Possible Gotcha

This lesson teaches us about some of the rules and conventions of working with Redux. Instead of directly accessing the state, we use the store to receive the state as props.

In the just concluded Hello World example, a possible solution you may have come up with for grabbing the state from the STORE may look like this:

```
class App extends Component {
  state = store.getState();
  render() {
    return <HelloWorld tech={this.state.tech} />;
  }
}
```

What do you think? Will this work? Just as a reminder, the following two ways are correct ways to initialise a React component's state.

(a)

```
class App extends Component {
 constructor(props) {
   super(props);
   this.state = {}
  }
}
```

(b)

```
class App extends Component {
  state = {}
}
```

So, back to answering the question, yes, the solution will work just fine.

`store.getState()` will grab the current state from the *Redux STORE*.

However, the assignment, `state = store.getState()` will assign the state gotten from Redux to that of the < App /> component.

gotten from Redux to that of the "App" component.

By implication, the return statement from render i.e `< HelloWorld tech={this.state.tech} />` will be valid.

Note that this reads `this.state.tech` NOT `store.getState().tech`. Even though this works, it is against the ideal philosophy of Redux.

If within the app, you now run, `this.setState()`, the App's state will be updated without the help of REDUX.

This is the default React mechanism, and it isn't what you want. You want the state managed by the Redux STORE to be the single source of truth.

Whether you're retrieving state, as in `store.getState()` or updating/changing state (as we'll cover later) you want that to be entirely managed by Redux, not by `setState()`.

Since Redux manages the app's state, all you need to do is feed in the state from the Redux STORE as props to any required component.

Another big question you're likely asking yourself is, why did I have to go through all this stress just to have the state of my App managed by Redux?

Reducer, Store, createStore blah, blah, blah ...

Yeah, I get it.

I felt that way too.

However, consider the fact that you do not just go to the bank and NOT follow a due process for withdrawing your own money. It's your money, but you do have to follow a due process.

The same may be said for Redux.

Redux has its own "process" of doing things. We've got to learn how that works - and hey, you're not doing bad!