Lesson:

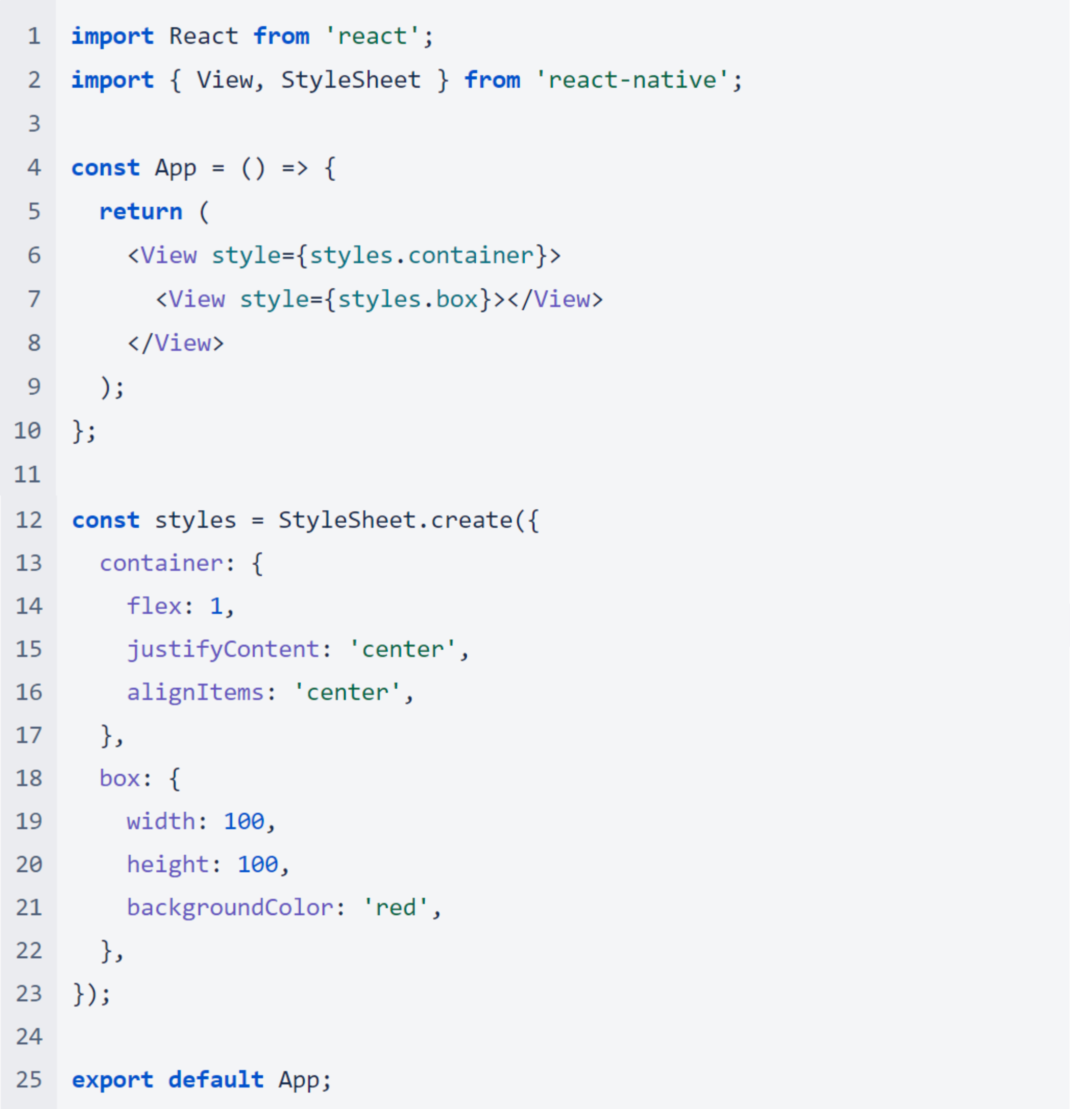
View Component

PHP
CSS
HTML
<HTML>
<HEAD>
<TITLE> Lorem ipsum </TITLE>
</HEAD>
<BODY>
<H1> Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam</H1>
<P> <IMG SRC= "Lorem ipsum"></P>

SKILLS

The View component is a fundamental building block in React Native used for creating the layout and structure of your app. It acts as a container and can be styled to control its appearance and positioning.
Here are a few examples of using the View component:

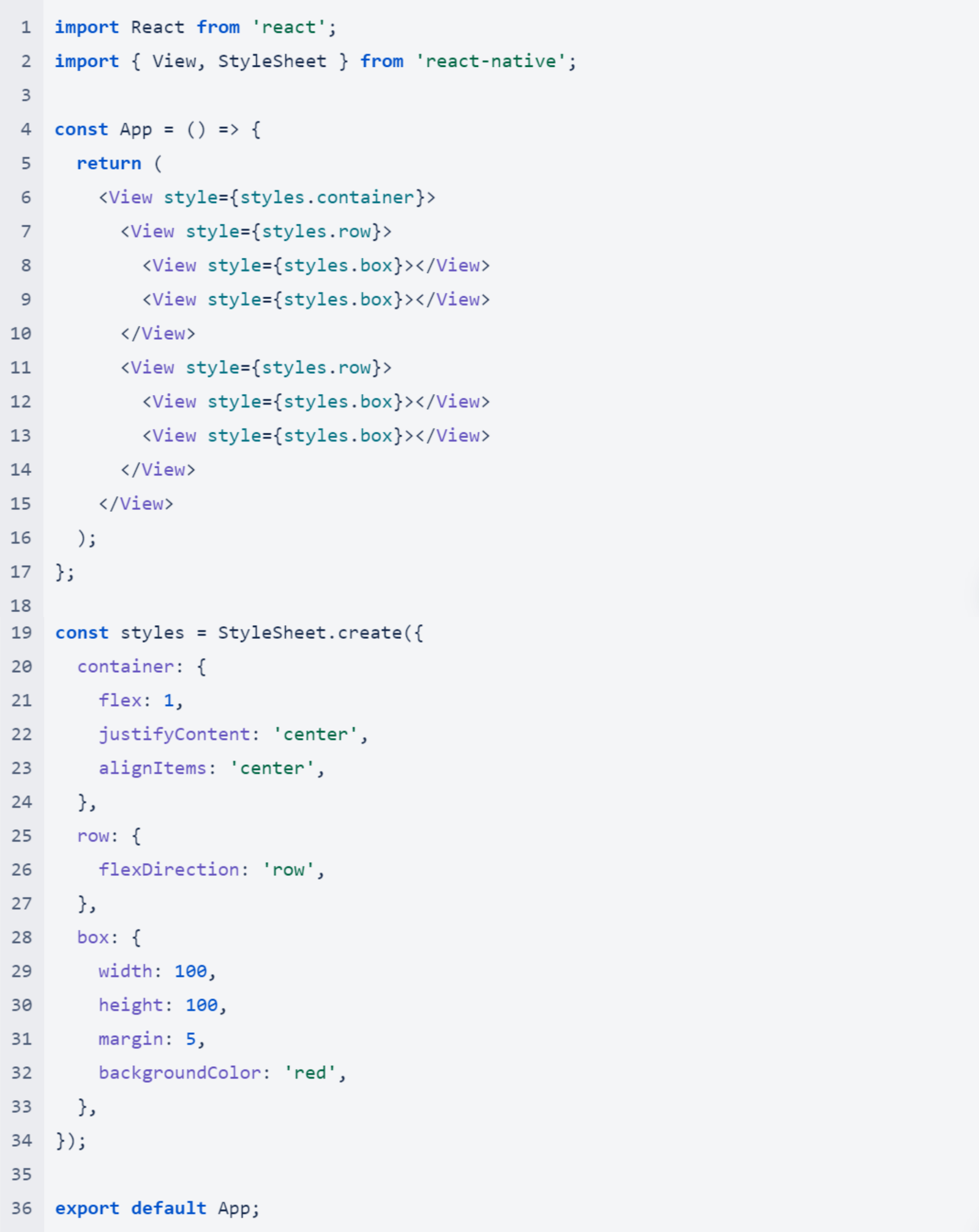
Example 1: Simple View Layout



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

const App = () => {
  return (
    <View style={styles.container}>
      <View style={styles.box}></View>
    </View>
  );
};

const styles = StyleSheet.create({
  container: {
    flex: 1,
    justifyContent: 'center',
    alignItems: 'center',
  },
  box: {
    width: 100,
    height: 100,
    backgroundColor: 'red',
  },
});

export default App;
```

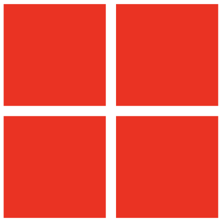
In this example, we have a parent View component called container which is set to flex: 1. This allows it to take up the available space on the screen. The justifyContent and alignItems properties are used to center its content vertically and horizontally. Inside the container, there's another View component called box which represents a red square with a width and height of 100 units.

Example 2: Nested Views

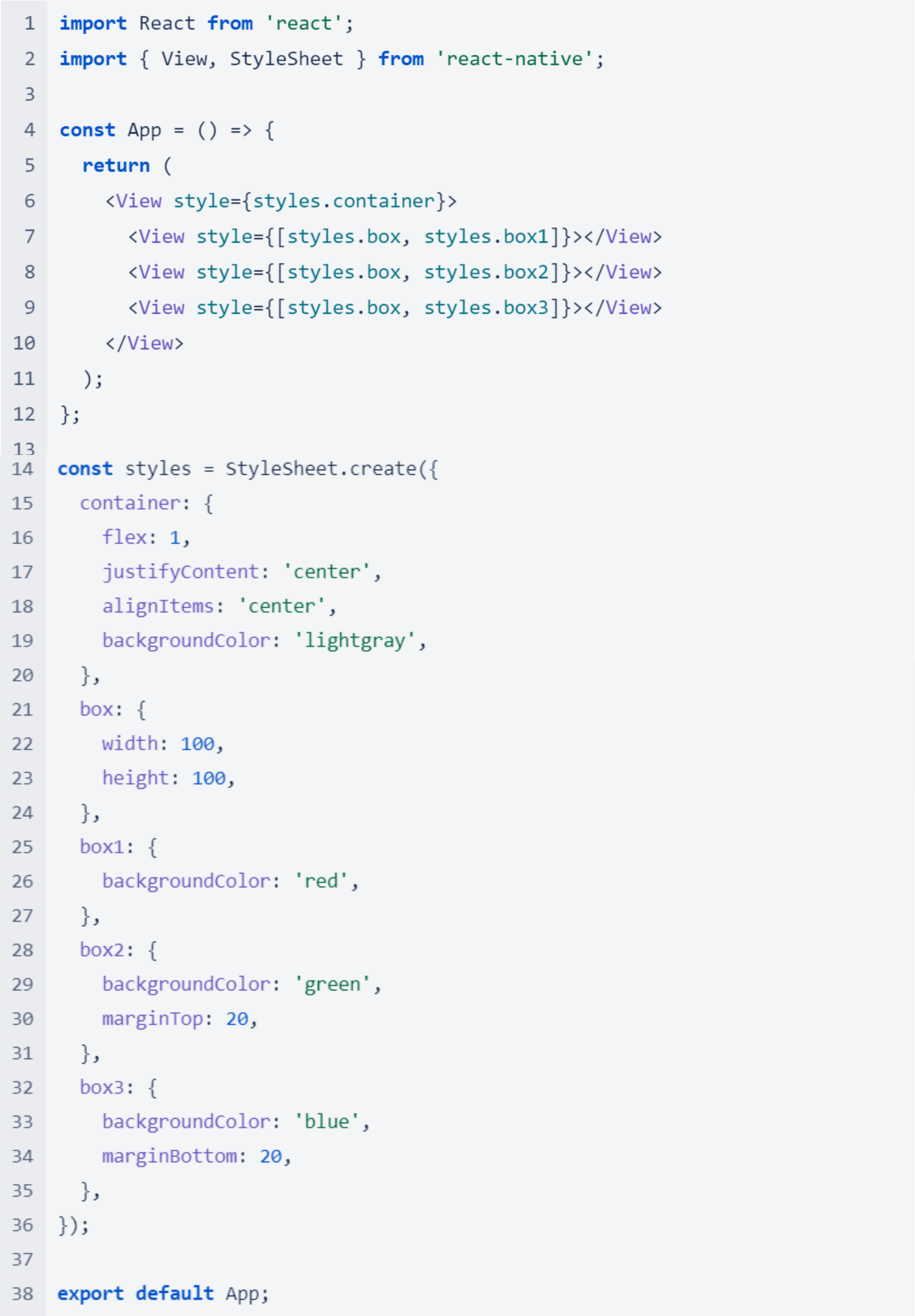


```
import React from 'react';
import { View, StyleSheet } from 'react-native';

const App = () => {
  return (
    <View style={styles.container}>
      <View style={styles.row}>
        <View style={styles.box}></View>
        <View style={styles.box}></View>
      </View>
      <View style={styles.row}>
        <View style={styles.box}></View>
        <View style={styles.box}></View>
      </View>
    </View>
  );
};

const styles = StyleSheet.create({
  container: {
    flex: 1,
    justifyContent: 'center',
    alignItems: 'center',
  },
  row: {
    flexDirection: 'row',
  },
  box: {
    width: 100,
    height: 100,
    margin: 5,
    backgroundColor: 'red',
  },
});

export default App;
```

In this example, we have a parent **View** component called **container** which is styled to center its content. Inside the **container**, there are two **row** components, each representing a row of box components. By using the **flexDirection**: **'row'** property, the **box** components are arranged horizontally within each row. The **margin** property adds some spacing between the boxes.



Example 3: Using View for Styling



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

const App = () => {
  return (
    <View style={styles.container}>
      <View style={[styles.box, styles.box1]}></View>
      <View style={[styles.box, styles.box2]}></View>
      <View style={[styles.box, styles.box3]}></View>
    </View>
  );
};

const styles = StyleSheet.create({
  container: {
    flex: 1,
    justifyContent: 'center',
    alignItems: 'center',
    backgroundColor: 'lightgray',
  },
  box: {
    width: 100,
    height: 100,
  },
  box1: {
    backgroundColor: 'red',
  },
  box2: {
    backgroundColor: 'green',
    marginTop: 20,
  },
  box3: {
    backgroundColor: 'blue',
    marginBottom: 20,
  },
});

export default App;
```

In this example, we have a parent View component called container which is styled with a gray background color. Inside the container, there are three box components, each with a different background color and some additional styling. By using an array of styles, we can apply multiple styles to a single component.

These examples showcase how the View component can be used to create the layout and structure of your app. With the flexibility of styling and nesting, you can build complex UIs by combining and nesting multiple View components.