

# REACT NATIVE

Core React Native components

Basic Element

# React Native API

- Over 30 UI Primitives

View

Text

ScrollView

Image

TextInput

Switch

<https://reactnative.dev/docs/components-and-apis>

<https://docs.expo.dev/versions/latest>

HIENTH

# React Native API

- Mapping to Web UI primitives

View == div

Text == span / p / h\*

ScrollView == div + styling

Image == img

TextInput == input

# React Native API

- Primitives not globally in scope, must be imported

```
import {  
  View,  
  Text,  
  ScrollView,  
  Image,  
  TextInput,  
  Switch  
} from 'react-native';
```

# React Native API

- Device APIs
  - CameraRoll
  - Keyboard
  - NetInfo
  - Vibration
  - PanResponder

# Setup Project (15 min)

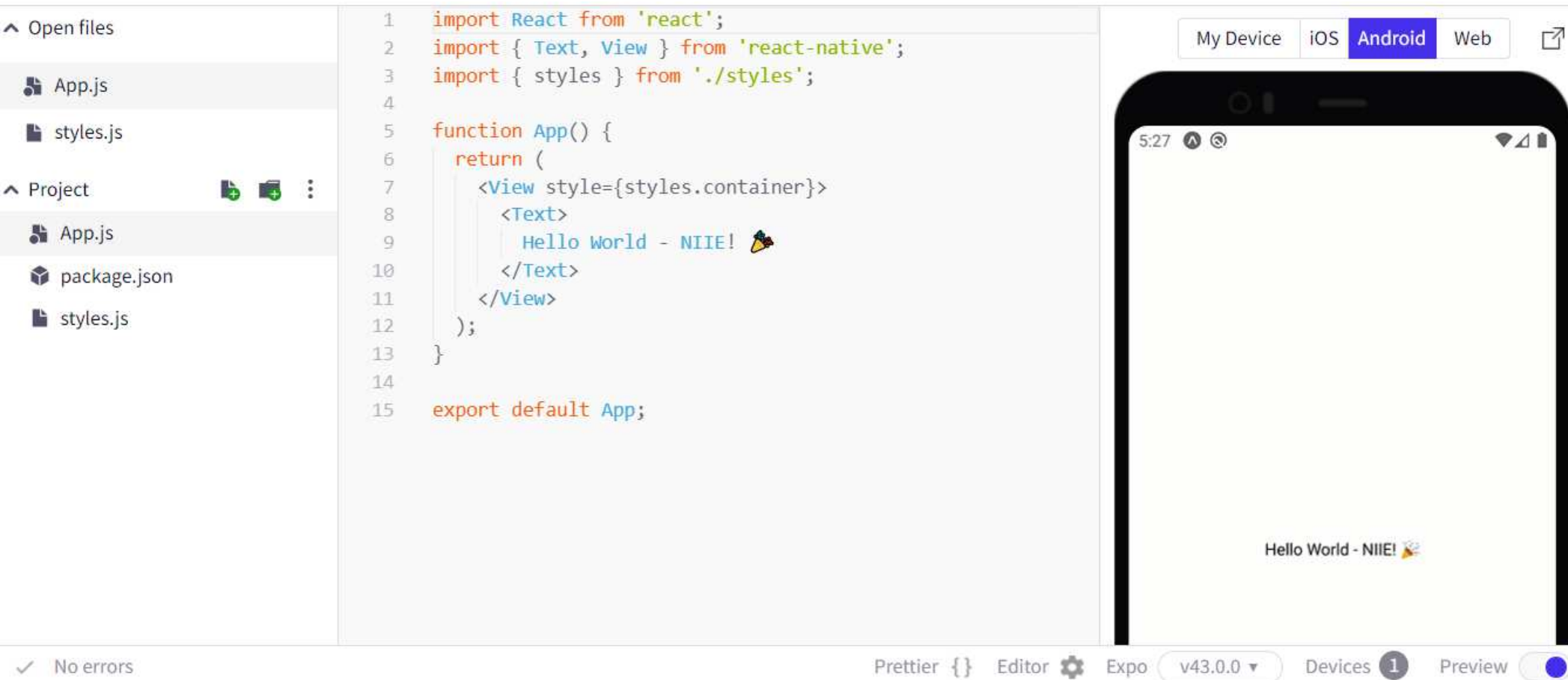
- Prerequisites
  - Expo CLI or React Native CLI is installed.
- Create new project flexbox with Expo
  - (Both) `$ expo init flexbox`
  - (Both) `$ cd flexbox`
- Run app
  - (Mac) `$ expo run:ios`
  - (Win) `$ expo run:android`

<https://docs.expo.dev/bare/hello-world>

**HIENLTH**

# Setup Project

- Setup the Screen like this



# JSX

- JSX is a JavaScript syntax extension that looks similar to XML
- We use a JSX to write User Interface (UI)
- JSX use camelCase
- We use JSX at the `render ()` function of a React component.



# JSX Syntax

`<Text>Hello World!</Text>`

Tag name: Text      Opening Tag      Closing Tag      Tag body

`<Image  
style={{height:100, width:100}}  
source={{uri: 'https://facebook.github.io/react/img/logo_og.png'}}  
>`

Self Closing Tag      Attribute Name      Attribute Value

# Attribute Value

- Using JavaScript Expression as Attribute Value, Use { }

```
<TextInput  
  style = {{height: 40, borderColor: 'gray', borderWidth: 1}}  
  value = 'Useless TextInput'  
>
```

- Using String as Attribute Value, Use ' '

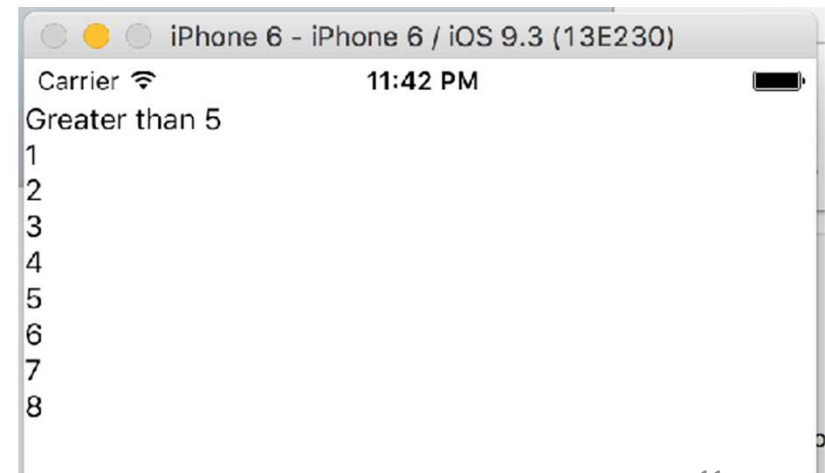
# Puttin JS Logic in JSX

```
class flexbox extends Component {  
  render() {  
    let number = 3;  
    return (  
      <View>  
        <View style={{ height: 20 }}></View>  
        <Text>{number > 5 ? 'Greater than 5' : 'Less than 5'}</Text>  
        {...Array(number).map((x, i) =>  
          <Text>{i + 1}</Text>  
        )}  
      </View>  
    );  
  }  
}
```



# Puttin JS Logic in JSX

```
class flexbox extends Component {  
  render() {  
    let number = 8;  
    return (  
      <View>  
        <View style={{ height: 20 }}></View>  
        <Text>{number > 5 ? 'Greater than 5' : 'Less than 5'}</Text>  
        {[...Array(number)].map((x, i) =>  
          <Text>{i + 1}</Text>  
        )}  
      </View>  
    );  
  }  
}
```



HIENTH

# Comment

- To comment in JSX, put it between `{ /* */ }`, `{ // ... \n }`

```
class flexbox extends Component {  
  render() {  
    let number = 3;  
    return (  
      <View>  
        { //<View style={{ height: 20 }}></View>  
      }  
      <Text>{number > 5 ? 'Greater than 5' : 'Less than 5'}</Text>  
      { /*{[...Array(number)].map((x, i) =>  
        <Text>{i + 1}</Text>  
      )} */ }  
    </View>  
  );  
}
```

# One Outmost Parent Tag Rules

OK, Only one outmost parent tags: View

```
class flexbox extends Component {  
  render() {  
    return (  
      // OK  
      <View>  
        <View style={{height:20}}></View>  
        <Text>Content 1</Text>  
        <Text>Content 2</Text>  
      </View>  
    );  
  }  
}
```

Carrier  
Content 1  
Content 2

BAD! Multi outmost parent tags: Text, Text

```
class flexbox extends Component {  
  render() {  
    return (  
      // Bad!  
      <Text>Content 1</Text>  
      <Text>Content 2</Text>  
    );  
  }  
}
```

iPhone 6 - iPhone 6 / iOS 9.3 (13E230)

**SyntaxError /Users/kobkrit/git/flexbox/  
renderJSX.js: Adjacent JSX elements must  
be wrapped in an enclosing tag (19:6)**

# Basic Element

```
<View></View>
```

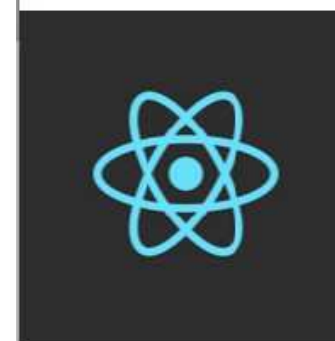
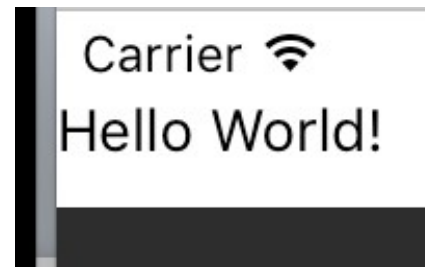
```
<Text>Hello World!</Text>
```

```
<Image  
  style={{height:100, width:100}}  
  source={{uri: 'https://facebook.github.io/react/img/logo_og.png'}}  
>
```

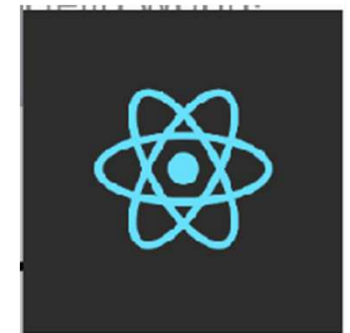
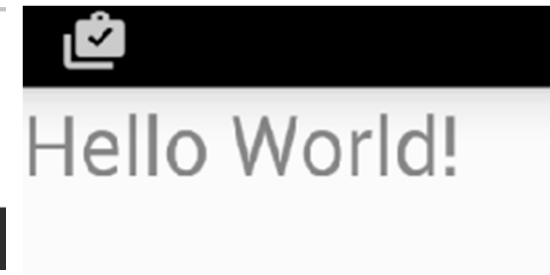
```
<Switch />
```

HIEUNLTH

iOS  
(Container)



Android  
(Container)

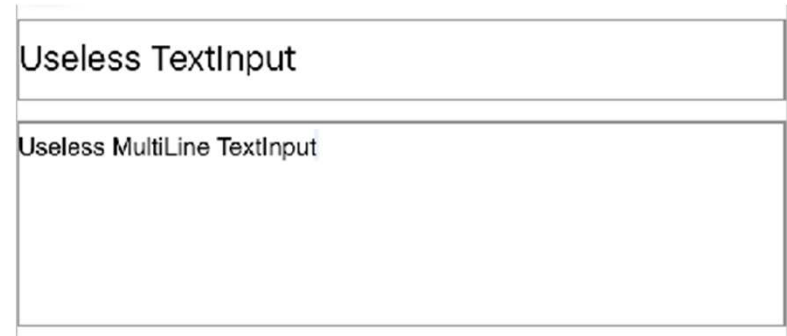


## Basic Elements (cont)

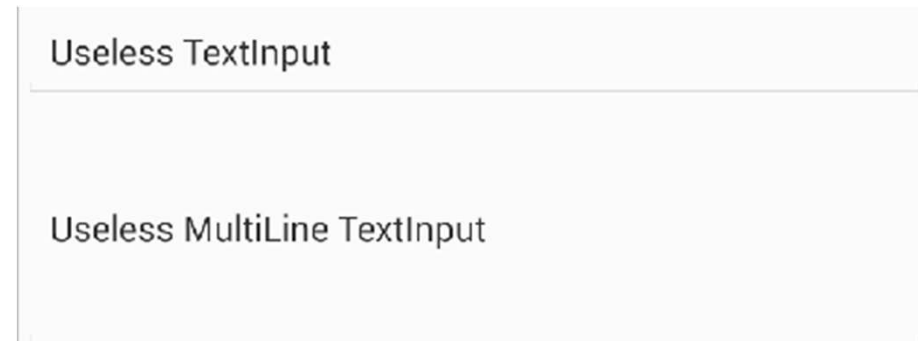
```
<TextInput  
  style={{height:40, borderColor:'gray',  
borderWidth: 1}}  
  value='Useless TextInput'  
>
```

```
<TextInput  
  multiLine={true} numberOfLine={4}  
  style={{height:100, borderColor:'gray',  
borderWidth: 1}}  
  value='Useless MultiLine TextInput'  
>
```

iOS



Android



HIEUNLTH



## Basic Elements (cont)

```
<TouchableOpacity onPress={() => {}}
  style={{borderColor:'#f00',
  backgroundColor:'#faa', borderWidth:
  1, padding: 10}}>
  <Text>Touch me for Opacity!</Text>
</TouchableOpacity>
```

```
<TouchableHighlight onPress={() => {}}
  underlayColor='#f00a'
  style={{borderColor:'#f00',
  backgroundColor:'#faa', borderWidth: 1,
  padding: 10}}>
  <Text>Touch me for Highlight!</Text>
</TouchableHighlight>
```

iOS

Touch me for Opacity!

Touch me for Highlight!

iOS & Android: Tapping

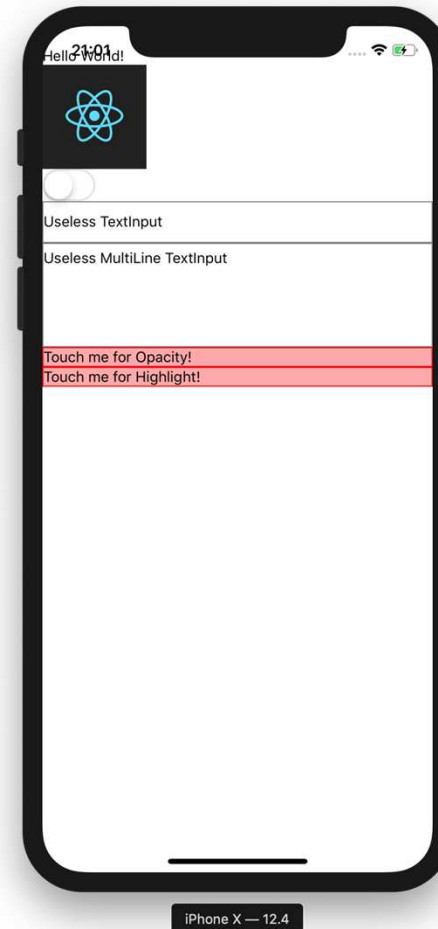
Touch me for Opacity!

Touch me for Highlight!

HIENLTH

# JSX Example

```
render() {  
  return (  
    <View>  
      <View style={{height: 20}}/>  
      <Text>Hello World!</Text>  
      <Image  
        style={{height: 100, width: 100}}  
        source={{uri: 'https://facebook.github.io/react/logo-og.png'}}  
      />  
      <Switch/>  
      <TextInput  
        style={{height:40, borderColor: 'gray', borderWidth: 1}}  
        value='Useless TextInput'  
      />  
      <TextInput  
        multiline={true}  
        numberOfLines={4}  
        style={{height:100, borderColor: 'gray', borderWidth: 1}}  
        value='Useless MultiLine TextInput'  
      />  
      <TouchableOpacity  
        onPress={() => {}}  
        style={{borderColor: '#f00', backgroundColor: '#faa', borderWidth: 1}}>  
        <Text>Touch me for Opacity!</Text>  
      </TouchableOpacity>  
      <TouchableHighlight  
        onPress={() => {}}  
        underlayColor='#f00a'  
        style={{borderColor: '#f00', backgroundColor: '#faa', borderWidth: 1}}>  
        <Text>Touch me for Highlight!</Text>  
      </TouchableHighlight>  
    </View>  
  );  
}
```



iPhone X — 12.4

*Thank  
you!*

## Exercise

- Design Login UI includes 2 input text: username, password, and button
  - Hint: Using minimum components: View, Text, TextInput, TouchableOpacity
  - Use and edit basic CSS
- Open Snack Expo to do exercise
- Duration: 30 minutes (lecturer will correct student)
- Individual exercise