Styling the Form

In this lesson, we'll give our form a more familiar appearance.

This is going to go by a lot quicker than previous lessons. I believe proficiency in styling increases logarithmically to practice.
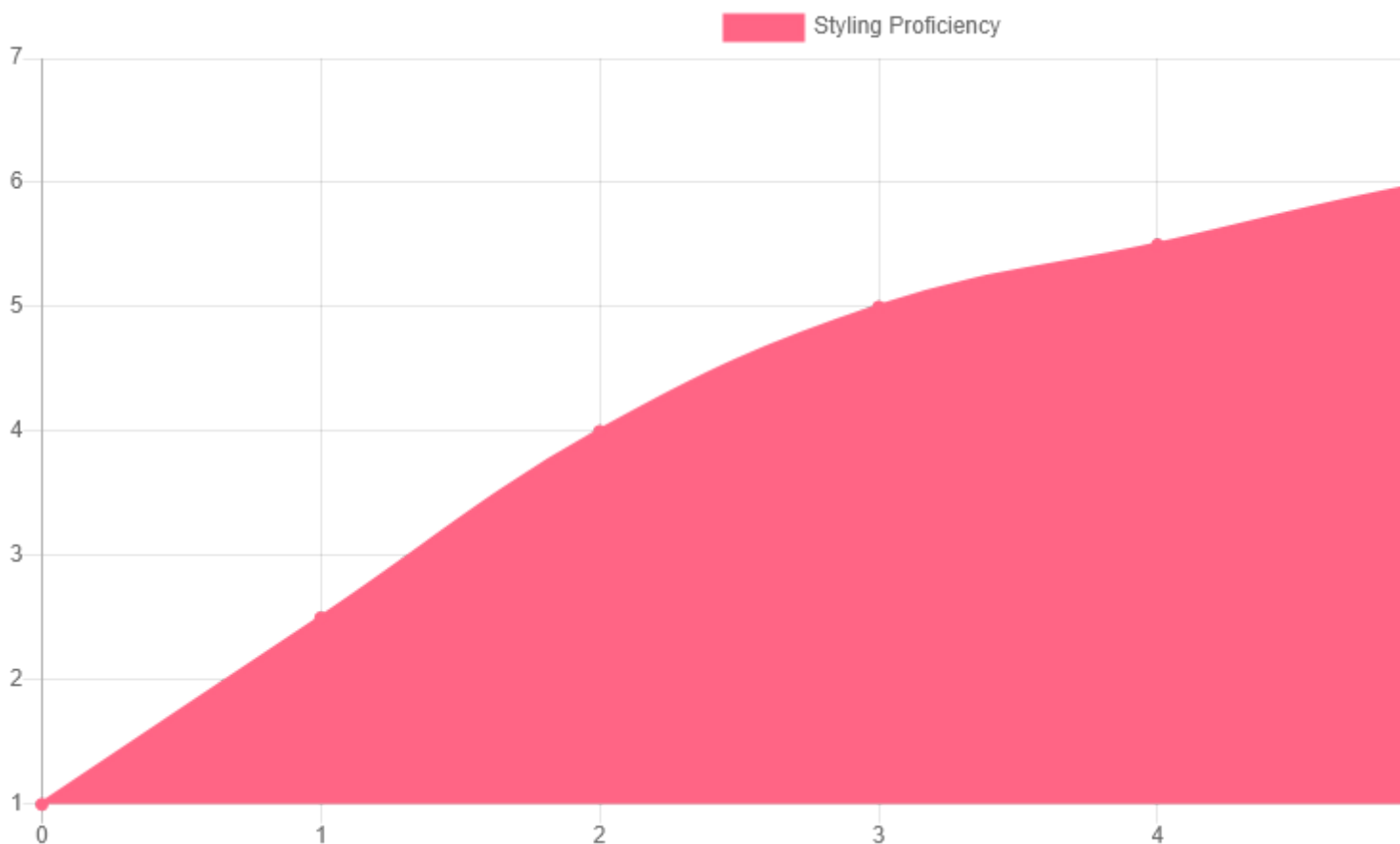
The chart uses a very loose definition of proficiency, but it shows the idea. Your improvement after implementing your thousandth component is going

to be a lot less than your tenth. The learning path for most programming languages follows this pattern, but improvement is especially quick with CSS styling, and a high proficiency can be achieved with a relatively small amount of practice.

This also has a non obvious implication: be wary of using advanced features. The low barrier of entry means other developers can come along and make stylistic changes even if they don't regularly work with CSS. Even within your frontend team, not everyone will be working in CSS, and the logarithmic learning curve probably means a lot of them won't be aware of the intricacies of highly advanced CSS tricks. If you can stick to the subset of CSS that most people know, development and debugging will be faster.

You should continue to attempt to read and understand every line of CSS, but, hopefully, fewer things will seem confusing, and you'll be nodding in agreement as you read these rules.

Styling

We're going from this to this:

To approximately get where we want, it looks like we should...

- Limit the width of the wrapper, grey background, some padding

- Bold headers, a tiny bit of padding below
- Inputs divide width evenly, so if one input, takes entire width; if two, each takes half, etc
- Space is *created* for the error messages. They look like lighter font weights too
- Remove the default border for `fieldset` elements

Output
HTML
CSS (SCSS)

Name

First   Last

Choose your username

Create a password

Confirm your password

Birthday

Month   Day   Year

Mobile phone

Your current email address

Changelist

- Group the inputs under a `signup__field__inputs` div. We want to specifically split the inputs evenly, and this facilitates that through:
- Applying flexbox styling. This is just a display type that lets us scale things more automatically. There are all sorts of special rules with it, and

I'd even go so far as to consider it advanced CSS. In this case, we have

inputs that can be 33% of the width, 50%, and others, so to be more programmatic, we'll take advantage of flexbox display.

- `flexbox-grow` says expand to fit, `flexbox-shrink` says shrink to fit.
- `flex-direction` specifies the orientation that expanding and shrinking occurs.
- We still specify a width for the default

- Miscellaneous font and sizing

That gets our "80%," which is all we were looking to achieve for now!