

Lab 4 : JavaScript for Mobile Device Programming

1. ให้นักศึกษาทำการสร้าง New Project ใหม่ ชื่อ ProjectCal ใน Folder Mobile\<รหัสนักศึกษา>\ProjectCal แบบ Blank Project โดยใช้คำสั่ง

Expo init <path\folder\StudentID\Project name>

2. ให้นักศึกษาทำการสร้างโปรแกรมเครื่องคิดเลขบนอุปกรณ์เคลื่อนที่ ดังแสดงตามรูปข้างล่างนี้



บันทึกผลการทดลอง :

```
import { StatusBar } from "expo-status-bar";
import React, { Component } from "react";
import { StyleSheet, Text, TouchableOpacity, View, TextInput, Button } from "react-native";

const calNum = [[1, 2, 3], [4, 5, 6], [7, 8, 9], [".", 0, "="]]
const calOp = [['DEL'], ['+'], ['-'], ['*'], ['/']]

class App extends Component {

  constructor(props) {
    super(props)
    this.initialState = {
      displayValue: '0',
      result: ""
    }
  }
}
```

```

    this.state = this.initialState;
  }
  // handleInput = (input) => {
  //   const { displayValue } = this.state;
  // }

  _onInputButtonPressed(input) {
    // alert(input)
    if (input == '=') {
      let check = 0
      check = this.state.result.length - 1;
      if (
        this.state.result.charAt(check) == '+' ||
        this.state.result.charAt(check) == '-' ||
        this.state.result.charAt(check) == '*' ||
        this.state.result.charAt(check) == '/'
      ) {
        return alert('รูปแบบไม่ถูกต้อง')
      }
      this.setState({
        result: eval(this.state.result).toString()
      })
      return
    }
    if (input == 'DEL') {
      this.setState({
        result: ""
      })
      return
    }
    if (input === ".") {
      let check = 0
      check = this.state.result.length - 1;
      if (this.state.result.charAt(check) == '.')
      ) {
        return
      }
    }
  }

  if (input === "+" || input === "-" || input === "*" || input === "/") {
    let check = 0
    check = this.state.result.length - 1;
    if (
      this.state.result.charAt(check) == '+' ||
      this.state.result.charAt(check) == '-' ||

```

```

        this.state.result.charAt(check) == '*' ||
        this.state.result.charAt(check) == '/'
    ) {
        return
    }
}
this.setState({
    result: this.state.result + (input + "")
})
}

renderButtonSet() {
    let layouts = calNum.map((buttonRows, index) => {
        let rowItem = buttonRows.map((buttonItem, buttonIndex) => {
            return (<TouchableOpacity
                key={buttonIndex}
                style={styles.container1}
                // onPress={() => handleOnPress(buttonItem)}
                onPress={this._onInputButtonPressed.bind(this, buttonItem)}
            >{
                <Text style={styles.textValue}>{buttonItem}</Text>

            }
            </TouchableOpacity>)
        });

        return <View style={styles.inputRow} key={'row-' + index}>{rowItem}</View>
    })
    console.log("Hello")
    return layouts
}

renderOpButtonSet() {
    let layouts = calOp.map((buttonRows, index) => {
        let rowItem = buttonRows.map((buttonItem, buttonIndex) => {
            return (<TouchableOpacity
                key={buttonIndex}
                style={styles.container1}
                // onPress={() => handleOnPress(buttonItem)}
                onPress={this._onInputButtonPressed.bind(this, buttonItem)}
            >{
                <Text style={styles.textValue}>{buttonItem}</Text>

```

```

    }
    </TouchableOpacity>)
  });

  return <View style={styles.inputRow} key={'row-' + index}>{rowItem}</View>
})
console.log("Hello1")
return layouts
}

render() {
  return (
    <View style={styles.container}>
      <View style={styles.resultContainer}>
        <Text style={styles.resultText}>{this.state.result}</Text>
      </View>

      <View style={styles.inputContainer}>

        <View style={styles.numButton}>
          {this.renderButtonSet()}
        </View>

        <View style={styles.opButton}>
          {this.renderOpButtonSet()}
        </View>

      </View>
    </View>
  );
}

const styles = StyleSheet.create({
  container: {
    flex: 1,
    // backgroundColor: '#fff',
    // alignItems: 'center',
    // justifyContent: 'center',
  },
  container1: {
    flex: 1,
    justifyContent: 'center',
    alignItems: 'center',
  },
});

```

```

    backgroundColor: 'black',
    margin: 1
  },
  resultContainer: {
    flex: 2,
    backgroundColor: 'rgb(37, 39, 41)',
    justifyContent: 'center',
  },
  inputContainer: {
    flex: 6,
    backgroundColor: 'rgb(37, 39, 41)',
    flexDirection: 'row',
  },
  numButton: {
    flex: 8,
    // backgroundColor: 'black',
  },
  resultText: {
    fontSize: '4em',
    textAlign: 'right',
    fontWeight: '5',
    padding: 20,
    color: 'white'
  },
  inputRow: {
    flex: 1,
    flexDirection: 'row',
  },
  textValue: {
    color: 'white',
    fontSize: 40,
    fontWeight: '5',
  },
  opButton: {
    flex: 3,
  },
},
});

export default App;

```