Core Components
React Native

What is React Native

- an open source framework for building Android and iOS applications using React and the app platform’s native capabilities.

Components [1]

- React Native is like React, but it uses native components instead of web components as building blocks.
- Basic React concepts
  - JSX
  - Components
  - State
  - props

Basic React concepts [1]

- **JSX**

```
<View><Text>Hello world!</Text></View>
```

- This is JSX - a syntax for embedding XML within JavaScript.
- JSX lets you write your markup language inside code.
- It looks like HTML on the web, except instead of web things like <div> or <span>
- In this case, <Text>

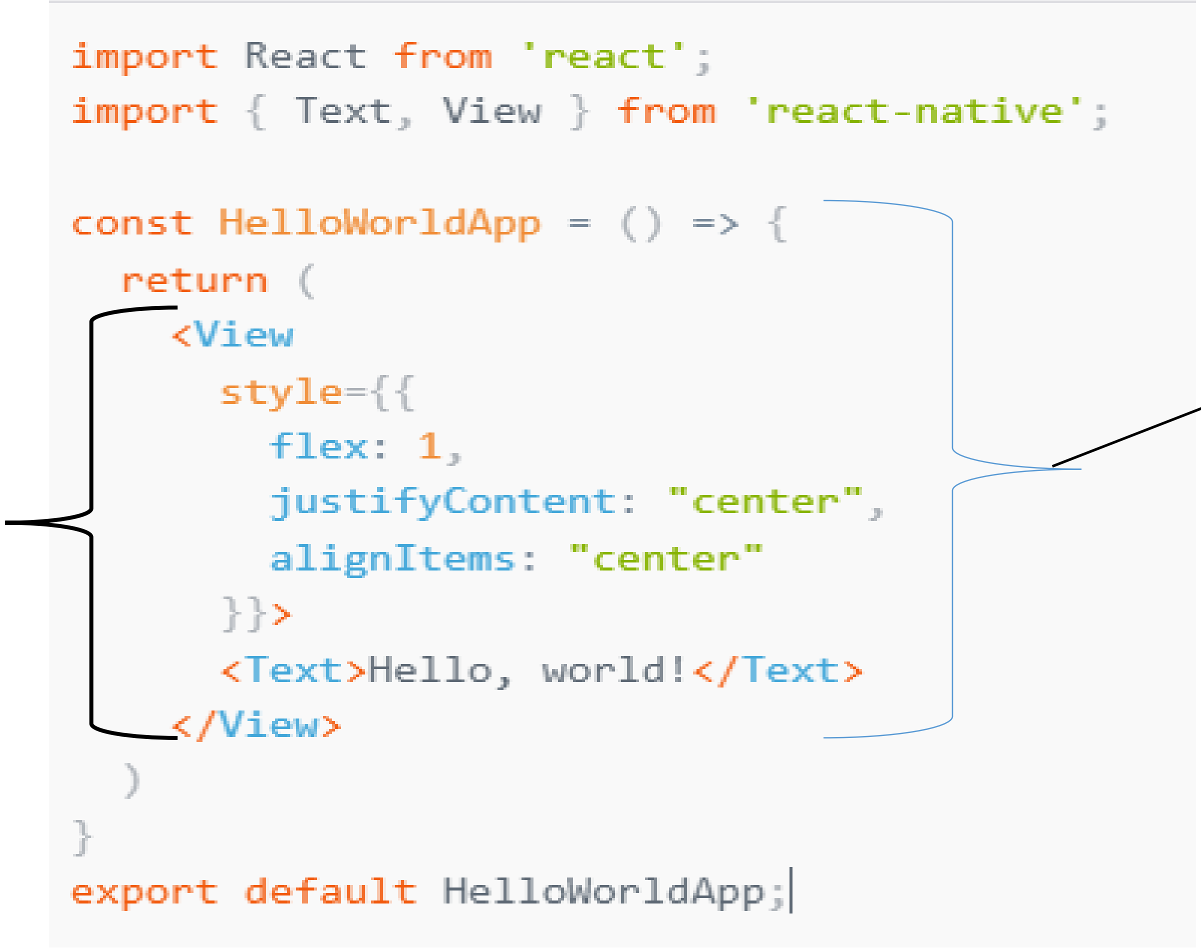
Basic React concepts[1]

- Components
  - Basic building block
  - Anything on the screen is some sort of component.

- Props
  - short for properties
  - Customize components
  - Should not be modified

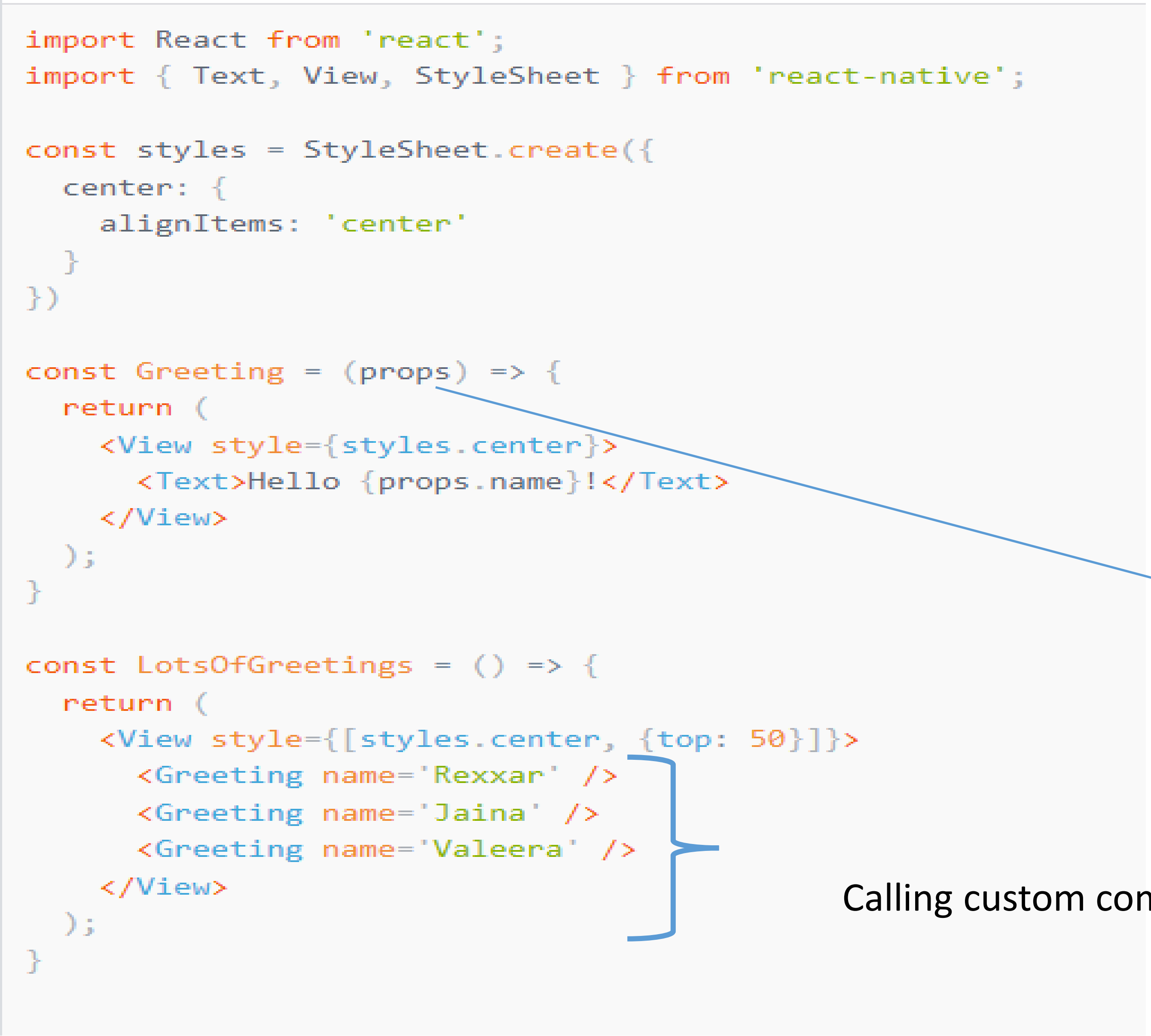
- State
  - allows React components to change their output over time
  - State changes in response to user Input, network response or anything else.

Basic component



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

const HelloWorldApp = () => {
  return (
    <View
      style={{
        flex: 1,
        justifyContent: "center",
        alignItems: "center"
      }}>
      <Text>Hello, world!</Text>
    </View>
  )
}
export default HelloWorldApp;
```

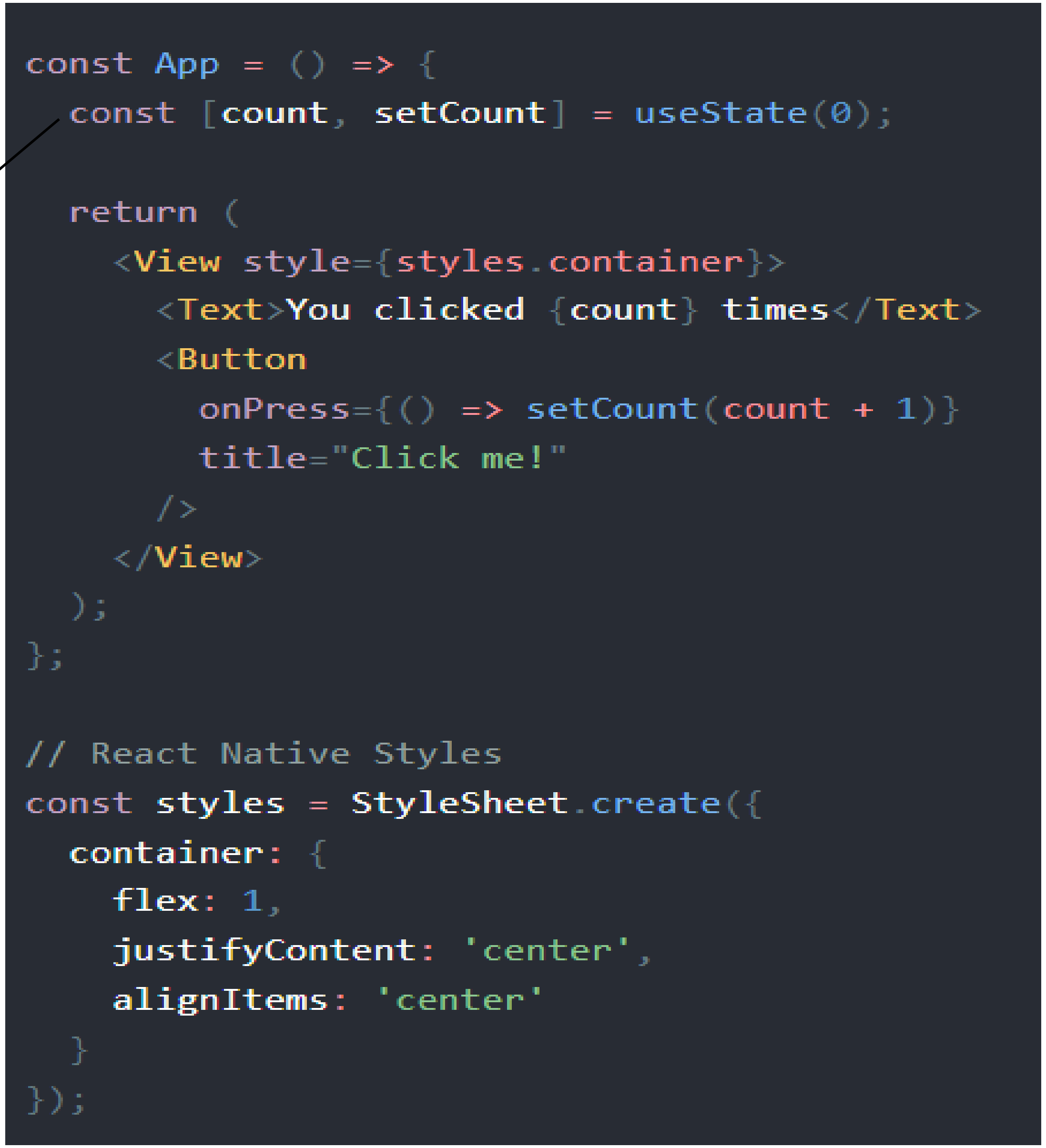
Component(just a function)



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

const styles = StyleSheet.create({
  center: {
    alignItems: 'center'
  }
})

const Greeting = (props) => {
  return (
    <View style={styles.center}>
      <Text>Hello {props.name}!</Text>
    </View>
  );
}

const LotsOfGreetings = () => {
  return (
    <View style={[styles.center, {top: 50}]}>
      <Greeting name='Rexxar' />
      <Greeting name='Jaina' />
      <Greeting name='Valeera' />
    </View>
  );
}
```

props

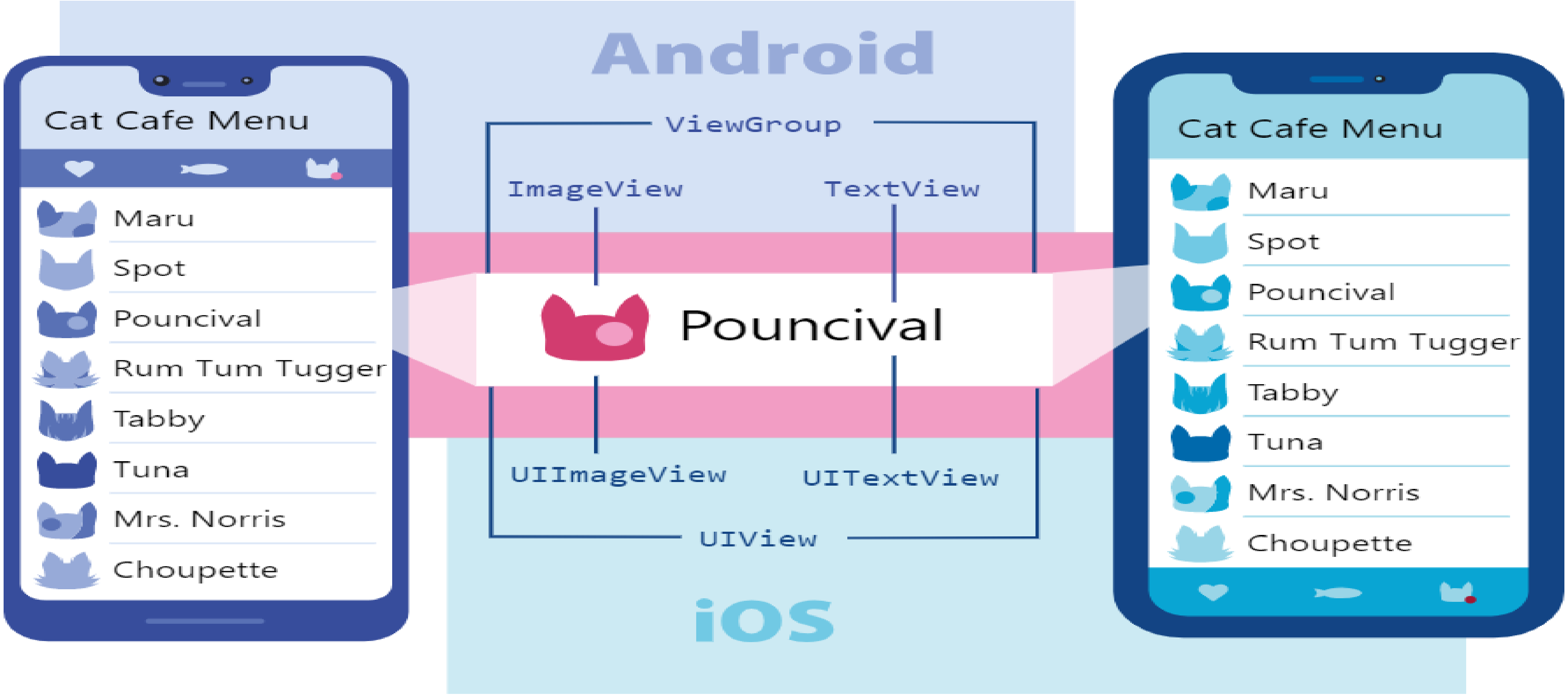
Calling custom components



```
const App = () => {
  const [count, setCount] = useState(0);

  return (
    <View style={styles.container}>
      <Text>You clicked {count} times</Text>
      <Button
        onPress={() => setCount(count + 1)}
        title="Click me!"
      />
    </View>
  );
};

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

state

how components work in React Native?

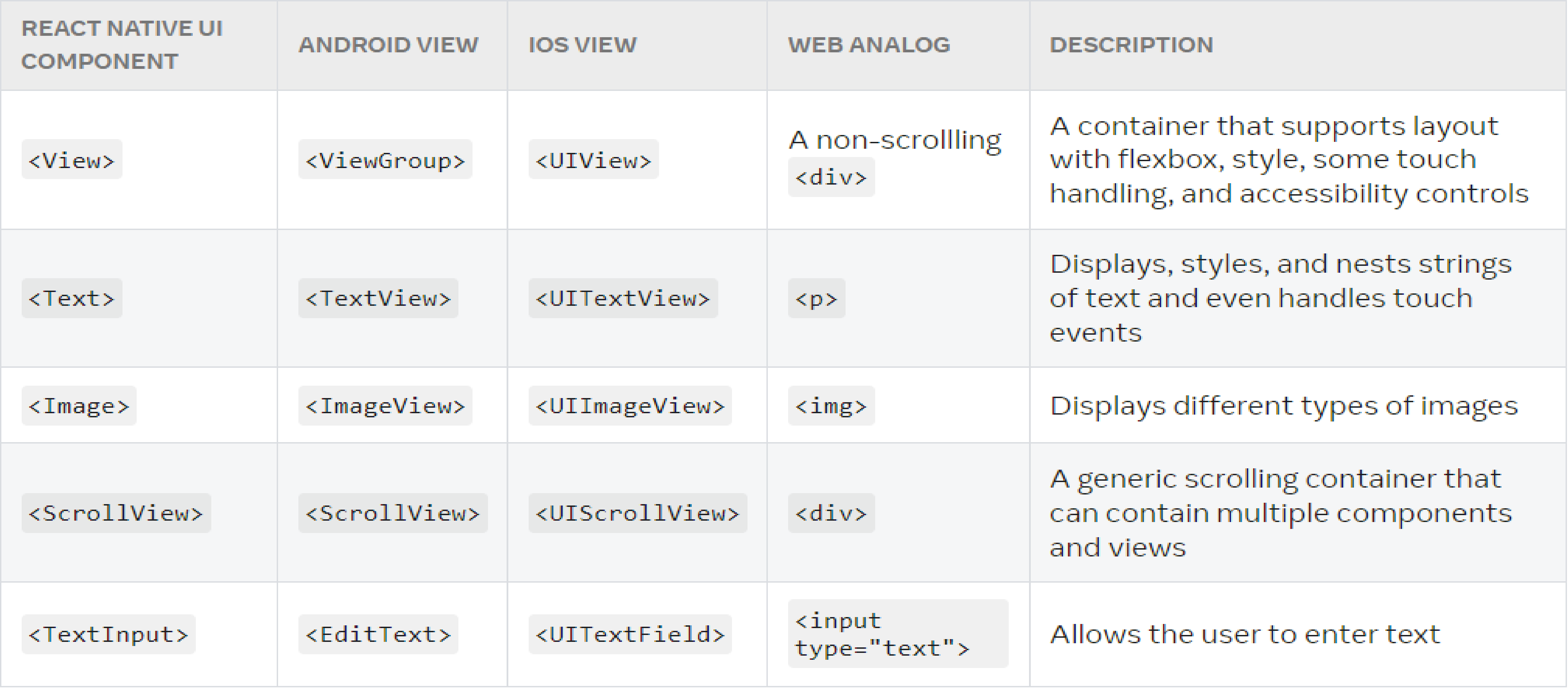


Core/Basic Components

React Native comes with a set of essential, ready-to-use Native Components

- View
- Text
- Image
- Button
- Touchables
- Alert

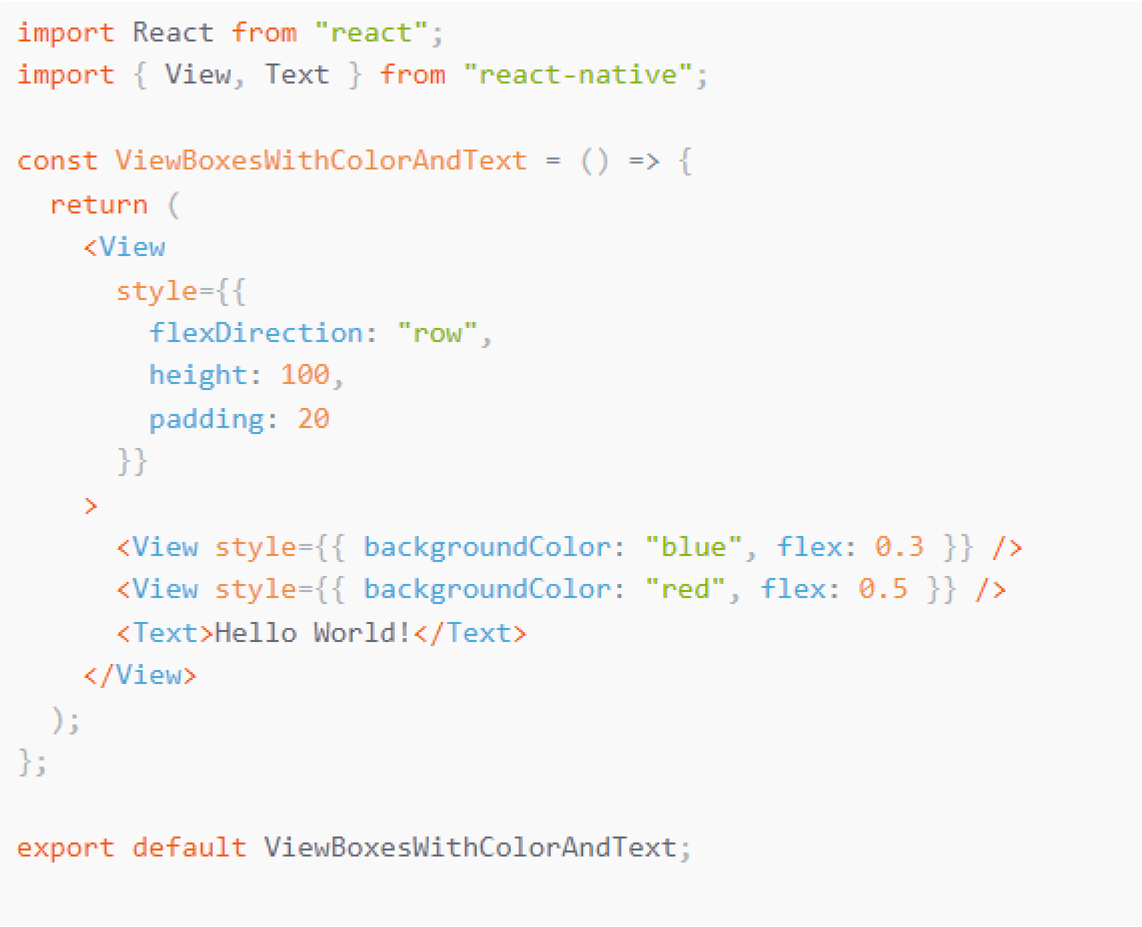
Cross platform working[2]



REACT NATIVE UI COMPONENT	ANDROID VIEW	IOS VIEW	WEB ANALOG	DESCRIPTION
`<View>`	`<ViewGroup>`	`<UIView>`	A non-scrollling `<div>`	A container that supports layout with flexbox, style, some touch handling, and accessibility controls
`<Text>`	`<TextView>`	`<UITextView>`	`<p>`	Displays, styles, and nests strings of text and even handles touch events
`<Image>`	`<ImageView>`	`<UIImageView>`	`<img>`	Displays different types of images
`<ScrollView>`	`<ScrollView>`	`<UIScrollView>`	`<div>`	A generic scrolling container that can contain multiple components and views
`<TextInput>`	`<EditText>`	`<UITextField>`	`<input type="text">`	Allows the user to enter text

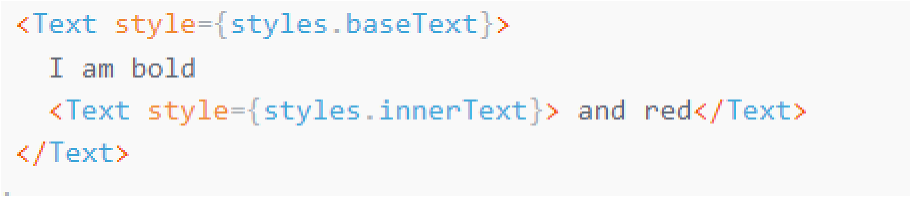
View

- basic building block of UI
- used to display text, images, or respond to user input
- views can contain other views.
- Maps to UIView, <div>, android.view



```
import React from "react";
import { View, Text } from "react-native";

const ViewBoxesWithColorAndText = () => {
  return (
    <View
      style={{
        flexDirection: "row",
        height: 100,
        padding: 20
      }}
    >
      <View style={{ backgroundColor: "blue", flex: 0.3 }} />
      <View style={{ backgroundColor: "red", flex: 0.5 }} />
      <Text>Hello World!</Text>
    </View>
  );
};

export default ViewBoxesWithColorAndText;
```

Text



```
<Text style={styles.baseText}>
  I am bold
  <Text style={styles.innerText}> and red</Text>
</Text>
```

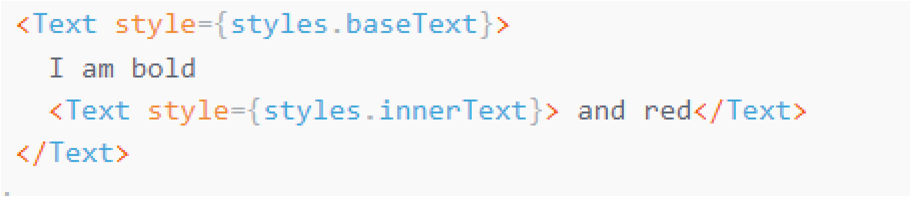
- Displays text
- supports nesting, styling, and touch handling.
- must wrap all the text nodes inside of a <Text> component.
- Props
  - numberOfLines
  - onPress

Nested Text

```
<Text style={styles.baseText}>
     I am bold and red
     <Text style={styles.innerText}> and red</Text>
   </Text>
```

TextInput [3]

- foundational component for inputting text into the app via a keyboard.
- Props:
  - autoComplete
  - autoCorrect
  - Editable
  - keyboardType
  - onChangeText



```
<Text style={styles.baseText}>
  I am bold
  <Text style={styles.innerText}> and red</Text>
</Text>
```

Image [5]

- Display images both local and network
- Local Image:
  - <Image source = {require(‘’)}/>
- Network Images:
  - <Image source = {{uri:‘’}}
- Props
  - blurRadius
  - defaultSource

Buttons [4]

- render nicely on any platform.
- Required props
  - onPress
  - Title
  - color

Touchables

- Wrapper to make views respond to touches
  - TouchableHighlight
  - TouchableOpacity
  - TouchableWithoutFeedback

References

[1] https://reactnative.dev/docs/tutorial

[2] https://reactnative.dev/docs/intro-react-native-components

[3] https://reactnative.dev/docs/textinput

[4] https://reactnative.dev/docs/button

[5] https://reactnative.dev/docs/image