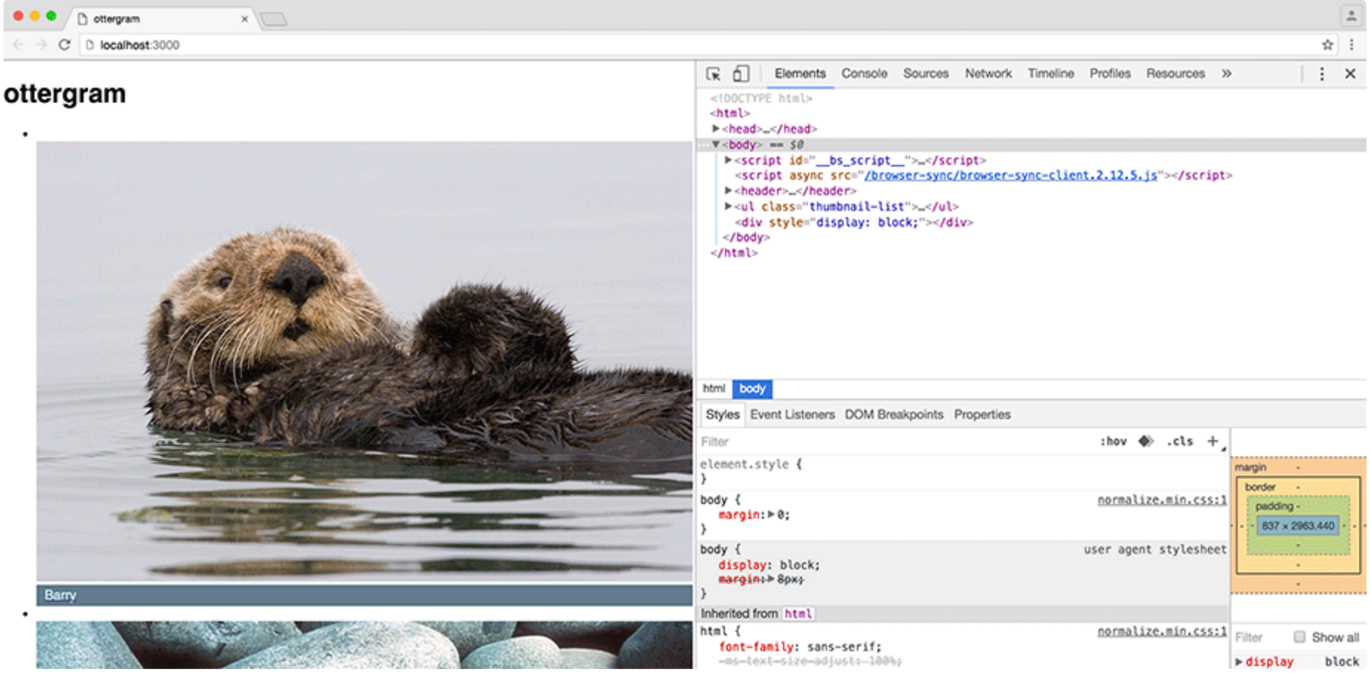
```
.thumbnail-title {
  display: block;
  margin: 0;
  padding: 4px 10px;

  background: rgb(96, 125, 139);
  color: rgb(202, 238, 255);
}
```

The three declarations you have added all affect an element's *box*. For every HTML tag that has a visual representation, the browser draws a rectangle to the page. The browser uses a scheme called the *standard box model* (or just "box model") to determine the dimensions of that rectangle.

The box model

To understand the box model, you are going to look at its representation in the DevTools. Save `styles.css`, switch to Chrome, and make sure the DevTools are open (Figure 3.6).

Figure 3.6 Exploring the box model



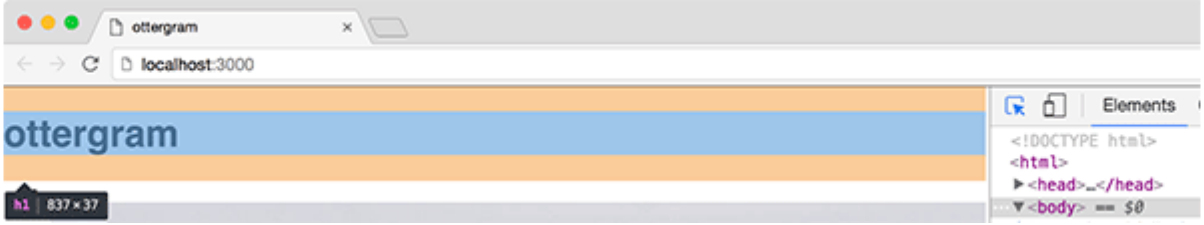
```
<!DOCTYPE html>
<html>
 ▶<head>…</head>
…▼<body> == $0
   ▶<script id="__bs_script__">…</script>
    <script async src="/browser-sync/browser-sync-client.2.12.5.js"></script>
   ▶<header>…</header>
   ▶<ul class="thumbnail-list">…</ul>
    <div style="display: block;"></div>
  </body>
</html>
```

```
element.style {
}
body {                                  normalize.min.css:1
  margin:▶0;
}
body {                                  user agent stylesheet
  display: block;
  ~~margin:▶ 8px;~~
}
Inherited from html
html {                                  normalize.min.css:1
  font-family: sans-serif;
  ~~-ms-text-size-adjust: 100%;~~
```

```
▶display     block
```

Click the ⬚ button in the upper-left of the elements panel. This is the *Inspect Element* button. Now move your cursor over the word "ottergram" on the web page. As you hover over the word, the DevTools surrounds the heading with a blue- and peach-colored rectangle (Figure 3.7).

Figure 3.7 Hovering over the heading

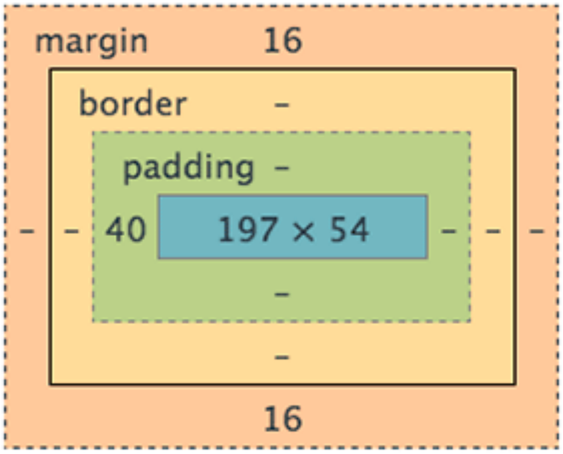


```
<!DOCTYPE html>
<html>
 ▶<head>…</head>
…▼<body> == $0
```

Click the word "ottergram" on the web page. Although you no longer see the multicolored overlay, the element is now selected and the DOM tree view in the elements panel will expand to show and highlight the corresponding `<h1>` tag.

The rectangular diagram in the lower-right of the elements panel represents the box model for the `h1` element. You can see that the regions of the diagram have some of the same colors as the rectangle you saw overlaying the heading when you inspected it (Figure 3.8).

Figure 3.8 Viewing the box model for an element



The box model incorporates four aspects of the rectangle drawn for an element (which the DevTools renders in four different colors in the diagram).

content (shown in blue)

the visual content – here, the text

padding (shown in green)

transparent space around the content

border (shown in yellow)

a border, which can be made visible, around the content and padding

margin (shown in peach)

transparent space around the border

The numbers in Figure 3.8 are *pixel* values; a pixel is a unit corresponding to the smallest rectangular area of a computer screen that can display a single color. In the case of the `h1` element, the content area has been allocated an area of 197 pixels by 54 pixels (your values may be different, depending on the size of your browser window). There is padding of 40 pixels on the left side. The border is set at 0, and there is a margin of 16 pixels above and below the element.

Where did that margin value come from? Each browser provides a default stylesheet, called the *user agent stylesheet* , in case an HTML file does not specify one. Styles that you specify override the defaults. Because you have not specified values for the `h1` element's box, the default styles have been applied.

Now you are ready to understand the styling declarations you added:

```
.thumbnail-title {
  display: block;
  margin: 0;
  padding: 4px 10px;

  background: rgb(96, 125, 139);
  color: rgb(202, 238, 255);
}
```

The `display: block` declaration changes the box for all elements of the class `.thumbnail-title` so that they occupy the entire width allowed by their containing element. (Notice in Figure 3.6 that the background color for the titles now covers a wider area.) Other `display` values, such as the `display: inline` property you will see later, make an element's width fit to its content.

You also set the margin for the thumbnail titles to 0 and the padding to two different values: 4px and 10px (*px* is the abbreviation for "pixels"). This sets the padding to specific pixel values, overriding the default size set by the user agent stylesheet.

Padding, margin, and certain other styles can be written as *shorthand properties* , in which one value is applied to multiple properties. You are taking advantage of this here: When two values are provided for the padding, the first is applied to both vertical values (top and bottom) and the second is applied to both horizontal values (left and right). It is also possible to provide a single value to be applied to all four sides or to specify a separate value for each side.

To sum up, your new declarations say that the box for all elements of the `.thumbnail-title` class will fill the width of its container with no margin and with padding that is 4 pixels at the top and bottom and 10 pixels at the left and right sides.