Lecture 6: Navigation

Brent Vatne & Eric Vicenti

Previous Lecture

- User input with `TextInput`
- Simple input validation
- `KeyboardAvoidingView`
- Debugging
    - Errors and warnings
    - Chrome Developer Tools
    - React Native Inspector with react-devtools
- Installing external libraries with `npm`

What is navigation?

- Navigation is a broad term that covers topics related to how you move between screens in your app
- Web navigation is oriented around URLs
- Mobile apps do not use URLs for navigating *within* the app*
- Navigation APIs completely different on iOS and Android
  - Several React Native libraries provide a platform agnostic alternative
  - We will talk about one of them today, React Navigation

* Linking into a mobile app with a URL is known as deep linking: https://v2.reactnavigation.org/docs/deep-linking.html

React Navigation and alternatives

- Two distinct approaches
  1. Implement mainly in JavaScript with React
  2. Implement mostly in native, expose an interface to JavaScript for existing native navigation APIs
- React Navigation takes approach #1

Read more at https://v2.reactnavigation.org/docs/pitch.html and https://v2.reactnavigation.org/docs/alternatives.html

Install React Navigation

- `npm install react-navigation@2.0.0-beta.5 --save`
- This will install the latest *pre-release* version at the time of writing. Typically you would just write `npm install react-navigation --save` to use the latest *stable* version.
- If you refer back to this in the future, keep in mind that this material is all specific to the 2.x series of releases.

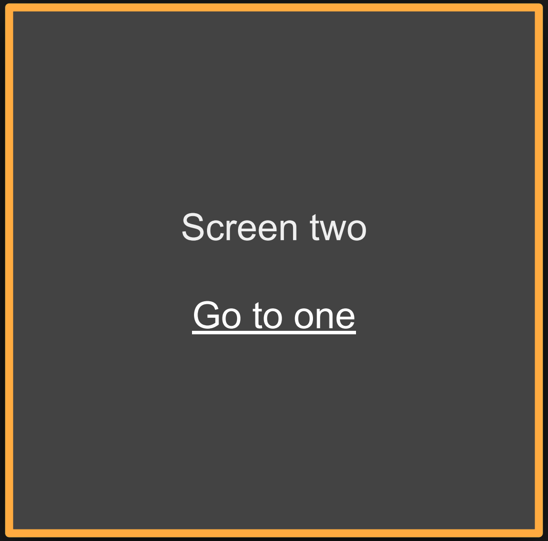
Navigators, routes, and screen components

- A navigator is a component that implements a navigation pattern (eg: tabs)
- Each navigator must have one or more routes.
    - A navigator is a parent of a route.
    - A route is a child of a navigator.
- Each route must have a name and a screen component.
    - The name is usually unique across the app
    - The screen component is a React component that is rendered when the route is active.
    - The screen component can also be another navigator.

Switch Navigator

- Display one screen at a time
- Inactive screens are unmounted
- The only action a user can take to switch from one route to another

Switch Navigator



Screen two

Go to one

Creating a navigator

```
import { createSwitchNavigator } from 'react-navigation';

const AppNavigator = createSwitchNavigator({
  "RouteNameOne": ScreenComponentOne,
  "RouteNameTwo": ScreenComponentTwo,
});
```

Rendering a navigator

```
const AppNavigator = createSwitchNavigator({
  "RouteNameOne": ScreenComponentOne,
  "RouteNameTwo": ScreenComponentTwo,
});

export default class App extends React.Component {
  render() {
    return <AppNavigator />
  }
}
```

- `createSwitchNavigator` is a function that returns a React component
- We render the component in our root App component. Usually we only *explicitly* render one navigator per app because navigators are composable.

Higher order components

- `createSwitchNavigator` is a Higher Order Component: it is a function that returns a React component.
- "A higher-order component (HOC) is an advanced technique in React ***for reusing component logic***."
- This is similar to higher order functions, which are functions that either take functions as arguments or return a function as a result.

Read more at https://reactjs.org/docs/higher-order-components.html

Navigating to another route

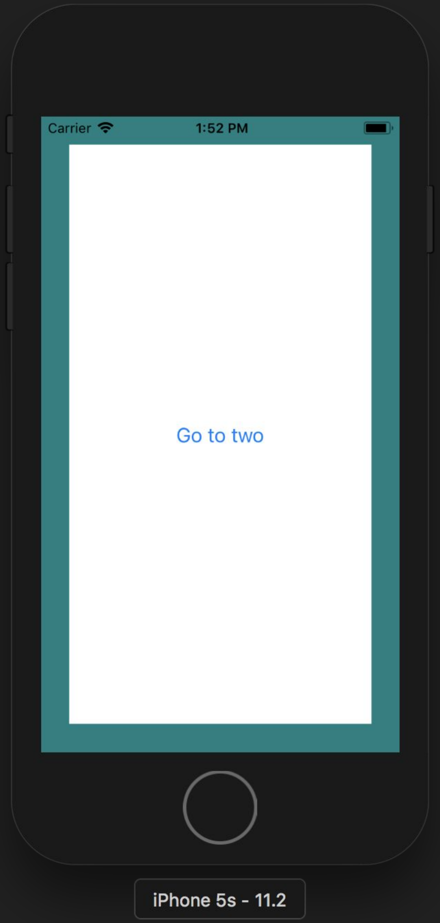
```
class ScreenComponentOne extends React.Component {
 render() {
   return (
     <Button
       title="Go to two"
       onPress={() => this.props.navigation.navigate('RouteNameTwo')}
     />
   );
 }
}
```

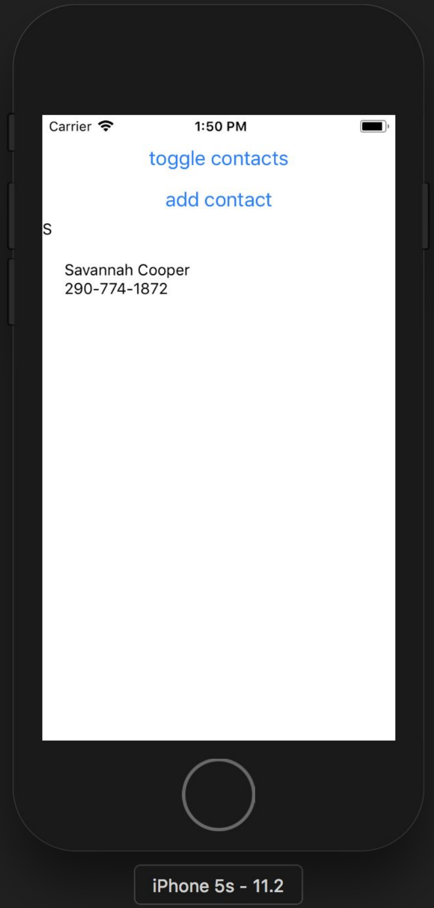
The navigation prop

- **`navigate(..)`**
- `goBack(..)`
- `setParams(..)`
- `getParam(..)`
- `dispatch(..)`
- `isFocused(..)`
- `addListener(..)`
- `state`

* The navigation prop is passed in to the screen component for each route.

Full reference: https://v2.reactnavigation.org/docs/navigation-prop.html


Carrier
1:52 PM
Go to two
iPhone 5s - 11.2



S

Savannah Cooper
290-774-1872

screenProps

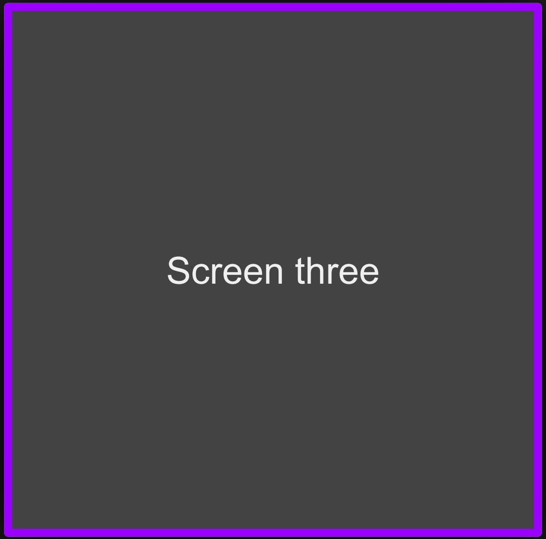
```
export default class App extends React.Component {
  render() {
    return <AppNavigator screenProps={/* object here */} />
  }
}
```

- Made available to every screen component in the navigator.
- Perfectly fine for very small applications and prototyping but inefficient for most meaningful applications - **every route in your app will re-render when `screenProps` changes**. Use a state management library or the React Context API instead.

Stack Navigator

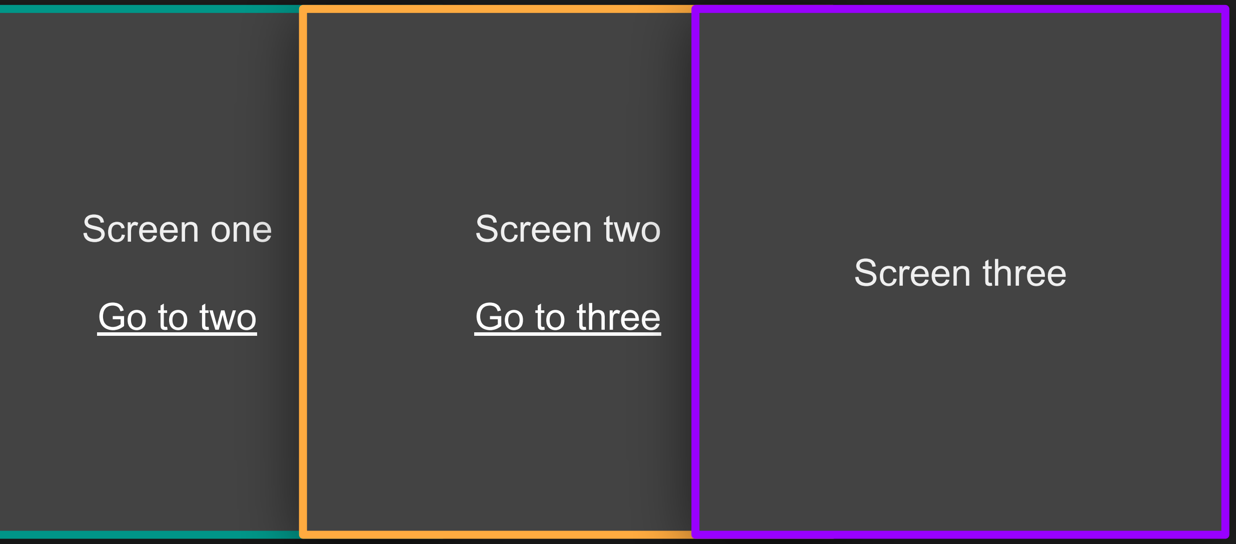
- Display one screen at a time
- The state of inactive screens is **maintained** and they remain mounted
- **Platform-specific layout, animations, and gestures**
  - Screens are stacked on top of each other
  - iOS: screens slide in from right to left, can be dismissed with left to right gesture. Modal screens slide in from bottom to top, can be dismissed with top to bottom gesture.
  - Android: screens fade in on top of each other, no dismiss gesture. Hardware back button dismisses the active screen.
- Users can push and pop items from the stack, replace the current item, and various other

Stack Navigator



Screen three

Stack Navigator



Screen one

Go to two

Screen two

Go to three

Screen three

Creating a StackNavigator

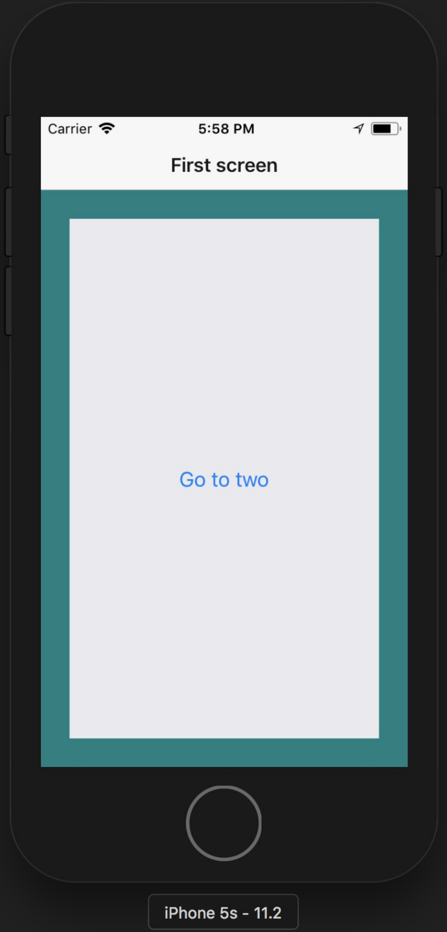
```
import { createStackNavigator } from 'react-navigation';

const AppNavigator = createStackNavigator({
 "RouteNameOne": ScreenComponentOne,
 "RouteNameTwo": ScreenComponentTwo,
});
```

Navigating to another route

```
class ScreenComponentOne extends React.Component {
 render() {
   return (
     <Button
       title="Go to two"
       onPress={() => this.props.navigation.navigate('RouteNameTwo')}
     />
   );
 }
}
```

Returning to the previously active route

```
class ScreenComponentThree extends React.Component {
 render() {
   return (
     <Button
       title="Go back"
       onPress={() => this.props.navigation.goBack()}
     />
   );
 }
}
```


Carrier
5:58 PM
First screen
Go to two
iPhone 5s - 11.2

Configuring `navigationOptions`

- `headerTitle`
- `headerStyle`
- `headerTintColor`
- `headerLeft`
- `headerRight`

Full list: https://v2.reactnavigation.org/docs/stack-navigator.html#navigationoptions...

Using params to pass state between routes

- `navigate` with params

```
this.props.navigation.navigate('RouteName', {
  paramName: 'value-of-param'
});
```

- `setParams` to update params for the route

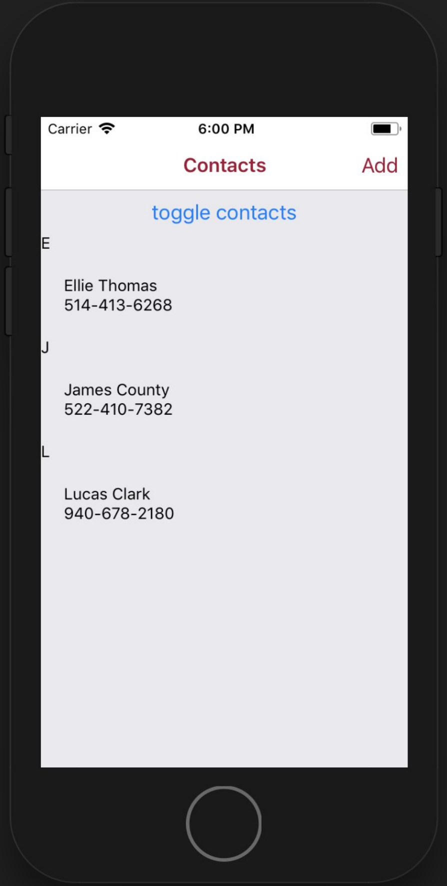
```
this.props.navigation.setParams({
 paramName: 'new-value-of-param',
});
```

- `getParam` to read a param

```
this.props.navigation.getParam('paramName', 'default-value');
```

```
// It's time for us to take a short break
this.props.navigation.navigate('BreakTime');
```

```
// Break time is over
this.props.navigation.goBack();
```



E

Ellie Thomas
514-413-6268

J

James County
522-410-7382

L

Lucas Clark
940-678-2180

iPhone 5s - 11.2

Add button to header with `navigationOptions`

- `headerLeft`
- `headerRight`

Full list: https://v2.reactnavigation.org/docs/stack-navigator.html#navigationoptions...

```
// Jump to a screen, identified by route name
navigate('MyRouteName', { paramName: 'param-value' });

// “Push” a new screen, even if it already is in the stack
push('MyRouteName');
```

Stack specific navigation actions

- `push(..)`
- `pop(..)`
- `popToTop(..)`
- `replace(..)`

More information: https://v2.reactnavigation.org/docs/navigation-prop.html#...

Composing navigators

- Navigators can be composed when one type of navigation visually appears to be inside another navigator
- A navigator can be the Screen Component of another navigator
- The app should only contain one top-level navigator
- You can `navigate()` to any route in the app
- `goBack()` works for the whole app, supports Android back button

Composing navigators

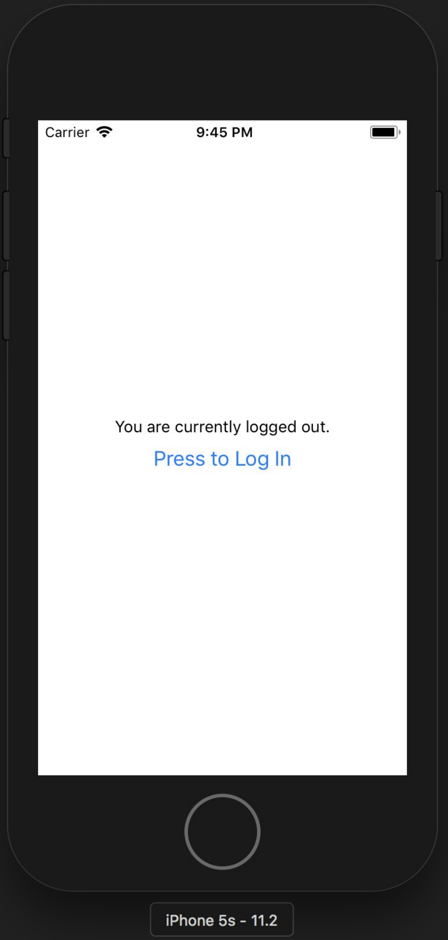
```
const MyStackNavigator = createStackNavigator({
  "Home": HomeScreen,
  "AddContact": AddContactScreen,
});
const AppNavigator = createSwitchNavigator({
  "Login": LoginScreen,
  "Main": MyStackNavigator,
});
```

Do not render a navigator inside a screen

```
class MyScreen extends React.Component {
  render() {
    return <MyStackNavigator />;
  }
}
```

Instead, set as a screen within the `AppNavigator`

```
const AppNavigator = createSwitchNavigator({
  "Main": MyStackNavigator,
});
```

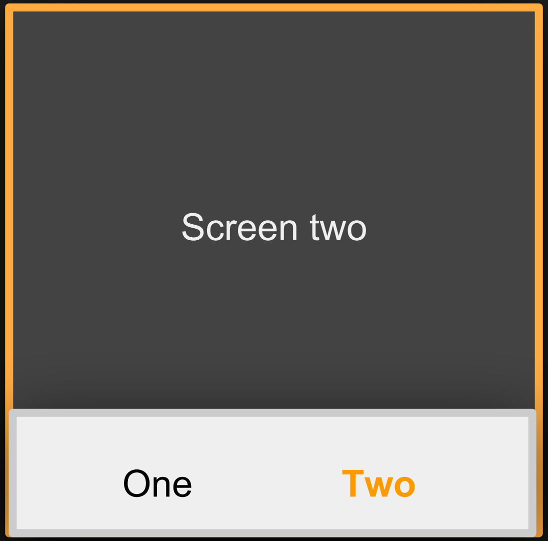


You are currently logged out.

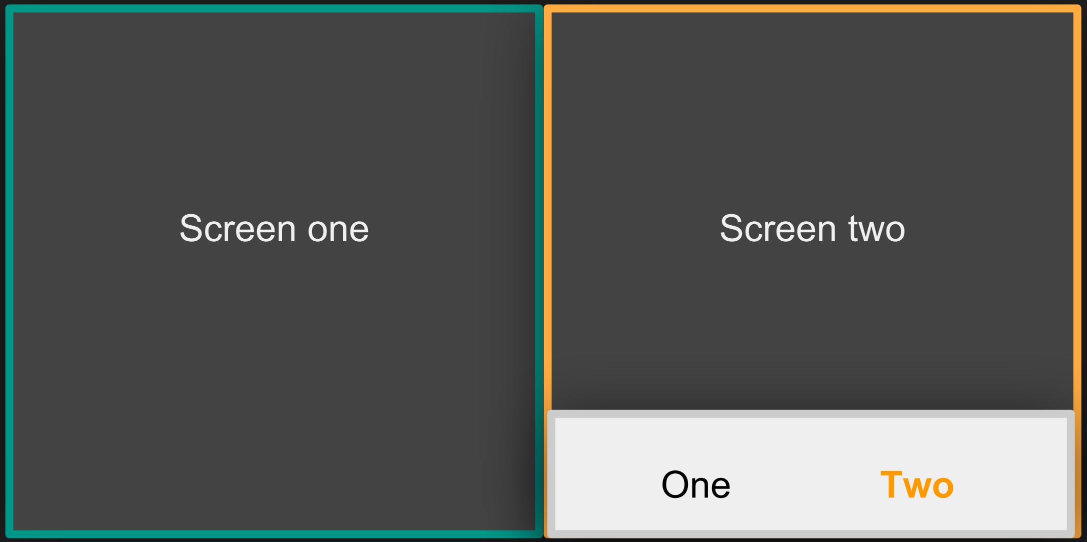
Tab navigators

- Display one screen at a time
- The state of inactive screens is maintained
- Platform-specific layout, animations, and gestures
    - `createMaterialTopTabNavigator`
    - `createMaterialBottomTabNavigator`
    - `createBottomTabNavigator`
- The `navigate()` action is used to switch to different tabs
- `goBack()` can be called to go back to the first tab
    - The tab navigator `goBack` behavior is configurable

Tab navigators

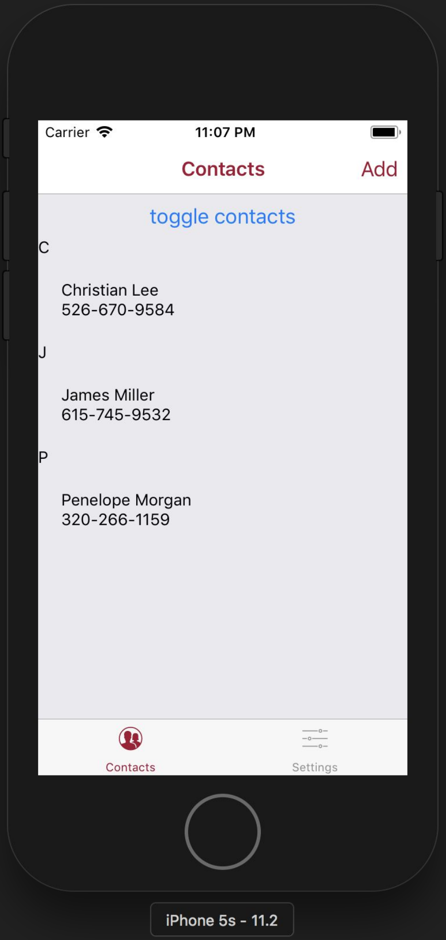


Tab navigators



Creating a tab navigator

```
const AppNavigator = createBottomTabNavigator({
  "TabOne": ScreenComponentOne,
  "TabTwo": ScreenComponentTwo,
});

export default class App extends React.Component {
  render() {
    return <AppNavigator />
  }
}
```



C

Christian Lee
526-670-9584

J

James Miller
615-745-9532

P

Penelope Morgan
320-266-1159

Configure tab bar settings

```
const MainTabs = createBottomTabNavigator(
 {
   ...
 },
 {
   tabBarOptions: {
     activeTintColor: "#a41034"
   }
 }
);
```

Full reference for `tabBarOptions`: https://v2.reactnavigation.org/docs/tab-navigator.html#...

Configure tab icons

```
MainStack.navigationOptions = {
 tabBarIcon: ({ focused, tintColor }) => (
   <Ionicons
     name={`ios-contacts${focused ? "" : "-outline"}`}
     size={25}
     color={tintColor}
   />
 )
};
```

Full reference of options: https://v2.reactnavigation.org/docs/tab-navigator.html#...

Use common icon packs

```
# Install it in your shell
npm install --save react-native-vector-icons
```

```
// Import a supported icon set in your code
import Ionicons from "react-native-vector-icons/Ionicons";
```

```
// Use it as a React component
<Ionicons name="md-checkmark" size={25} color="#000" />
```

See other icon sets that are included: https://expo.github.io/vector-icons/

React Navigation Resources

- React Navigation Documentation
- React Navigation API Reference
- NavigationPlayground example source code