1.3.6 Style the Sign-Up Form

Let's make our design team happy by following their mock-up to give the `<form>` some needed layout and styling.

Heading

Let's start by jazzing up the heading. This form needs to exude excitement and enthusiasm, and the current `<h3>` element leaves a bit to be desired.

There are several ways to select this element using CSS:

- We could give this `<h3>` element a class and select it as we did for the form container and the `<section>`.
- We could select the parent element and then the child element that's contained within it, as we did in the previous lesson in the `header` and `footer`.

We'll go with the second option for one reason: class attributes are used to distinguish or select one or more set of elements from other elements. Since there is only one `<h3>`, creating a new class is a bit of an overkill. Simplicity is a great rule of thumb to go by when possible.

In `style.css`, adjust this `<h3>` element to make it a bit larger, and set the `margin` to 0 to reduce any possible word wrapping. The CSS is listed below, but try it on your own before you peek!

Text Spacing

Next, add some space to the `<p>` element to make it more pleasant to read, specifically on the top and bottom margins.

Input Fields

Now we will give our input fields some style and panache. First let's get each one to exist on its own line by making the `font-size` bigger with a nice `border`.

Part of having a good website is making it easy for a visitor to navigate. When it comes to user interaction, it's really important to make it as simple and straightforward as possible. This means having clear demarcations on each line and to address what each input field is for.

PAUSE

If we selected `<input>` as our CSS selector, we would style the radio button and checkbox as well as the input fields, which isn't our goal. So how could we select just the input fields for this styling?

Let's use a class as our CSS selector to target a group of input fields while ignoring the rest. Let's add the class `form-input` to our `<input>` opening tags for just the first three inputs to target our input fields.

Hide Answer

The `<input>` elements in your HTML code should now look like this:

```
<label for="name">Enter full name:</label>
<input type="text" placeholder="Your Name" name="name" id="name" class

<label for="email">Enter email address:</label>
<input type="text" placeholder="Email Address" name="email" id="email"

<label for="phone">Enter a telephone number:</label>
<input type="text" placeholder="Phone Number" name="phone" id="phone"
```

Now let's add a border, display each line separately, make the font larger, extend each input field box the length of the form container, and use the navy font color we used for the `footer` font color.

Your CSS code should now look like this:

```
.hero-form h3 {
  font-size: 24px;
  margin: 0;
}

.hero-form p {
  margin: 5px 0 15px 0;
}

.form-input {
  border: 1px solid #024e76;
  display: block;
  padding: 7px 15px;
  font-size: 16px;
  color: #024e76;
  width: 100%;
  margin-bottom: 15px;
}
```

Let's unpack some of the properties in the `.form-input` selector and see how they affected the form input elements:

- `border`: A skinny 1px border helped emphasize our sign-up requests.
- `display`: This property instructs the browser to render the element either as a `<block>` or `inline`.

PAUSE

Can you name a few `<block>` elements we've used?

`<p>`, `<div>`, and all the `<h1>` to `<h6>`, to name a few.

Hide Answer

PAUSE

Can you name a few inline elements we've used?

`<span>`, `<a>`, and `<label>`, to name a few.

Hide Answer

Remember to save these changes and take a look at the new page's styles in the browser as we add them.

Radio Buttons

Adding some horizontal spacing for the `<label>` element in the radio button elements helps the user not misinterpret which selection is for which answer. Let's go ahead and do that now.

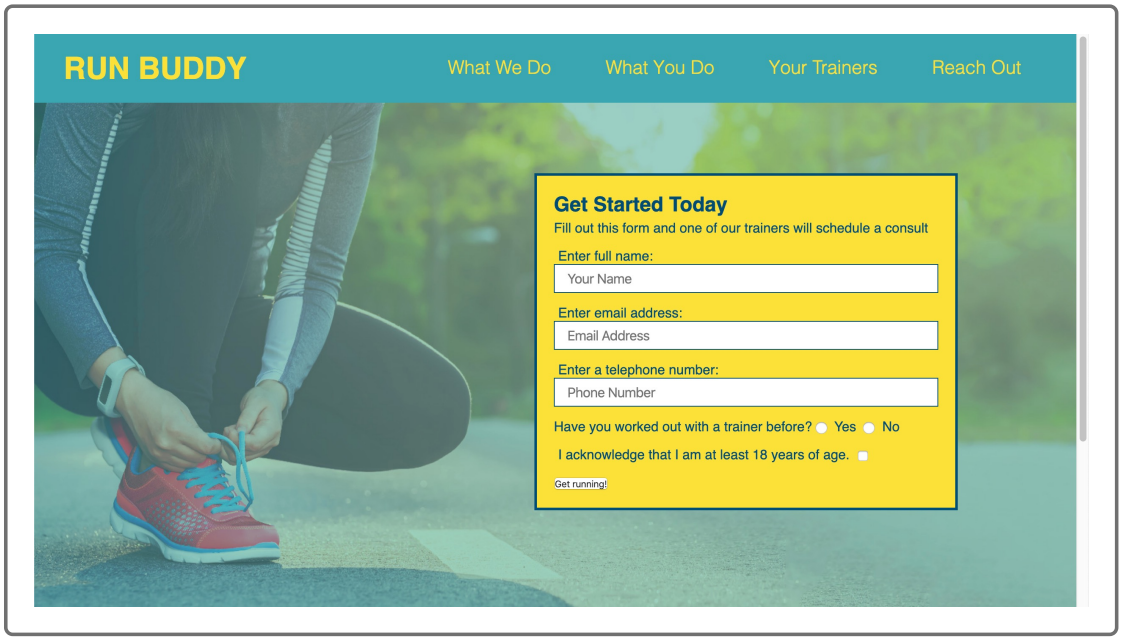
Add the following code to `style.css`:

```
.hero-form label {
  margin: 0 5px;
}
```

IMPORTANT

> It is important to try and implement these styles yourself. Syntax errors and red squiggly lines are helpful reminders that you might be missing or misplacing a character. Remember, you can always experiment in the Chrome DevTools until you get it just right.

Save your work and take a look at the browser to see what you've completed so far. It should look like this image:



Awesome! The `<section>` element content looks almost finished.

Button

The design team wants the "Get running" `button` to be navy (`#024e76`) and the font color yellow (`#fce138`), like this image shows us:

Let's style the `button` as requested and add sizes to the font and `padding`. Open Chrome's DevTools in the browser and select the `button` using the element selector or arrow icon. Then go to the Styles panel to experiment with some of the `button` element's properties. By clicking the declaration, the property or value can be changed. By clicking to the right of the declaration, a new line is added for a new declaration.

When you have the look that matches the mock-up, the CSS rule for the button should look something like this:

```
.hero-form button {
  color: #fce138;
  background-color: #024e76;
  border: none;
  padding: 10px 20px;
  font-size: 16px;
}
/* HERO STYLES END */
```

Let's save the file and refresh the browser. Excellent work—our section looks just like the mock-up!

You just completed a large section of the page, so it's a great time to add and commit your work using Git.

IMPORTANT

> Don't worry if you haven't nailed down all the concepts yet. In programming, practice doesn't make perfect—it makes developers. If you get stuck, become a Google-fu master. Your answer is out there; you just have to go find it.