Building the Form

Let's set up the basic framework for our auto validating form using the HTML form element.

WE'LL COVER THE FOLLOWING

- The HTML
- `<div>` vs `<label>`
- Switching to semantic tags
  - What's up with the birthday month?

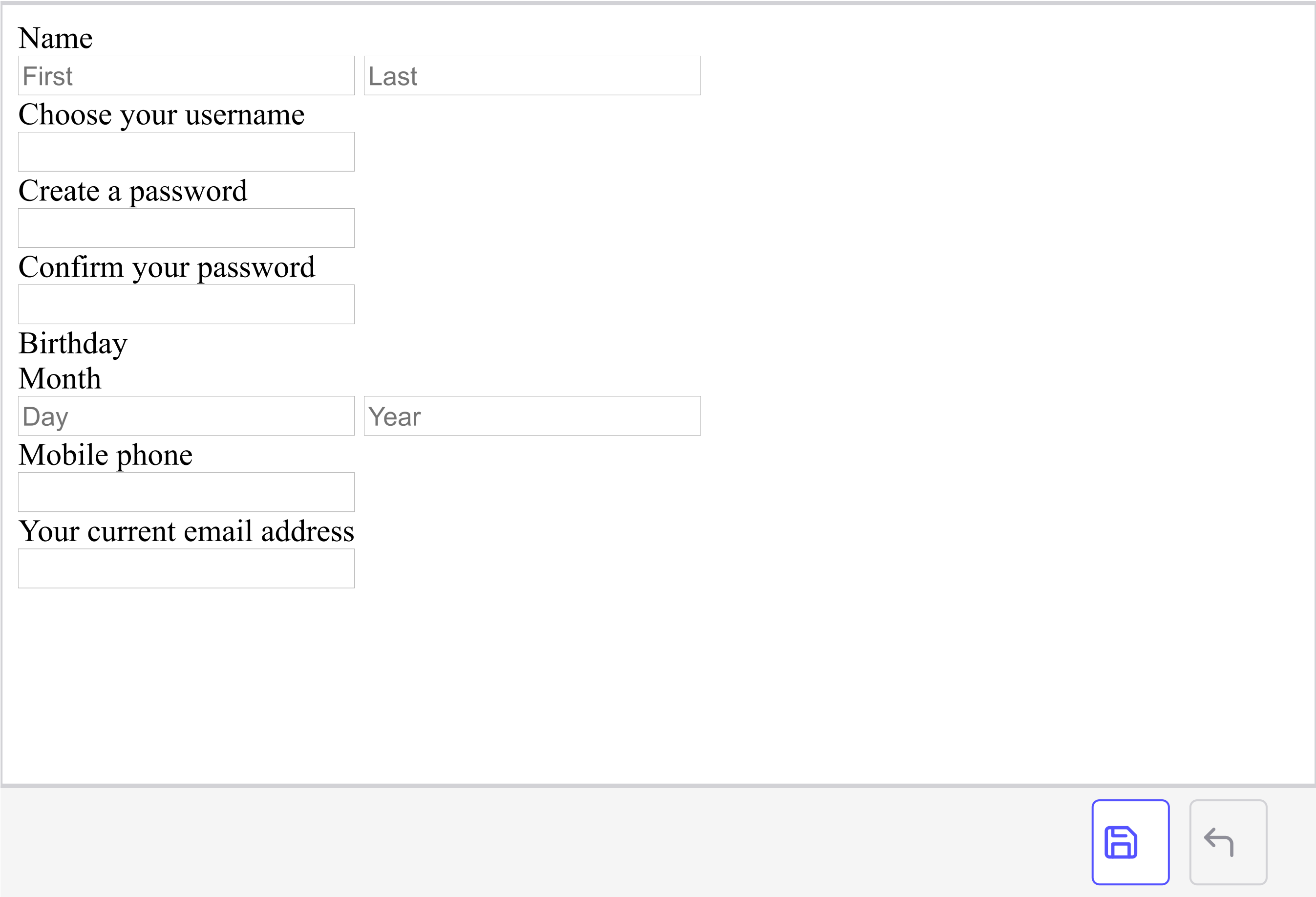
The HTML

As always, we'll start with the HTML and layout some basic elements for us to work with.

In general, as you're developing on the frontend, you'll find an iterative process with short feedback loops to be the fastest way of progressing. If you're building a JavaScript library, that means continuously `console.log`ing or writing small tests after completion of a piece. In our case, we're working towards validating inputs, so it makes sense to have some to work with.

Output

HTML



Name

First Last

Choose your username

Create a password

Confirm your password

Birthday

Month

Day Year

Mobile phone

Your current email address

`<div>` vs `<label>`

Forms are an exciting part of HTML with many surprises. You might have seen `label` and other semantic elements around in tutorials for forms, but what's the advantage of doing so over using plain `div`?

While different elements usually make no difference in how they're presented to the user, they can have slightly different user experiences. For example, you can define a `div` and an `input` and the `div` acts as a "label" for the div, but using a `label` element will give the additional functionality that when you click on the text, the input for that label is focused. Of course, this can be powered by JavaScript, too.

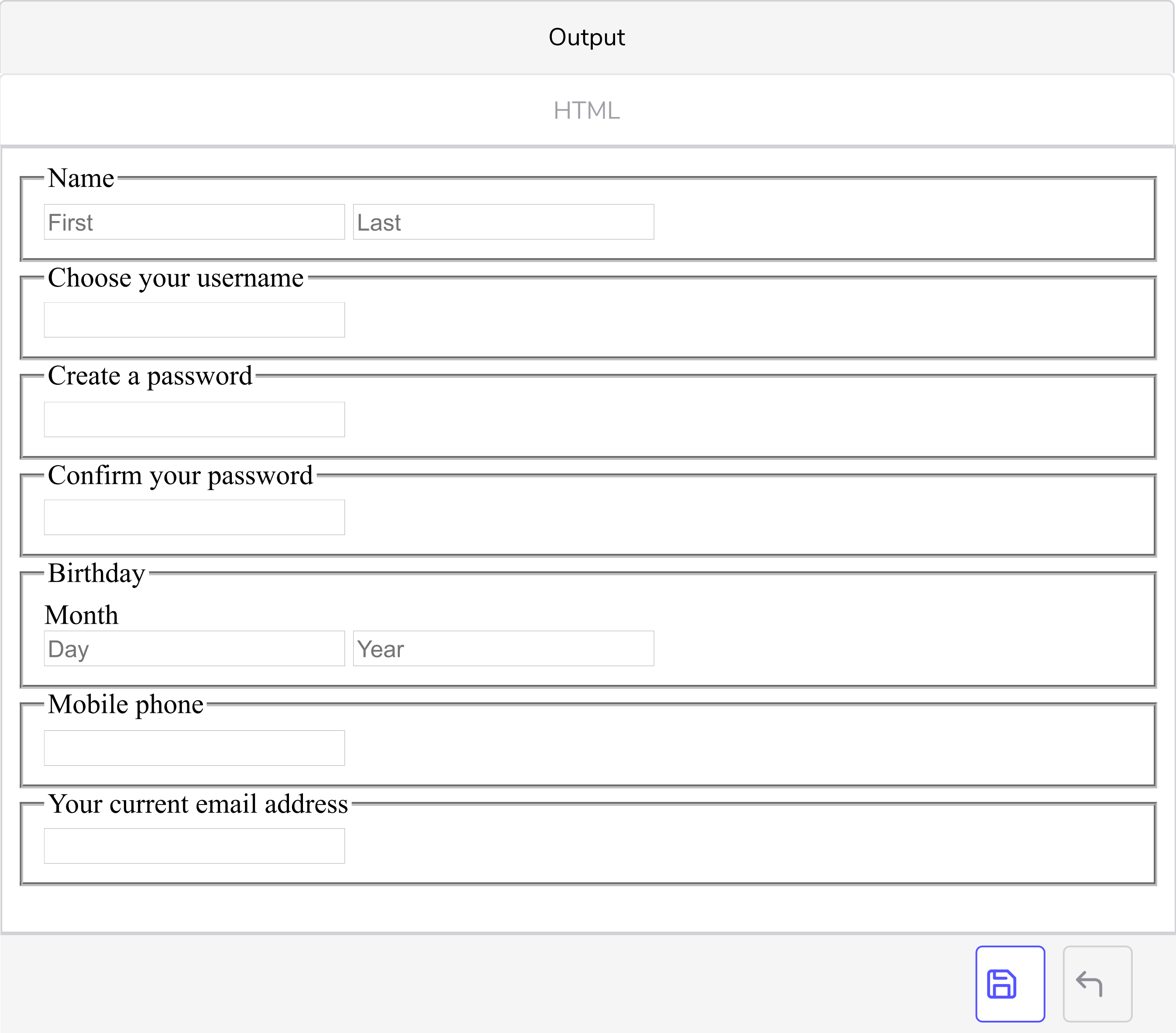
Another benefit that semantic labels give is greater accessibility. For example, screen readers for the visually-impaired will interpret different tags uniquely. This is often a neglected aspect of web development but one for which companies often have entire teams. We won't touch on this too much, but keep in mind that you should take care to test your web app on screen readers and such.

Finally, semantic elements can have platform-specific ramifications. When

Finally, semantic elements can have platform specific ramifications. When you wrap a set of inputs within a `form`, your mobile browsers will often add a special button like “Go” on the keyboard displayed for users to easily submit.

Switching to semantic tags

Since there’s a little downside, let’s switch our implementation to use semantic tags.



Output

HTML

Name

First Last

Choose your username

Create a password

Confirm your password

Birthday

Month

Day Year

Mobile phone

Your current email address

The default styling has changed a bit, but that’ll be easy customization. The entire thing is wrapped in a `form` element, each distinctive input set is wrapped in a `fieldset`, headers are within `legend`s, and inputs get an `aria-label` tag (for accessibility). This is *not* comprehensive towards checking all the boxes for accessibility, but it’s a start.

What’s up with the birthday month?

Hopefully, you looked at the HTML and noticed that the month still uses a `div` element and a hidden input field. This should seem strange to you. It’s one element and a hidden input field. This should seem strange to you. It's one area where I believe HTML and CSS can be improved. As of this writing, there isn't a way to really customize the dropdown menu of a `select` element (the element that gives a dropdown menu in forms) with plain HTML and CSS. You need JavaScript to do so. So we'll use a `div` as the presentation layer of selecting months, dynamically show a dropdown menu with JavaScript when clicked, and whichever menu item is selected, we'll change both the div and the hidden input (so that it's included in the data set when submitted).

In the next lesson, we'll learn about validation checks and errors that will become part of the form.