Universidad Politecnica de Queretaro
Tecnologias de la Informacion e Innovacion Digital
Fecha de entrega:16/10/25 Grupo:TIID211

Materia:Desarrollo Apps Moviles

Profesor: Ivan Isay Guerra Lopez

Alumno:

Irineo Atanacio Diego

Fecha de entrega:16/10/25

Universidad Politecnica de Queretaro
Tecnologias de la Informacion e Innovacion Digital
Fecha de entrega:16/10/25 Grupo:TIID211

En esta práctica usamos el codigo llamado BotonesScreen que permite al usuario encender o apagar un estado representado por la variable Prendido

utiliza los siguientes elementos de React Native:

useState: Para manejar el estado de la variable Prendido

TouchableOpacity: Para crear botones de encendido y apagado

Switch: Para alternar el estado con un interruptor

StyleSheet: Para aplicar estilos personalizados a los componentes

Aquí se importan los módulos que necesitamos:

React: para poder usar JSX y los hooks

useState: hook que permite guardar y actualizar los valores “prendido” o “apagado

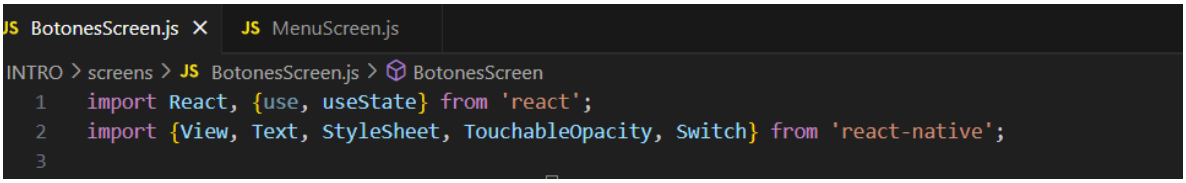
View: contenedor principal (como un en HTML)

Text: para mostrar texto en pantalla

StyleSheet: para crear estilos

TouchableOpacity: crea botones táctiles con efecto visual al presionar

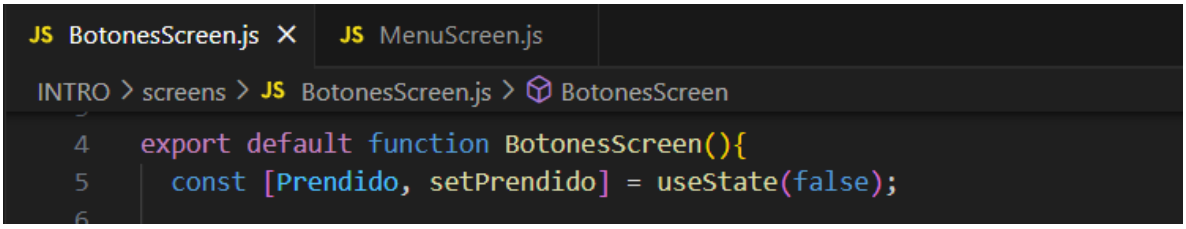
Switch: componente tipo interruptor (encendido/apagado)



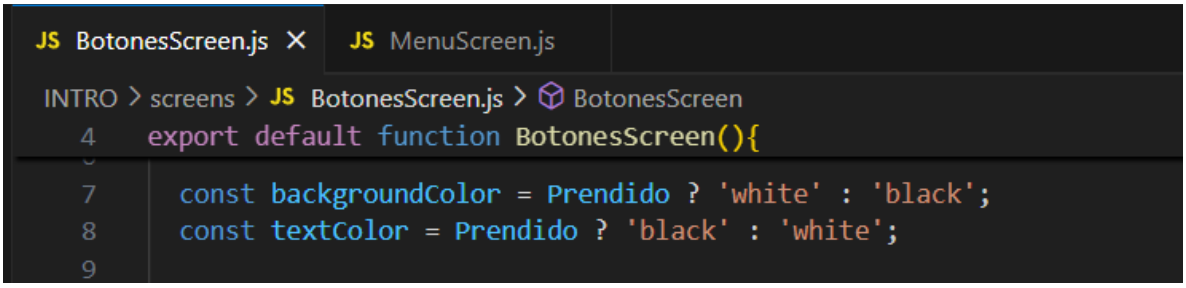
Universidad Politecnica de Queretaro
Tecnologias de la Informacion e Innovacion Digital
Fecha de entrega:16/10/25 Grupo:TIID211

Se define un componente funcional llamado BotonesScreen

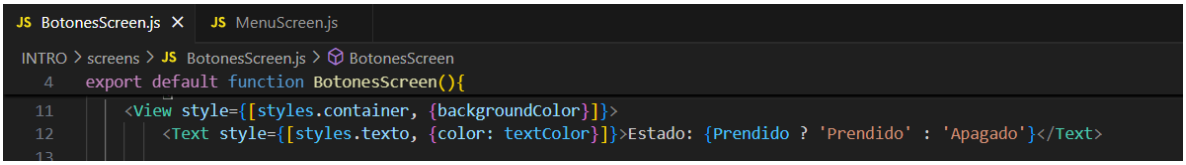
Se crea una variable de estado llamada Prendido



Estas dos líneas se usan para decidir el color del fondo y del texto según el estado

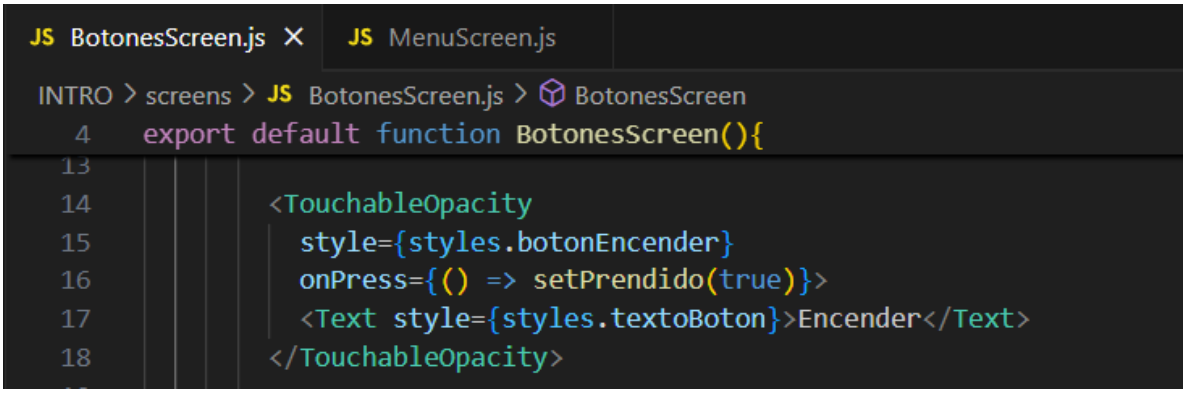


Muestra el texto “Estado: Prendido” o “Estado: Apagado” según el valor del estado
También cambia el color del texto dinámicamente

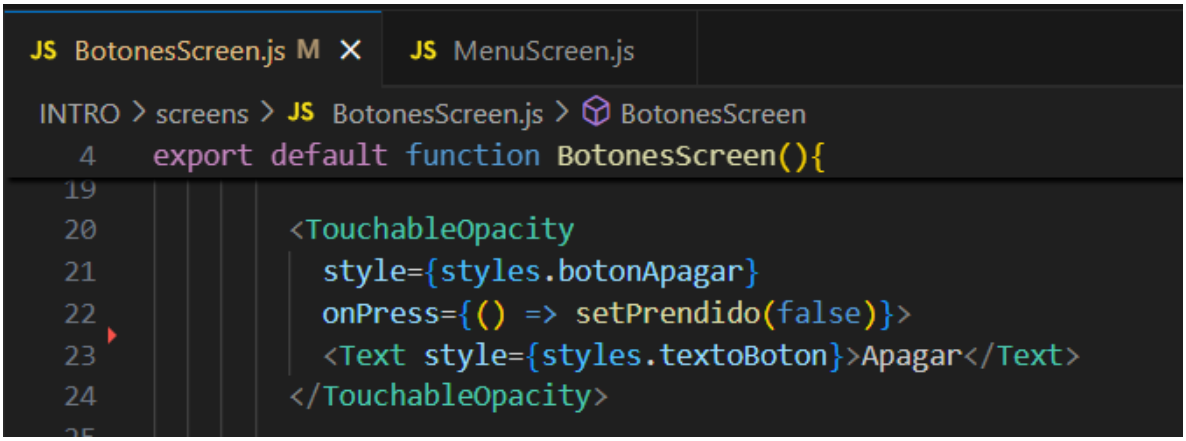


Universidad Politecnica de Queretaro
Tecnologias de la Informacion e Innovacion Digital
Fecha de entrega:16/10/25 Grupo:TIID211

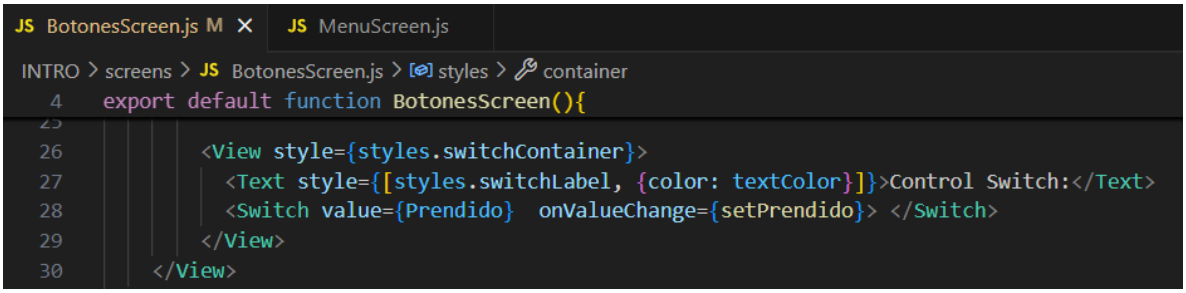
Se rea un botón táctil verde que al presionarlo cambia el estado a encendido



Igual al botón anterior, pero con color rojo y que apaga el estado



Aquí se crea un interruptor que también controla el mismo estado



Y por último esta seria la parte de los estilos

JS BotonesScreen.js M JS MenuScreen.js

INTRO > screens > JS BotonesScreen.js > styles > container

```
33  const styles = StyleSheet.create({
34    container: {
35      flex: 1,
36      justifyContent: 'center',
37      alignItems: 'center',
38      backgroundColor: 'hsla(209, 73%, 88%, 1.00)',
39    },
40    texto : {
41      fontSize: 20,
42      fontWeight: 'bold',
43      marginBottom: 30,
44    },
45    botonEncender: {
46      backgroundColor: 'green',
47      paddingVertical: 10,
48      paddingHorizontal: 25,
49      borderRadius: 8,
50      marginBottom: 20,
51    },
52    botonApagar: {
53      backgroundColor: 'red',
54      paddingVertical: 10,
55      paddingHorizontal: 25,
56      borderRadius: 8,
57      marginBottom: 20,
58    },
59    textoBoton: {
60      color: 'white',
61      fontSize: 16,
62      fontWeight: '600',
63    },
64    switchContainer: {
65      flexDirection: 'row',
66      alignItems: 'center',
67    },
68    switchLabel: {
69      fontSize: 16,
70      marginRight: 10,
71    },
72  })
```

Universidad Politecnica de Queretaro
Tecnologias de la Informacion e Innovacion Digital
Fecha de entrega:16/10/25 Grupo:TIID211

Conclusión:

En esta práctica comprendimos mejor el uso del useState para controlar el estado de los componentes y cómo los botones y switches pueden modificar dinámicamente la interfaz de usuario en React Native.