Weather app in React

Starting to write our first React.js application, we learn how to structure our app and how to fetch data from an API!

Let's use our knowledge to write an actual app! What we'll build is a weather application that'll display the current conditions and a 7 day forecast. Again, here is a live preview of what we're going to build:

Let's use the knowledge that we gained in the first part of this course and create a barebones *Hello World Weather app* that displays a heading saying

“Weather”!

*As we are now going to write a larger App, we are also going to split our code into multiple files. Now **index.js** will be the entry point and include the logic for rendering the App component. We will start writing our own code in **App.js***

We’ll need a bit of styling to make sure our app looks good. We’ve prepared that for you so you can focus on React. Just view the styling in the above example if you are interested.

We’ll also need to be able to tell our app for which location we want the weather, so let’s add a form with an input field and label that says “City, Country”!

> We nest the `input` inside the `label` so the input is focussed when users click on the label!

When entering something into the input field and pressing “Enter”, the page refreshes and nothing happens. What we really want to do is fetch the data when a city and a country are input. Let’s add an `onSubmit` handler to the `form` and a `fetchData` function to our component!

If you enter a City/Country name in the above form and press Enter, you will see the console.log statement in your browser console

By running `evt.preventDefault()` in fetchData (which is called when we press enter in the form), we tell the browser to not refresh the page and instead ignore whatever it wanted to and do what we tell it to. Right now, it logs “fetch weather data!” in the console over and over again whenever you submit the form. How do we get the entered city and country in that function though?

By storing the value of the text input in our local component state, we can grab it from that method. When we do that, we make our input a so-called

“controlled input”.

We’ll store the currently entered location in `this.state.location`, and add a utility method to our component called `changeLocation` that is called `onChange` of the text input and sets the state to the current text:

```
class App extends React.Component {
  fetchData = (evt) => { /* … */ };

  changeLocation = (evt) => {
    this.setState({
      location: evt.target.value
    });
  };

  render() {
    return (
      <div>
        <h1>Weather</h1>
        <form onSubmit={this.fetchData}>
          <label>I want to know the weather for
            <input
              placeholder={"City, Country"}
              type="text"
              value={this.state.location}
              onChange={this.changeLocation}
            />
          </label>
        </form>
      </div>
    );
  }
}
```

As mentioned in Chapter 1, when saving anything to our local state, we have to predefine it. Let’s do that:

```
class App extends React.Component {
  state = {
    location: ''
  };

  ...
}
```

In our fetchData function, we can then access `this.state.location` to get the current location:

```
class App extends React.Component {
  state = { /* … */ };

  fetchData = (evt) => {
    evt.preventDefault();
    console.log('fetch data for', this.state.location);
  };

  changeLocation = (evt) => { /* … */ };

  render() { /* … */ }
}
```

Here's the full working example of what we've learned so far

Now, whichever location you enter it should log "fetch data for <YourCity>, <YourCountry>!