More on CSS Modules

CSS Modules are a relatively new concept (you can dive super-deep into them here: https://github.com/css-modules/css-modules). With CSS modules, you can write normal CSS code and make sure, that it only applies to a given component.

It's not using magic for that, instead it'll simply **automatically generate unique CSS class names** for you. And by importing a JS object and assigning classes from there, you use these dynamically generated, unique names. So the imported JS object simply exposes some properties which hold the generated CSS class names as values.

Example:

In Post.css File

```
.Post {
    color: red;
}
```

In Post Component File

```
import classes from './Post.css';

const post = () => (
    <div className={classes.Post}>...</div>
);
```

Here, `classes.Post` refers to an automatically generated `Post` property on the imported `classes` object. That property will in the end simply hold a value like `Post__Post__ah5_1` .

So your `.Post` class was automatically transformed to a different class (`Post__Post__ah5_1`) which is unique across the application. You also can't use it accidentally in other components because you don't know the generated string! You can only access it through the `classes` object. And if you import the CSS file (in the same way) in another component, the `classes` object there will hold a `Post` property which yields a **different** (!) CSS class name. Hence it's scoped to a given component.

By the way, if you somehow also want to define a global (i.e. un-transformed) CSS class in such a `.css` file, you can prefix the selector with `:global` .

Example:

```
:global .Post { ... }
```

Now you can use `className="Post"` anywhere in your app and receive that styling.

Debugging & Error Boundries:

Useful Resources & Links

- Error Boundaries: https://reactjs.org/docs/error-boundaries.html
- Chrome Devtool Debugging: https://developers.google.com/web/tools/chrome-devtools/javascript/