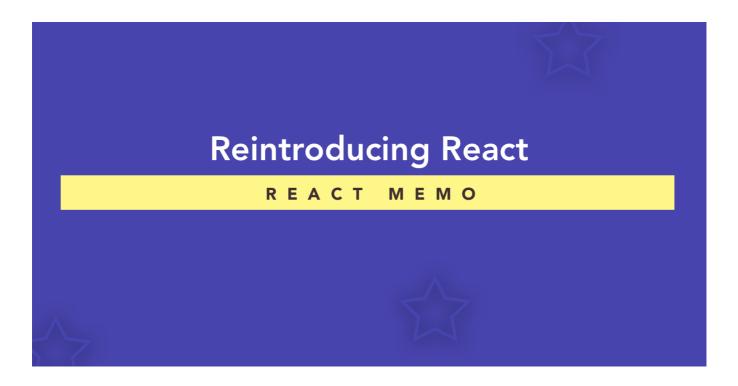
React.memo === Functional PureComponent

We will discuss why refactoring is not the most suitable method for larger React apps.



A few weeks ago, John refactored the Benny component to a PureComponent.

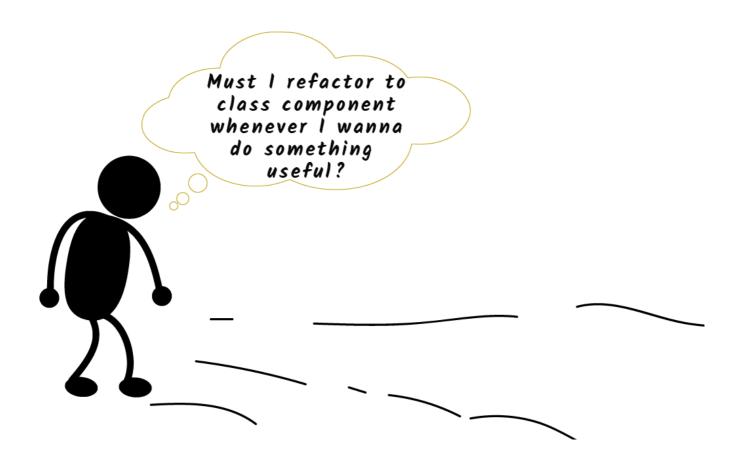
Here's what his change looked like:

```
+ import { PureComponent } from 'react'

- class Benny extends Component {
+ class Benny extends PureComponent {
   render () {
      return <Consumer>
      {position => <svg />}
      </Consumer>
   }
}
```

That looks good.

However, the only reason he refactored the Benny component to a class component was because he needed to extend React.PureComponent.



The solution works, but what if we could achieve the same effect without having to refactor from functional to class components?

Refactoring large components just because you need a specific React feature isn't the most pleasant of experiences.

In the next lesson, let's see how we can use React.memo instead of extending class from PureComponent.