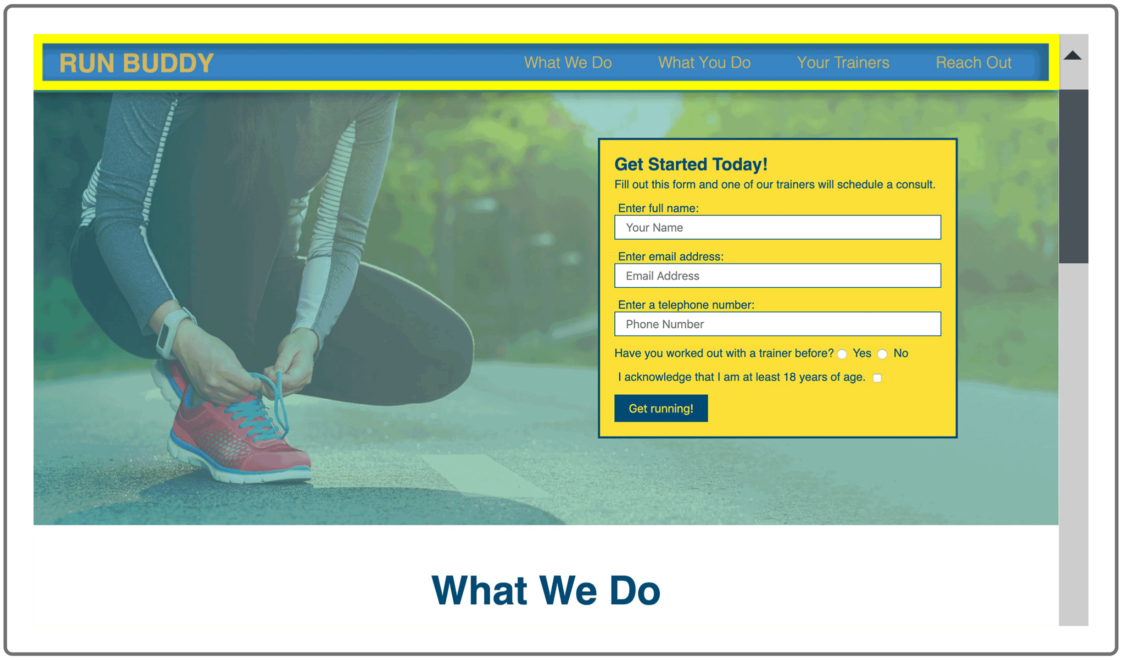
2.5.5 Make the Header "Sticky"

We've managed to spruce up the page a little bit, but the Run Buddy team is getting antsy about the header. It still doesn't look like the new mock-up. The design team also wants the header to be "sticky," meaning it looks like part of the hero at first but then follows the page as it scrolls.

Here's a demo of that behavior:



Get Started Today!

Fill out this form and one of our trainers will schedule a consult.

Enter full name: Your Name

Enter email address: Email Address

Enter a telephone number: Phone Number

Have you worked out with a trainer before? ○ Yes ○ No

I acknowledge that I am at least 18 years of age. □

What We Do

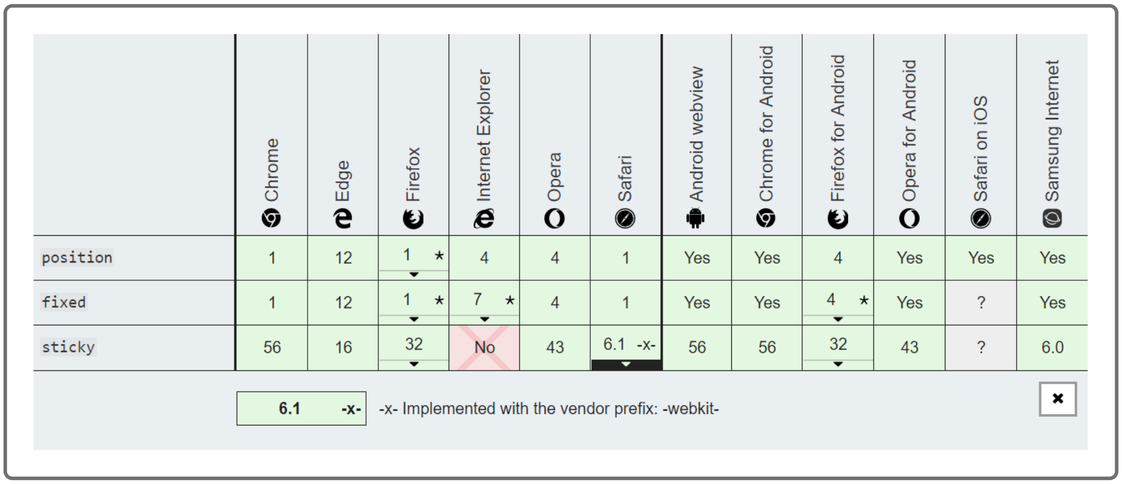
Let's start with the stickiness aspect. In the previous module, we talked about different types of positioning: relative, absolute, fixed, etc. There's a new position in CSS called **sticky** that's kind of like a hybrid position. An element that's sticky will behave relatively until the viewport scrolls past it. Then the element will switch to fixed and remain visible even as the user scrolls farther down.

Add the following declarations to the `header`, then scroll up and down in the browser and see what happens:

```
position: sticky;
top: 0;
```

The `top` property is important because this tells the browser where to "stick" the element. We always want the header to be at the very top of the viewport, so `top: 0;` makes the most sense.

Sticky headers are pretty cool, but it's important to know that not all browsers support this feature of CSS. Looking at the **position compatibility chart from MDN (https://developer.mozilla.org/en-US/docs/Web/CSS/position#Browser_compatibility)**, we can see that position sticky doesn't work at all in Internet Explorer:



	Chrome	Edge	Firefox	Internet Explorer	Opera	Safari	Android webview	Chrome for Android	Firefox for Android	Opera for Android	Safari on iOS	Samsung Internet
`position`	1	12	1 *	4	4	1	Yes	Yes	4	Yes	Yes	Yes
`fixed`	1	12	1 *	7 *	4	1	Yes	Yes	4 *	Yes	?	Yes
`sticky`	56	16	32	No	43	6.1 -x-	56	56	32	43	?	6.0

6.1 -x- -x- Implemented with the vendor prefix: -webkit-

REWIND

MDN's compatibility chart is a similar tool to the **Can I Use website** (https://caniuse.com/), which we learned about at the beginning of this module.

Fortunately, Run Buddy has determined that none of its users are on IE. However, they do want support for Safari, and that comes with a caveat: we'll have to use a **vendor prefix**.

But what's a vendor prefix? Before certain CSS properties become standard, browsers will sometimes implement them in a more experimental capacity using a browser-specific prefix. When the CSS feature is standardized, later versions of the browser drop the prefix.

For example, the `border-radius` property used to require vendor prefixes, as shown here:

```
.rounded-corners {
  -webkit-border-radius: 10px; /* old versions of chrome and safari */
  -moz-border-radius: 10px; /* old versions of firefox */
  border-radius: 10px; /* standardized browsers */
}
```

DEEP DIVE ▲

DEEP DIVE

Because browsers are built by different companies, their underlying **engines** behave differently. Browser engines read the HTML code of a webpage and convert it into something the user can actually see and interact with. Safari's engine is called WebKit. Chrome was originally based on WebKit but has become its own thing, now called Blink. Recently, Microsoft adopted Blink as the engine for the Edge browser. Clearly, a lot of sharing happens across these engine projects, but discrepancies still exist, of course.

To learn more, read the **Wikipedia article on browser engines (https://en.wikipedia.org/wiki/Browser_engine)**.

Until Safari joins the cool kids, we'll need to add a vendor prefix to the sticky header to make sure it will be supported by that browser. The `header` CSS rule should now include the following three declarations:

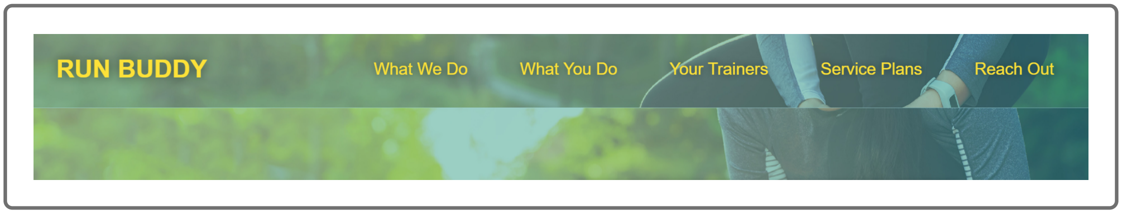
```
position: -webkit-sticky;
position: sticky;
top: 0;
```

Now that the header is sticky, we can style it to make it look like part of the hero. Keep in mind that the header isn't really part of the hero; they must be separate elements for the stickiness to work. If done correctly, the user won't notice.

Copy the following background properties from `.hero` into `header`:

```
background-image: url("../images/hero-bg.jpg");
background-size: cover;
background-position: center;
```

That got us close, but the hero and header aren't lining up as nicely as they do in the mock-up, as you can see here:



We need the background images of these two elements to essentially be in the same place. Fortunately, we can do just that by using the declaration `background-attachment: fixed;`, which places the image relative to the viewport instead of the element.

Let's add `background-attachment` to the `header` and `.hero` rules. At the same time, add or change their `background-position` values to `80%` to better match the mock-up. Both declarations should look like this:

```
background-attachment: fixed;
background-position: 80%;
```

If you test the site again on a mobile device, however, you'll quickly notice that the sticky header covers up a lot of valuable space. For mobile devices, it would be a better user experience to revert back to a normal header.

In the `@media screen and (max-width: 980px)` query, add `position: relative` to the `header` rule to undo the stickiness for smaller screen sizes.

There's still plenty more to do, but let's pause for a knowledge check:

Given the declaration `box-shadow: 1px 2px 3px blue;` , what does each value mean?

○ Blur radius, horizontal offset, vertical offset, and color.

○ Blur radius, vertical offset, horizontal offset, and color.

○ Horizontal offset, vertical offset, blur radius, and color.

○ Vertical offset, horizontal offset, blur radius, and color.

Check Answer

Which is an example of a vendor prefix?

○ -webkit-

○ ::before

○ position: sticky

○ #024e76

Check Answer

Finish ▸