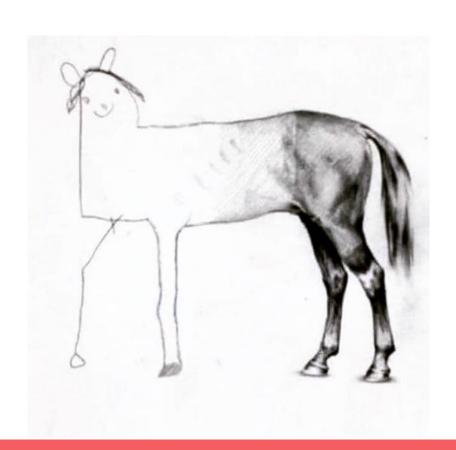
# React Native

@tinc2k

## [horse]



## **Big Picture**

- JavaScript
  - O ES6+
  - O npm
- React
  - O props
  - O state, setState()
  - O JSX
  - O render() & virtual DOM
  - component lifecycle
- HTML5
  - O flexbox
  - O fetch
- React Native
  - $\circ$

### **ES6+**

```
let, const
• for...of, for...in for (Let x of items) {...}
 arrow functions Let records = list.find(r => r.id === id);
classes
                class Hello extends React.Component {...}

    async/await
    Let records = await store.fetchRecords(id);

   array & object spread syntax < Component {...this.props} />
   destructuring
                                  const \{a, b, c\} = obj;
   modules
     export default class Record extends React.Component {...
     import { StyleSheet, Text, View } from 'react-native';
```

### React

- declarative -> predictable -> confidence -> reliability
- components
- props
- state, setState()
- render(), Virtual DOM, reconciliation
- JSX
- mounting lifecycle
  - O constructor, componentWillMount, componentDidMount, render
- updating lifecycle
  - componentWillReceiveProps, shouldComponentUpdate, componentWillUpdate, render, componentDidUpdate

```
class Hello extends React.Component {
   constructor(props) {
      super(props);
   }
   render() {
      return <h1>Hello {this.props.name}.</h1>;
   }
}
```

```
function Hello(props) {
    return <h1>{props.name}</h1>;
}

const Hello = (props) => (<h1>Hello {props.name}</h1>);
```

```
class Hello extends Component {
 state = { counter: 0 };
  componentDidMount() {
   setInterval(() => this.setState({ counter: this.state.counter + 1 }), 1000);
 render() {
   return (
     <div>
       <h1>Hello {this.props.name}.</h1>
       This screen is open for {this.state.counter} seconds.
     </div>
```

```
class Hello extends Component {
 state = { counter: 0 };
  componentDidMount() {
    setInterval(() => this.setState({ counter: this.state.counter + 1 }), 1000);
 render() {
   return (
      <View>
        <Text>Hello, {this.props.name}.</Text>
        <Text>This screen is open for {this.state.counter} seconds.</Text>
      </View>
```

## **Core Components**

FlatList, SectionList, ListView

## **Styling & StyleSheet**

- dimensions: fixed | flex
- layout
  - flex, flexDirection, justifyContent, alignItems
  - o width, height
  - o margin, padding
- visual (View)
  - color, backgroundColor
  - borderWidth, borderColor, borderRadius
  - opacity
- visual (Text)
  - o fontFamily, fontSize, fontStyle, fontWeight, lineHeight, textAlign...

### **Flexbox**

- flexDirection
  - column | row
- justifyContent
  - o flex-start | center | flex-end | space-around | space-between | space-evenly
- alignItems
  - o flex-start | center | flex-end | stretch
- flex: n
- http://www.reactnativeexpress.com/flexbox

## **UI, Native & Custom Components**

#### User Interface

Button, Picker, Slider, Switch

#### iOS

 ActionSheetIOS, AlertIOS, DatePickerIOS, ImagePickerIOS, NavigatorIOS, ProgressViewIOS, PushNotificationIOS, SegmentedControlIOS, TabBarIOS

#### Android

BackHandler, DatePickerAndroid, DrawerLayoutAndroid, PermissionsAndroid,
 ProgressBarAndroid, TimePickerAndroid, ToastAndroid, ToolbarAndroid, ViewPagerAndroid

#### Platform-specific

- file extensions: Button.ios.js | Button.android.js
- Platform module: Platform.OS, Platform.select, Platform.Version

### **Touch**

- Native props
  - o <Button onPress={this.\_onPressButton} title="Press Me" />
- Touchables
  - TouchableHighlight, TouchableOpacity, TouchableWithoutFeedback

```
<TouchableOpacity onPress={this.handlePress}>
  <View style={styles.article}></View>
  </TouchableOpacity>
```

Gesture Responder System

### **Animation**

Animated API

```
new Animated.Value(0)
Animated.timing(foo, { toValue: 1, duration: 1000 }).start()

<Animated.View style={{...this.props.style, opacity: fadeAnim }}>
```

LayoutAnimation

## **Navigation**

npm install --save react-navigation

- StackNavigator
- route configuration
- nesting
- this.props.navigation

```
let { navigation } = this.props;
navigation.navigate('Details');
navigation.navigate('Details', { id: 15 });
let id = navigation.state.params.id;
```

## Networking

fetch()

```
const response = await fetch(uri);
const json = await response.json();
```

- XMLHttpRequest
- WebSockets

### **Data**

- Component State
- Redux
- GraphQL
- Realm

## **Thinking in React**

- 1. mock-first
- 2. break into components
- 3. build static, stateless version
- 4. identify state data
- 5. identify state location
- 6. add inverse data flows

## Comparison

#### Params

- language
- ecosystem / community
- o convenience (toolchain, setup & build time, OS/hardware limitations, debugging/profiling)
- o upgradeability, maintainability

#### Competition

- Xamarin
- lonic
- Apache Cordova / PhoneGap

### **Pros**

- community
- battle-tested: Instagram, Facebook, Tesla, AirBnb, Skype
- iOS dev/debugging bez build/Xcode
- sophisticated gesture handling
- access to native capabilities
- constant updates

### Cons

- ES6+/npm/yarn/React/flexbox/fetch/Animatesjafldsjfsaljlskadsajlkfdsjfl!!!!!1
- constant updates
- npm | yarn
- <u>not</u> pixel perfect
- FIXED toolchain
- FIXED licensing

### **Toolchain**

- create-react-native-app & Expo
- babel, eslint, flow, watchman, jest, yarn...
- Windows, \*nix, macOS

# Thanks!

@tinc2k tinc2k@gmail.com