React Native - Basics - Text Input

- Dr Nick Hayward

A brief intro to the basics of text input in React Native app development.

Contents

Intro

React Native includes a default component to handle user text input.

The component `TextInput` is similar to a standard input field, allowing a user to simply enter any required text content.

General usage

To use `TextInput` with an app, we need to add the default module from React Native as part of the standard `import` statement,

e.g.

```
import {
  AppRegistry,
  Platform,
  StyleSheet,
  Text,
  TextInput,
  View
} from 'react-native';
```

The `TextInput` component includes a useful *prop*, `onChangeText`, which accepts a callback function for each time the text is changed in the input field.

Likewise, it also includes a complementary *prop*, `onSubmitEditing`, to handle text as it is submitted, again using a defined callback function.

Props usage

So, we might accept user text input for a given value, such as a name, place, &c.

Then, we can dynamically update the *view*.

So, we can initially setup our `TextInput` component as follows,

e.g.

```
<TextInput
  style={styles.textInput}
  placeholder={this.state.quoteInput}
  onChangeText={(quoteText) => this.setState({quoteText})}
/>
```

n.b. For styling this component, separate from the parent `View`, we need to ensure a minimum height of 40 to ensure the text is not cut off at the top of each character.

Props and State

This example relies upon calling and setting state for the app, relative to the `TextInput` and various `Text` components.

In a simple constructor for this app, we can pass required `props` and define intial values for `state`,

e.g.

```
export default class TextUpdater extends Component {
  constructor(props) {
    super(props);
    this.state = {
      quoteInput: 'enter a favourite quotation...',
      quoteText: 'the unexamined life is not worth living...'
    };
  }
}
```

We can then use the properties on `state` to set initial values for the text input field and the text output,

e.g.

```
<TextInput
  style={styles.textInput}
  placeholder={this.state.quoteInput}
  onChangeText={(quoteText) => this.setState({quoteText})}
/>
```

and

```
<Text style={styles.content}>
        {this.state.quoteText}
</Text>
```

So, as a user enters their text in the input field, we can dynamically set state for the property `quoteText`. This will then trigger a request to update state, which will eventually create an update for parts of the app that use `state.quoteText`.

n.b. don't forget - `setState` may not automatically trigger an update to the contents of the rendered app. React Native will update the app via state when it is most efficient.