CS47: Cross-Platform Mobile Development

Lecture 5B: Introduction to Navigation

James Landay
Abdallah AbuHashem
Tiffany Manuel
Cisco Vlahakis
Vy Mai

https://cs47.stanford.edu


cs47-fall19.slack.com

Fall 2019

Administrivia

Assignment 4

Cancelled

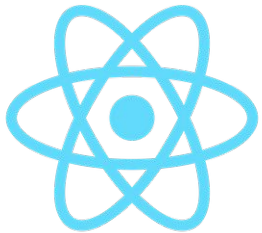
More than 2 absences?

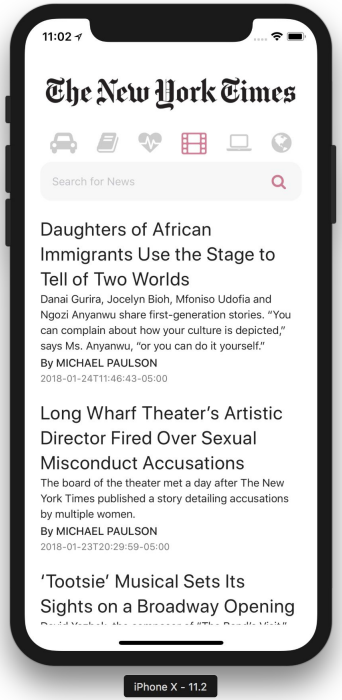
1- Talk to us asap

2- Do NOT miss any more classes with no excuse (we will fail you)

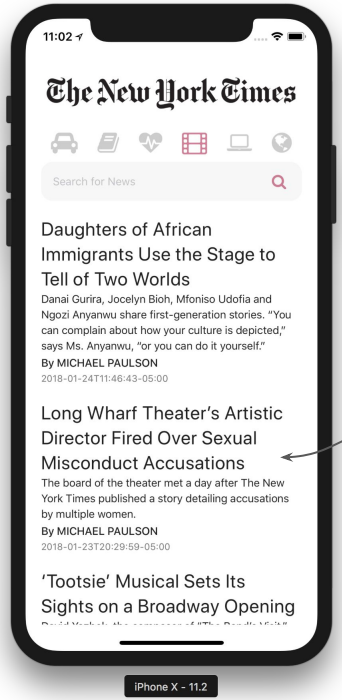
Overview for today

- Introduction to navigation
  - React Navigation
- How to do things with code?
  - Demo
    - Follow demo here: https://reflect.sh/jaded-design





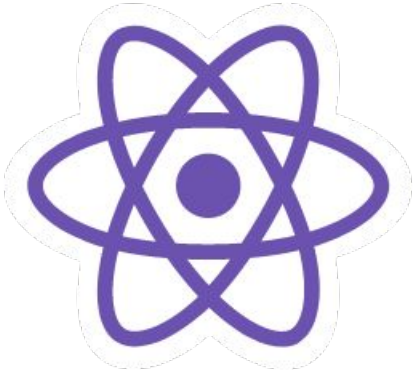
Single Screen App

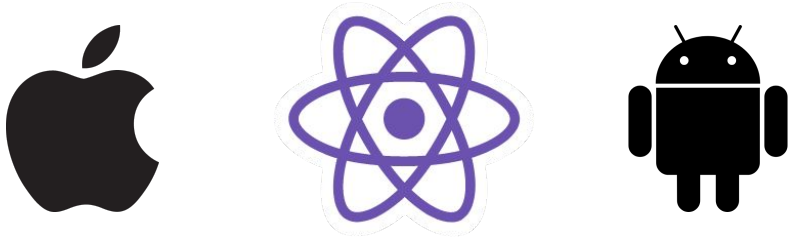


Single Screen App

What happens when you want to show a different screen?

Maybe a Settings screen, or an individual screen for one of those articles?





School of thought

React Native navigation should be based off “native” iOS and Android navigation components.

Usually a Java and Swift/Obj-C implementation

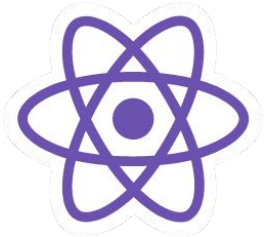
School of thought

React Native navigation should be based off “native” iOS and Android navigation components.

Usually a Java and Swift/Obj-C implementation

Native Navigation
http://airbnb.io/native-navigation/

React Native Navigation
https://github.com/wix/react-native-navigation

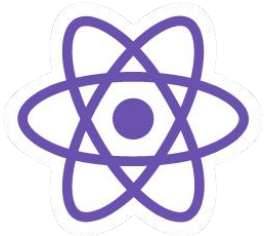


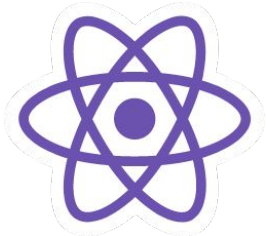
School of thought

React Native navigation should be based off as many existing components in the JavaScript layer.

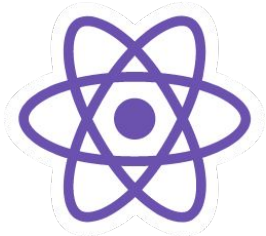
A JavaScript Implementation


Expo





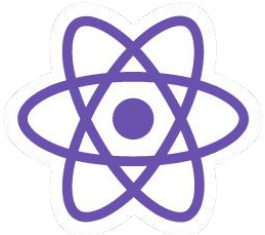
React Navigation
https://reactnavigation.org



React Navigation
https://reactnavigation.org

JavaScript implementation means:

- Greatest amount of customization
- Better fit for the RN ecosystem
- Room for growth independent of the navigation solutions that native platforms provide
- Integration with existing state-management systems like Redux

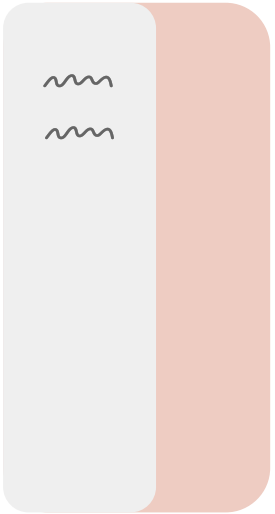


React Navigation

https://reactnavigation.org

Stack
Navigator

Tab
Navigator



Drawer
Navigator

Stack
Navigator

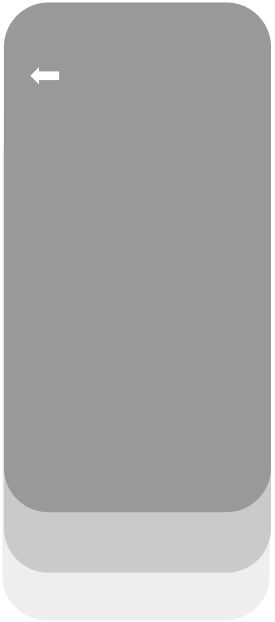
Tab
Navigator

Drawer
Navigator

Stack
Navigator

Tab
Navigator

Drawer
Navigator



Stack
Navigator

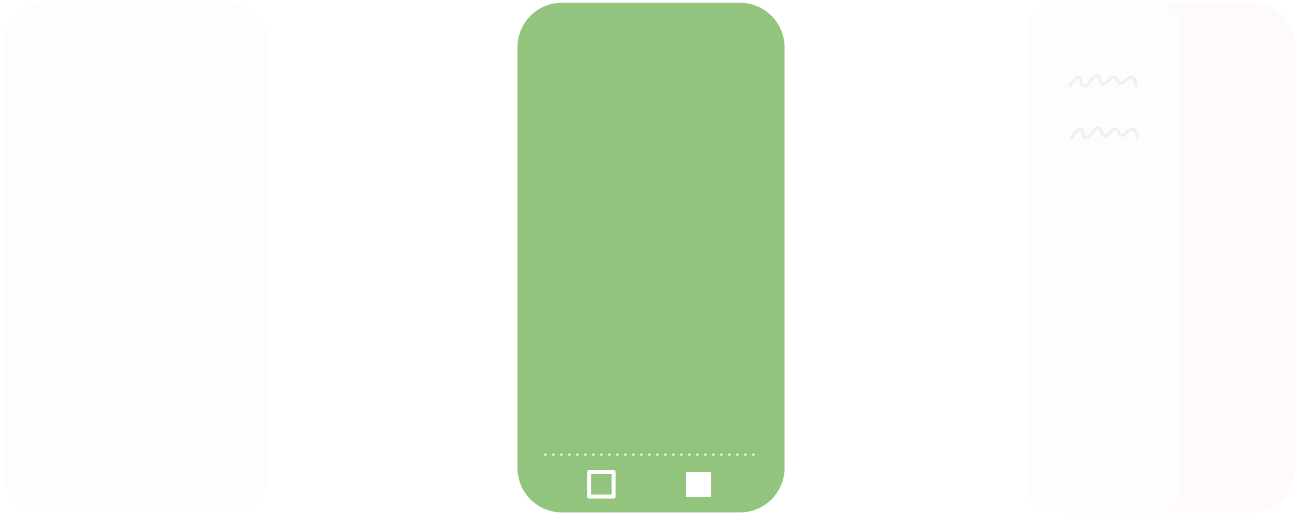
Tab
Navigator

Drawer
Navigator

Stack
Navigator

Tab
Navigator

Drawer
Navigator



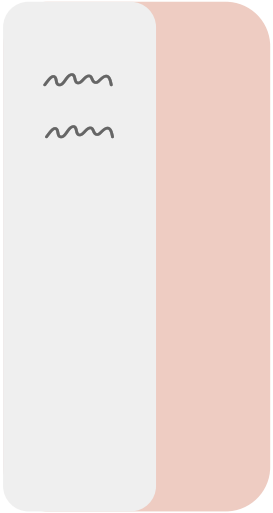
Stack
Navigator

Tab
Navigator

Drawer
Navigator

Stack
Navigator

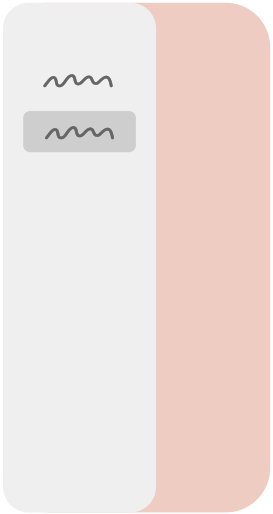
Tab
Navigator



Drawer
Navigator

Stack
Navigator

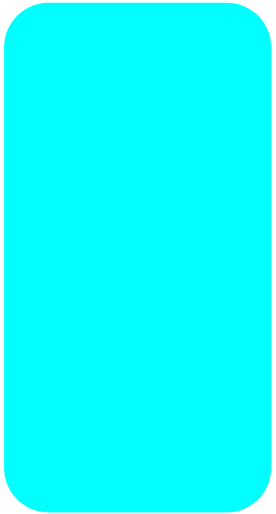
Tab
Navigator



Drawer
Navigator

Stack
Navigator

Tab
Navigator

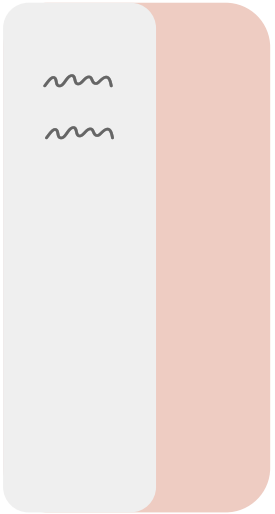


Drawer
Navigator

Live Demo

Stack
Navigator

Tab
Navigator



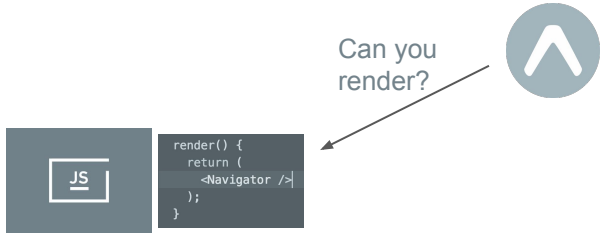
Drawer
Navigator

How does it work?

Can you
render?


```
render() {
  return (
    <Navigator />
  );
}
```

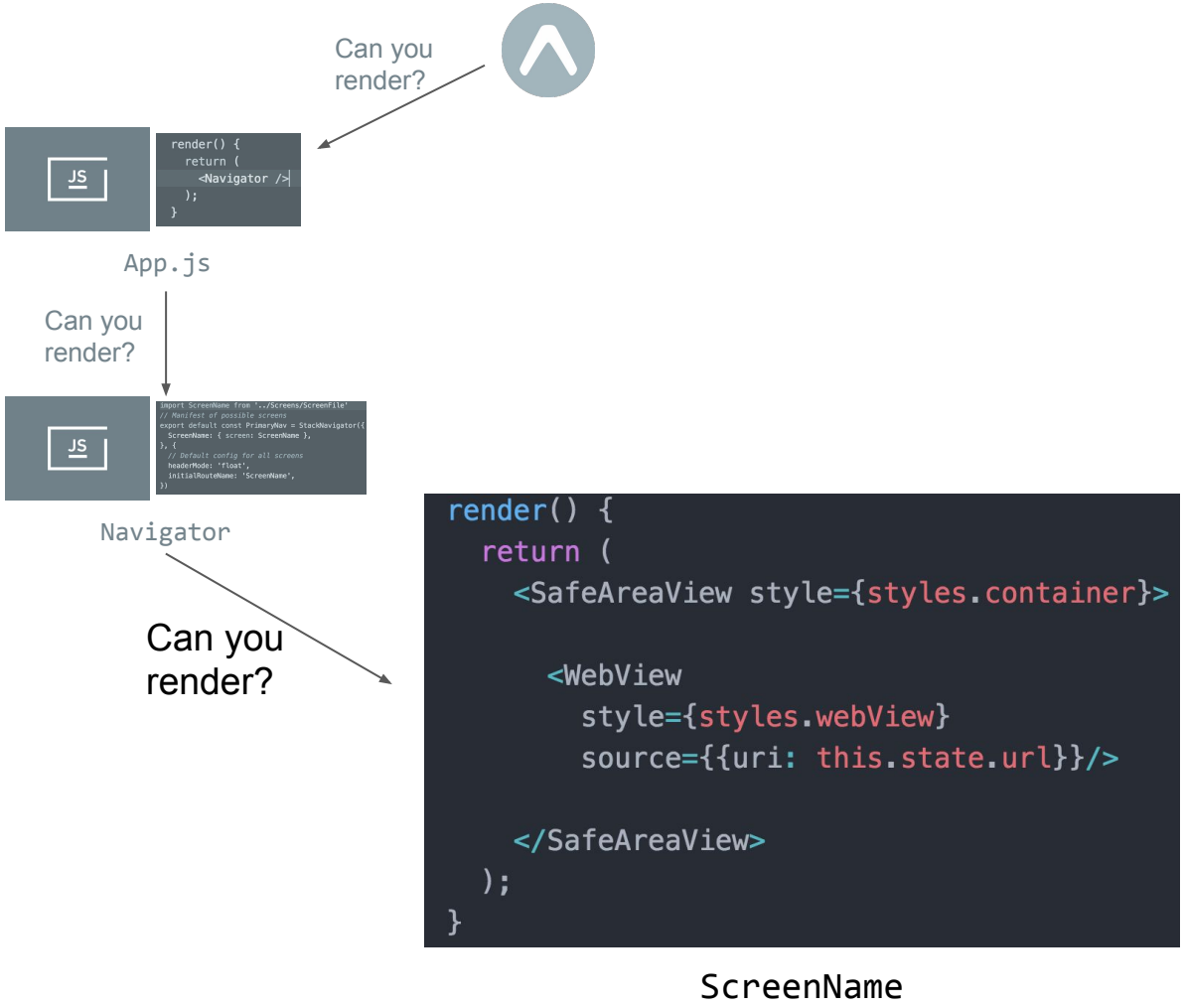
App.js



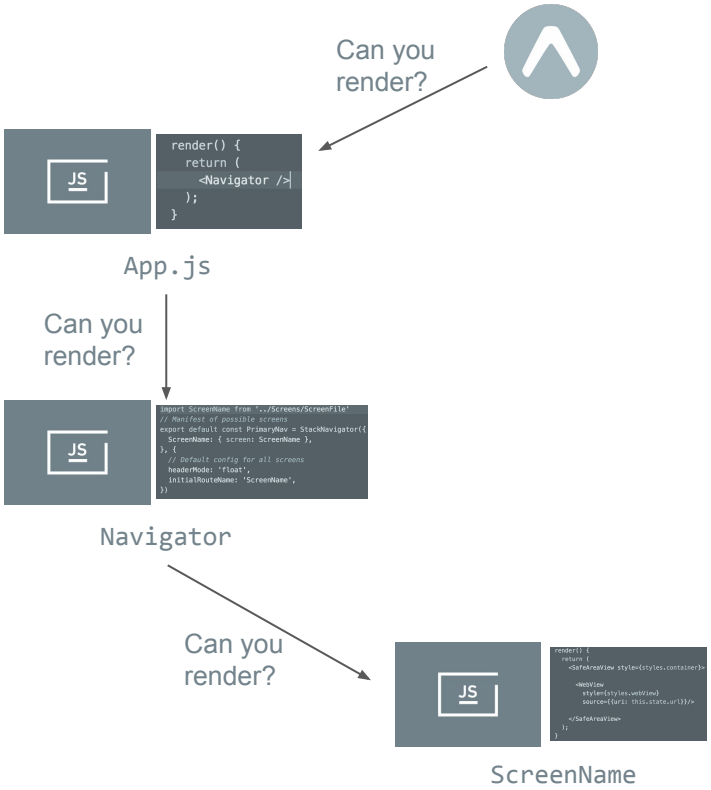
Can you
render?

```
import ScreenName from '../Screens/ScreenFile'
// Manifest of possible screens
export default const PrimaryNav = StackNavigator({
  ScreenName: { screen: ScreenName },
}, {
  // Default config for all screens
  headerMode: 'float',
  initialRouteName: 'ScreenName',
})
```

Navigator


Can you render?
JS
render() {
return (
<Navigator />
);
}
App.js
Can you render?
JS
Navigator
Can you render?
render() {
return (
<SafeAreaView style={styles.container}>
<WebView
style={styles.webView}
source={{uri: this.state.url}}/>
</SafeAreaView>
);
}
ScreenName

How does it work?



Really??

Really??

Kinda…
Sorta...

```
import ScreenName from '../Screens/ScreenFile'
// Manifest of possible screens
export default const PrimaryNav = StackNavigator({
  ScreenName: { screen: ScreenName },
}, {
  // Default config for all screens
  headerMode: 'float',
  initialRouteName: 'ScreenName',
})
```

Navigator

```
import ScreenName from '../Screens/ScreenFile'
// Manifest of possible screens
export default const PrimaryNav = StackNavigator({
  ScreenName: { screen: ScreenName },
}, {
  // Default config for all screens
  headerMode: 'float',
  initialRouteName: 'ScreenName',
})
```

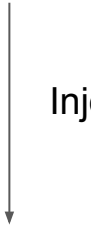
Navigator

Injects

```
<OtherComponents>
  <MainScreen navigation={...}>
<OtherComponents>
```

```
import ScreenName from '../Screens/ScreenFile'
// Manifest of possible screens
export default const PrimaryNav = StackNavigator({
  ScreenName: { screen: ScreenName },
}, {
  // Default config for all screens
  headerMode: 'float',
  initialRouteName: 'ScreenName',
})
```

Navigator



Injects

Tries to get

```
<OtherComponents>
  <MainScreen navigation={...}>
<OtherComponents>
```

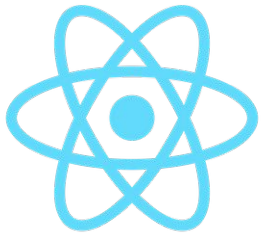
```
static navigationOptions = ({ navigation }) => {
  var {params} = navigation.state;
   //Some code to get values
   var options = {
     //key : //Value
   }
   return options;
};
```

Really??

Really??

Overview for today

- Introduction to navigation
  - React Navigation
- How to do things with code?
  - Demo
    - Follow demo here: https://reflect.sh/jaded-design



Deeper Dive

More on Stack Navigators

this.props.navigation.

- navigate('RouteName')
    - If screen is in stack, navigates to screen; else pushes to stack
- push('RouteName')
- .goBack()
- .popToTop()

More on Stack Navigators

this.props.navigation.

- navigate('RouteName')
- push('RouteName')
    - Pushes route to stack irregardless if it is already in the stack
- .goBack()
- .popToTop()

More on Stack Navigators

this.props.navigation.

- navigate('RouteName')
- push('RouteName')
- .goBack()
    - A way to go back from within the class; a button for example
- .popToTop()

More on Stack Navigators

this.props.navigation.

- navigate('RouteName')
- push('RouteName')
- .goBack()
- .popToTop()
    - Goes back to the first screen in the stack

Demo

Params

- Very Similar to Props
  - Passed in With Navigation
    - this.props.navigation.navigate('RouteName', { /* params go here */ })
    - this.props.navigation.navigate('Opinions', {myParam: ‘Dogs are Overrated’});
  - Getting Params
    - const paramBam = this.props.navigation.getParam(‘myParam’ , ‘default’)
    - this.props.navigation.state.params
      - Null if no params supplied

Headers

- static navigationOptions

```
class HomeScreen extends React.Component {
  static navigationOptions = {
    title: 'Home',
    headerStyle: {
      backgroundColor: '#f4511e',
    },
    headerTintColor: '#fff',
    headerTitleStyle: {
      fontWeight: 'bold',
    },
  };

  /* render function, etc */
}
```

Headers

- static navigationOptions
  - Configuration object (static property of component)

```
class DetailsScreen extends React.Component {
  static navigationOptions = ({ navigation }) => {
    return {
      title: navigation.getParam('otherParam', 'A Nested Details Screen'),
    };
  };

  /* render function, etc */
}
```

Header Buttons

- NavigationOptions is a static property of the component

Header Buttons

- NavigationOptions is a static property of the component
  - You can’t just call a function from the header

Header Buttons

- NavigationOptions is a static property of the component
  - You can’t just call a function from the header

```
class HomeScreen extends React.Component {
  static navigationOptions = {
    headerTitle: <LogoTitle />,
    headerRight: (
      <Button
        onPress={() => alert('This is a button!')}
        title="Info"
        color="#fff"
      />
    ),
  };
}
```

Header Buttons

- NavigationOptions is a static property of the component
  - You can’t just call a function from the header

```
class HomeScreen extends React.Component {
  static navigationOptions = {
    headerTitle: <LogoTitle />,
    headerRight: (
      <Button
        onPress={() => alert('This is a button!')}
        title="Info"
        color="#fff"
      />
    ),
  };
}
```

← Will not work

Header Buttons

- Instead you use params, and treat them like state
  - You set/update a param in the class, and access that param from the navigation options to call the function

Header Buttons

```
class HomeScreen extends React.Component {
  static navigationOptions = ({ navigation }) => {
    return {
      headerTitle: () => <LogoTitle />,
      headerRight: () => (
        <Button
          onPress={navigation.getParam('increaseCount')}
          title="+1"
          color="#fff"
        />
      ),
    };
  };

  componentDidMount() {
    this.props.navigation.setParams({ increaseCount: this._increaseCount });
  }

  state = {
    count: 0,
  };

  _increaseCount = () => {
    this.setState({ count: this.state.count + 1 });
  };

  /* later in the render function we display the count */
}
```

Demo

CS47: Cross-Platform Mobile Development

Lecture 5B: Introduction to Navigation

James Landay
Abdallah AbuHashem
Tiffany Manuel
Cisco Vlahakis
Vy Mai

https://cs47.stanford.edu


cs47-fall19.slack.com

Fall 2019