**React Native Notes**

**Introduction**

* **React** **Native** is a Framework for building mobile apps using the React Library and JavaScript.
* Cross Platform Ability to Convert Code into an Android/IOS App.
* Uses the same fundamental building blocks as a native app built with **Swift/Objective-C/Java**.

**UI Component Examples**

* **TextInput, Picker, Switch, Slider**
* **Touchable Elements**
* **ListView & ScrollViewAlerts, Modals, ProgressBars**
* **StatusBars**
* **TabBarIOS**
* **ToolbarAndroid**

**Setting Up (Development Environment)**

**Windows**

* **Android Studio**
* **Android SDK**
* **Android AVD**

**Mac**

* **Xcode**
* **Simulator**

**Standard React Native Component**

**Note:** Type **“rnc”** and press Tab to automatically load the default *React Native Component* Boiler Plate as shown below:

import React, { Component } from 'react'

import { Text, View } from 'react-native'

export default class MyApp extends Component {

render() {

return (

<View>

<Text> Hello </Text>

</View>

)

}

}

**React Native CLI Quickstart**

**Step 1:** You will need Node, Python2, a JDK, and Android Studio.

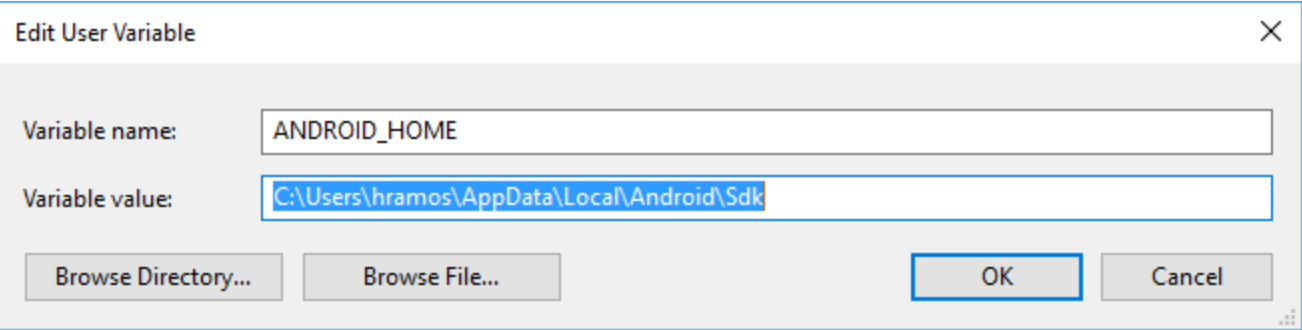
***(React recommends installing Node and Python2 via Chocolatey, a popular package manager for Windows.)***

In Command Line type: ***choco install -y nodejs.install python2 jdk8***

**Step 2:** Install the React Command Line Interface:

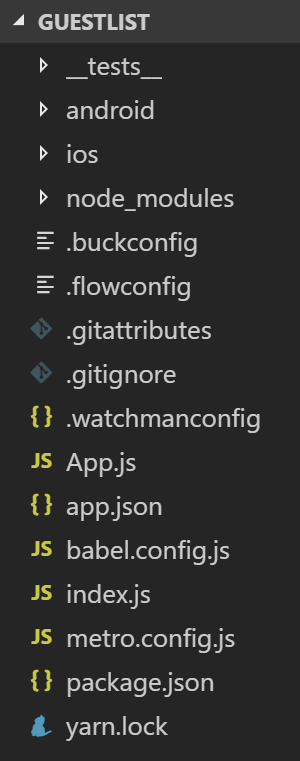
In Command Line, Type: ***npm install -g react-native-cli***

**Step 3:** Create a new environment variable called **ANDROID\_HOME that points to your Android SDK.**



**Step 4:** Create a New Application:

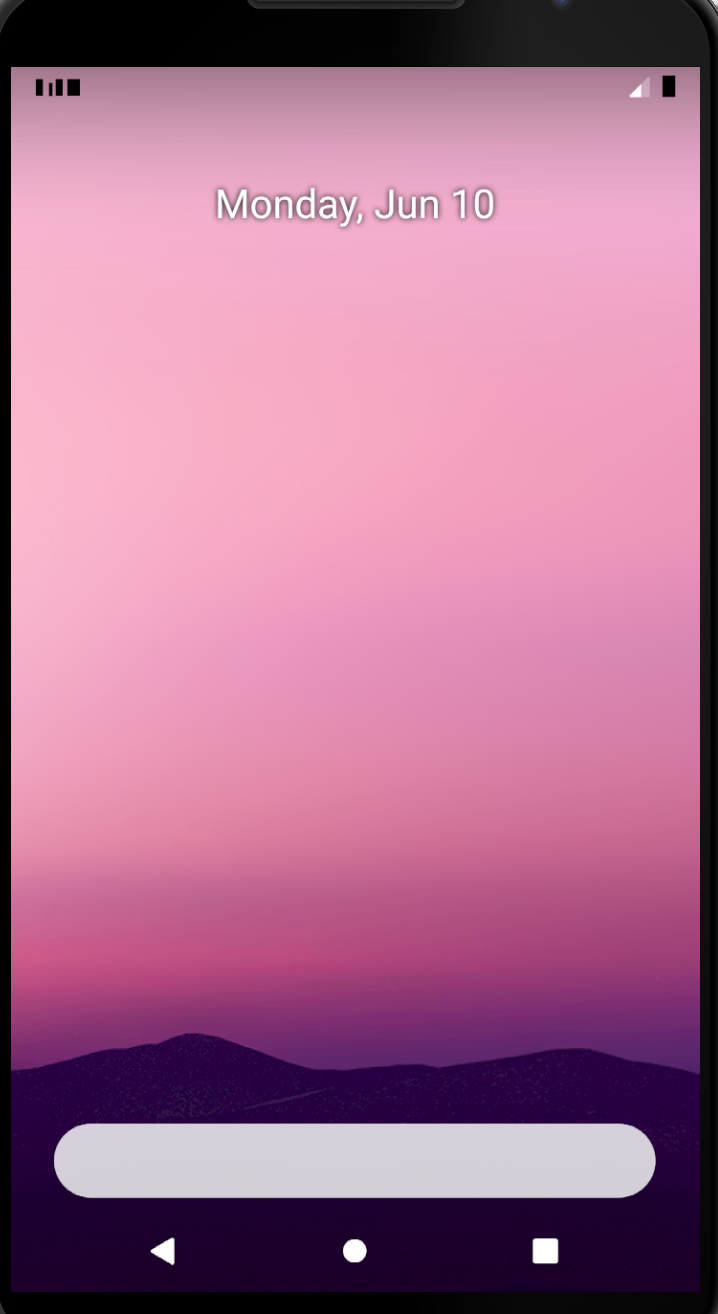
**In Command Line, type:** react-native init GuestList



**Step 5:** Connect your phone or Run Android Virtual Device, (Using Android Studio) which allows you to emulate an Android device on your computer.

**Step 6:** Open **Android Studio** and run an **Android Virtual Device** such as the Nexus 6:

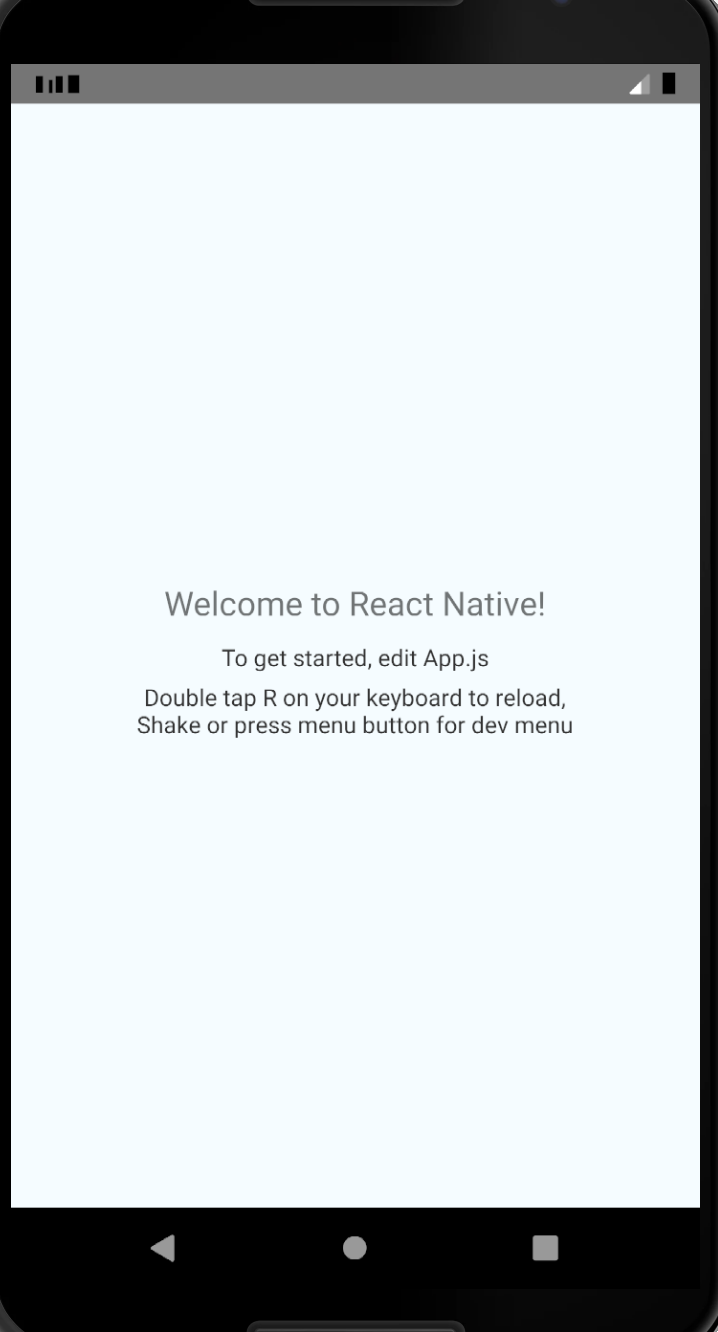
**Note:** Here is an Example of a running Android Virtual Device:



**Step 7:** Run your application:

**In Command Line, type:** react-native run-android

**Note:** Here is an Example of the Starter React Native App running on the Android Virtual Device:



**App.js File:**

import React, {Component} from 'react';

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

const instructions = Platform.select({

ios: 'Press Cmd+R to reload,\n' + 'Cmd+D or shake for dev menu',

android:

'Double tap R on your keyboard to reload,\n' +

'Shake or press menu button for dev menu',

});

export default class App extends Component{

render() {

return (

<View style={styles.container}>

<Text style={styles.welcome}>Welcome to React Native!</Text>

<Text style={styles.instructions}>To get started, edit App.js</Text>

<Text style={styles.instructions}>{instructions}</Text>

</View>

);

}

}

const styles = StyleSheet.create({

container: {

flex: 1,

justifyContent: 'center',

alignItems: 'center',

backgroundColor: '#F5FCFF',

},

welcome: {

fontSize: 20,

textAlign: 'center',

margin: 10,

},

instructions: {

textAlign: 'center',

color: '#333333',

marginBottom: 5,

},

});

**index.js File:**

/\*\*

\* @format

\*/

import {AppRegistry} from 'react-native';

import Wow from './Wow';

import {name as appName} from './app.json';

AppRegistry.registerComponent(appName, () => Wow);

**Step 8: Update App.Js to:**

import React, {Component} from 'react';

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

export default class guests extends Component{

render() {

return (

<View>

<Text>Hello World</Text>

</View>

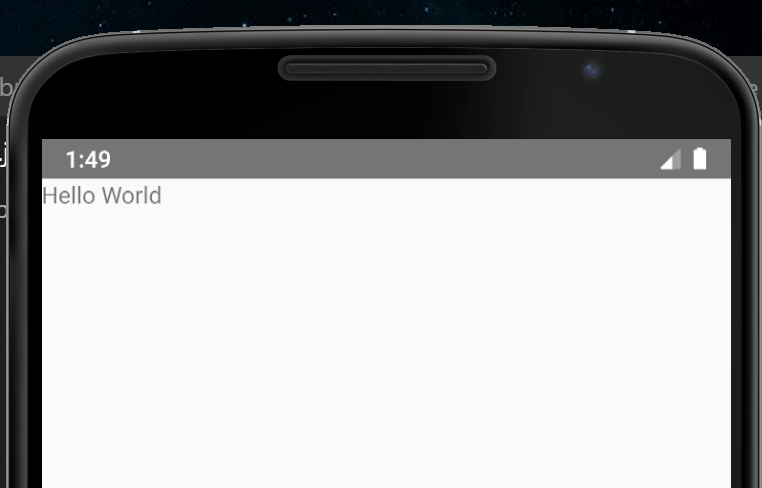
);

}

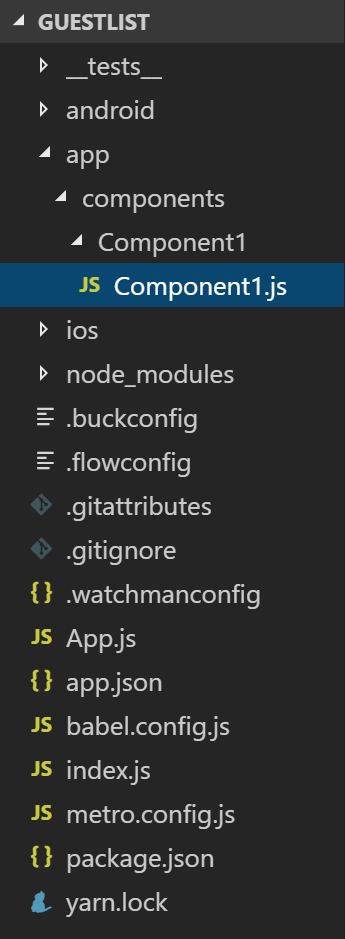
}

AppRegistry.registerComponent('guests', () => guests);

**Note: Server View:**



**Step 9:** Create a folder called **“app”** and then a folder inside of it called **“components”** and then a folder inside of that called **“Component1”** and then a file inside of that called **“Component1.js”.**



**Step 10:** Add this code to **“Component1.js”:**

import React, {Component} from 'react';

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

export default class Component1 extends Component{

render() {

return (

<View>

<Text>This is Component 1</Text>

</View>

);

}

}

AppRegistry.registerComponent('Component1', () => Component1);

**Step 11: Delete the “Hello World” text code and** Add **<Component1 />** to **“App.js”:**

import React, {Component} from 'react';

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

// import {Platform, StyleSheet, Text, View} from 'react-native';

import Component1 from './app/components/Component1/Component1';

export default class guests extends Component{

render() {

return (

Delete

<View>

<Text>Hello World</Text>

<Component1 />

</View>

);

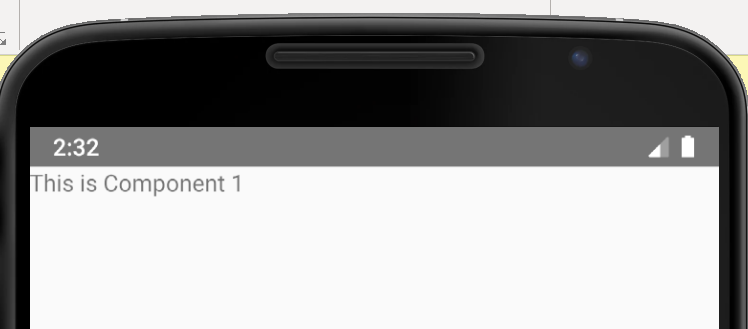
}

**Add**

}

AppRegistry.registerComponent('guests', () => guests);

**Note: Server View:**



**Props**

**Step 11:** Add message prop to **Component1** in the **App.js** File.

**App.js File:**

import React, {Component} from 'react';

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

// import {Platform, StyleSheet, Text, View} from 'react-native';

import Component1 from './app/components/Component1/Component1';

export default class guests extends Component{

render() {

return (

<View>

<Component1 message = "Welcome"/>

</View>

);

}

**Add**

}

AppRegistry.registerComponent('guests', () => guests);

**Step 12:** Add **{this.props.message}** code in the **Component1.js** File.

*(Passing a property into* ***Component1*** *and displaying it.)*

**Component1.js File:**

import React, {Component} from 'react';

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

export default class Component1 extends Component{

render() {

return (

<View>

<Text>This is Component 1</Text>

<Text>{this.props.message}</Text>

</View>

);

**Add**

}

}

AppRegistry.registerComponent('Component1', () => Component1);

**Note:** Server View:



**State**

**Step 13:** Add the following code to the **Component1.js** File:

**Component1.js File:**

import React, {Component} from 'react';

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

export default class Component1 extends Component{

constructor(){

super();

this.state = {

name: 'Ahmed'

}

}

**Add**

render() {

return (

<View>

<Text>This is Component 1</Text>

<Text>{this.props.message}</Text>

<Text>{this.state.name}</Text>

</View>

);

}

}

AppRegistry.registerComponent('Component1', () => Component1);

**Note:** Server View:



**Step 14:** Add Boolean called **showName** set to true. Create a variable called name, if **showName** is true, then display **this.state.name**, else show **‘No name.’**

**Component1.js File:**

import React, {Component} from 'react';

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

**Add**

export default class Component1 extends Component{

constructor(){

super();

this.state = {

name: 'Ahmed'

showName: true

**If this True, then show this, else show:**

}

}

render() {

let name = this.state.showName ? this.state.name : 'No name';

return (

<View>

<Text>This is Component 1</Text>

<Text>{this.props.message}</Text>

<Text>{this.state.name}</Text>

</View>

);

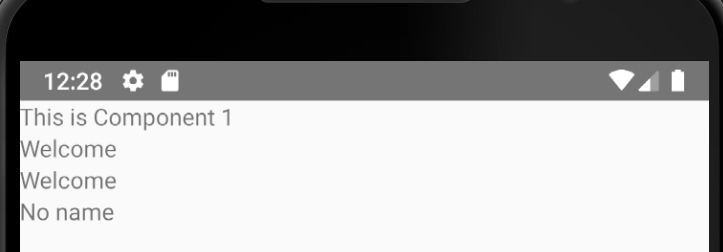
}

}

Delete

AppRegistry.registerComponent('Component1', () => Component1);

**Note:** Server View when showName: **true** **Note:** Server View when showName: **false**



**Step 15:** Add Props and Access through this.state:

**Component1.js File:**

import React, {Component} from 'react';

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

export default class Component1 extends Component{

constructor(props){

super(props);

**Add**

this.state = {

name: 'Ahmed',

showName: true,

message: this.props.message

}

}

render() {

let name = this.state.showName ? this.state.name : 'No name';

return (

<View>

<Text>This is Component 1</Text>

<Text>{this.props.message}</Text>

<Text>{this.state.message}</Text>

<Text>{name}</Text>

</View>

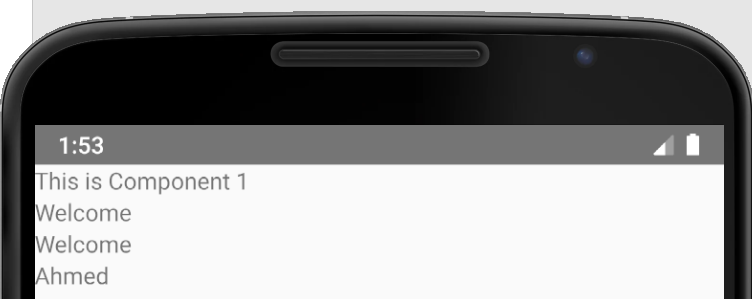
);

}

}

AppRegistry.registerComponent('Component1', () => Component1);

**Note:** Server View:



**Default Props - *(Add Props to Component File Instead)***

**Step 16:** Add Props to current Component:

**Component1.js File:**

import React, {Component} from 'react';

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

export default class Component1 extends Component{

constructor(props){

super(props);

this.state = {

name: 'Ahmed',

showName: true,

message: this.props.message

}

}

**Add**

static defaultProps = {

message: 'Welcome'

}

render() {

let name = this.state.showName ? this.state.name : 'No name';

return (

<View>

<Text>This is Component 1</Text>

<Text>{this.props.message}</Text>

<Text>{this.state.message}</Text>

<Text>{name}</Text>

</View>

);

}

}

AppRegistry.registerComponent('Component1', () => Component1);

**Step 17:** Delete ‘message’ prop in **App.js** file:

**App.js File:**

import React, {Component} from 'react';

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

// import {Platform, StyleSheet, Text, View} from 'react-native';

import Component1 from './app/components/Component1/Component1';

export default class guests extends Component{

render() {

return (

Delete

<View>

<Component1 message = "Welcome"/>

</View>

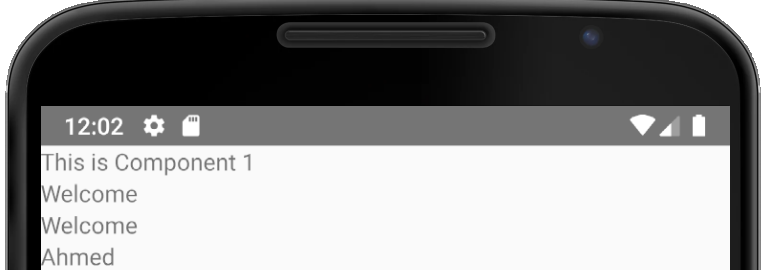
);

}

}

AppRegistry.registerComponent('guests', () => guests);

**Note:** Server View:



**Console.logging in React-Native**

**Step 18:** Add a console.log message: “***console.log(‘Testing Console.log);****”*

**Component1.js File:**

import React, {Component} from 'react';

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

export default class Component1 extends Component{

constructor(props){

super(props);

this.state = {

name: 'Ahmed',

showName: true,

message: this.props.message

}

}

static defaultProps = {

message: 'Welcome'

}

**Add**

render() {

console.log('Testing Console.log');

let name = this.state.showName ? this.state.name : 'No name';

return (

<View>

<Text>This is Component 1</Text>

<Text>{this.props.message}</Text>

<Text>{this.state.message}</Text>

<Text>{name}</Text>

</View>

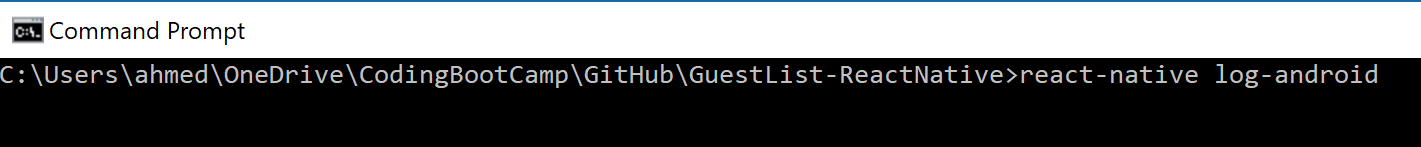
);

}

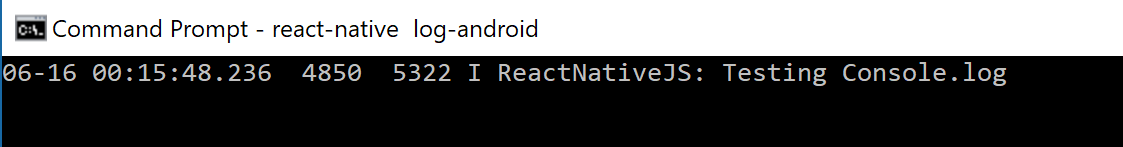
}

AppRegistry.registerComponent('Component1', () => Component1);

**Step 19:** In Command Line, run “***react-native log-android***”

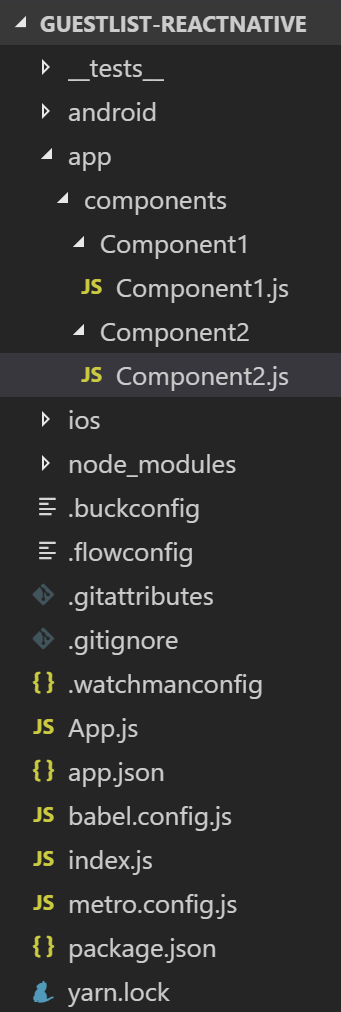


**Note:** Refresh Android Virtual Device to Refresh Console.



**Styles**

**Step 20:** Create a folder called **“Component2”** and then a file inside of that called **“Component2.js”.**



**Add the following code to Component2.js File:**

**Note\*:** Type “rnc” and press Tab to quickly load a starter template.

**Component2.js File:**

import React, {Component} from 'react';

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

export default class Component2 extends Component{

render() {

return (

<View>

<Text>Component2</Text>

</View>

);

}

}

AppRegistry.registerComponent('Component2', () => Component2);

**Step 21:** Add **In-line Styling** Code:

**Component2.js File:**

import React, {Component} from 'react';

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

**Add**

export default class Component2 extends Component{

render() {

return (

<View style = {{backgroundColor:"lightgray"}}>

<Text style = {{fontSize: 28, textAlign: 'center', margin: 10, color:'red'}}>Component2</Text>

</View>

);

}

}

AppRegistry.registerComponent('Component2', () => Component2);

**Step 22:** Add Component to App.js File:

**App.js File:**

import React, {Component} from 'react';

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

import Component1 from './app/components/Component1/Component1';

import Component2 from './app/components/Component2/Component2';

export default class guests extends Component{

render() {

return (

<View>

<Component1 />

**Add**

<Component2 />

</View>

);

}

}

AppRegistry.registerComponent('guests', () => guests);

**Note:** Server View:



**Step 23:** ImportStylesheet and enter the styling css code in a constant instead.

**Component2.js File:**

import React, {Component} from 'react';

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

export default class Component2 extends Component{

render() {

**Add**

return (

<View style = {styles.theView}>

<Text style = {styles.theText}>Component2</Text>

</View>

);

}

}

const styles = StyleSheet.create({

theView: {

backgroundColor:"lightgray"

},

theText: {

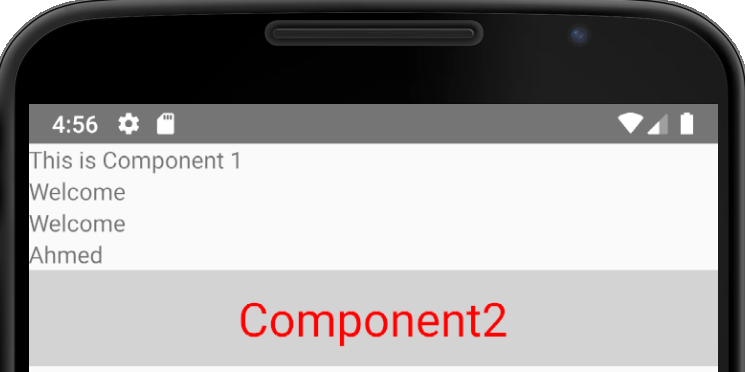
color: '#FFFFFF', fontSize: 28, textAlign: 'center', margin: 10, color:'red'

}

})

AppRegistry.registerComponent('Component2', () => Component2);

**Note:** Server View:



**Component2.js File:**

import React, {Component} from 'react';

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

export default class Component2 extends Component{

render() {

return (

<View>

<View style = {styles.theView}>

<Text style = {styles.theText}>Component2</Text>

</View>

**Add**

<View style = {styles.container}>

<View style = {styles.v1}>

<Text>View1</Text>

</View>

<View style = {styles.v2}>

<Text>View2</Text>

</View>

<View style = {styles.v3}>

<Text>View3</Text>

</View>

</View>

</View>

);

}

}

const styles = StyleSheet.create({

theView: {

backgroundColor:"lightgray"

},

theText: {

color: '#FFFFFF', fontSize: 28, textAlign: 'center', margin: 10, color:'red'

},

container: {

height:100

},

v1: {

flex: 1,

backgroundColor: 'red',

**Add**

padding:10,

borderRadius: 10,

borderWidth: 1,

borderColor: 'black'

},

v2: {

flex: 1,

backgroundColor: 'white',

padding:10,

borderRadius: 10,

borderWidth: 1,

borderColor: 'black',

},

v3: {

flex: 1,

backgroundColor: 'lightblue',

padding:10,

borderRadius: 10,

borderWidth: 1,

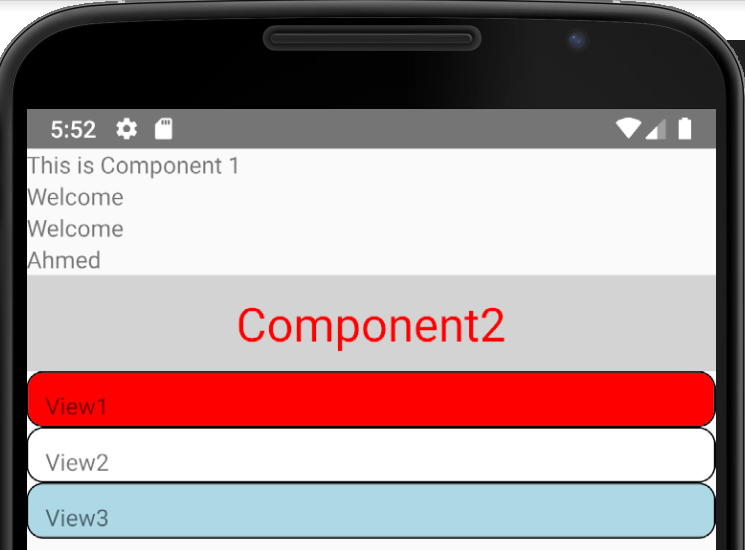
borderColor: 'black'

}

})

AppRegistry.registerComponent('Component2', () => Component2);

**Note:** Server View:



**Step 24:** Add **flexDirection: ‘row’** to container css code:

container: {

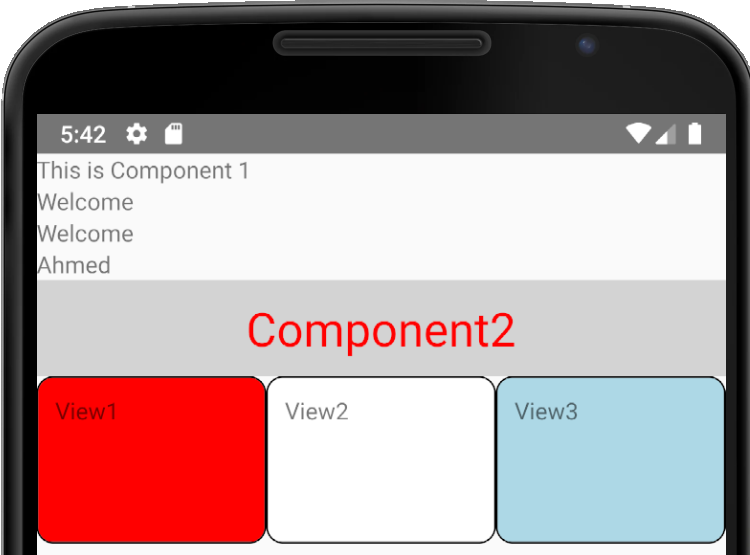
height:100,

**Add**

flexDirection:'row'

},

**Note:** Server View:



**Step 25:** Add CSS Styling to Text:

import React, {Component} from 'react';

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

export default class Component2 extends Component{

render() {

return (

<View>

<View style = {styles.theView}>

<Text style = {styles.theText}>Component2</Text>

</View>

<View style = {styles.container}>

<View style = {styles.v1}>

<Text style = {styles.vText}>View1</Text>

</View>

<View style = {styles.v2}>

<Text>View2</Text>

</View>

<View style = {styles.v3}>

<Text style = {styles.vText}>View3</Text>

**Add**

</View>

</View>

</View>

);

}

}

const styles = StyleSheet.create({

theView: {

backgroundColor:"lightgray"

},

theText: {

color: '#FFFFFF', fontSize: 28, textAlign: 'center', margin: 10, color:'red'

},

container: {

height:100,

flexDirection:'row'

},

v1: {

flex: 1,

backgroundColor: 'red',

padding:10,

borderRadius: 10,

borderWidth: 1,

borderColor: 'black'

},

v2: {

flex: 1,

backgroundColor: 'white',

padding:10,

borderRadius: 10,

borderWidth: 1,

borderColor: 'black',

},

v3: {

flex: 1,

backgroundColor: 'lightblue',

padding:10,

borderRadius: 10,

borderWidth: 1,

borderColor: 'black'

},

vText: {

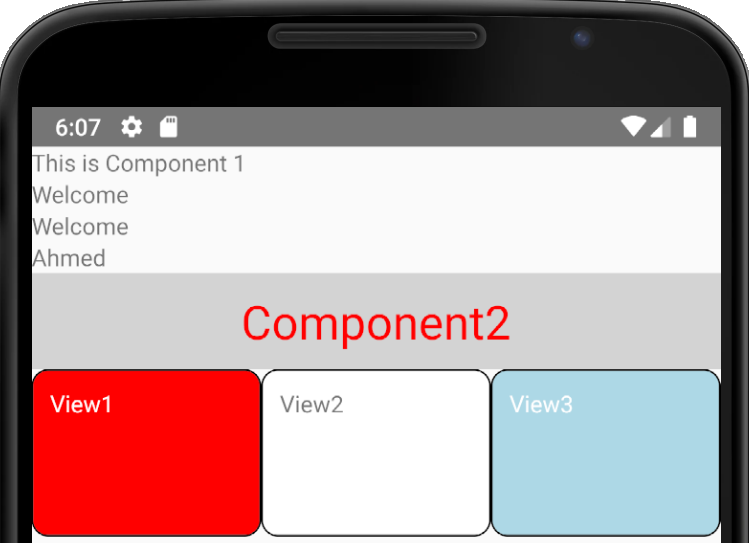
color: 'white'

}

})

AppRegistry.registerComponent('Component2', () => Component2);

**Note:** Server View:



**Touchable Highlights**

**Step 26:** Add Touchable Highlights to View Panel

import React, {Component} from 'react';

import {AppRegistry, Text, View, StyleSheet, TouchableHighlight} from 'react-native';

export default class Component2 extends Component{

onPress(){

console.log('Pressed Area');

}

render() {

**Add**

return (

<View>

<View style = {styles.theView}>

<Text style = {styles.theText}>Component2</Text>

</View>

<View style = {styles.container}>

<TouchableHighlight style = {styles.v1} onPress={this.onPress}>

<View>

<Text style = {styles.vText}>View1</Text>

</View>

</TouchableHighlight>

<View style = {styles.v2}>

<Text>View2</Text>

</View>

<View style = {styles.v3}>

<Text style = {styles.vText}>View3</Text>

</View>

</View>

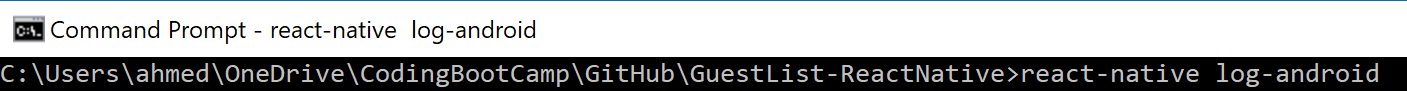
</View>

);

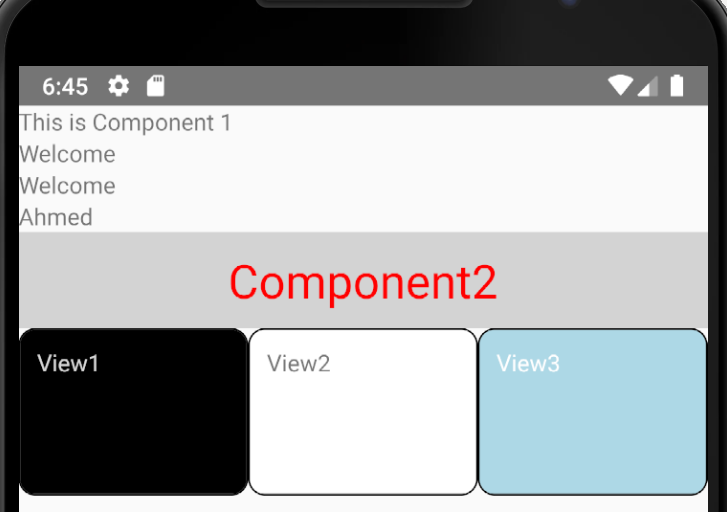
}

}

**Step 27:** Pull up Console by running “***react-native log-android***” in your React-Native Folder:



**Note:** Server View: ***(View1 Panel Changes color to black when clicked on.)***



**Note:** Every time View1 is clicked, the message “Pressed Area is logged to the console: