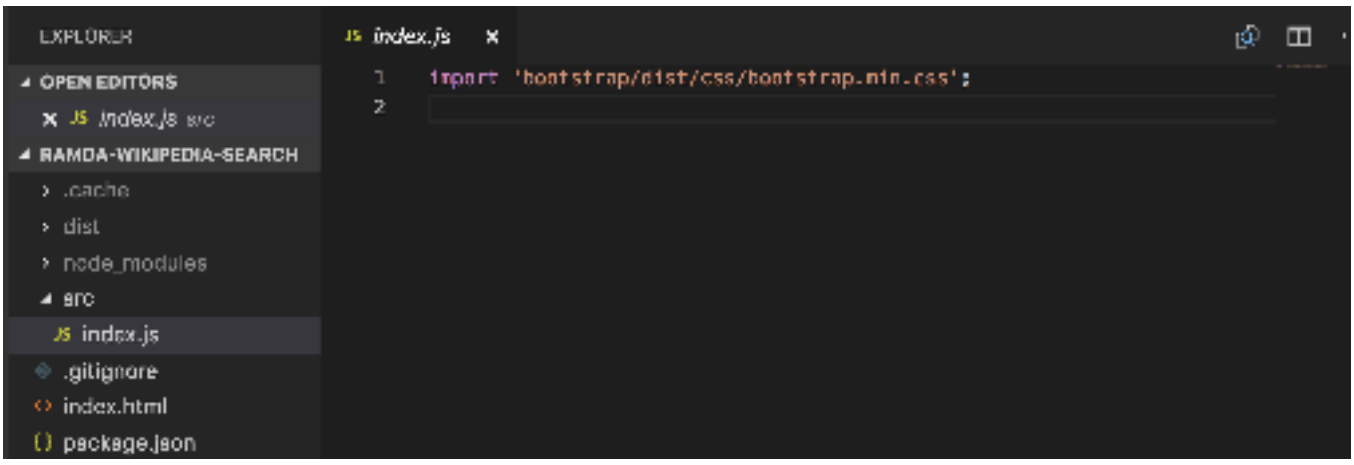
Getting the Input Value

Using Ramda to get and trim the input's text value, setting a default if nothing's passed. (5 min. read)

Here's the initial app.

To capture the user's input as they type, our `input` element needs an event listener.

Your `src/index.js` file is already hooked up and ready to go. You'll notice we imported Bootstrap for styling.



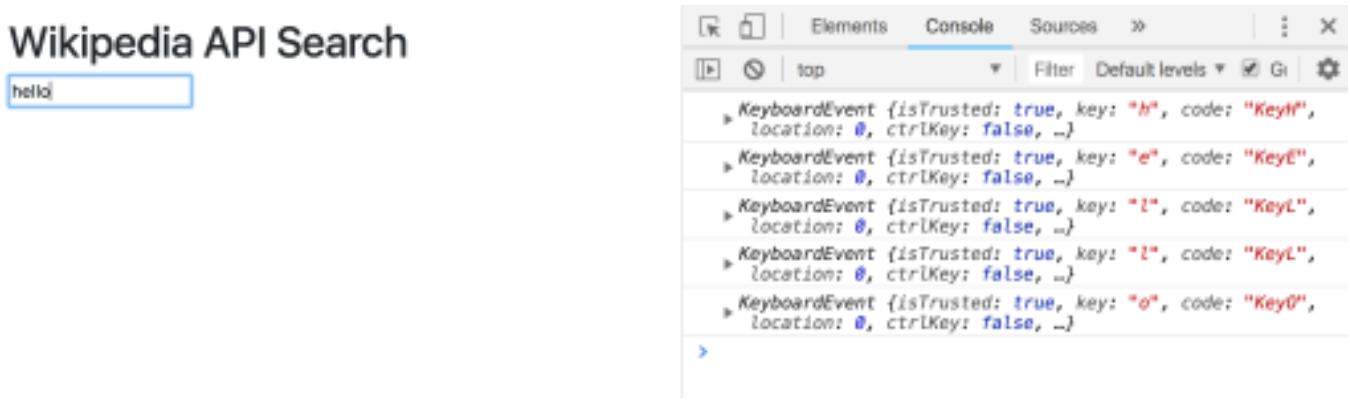
Let's add a dummy event listener to get things going.

```
import 'bootstrap/dist/css/bootstrap.min.css';

const inputElement = document.querySelector('input');

inputElement.addEventListener('keyup', console.log);
```

This `console.log`s whatever you type. You can find the logs in DevTools of course.



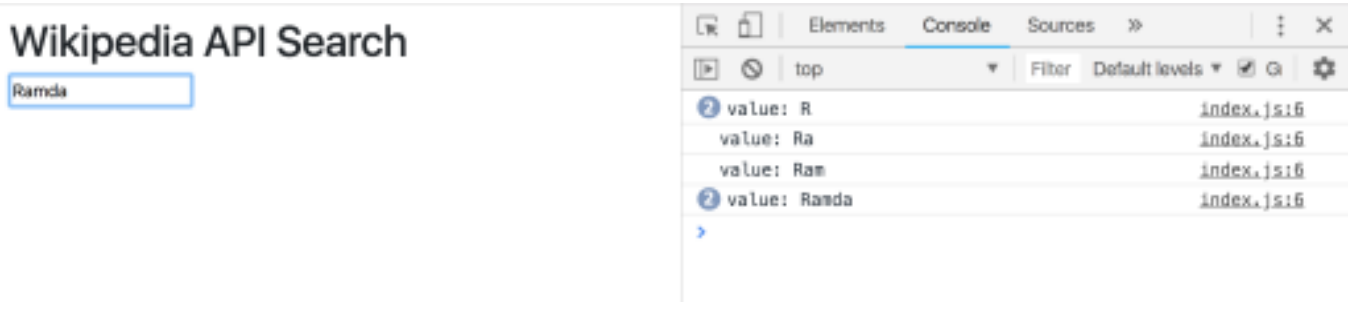
You probably know that the standard way to access an `input`'s value is through `event.target.value`. Let's edit the event handler to log it out.

```
import 'bootstrap/dist/css/bootstrap.min.css';

const inputElement = document.querySelector('input');

inputElement.addEventListener('keyup', event => {
  console.log('value:', event.target.value);
});
```

Now it shows the value.



How can Ramda help us achieve the following?

- Grab `event.target.value`
- Trim the output (strip leading/trailing whitespace)

- Default to empty string if `undefined`

The `pathOr` function can actually handle the first and third bullet points. It takes three parameters: the default, the path, and the data.

So the following works perfectly

```
import { pathOr } from 'ramda';

const getInputValue = pathOr('', ['target', 'value']);
```

If `event.target.value` is `undefined`, we'll get an empty string back!

Ramda also has a `trim` function, so that solves our whitespace issue.

```
import { pathOr, trim } from 'ramda';

const getInputValue = (event) => trim(pathOr('', ['target', 'value'], event));
```

Oh come on, we know better...let's combine them with `pipe`!

```
import { pathOr, pipe, trim } from 'ramda';

const getInputValue = pipe(
  pathOr('', ['target', 'value']),
  trim
);
```

We now have a mini pipeline that takes an `event` object, grabs its `target.value`, defaults to `''`, and trims it.

Beautiful.

I recommend storing this in a separate file.

index.js

getInputValue.js

```
import { pathOr, pipe, trim } from 'ramda';

export default pipe(
  pathOr('', ['target', 'value']),
  trim
);
```