Lesson:

Button

PW SKILLS

The Button component in React Native is a pre-styled button that allows users to trigger actions or events when pressed. It provides a simple way to add interactive elements to your application without the need for custom styling.

Using the Button Component

Here's an example of using the Button component:

```
import React from 'react';
import { View, Button, StyleSheet } from 'react-native';

const App = () => {
  const handlePress = () => {
    console.log('Button pressed!');
  };

  return (
    <View style={styles.container}>
      <Button title="Press Me" onPress={handlePress} />
    </View>
  );
};

const styles = StyleSheet.create({
  container: {
    flex: 1,
    justifyContent: 'center',
    alignItems: 'center',
  },
});

export default App;
```

In this example, we have a Button component rendered inside a View component. The title prop is used to set the text displayed on the button. The onPress prop is used to specify the function to be called when the button is pressed. In this case, we have defined a handlePress function that logs a message to the console when the button is pressed.

Press Me

Platform-specific Styling

The Button component automatically applies the platform-specific styling, which means that the appearance of the button will match the design guidelines of the operating system running the app. On iOS, it will have a rounded shape with a background color, while on Android, it will have a rectangular shape with a different background color. This ensures a consistent look and feel across different devices.

The platform-specific styling of the Button component is achieved through the underlying native components provided by React Native. The native components are mapped to the corresponding UI elements of each platform, allowing for seamless integration with the platform's design standards.

Icon Button

To create an icon button using the Button component, you can leverage other components like Icon from popular libraries such as React Native Vector Icons. Here's an example:

```
import React from 'react';
import { View, Button, StyleSheet } from 'react-native';
import Icon from 'react-native-vector-icons/FontAwesome';

const App = () => {
  const handlePress = () => {
    console.log('Button pressed!');
  };

  return (
    <View style={styles.container}>
       <Icon.Button
        name="star"
        backgroundColor="#3b5998"
        size='20'
        color="white"
      >
        Click here!
      </Icon.Button>
    </View>
  );
};

```

```
24 const styles = StyleSheet.create({
25   container: {
26     flex: 1,
27     justifyContent: 'center',
28     alignItems: 'center',
29   },
30 });
31
32 export default App;
```

In this example, we have imported the Icon component from the React Native Vector Icons library. We then use the Icon component. The name prop of the Icon component specifies the icon to be displayed (in this case, a star), while the size and color props determine the size and color of the icon.

By combining the Button component with other components like Icon, you can create customized button components with icons that fit your application's design and requirements.