พัฒนา Mobile Apps ด้วย React Native

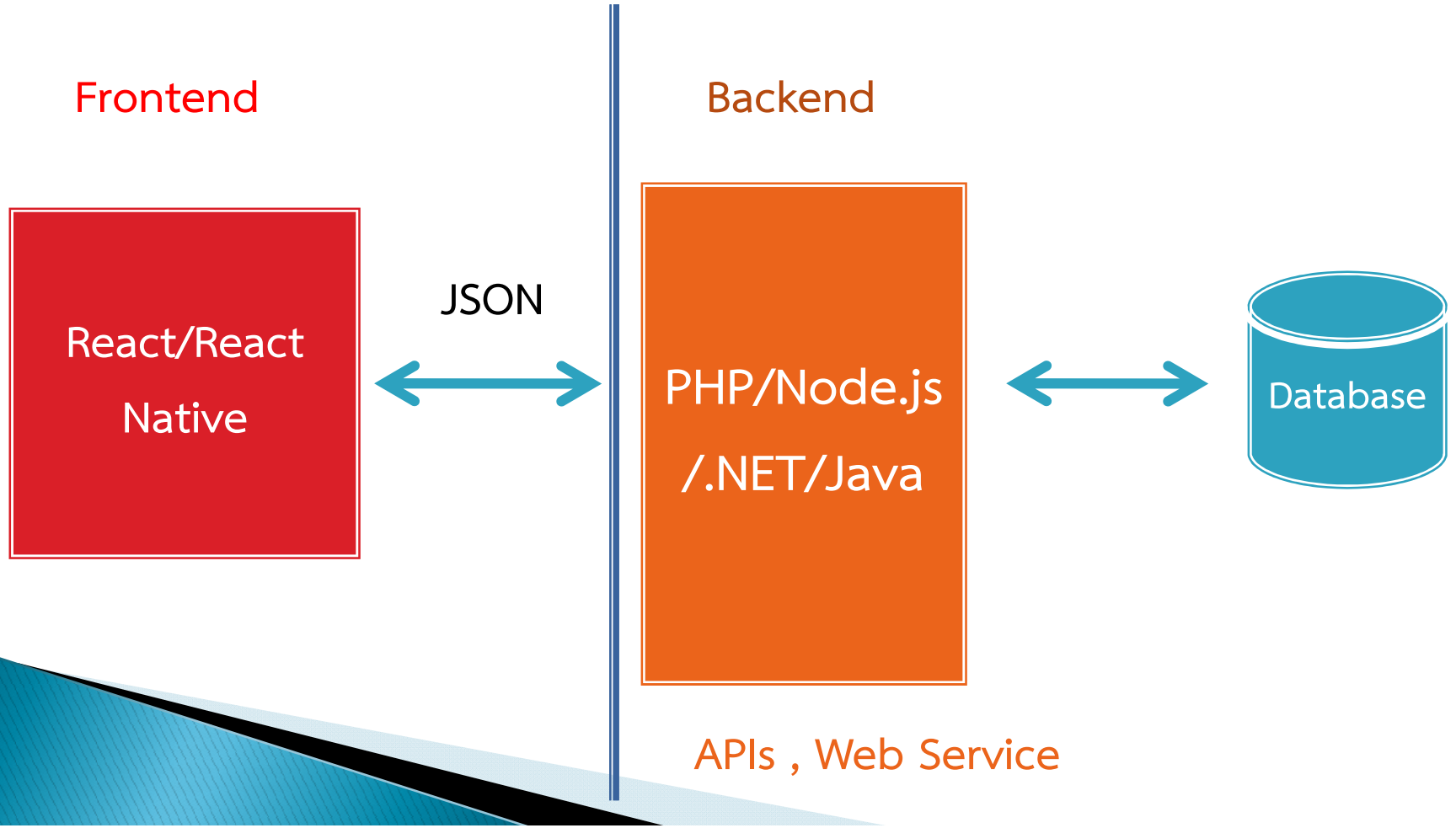
Workshop: ระบบจัดการข้อมูลข่าวสาร

โค้ชเอก

Codingthailand.com

ภาพใหญ่ของหลักสูตรนี้

- พัฒนา Mobile Apps ด้วย **React Native (Workshop)**



เตรียมเครื่องเปิด Virtualization Technology ที่ BIOS

- เปิด Virtualization Technology (Intel VT-x) ที่ BIOS
  (กด F2 หรือ Delete หรือ Esc เพื่อเข้า BIOS)
  มองหาเมนู หรือคำว่า "VT", "Virtualization Technology",
  หรือ "VT-d."

  แล้วเปิดใช้งาน จากนั้นบันทึก แล้ว restart อีกครั้ง

โปรแกรมที่ต้องติดตั้งก่อน

- ติดตั้ง git ดาวน์โหลดได้ที่นี่ โดยคลิก next อย่างเดียวจนเสร็จเรียบร้อย
  https://git-scm.com/download/win

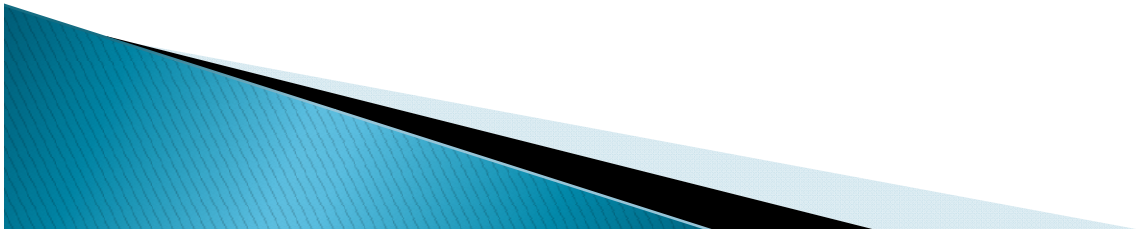
- ติดตั้ง Node.js เวอร์ชัน LTS โดยคลิก next อย่างเดียวจนเสร็จเรียบร้อย ตามลิงก์
  https://nodejs.org/

- ติดตั้ง Python เวอร์ชัน 2 เท่านั้น https://www.python.org/downloads/release/python-2715/

- ติดตั้ง Visual Studio Code คลิก next อย่างเดียวจนเสร็จเรียบร้อย ตามลิงก์ https://code.visualstudio.com/
- ติดตั้ง JRE 8 คลิก next อย่างเดียวจนเสร็จเรียบร้อย
  https://www.oracle.com/technetwork/java/javase/downloads/jre8-downloads-2133155.html
- ติดตั้ง Java JDK เวอร์ชัน 8 คลิก next อย่างเดียวจนเสร็จเรียบร้อย
  https://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html
- ติดตั้ง Android Studio (คลิก next อย่างเดียวจนเสร็จเรียบร้อย) https://developer.android.com/studio/

เช็คโปรแกรมที่ติดตั้ง

- node -v
- npm -v
- java -version
- git –version
- adb version

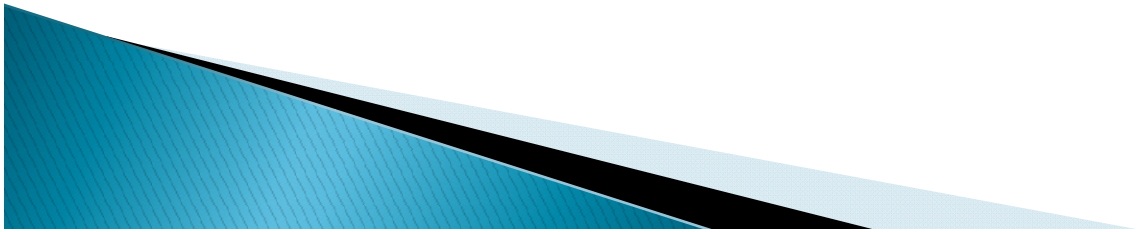


ลิงก์แนะนำการติดตั้ง React Native

- https://facebook.github.io/react-native/docs/getting-started.html

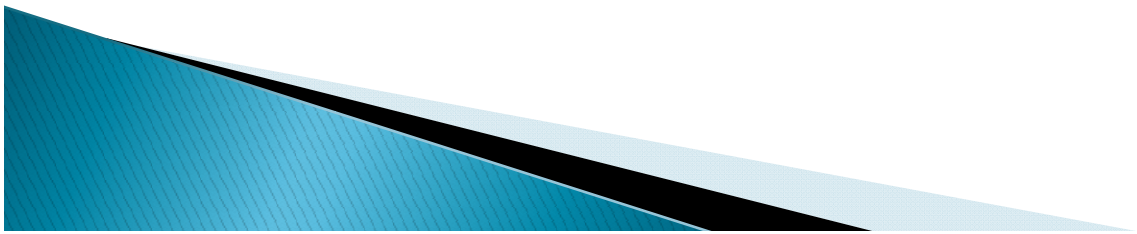
- เปิดคู่มือการอบรม เพื่อติดตั้ง SDK และ Emulator

- ติดตั้ง Visual Studio Code และ Extensions



ทดสอบการทำงานของโปรแกรมที่ติดตั้ง

- ตรวจสอบโปรแกรมต่างๆ

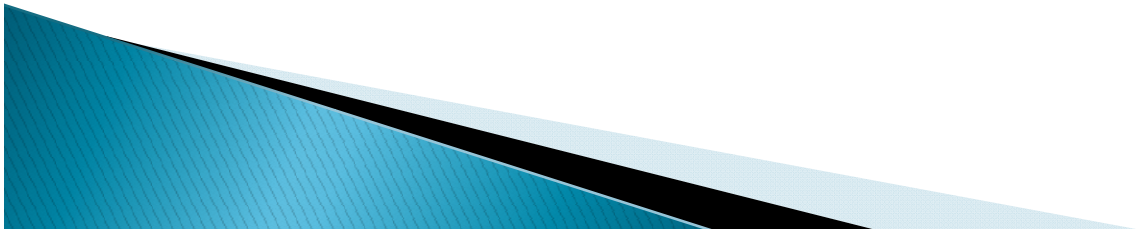


React คืออะไร

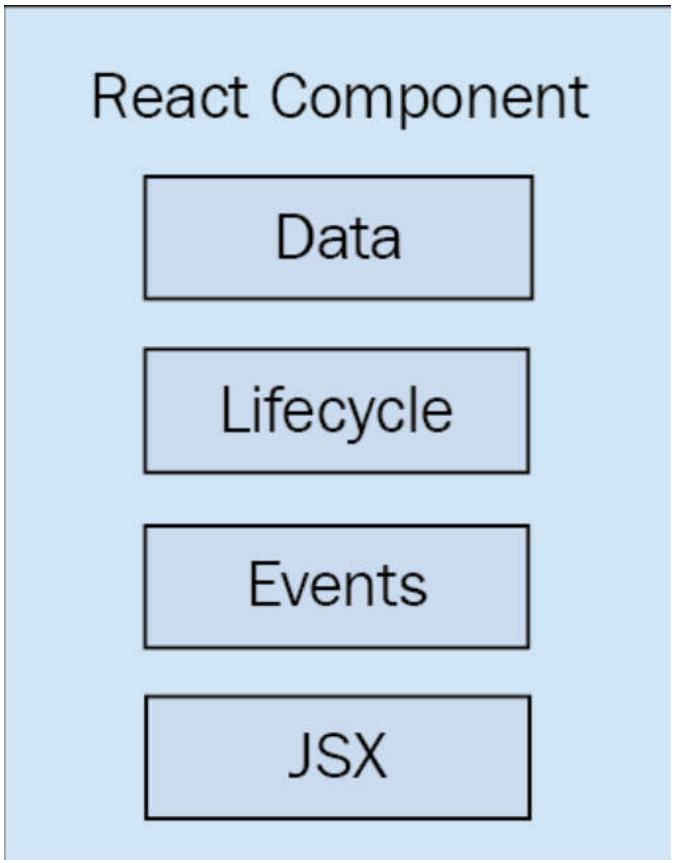
- **React** is a JavaScript library open sourced by and used within Facebook, and was originally used to **build user interfaces** for web applications.

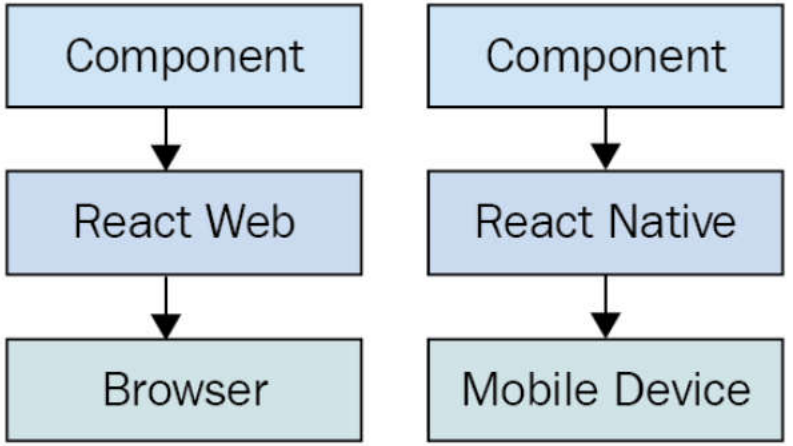
- **React** is a JavaScript library for building user interfaces across a variety of platforms.

- เว็บไซต์หลัก: https://reactjs.org/



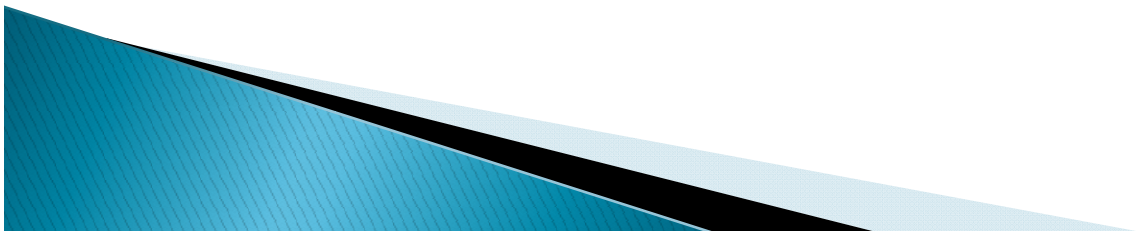
Components


React Component
Data
Lifecycle
Events
JSX


Component
Component
React Web
React Native
Browser
Mobile Device

React Concept

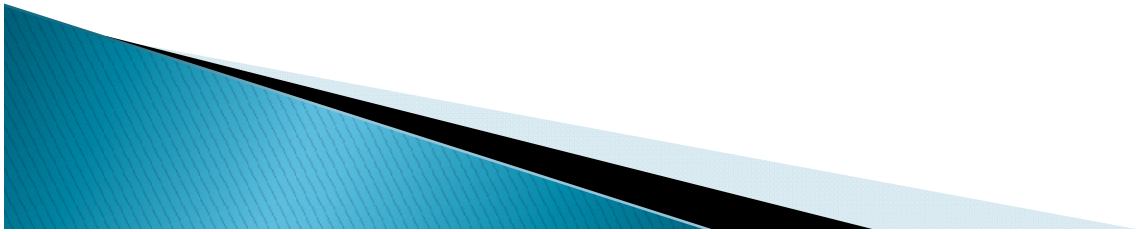
- JSX
- Components
- Props
- State
- Handling Events



JSX

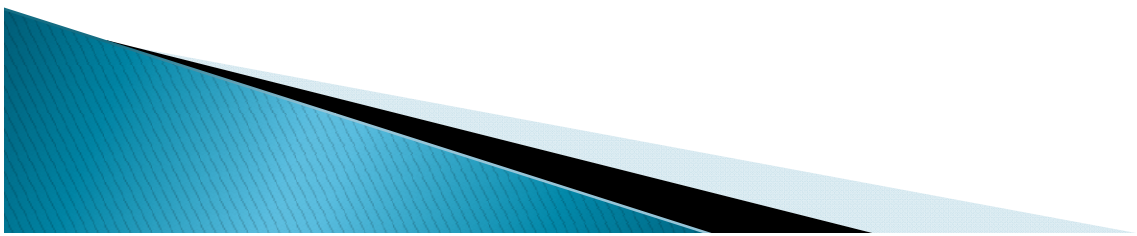
- https://reactjs.org/docs/introducing-jsx.html

- https://reactjs.org/docs/jsx-in-depth.html

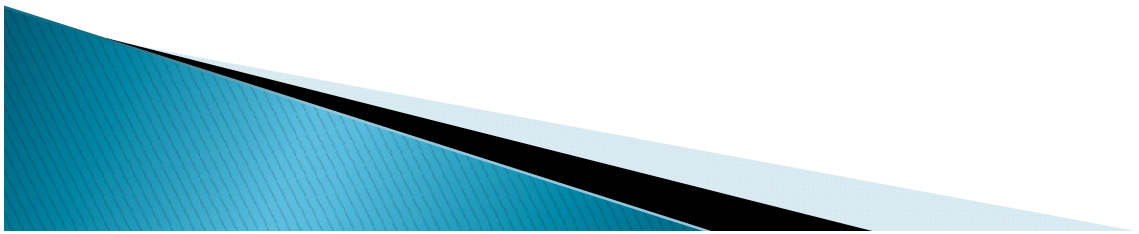


Components

- **Data**: This is data that comes from somewhere (the component doesn't care where), and is rendered by the component.
- **Lifecycle**: These are methods that we implement that respond to changes in the lifecycle of the component. For example, the component is about to be rendered.
- **Events**: This is code that we write for responding to user interactions.
- **JSX**: This is the syntax of React components used to describe UI structures.

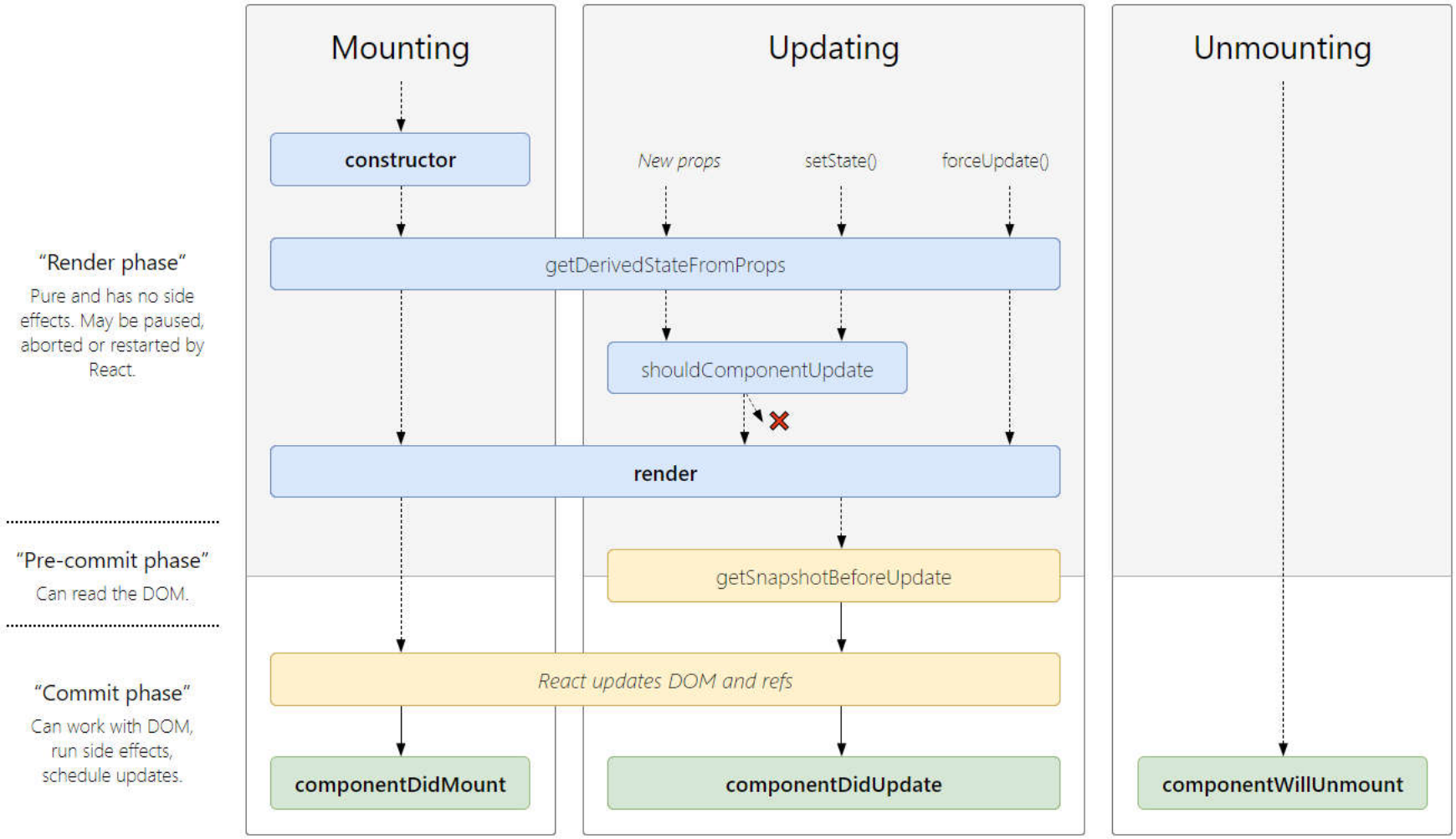


React Lifecycle Methods



The Component Lifecycle

http://projects.wojtekmaj.pl/react-lifecycle-methods-diagram/



The componentDidMount lifecycle method

- **componentDidMount** is called **exactly once, just after the component has been loaded.** This method is a good place **to fetch data with ajax calls**

```
class MainComponent extends Component {
  constructor() {
    super()
    this.state = { loading: true, data: {} }
  }
  componentDidMount() {
    // simulate ajax call
    setTimeout(() => {
      this.setState({
        loading: false,
        data: {name: 'Nader Dabit', age: 35}
      })
    }, 2000)
  }
  render() {
    if(this.state.loading) {
      return <Text>Loading</Text>
    }
    const { name, age } = this.state.data
    return (
      <View>
        <Text>Name: {name}</Text>
        <Text>Age: {age}</Text>
      </View>
    )
  }
}
```

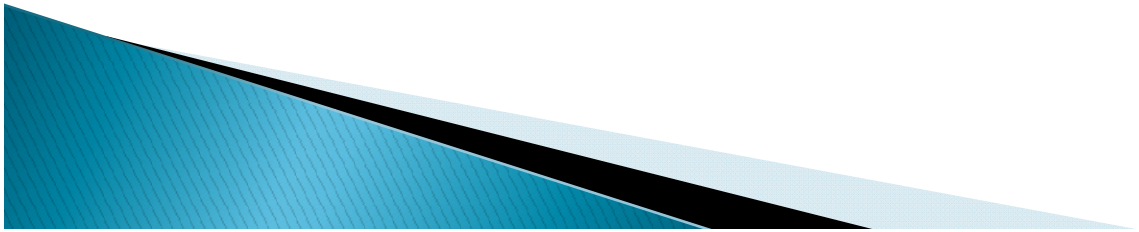
The componentWillUnmount lifecycle method

- **componentWillUnmount is called before the component is removed** from the application. Here, you can perform any necessary **cleanup, remove listeners, or remove timers** that were set up in componentDidMount.

```
class MainComponent extends Component {

   handleClick() {
      this._timeout = setTimeout(() => {
        this.openWidget();
      }, 2000);
    }
    componentWillUnmount() {
      clearTimeout(this._timeout);
    }
    render() {
      return <SomeComponent
                handleClick={() => this.handleClick()} />
    }
}
```

ทำความรู้จักกับ React Native

- **React Native** is a framework for building native mobile apps in JavaScript using the React JavaScript library and compiles to real native components.



Thinking in components

A Basic React Native Class

- There are two main types of React Native components, stateful and stateless.

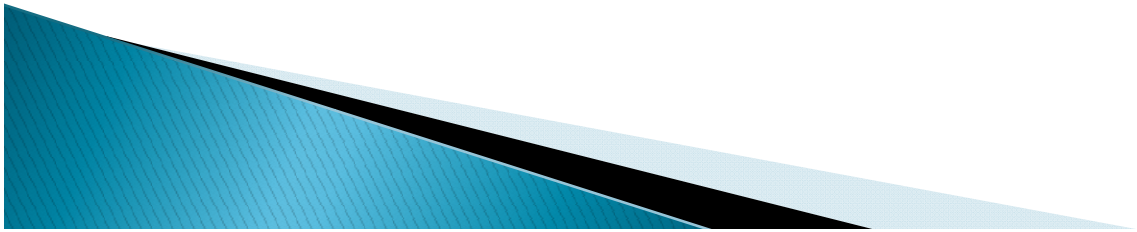
- Stateful component using ES6 class

```
class HelloWorld extends React.Component {
  constructor() {
    super();
    this.state = { name: 'Chris' }
  }

  render () {
    return (
      <SomeComponent />
    )
  }
}
```

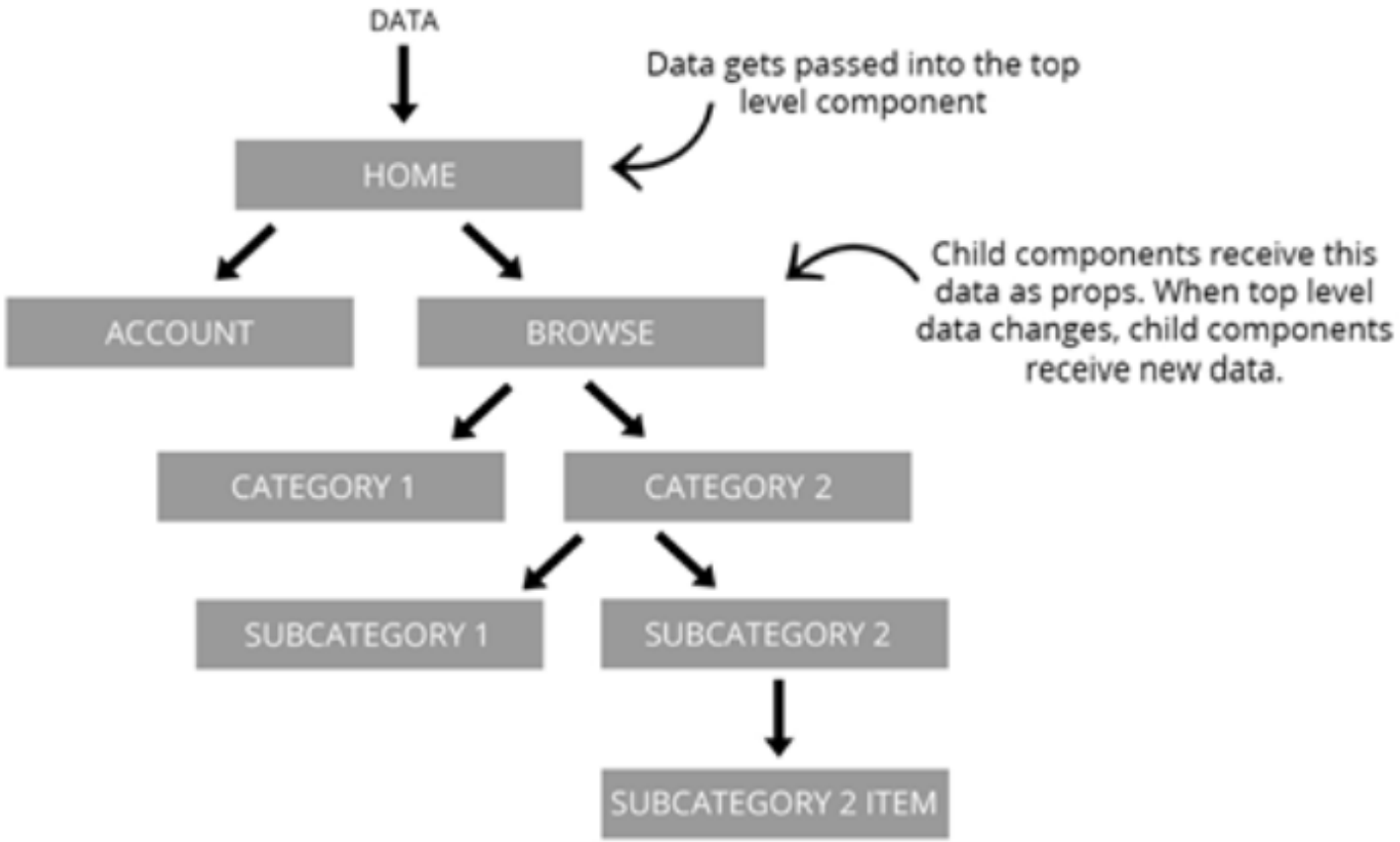
A Basic React Native Class

- Stateless component

```
const HelloWorld = () => (
  <SomeComponent />
)
```



How one-way data flow works

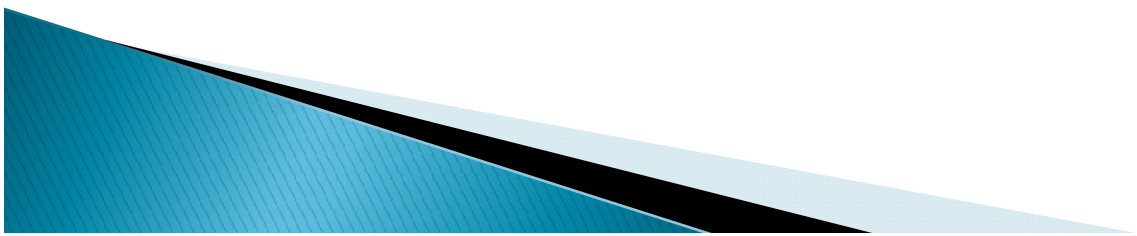


Managing component data using state

- state is declared when the component is created, and **its structure is a plain JavaScript object.**

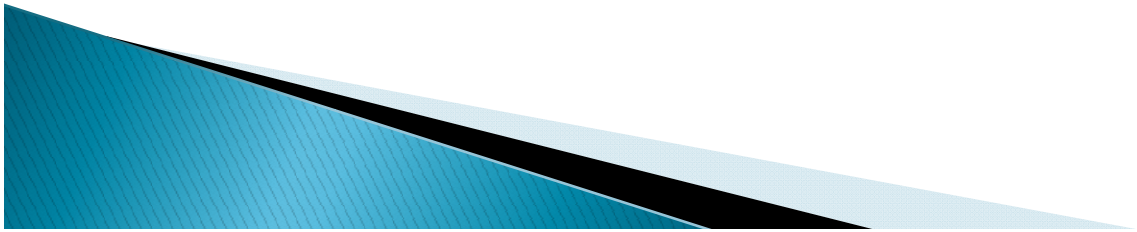
- State can be updated within the component using a function called **setState**

- When the state of a component changes using the setState function, **React rerenders the component.** If any child components are inheriting this state as props, then **all of the child components get rerendered as well.**



Setting *Initial State*

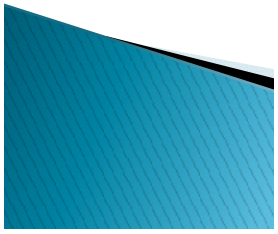
- State is initialized when the component is created in either the constructor or with a property initializer. Once the state is initialized, it is then available in the component **as this.state.**



Setting *Initial State*

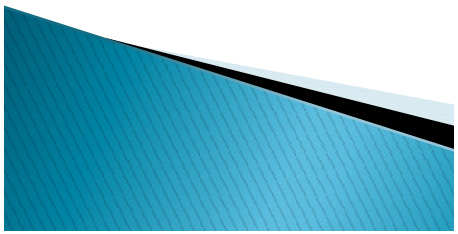
- Setting state with a property initializer

```
import React from 'react'

class MyComponent extends React.Component

  state = {
    year: 2016,
    name: 'Nader Dabit',
    colors: ['blue']
  }

  render() {
    return (
      <View>
        <Text>My name is: { this.state.name }</Text>
        <Text>The year is: { this.state.year }</Text>
        <Text>My colors are { this.state.colors[0] }</Text>
      </View>
    )
  }
}
```

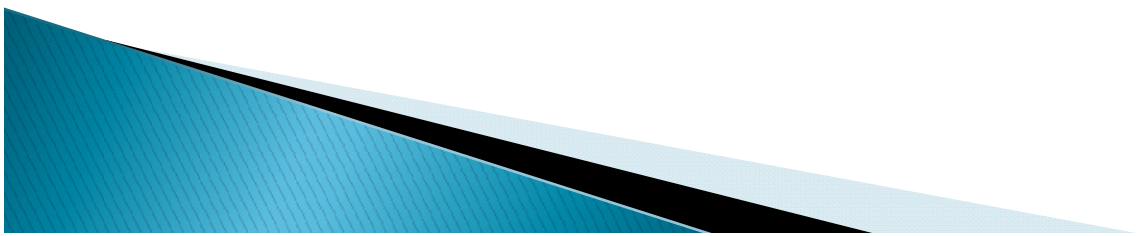


Updating state

```
class MyComponent extends Component {
    constructor(){
      super()
      this.state = {
        year: 2016,
      }
    }
    updateYear() {
      this.setState({
        year: 2017
      })
    }
    render() {
      return (
        <View>
          <Text
            onPress={() => this.updateYear()}>
            The year is: { this.state.year }
          </Text>
        </View>
      )
    }
}
```



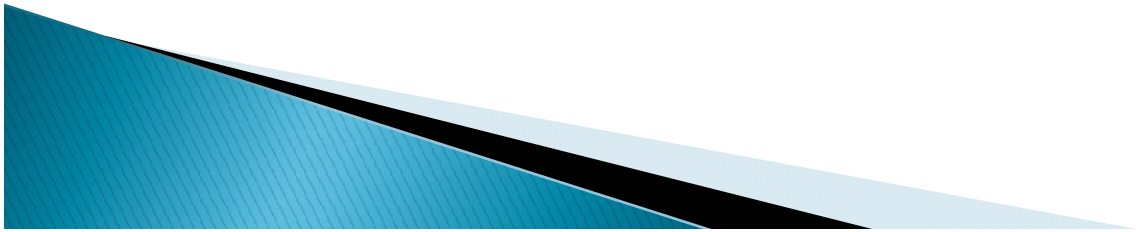
Managing component data using props



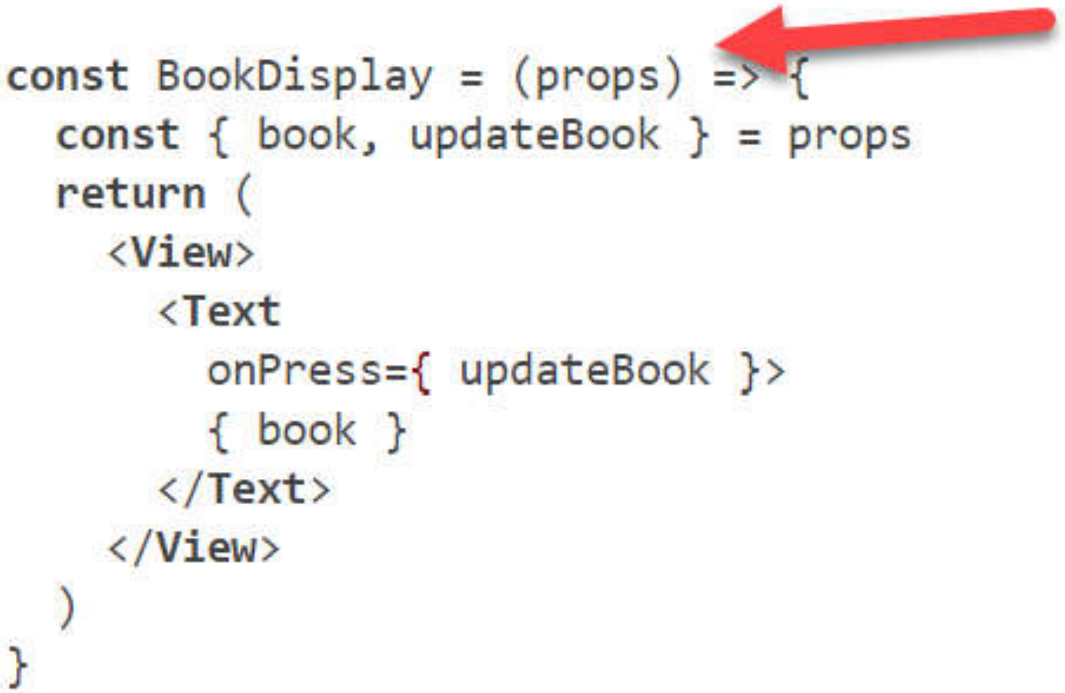
Managing component data using props

- **props** (short for properties) are **a component's inherited values or properties** that have been **passed down from a parent component.**

- "a way of passing data from parent to child."



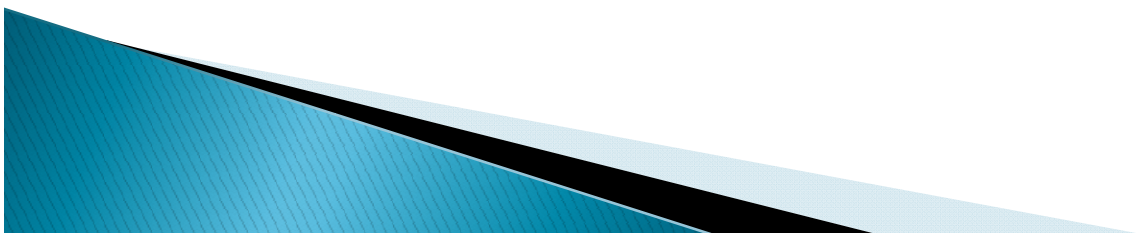
Props with stateless components



```
const BookDisplay = (props) => {
  const { book, updateBook } = props
  return (
    <View>
      <Text
        onPress={ updateBook }>
        { book }
      </Text>
    </View>
  )
}
```

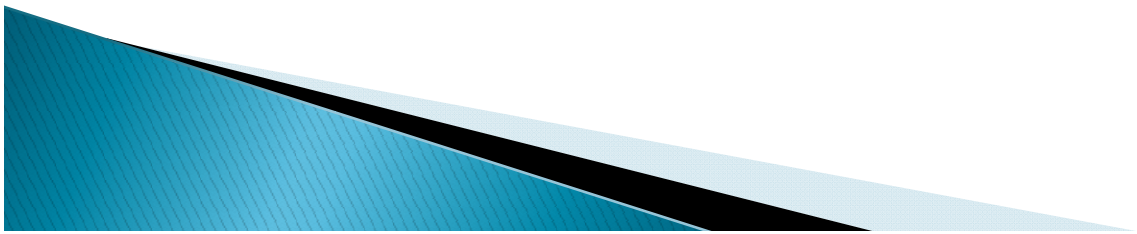
Destructuring props in a stateless component

```
const BookDisplay = ({ updateBook, book }) => {
  return (
    <View>
      <Text
        onPress={ updateBook }>
        { book }
      </Text>
    </View>
  )
}
```

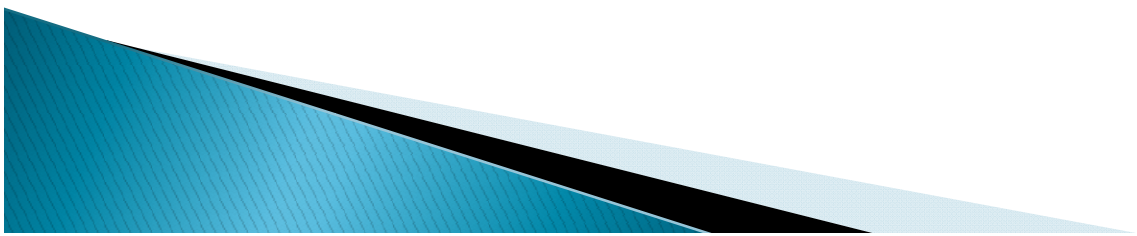


Static props

```
class MyComponent extends Component {
  render() {
    return (
      <BookDisplay book=”React Native in Action” />
    )
  }
}
class BookDisplay extends Component {
  render() {
    return (
      <View>
        <Text>{ this.props.book }</Text>
      </View>
    )
  }
}
```



Introduction to styling



Applying styles

- Using inline styles

```
import React, { Component } from 'react'
import { View } from 'react-native'

class App extends Component {
  render () {
    return (
      <View style={{marginLeft: 20}}>    // A
        <Text style={{fontSize: 18,color: 'red'}}>Some Text</Text> // B
      </View>
    )
  }
}
A An inline style applied to a React Native component.
B Multiple inline styles applied at once
```

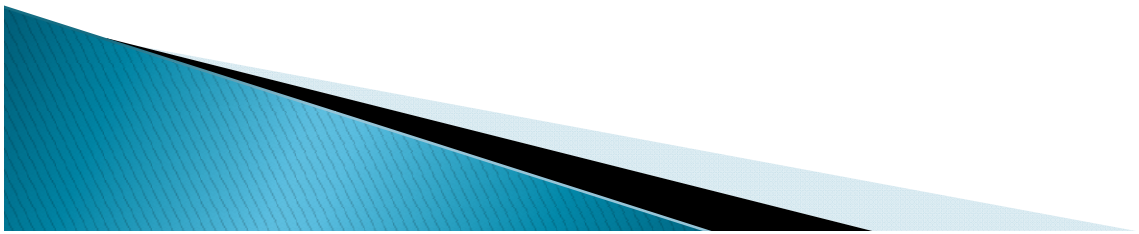
Referencing styles defined within a StyleSheet

```
import React, { Component } from 'react'
import { StyleSheet, View } from 'react-native'

class App extends Component {
  render () {
    return (
      <View style={styles.container}>     // A
        <Text style={[styles.message,styles.warning]}>Some Text</Text> //
      </View>
    )
  }
}
const styles = StyleSheet.create({
  container: {          // C
    marginLeft: 20
  },
  message: {            // C
    fontSize: 18
  },
  warning: {
    color: 'red'        // C
  }
})
A Referencing the container style defined within the styles stylesheet.
B Using an array to reference both the message and warning styles from th
C Defining the styles using StyleSheet.create
```

b

Organizing styles

- declaring stylesheets within the same file as the component.
- declaring stylesheets in a separate file, outside of the component.



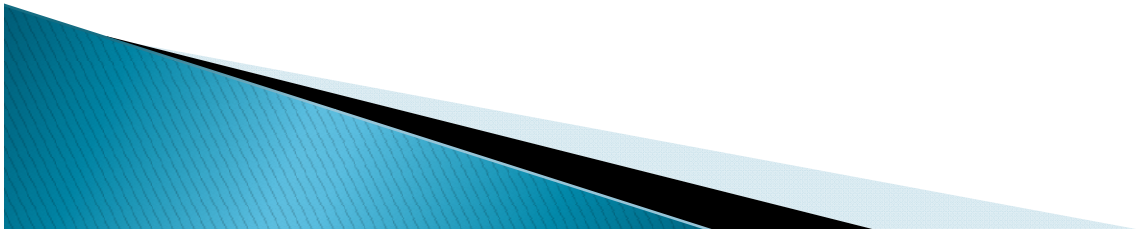
Externalizing a component's stylesheets (styles.js)

```
import { StyleSheet } from 'react-native'

const styles = StyleSheet.create({   // A
  container: {    // B
    marginTop: 150,
    backgroundColor: '#ededed',
    flexWrap: 'wrap'
  }
})

const buttons = StyleSheet.create({  // C
  primary: {     // D
    flex: 1,
    height: 70,
    backgroundColor: 'red',
    justifyContent: 'center',
    alignItems: 'center',
    marginLeft: 20,
    marginRight: 20
  }
})

export { styles, buttons }    // E
```

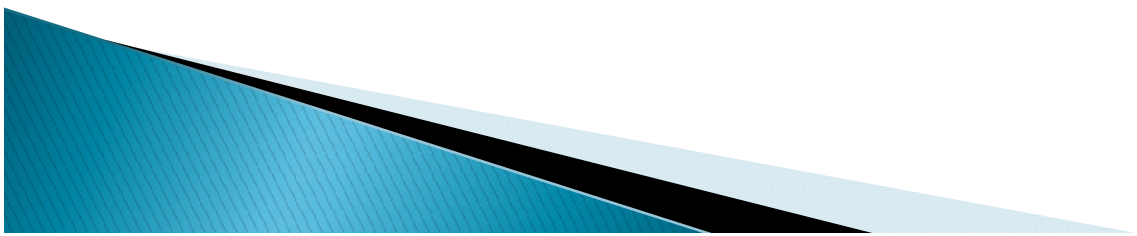
Externalizing a component's stylesheets (styles.js)

- Importing external stylesheets

```
import { styles, buttons } from './component/styles'    // A

<View style={styles.container}>    // B
  <TouchableHighlight style={buttons.primary} />    // C
    ...
  </TouchableHighlight>
</View>
A Importing multiple stylesheets exported from styles.js
B A reference to the styles.container style created in styles.js
C A reference to the buttons.primary style created in styles.js
```



Redux

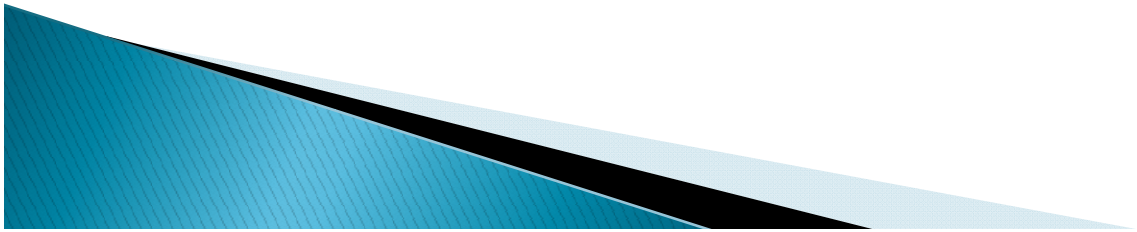


Redux และ React Redux

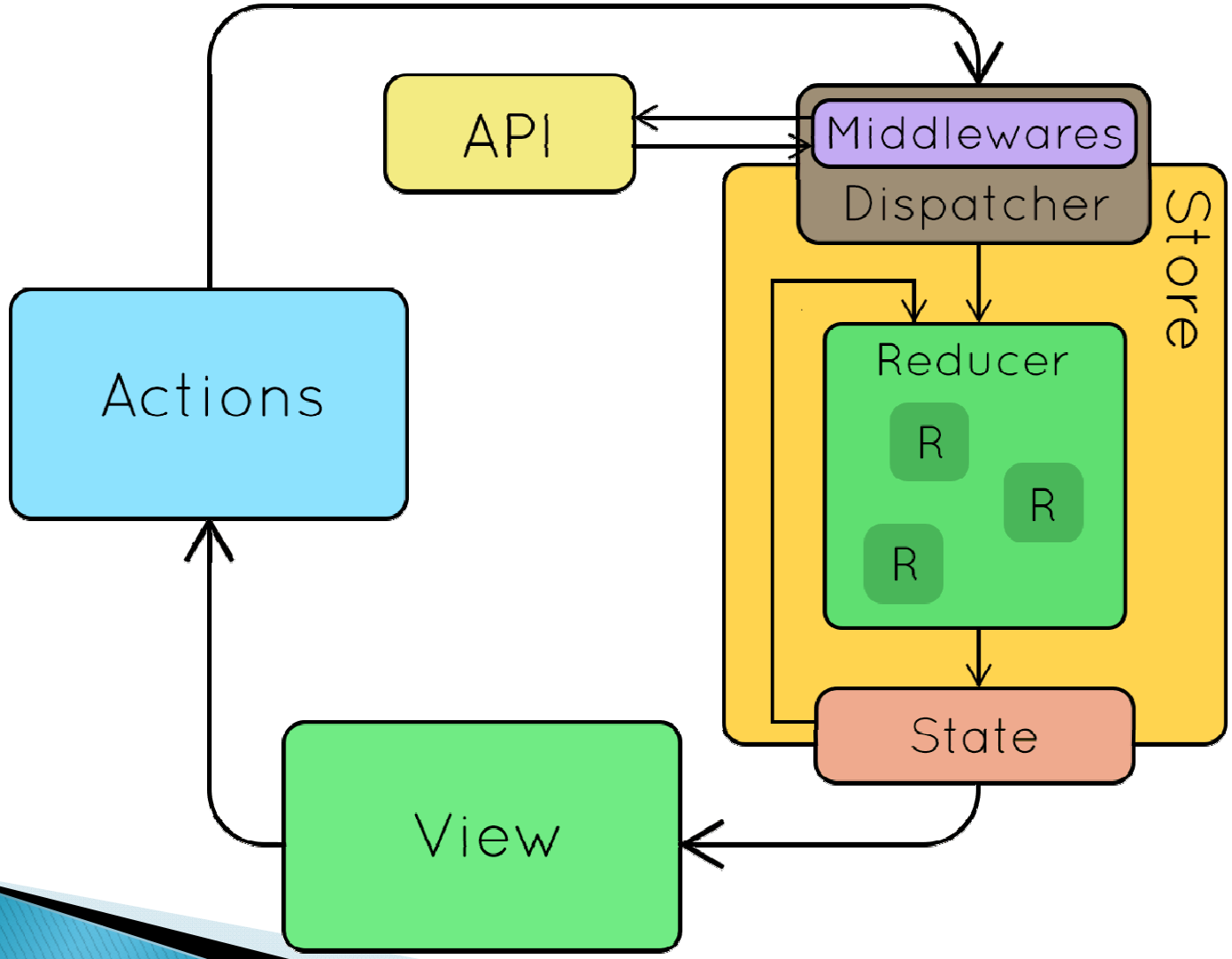
- npm install redux react-redux --save

- https://redux.js.org/
- Redux is a predictable state container for JavaScript apps.

- https://github.com/reduxjs/react-redux
- React Redux is the official React binding for Redux. It lets your React components read data from a Redux store, and dispatch actions to the store to update data.

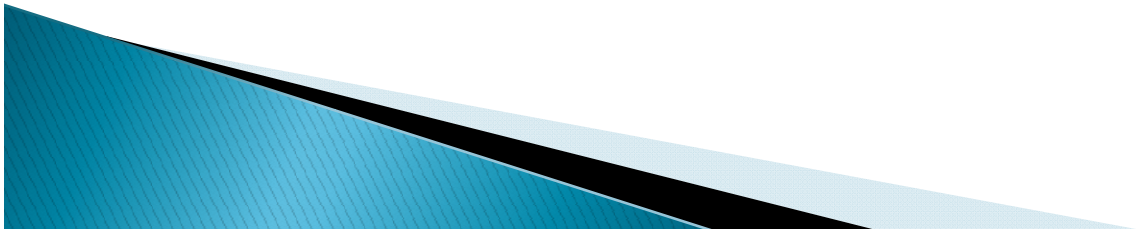


ภาพใหญ่ของ Redux



ดู Animation ภาพใหญ่ของ Redux ที่นี่

- https://github.com/reduxjs/redux/issues/653#issuecomment-216844781



The end