1.2.5   Set Up the Style Sheet

Now that we know the route we're taking to add CSS to our webpage, let's do it!

Here's a rundown of the steps we'll be taking:

1. Using our newfound knowledge of command-line tools, create a folder called `assets` (**Hint:** Use `mkdir` to create a folder, then use `ls` to print the contents and see if it worked!).

2. Move into the newly created `assets` folder using `cd` and then create another folder inside `assets` called `css`.

3. Move into the newly created `css` folder and create a file called `style.css`. Don't worry about putting anything in it just yet.

Now that our file is created, we need to tell our HTML document to read any applicable styles that `style.css` may have for it. We can do this by placing `<link rel="stylesheet" href="./assets/css/style.css" />` in between the `head` tags in our HTML document.

PAUSE

What do you think `href="./assets/css/style.css"` is telling our HTML document to do?

We are instructing the `<link>` tag to find the stylesheet called `style.css` in the `assets/css` subfolder and apply the CSS rules listed in that file.

[Hide Answer](#)

What we just did was use the HTML element `link` to tell our HTML document to go find a specific resource (file) and incorporate it into the document. It needs at least one attribute, `href`, which behaves similarly to how it works in `<a>` elements, but this one is serving a different purpose. This one is saying "find this file called `style.css` located in the `css` folder inside of the `assets` folder, read it, and incorporate any of its information into this HTML document." This is the first real instance where we've provided a value to an `href` that points to another file in our directory. This is what's known as **relative pathing**, and it's a very important concept so make sure to read about it below.

The other attribute, `rel`, is providing a little more context about what the HTML's **relationship** to the file being included through the `href` is supposed to be. In this case and most of the time, we'll use `rel="stylesheet"`, but as time goes on there may be situations where that will change.

> ## IMPORTANT
>
> Throughout your career, there will be almost daily instances where you will need to make one file look for and read another. We used this above with the `link` element's `href` value `"./assets/css/style.css"`.
>
> This was an example of relative pathing, which works well for us. The other option would be to put a hard set path (also known as an

absolute path) from the host computer's directory structure, which would look something like this:

```
<link rel="stylesheet" href="/Users/<username>/Desktop/run-buddy/
```

See a potential problem here? This path is very specific to someone's personal computer—it even has a username in it. If we were to put this code into production or share it with a teammate, the project's code would then live on a different computer entirely. The path in the example most likely does not exist on that computer, meaning any reference to it would break and the page would not load correctly.

The best solution for this is to use relative pathing. With relative pathing, when we push up the entire folder structure for the project all at once, the paths don't lose context as to where they are.

Let's test this and make sure it works by adding the following to `style.css`:

```
body {
    font-family: Helvetica, Arial, sans-serif;
    background-color: tomato;
}
```

If that turned the whole background of your page to red, then it works! Please remove the `background-color` style from the page as we won't need it anymore.

PAUSE

We'll get into explaining the above CSS syntax in a minute. Before we do, take a moment and think about what's happening here. In

CSS, you'll often see the word `body`. Where have we seen "body" before?

We used the word "body" to create the HTML `<body>` element in Lesson 1. The `<body>` element contains all the content that a user sees on a webpage.

Hide Answer