React Native

Text input

TextInput component

- subscribe to the onChangeText events to read the user input.
- other events: onSubmitEditing and onFocus
- Two methods exposed via the native element are .focus() and .blur()
  - will focus or blur the TextInput programmatically.
- some props are only available with multiline={true/false}
- border styles that apply to only one side of the element (e.g., borderBottomColor, borderLeftWidth, etc.) will not be applied if multiline=false.
- you can wrap your TextInput in a View

Basic input

```
import React, { Component } from 'react';
import { AppRegistry, TextInput } from 'react-native';


export default class UselessTextInput extends Component {
  constructor(props) {
    super(props);
    this.state = { text: 'Useless Placeholder' };
  }
  render() {
    return (
      <TextInput
        style={{height: 40, borderColor: 'gray', borderWidth: 1}}
        onChangeText={(text) => this.setState({text})}
        value={this.state.text}
      />
    );
  }
}
```

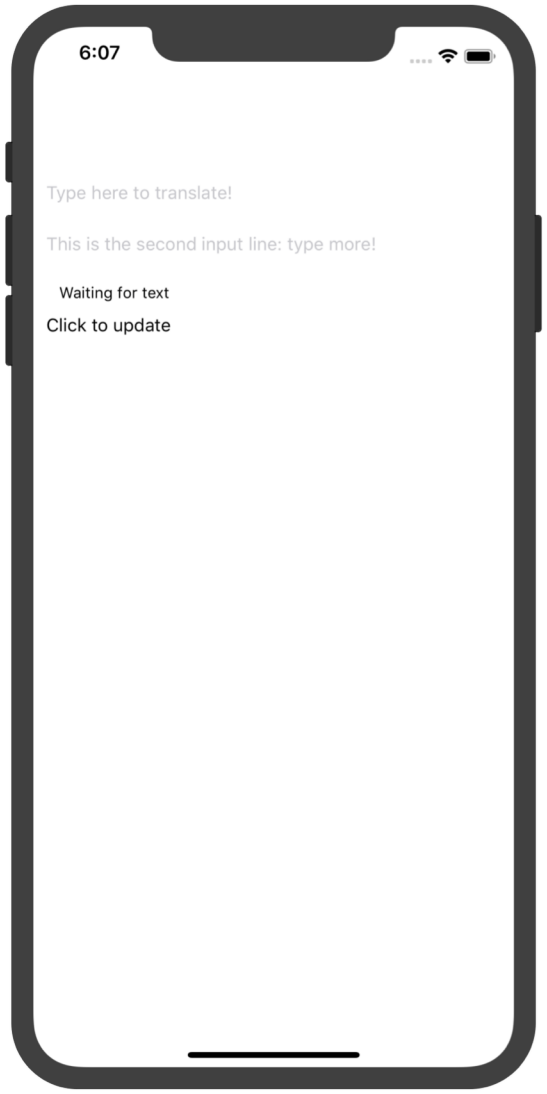
This is the name of the state variable that will receive the text from the TextInput Box.

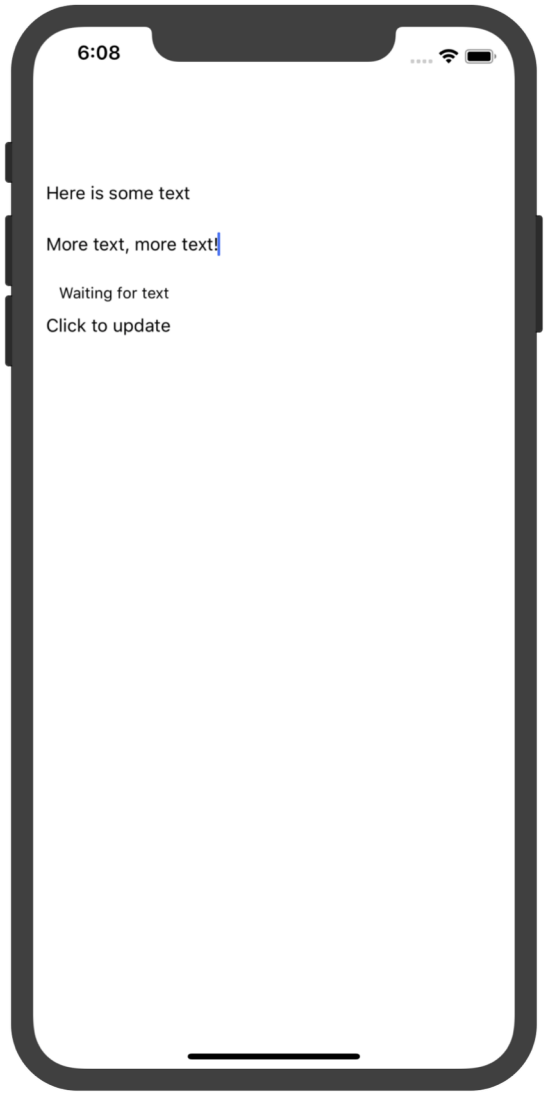
Example 2

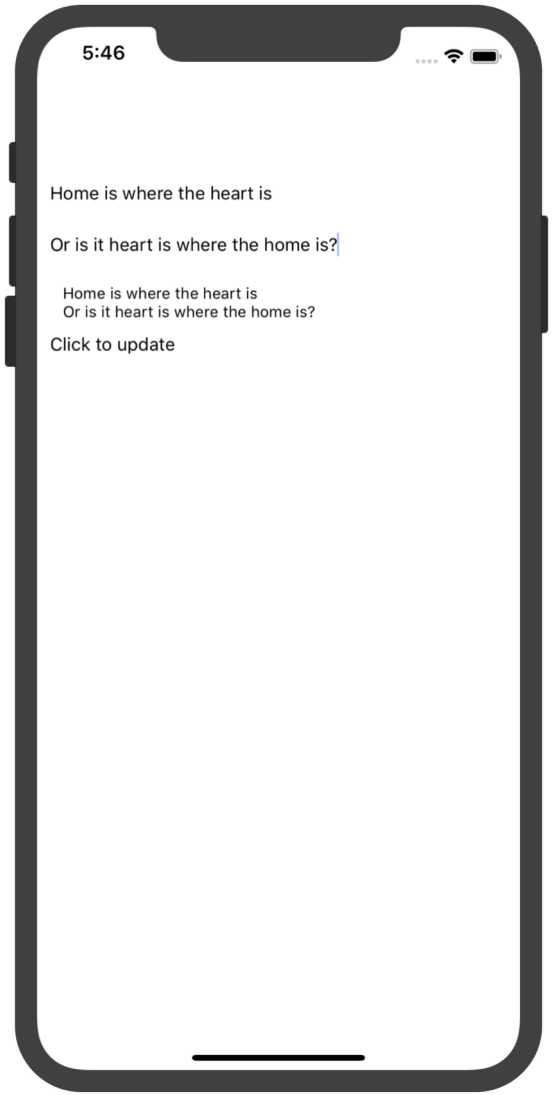
```
import React, { Component } from 'react';
import { AppRegistry, View, TextInput } from 'react-native';

class UselessTextInput extends Component {
  render() {
    return (
      <TextInput
        {...this.props} // Inherit any props passed to it; e.g., multiline, numberOfLines below
        editable = {true}
        maxLength = {40}
      />
    );
  }
}

export default class UselessTextInputMultiline extends Component {
  constructor(props) {
    super(props);
    this.state = {
      text: 'Useless Multiline Placeholder',
    };
  }
  // If you type something in the text box that is a color, the background will change to that
  // color.
  render() {
    return (
      <View style={{
        backgroundColor: this.state.text,
        borderBottomColor: '#000000',
        borderBottomWidth: 1 }}
      >
        <UselessTextInput
          multiline = {true}
          numberOfLines = {4}
          onChangeText={(text) => this.setState({text})}
          value={this.state.text}
        />
      </View>
    );
  }
}
```

Example 3...click to get text

Will put code in two separate files:
Home.js
App.js







This code is in file Home.js in the same directory as App.js

Example 3...click to get text

```
import React, { Component } from 'react';
import { AppRegistry, Text, TextInput, View } from 'react-native';
```

Must export this class for it to be found in App.js

```
export default class Home extends Component {
  constructor(props){
    super(props);
    this.state = { myState: this.props.origText, myState2: this.props.origText2,
myState3:'Waiting for text'};
  }
  updateState = () => {
      this.setState({ myState3: this.state.myState + '\n' + this.state.myState2})
  };
```

Initialize all 3 variables

Event handler for onPress event
Update myState3 with the values from the fields (stored in myState1 and myState2)

Two TextInput boxes; update different state as enter

```
render() {
  return (
  <View style={{padding: 10, marginTop: 100}}>
    <TextInput
      style={{height: 40}}
      placeholder="Type here to translate!"
      onChangeText={(myState) => this.setState({myState})}
    />
    <TextInput
      style={{height: 40}}
      placeholder="This is the second input line: type more!"
      onChangeText={(myState2) => this.setState({myState2})}
    />
    <Text style={{padding: 10, fontSize: 12}}>
      {this.state.myState3}
    </Text>
    <Text onPress = {this.updateState}>
        Click to update
    </Text>
  </View>
  );
 }
}
```

Text field for output

Text field to act as a button

This code is in file App.js

Main class

```
import React, { Component } from 'react';
import { AppRegistry, Text, TextInput, View } from 'react-native';
import Home from './Home.js'
export default class saveHome extends Component {
  constructor(props) {
    super(props);
    this.state = {text: ''};
  }
  render() {
    return (
        <Home origText='Home is where your apps are!' origText2='To be or not to be'/>
    );
  }
}
```

Must import the Home class. Note the './' to indicate that the file is in the same directory as the App.js file

Note: if you do nothing, keyboard should automatically dismiss when text is submitted (enter pressed)

Dismissing the keyboard

Must import the keyboard component

```
import React, { Component } from 'react';
import { Keyboard, TextInput } from 'react-native';

class Example extends Component {
  componentDidMount () {
    this.keyboardDidShowListener = Keyboard.addListener('keyboardDidShow', this._keyboardDidShow);
    this.keyboardDidHideListener = Keyboard.addListener('keyboardDidHide', this._keyboardDidHide);
  }

  componentWillUnmount () {
    this.keyboardDidShowListener.remove();
    this.keyboardDidHideListener.remove();
  }
```

Don't need listeners to dismiss keyboard

Must add keyboard listeners when this component is loaded
Remove listener when this component is unloaded.

Need these handlers since we use them

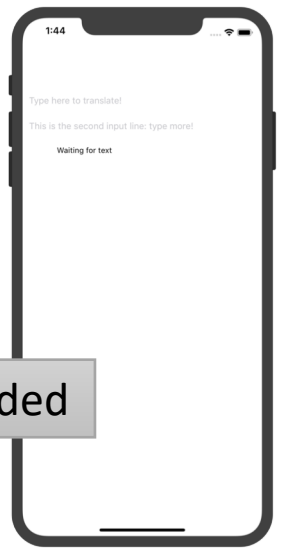
```
_keyboardDidShow () {


}
_keyboardDidHide () {


}
render() {
  return (
    <TextInput
      onSubmitEditing={Keyboard.dismiss}
    />
  ); } }
```

Dismiss keyboard when enter pressed

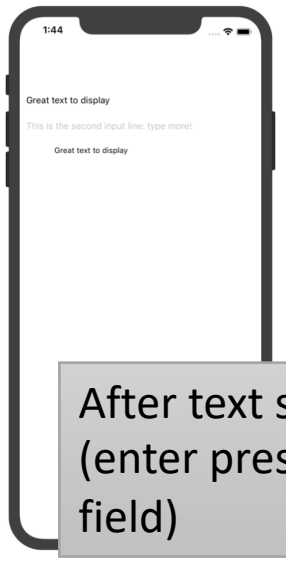
This is called from same App.js as example 3.

Example 4...using submit

```
import React, { Component } from 'react';
import { AppRegistry, Text, TextInput, View } from 'react-native';

export default class Home extends Component {
  constructor(props){
    super(props);
    this.state = { myState: this.props.origText, myState2:
this.props.origText2, myState3:'Waiting for text'};
  }
  updateState = event => {
      this.setState({ myState3: event.nativeEvent.text})
  };
 render() {
    return (
    <View style={{padding: 10, marginTop: 100}}>
      <TextInput
        style={{height: 40}}
        placeholder="Type here to translate!"
        onSubmitEditing ={this.updateState}
      />
      <TextInput
        style={{height: 40}}
        placeholder="This is the second input line: type more!"
        onSubmitEditing ={this.updateState}
      />
      <Text style={{padding: 10, fontSize: 12}}>
        {this.state.myState3}
      </Text>
    </View>
    );    }    }
```



When loaded



After text submitted (enter pressed in top field)