2.6.5 Animate the Hover

Now that we know what the end result looks like, we can define the starting point and bridge the gap between them with a `transition`. Let's start with the gradient because it involves editing fewer elements. The starting point is an `opacity` of 0, so add that to the pseudo-element rule (`.trainer-img::after`).

Next, we want to change the `opacity` to 1 when the mouse hovers over any part of the trainer `<article>`—not just the background image. Hold on, though. How can a parent's hover state affect a child? Consider for a moment the following CSS rule:

```
div span {
  color: blue;
}
```

What does this mean again? It means any `<span>` element that's inside a `<div>` element will have blue text. What about this example:

```
div:hover span {
  color: red;
}
```

This means any `<span>` element that's inside a `<div>` element *that's in a hover state* will have red text.

With that in mind, add a new CSS rule to the style sheet that sets the `opacity` of the pseudo-element (`.trainer-img::after`) to 1 when the parent (`.trainer`) is hovered over. You'll know it's working if you can hover over the image or the bio underneath to make the gradient appear.

Of course, the gradient still isn't animated, but now that the default state and hover state are set, it's only a matter of adding a half-second `transition` to the default state. Refer to the **MDN web docs on transitions (https://developer.mozilla.org/en-US/docs/Web/CSS/transition)** if you get stuck.

HIDE HINT

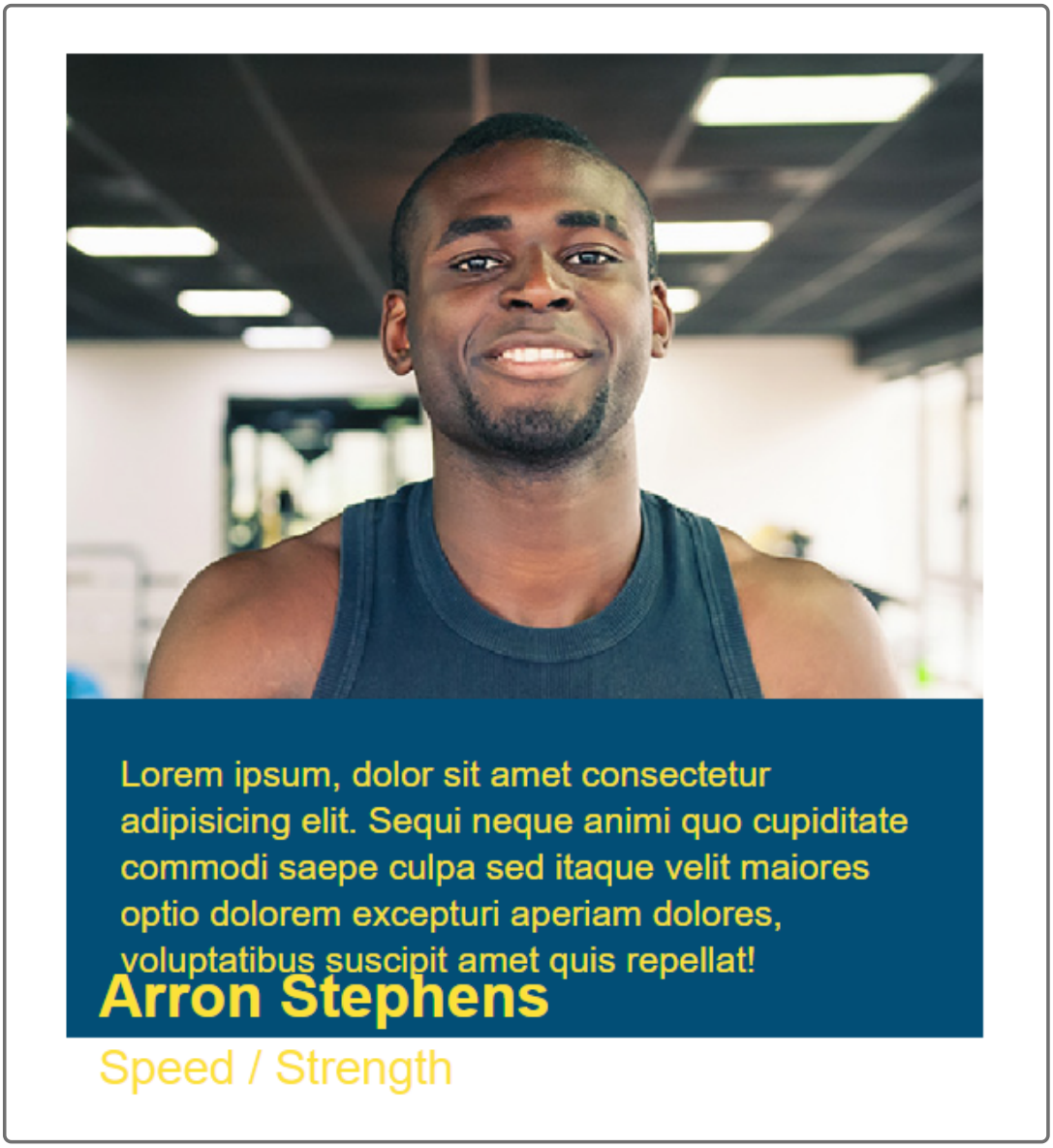
You'll only need to update the following CSS rule:

```
.trainer-img::after {
 opacity: 0;
 transition: ???;
}
```

Finally, we need to do something about the text. This is a somewhat misleading problem. It may seem like the text "comes out of nowhere," but the secret is that the text is always there. It just slides into view when we're ready for it.

By now, your `<h3>` and `<h4>` elements should have `position: relative` applied to them. Relative positioning will allow us to bump these elements

down and out of view using the `top` property. Try setting `top` to `200px` for both of them. That should result in this:



Hmmm. These elements are not very good at hide-and-seek. We can still see them! With the `overflow` property, however, we can hide any content that flows outside of its parent. Look at the **MDN web docs on overflow (https://developer.mozilla.org/en-US/docs/Web/CSS/overflow)** and see which value will accomplish this for us.

HIDE HINT

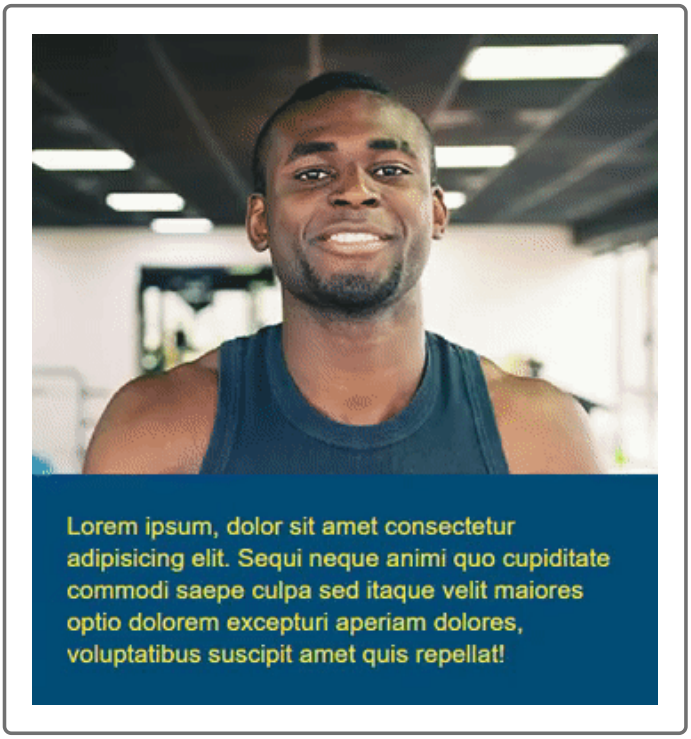
> The `overflow` property would be applied to the immediate parent, not the children you're actually trying to hide.

Like the pseudo-element's opacity, we can set `top` back to `0px` when the trainer `<article>` is being hovered over. Because both elements will have the same property and value, we can combine their selectors into one rule:

```
.trainer:hover .trainer-img h3, .trainer:hover .trainer-img h4 {

}
```

This is the behavior we should see:



Because the text isn't animated yet, it almost looks like we just changed the opacity/visibility instead of the position. Add a 0.6-second `transition`

to the following CSS rules so you can better see the elements slide in from `200px` to `0px`:

```
.trainer-img h3 {

}

.trainer-img h4 {

}
```

HIDE PRO TIP

Rather than setting `top` to `0px` on the hover state, you could use a different declaration: `transform: translate(-200px)`. It's probably hard to see in this example, but `translate()` can lead to smoother animations. For more details, see the **MDN web docs on translate (https://developer.mozilla.org/en-US/docs/Web/CSS/transform-function/translate)**.

Also keep in mind that the `<h4>` trails behind the `<h3>` for a fraction of a second. What additional transition property would help us set a delay on the `<h4>`? Remember to consult the **MDN web docs on transitions (https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Transitions)** for help.

Once you've got it, reference your work against the demo animation provided earlier. If things don't match up, double-check your `transition` values and CSS selectors. Above all else, leverage Chrome's DevTools!