**Lab 11**

**Code:** App.js

|  |  |
| --- | --- |
|  |  |
|  | import React, { Component } from 'react';  import { StyleSheet, Text, View, Button, TouchableOpacity } from 'react-native';  export default class App extends Component {  constructor() {  super()  this.state = {  resultText: "",  calculationText: ""  }  this.operations = ["DEL", '/', '\*', '-', '+']  }  calculateResult() {  const text = this.state.resultText  // parse text and calculate  this.setState({  calculationText: eval(text)  })  }  validate() {  const text = this.state.resultText  switch(text.slice(-1)) {  case '+':  case '-':  case '\*':  case '/':  return false  }  return true  }  buttonPressed(text) {  console.log(text)  if (text == "=") {  return this.validate() && this.calculateResult()  }  this.setState({  resultText: this.state.resultText + text  })  }  operate(operation) {  switch (operation) {  case "DEL":  let text = this.state.resultText.split('')  text.pop()  this.setState({  resultText: text.join('')  })  break  case '+':  case '-':  case '\*':  case '/':  const lastChar = this.state.resultText.split('').pop()  if (this.operations.indexOf(lastChar) > 0) return  if (text == "") return  this.setState({  resultText: this.state.resultText + operation  })  break  }  }  render() {  let rows = []  let nums = [[7, 8, 9], [4, 5, 6], [1, 2, 3], [".", 0, "="]]  for (let i = 0; i < 4; i++) {  let row = []  for (let j = 0; j < 3; j++) {  row.push(<TouchableOpacity key={nums[i][j]} onPress={() => this.buttonPressed(nums[i][j])}  style={styles.btn}>  <Text style={styles.btntext}>{nums[i][j]}</Text>  </TouchableOpacity>)  }  rows.push(<View key={i} style={styles.row}>{row}</View>)  }  ops = []  for (let i = 0; i < 5; i++) {  ops.push(<TouchableOpacity key={this.operations[i]} style={styles.btn} onPress={() => this.operate(this.operations[i])}>  <Text style={[styles.btnopstext, styles.white]}>{this.operations[i]}</Text></TouchableOpacity>)  }  return (  <View style={styles.container}>  <View style={styles.result}>  <Text ellipsizeMode='head' numberOfLines={1}  style={styles.resultText}>{this.state.resultText}</Text>  </View>  <View style={styles.calculation}>  <Text ellipsizeMode='head' numberOfLines={2}  style={styles.calculationText}>{this.state.calculationText}</Text>  </View>  <View style={styles.buttons}>  <View style={styles.numbers}>{rows}</View>  <View style={styles.operations}>  {ops}  </View>  </View>  </View>  );  }  }  const styles = StyleSheet.create({  container: {  flex: 1,  },  white: {  color: 'white',  },  btn: {  flex: 1,  alignItems: 'center',  alignSelf: 'stretch',  justifyContent: 'center'  },  btntext: {  fontSize: 36,  color: 'white',  fontFamily: 'sans-serif-light'  },  btnopstext: {  fontSize: 24,  color: 'white',  fontFamily: 'sans-serif-light'  },  resultText: {  fontSize: 70,  color: 'black',  fontFamily: 'sans-serif-light',  paddingEnd: 10,  paddingStart: 10,  },  calculationText: {  fontSize: 44,  color: 'grey',  fontFamily: 'sans-serif-light',  paddingEnd: 10,  paddingStart: 10,  marginBottom: 20  },  row: {  flexDirection: 'row',  flex: 1,  justifyContent: 'space-around',  alignItems: 'center'  },  result: {  flex: 2,  backgroundColor: 'white',  justifyContent: 'center',  alignItems: 'flex-end'  },  calculation: {  flex: 1,  backgroundColor: 'white',  justifyContent: 'center',  alignItems: 'flex-end'  },  buttons: {  flexGrow: 5,  flexDirection: 'row',  },  numbers: {  flex: 3,  backgroundColor: '#434343',  },  operations: {  flex: 1,  backgroundColor: '#636363',  justifyContent: 'space-around',  } |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  | }); |