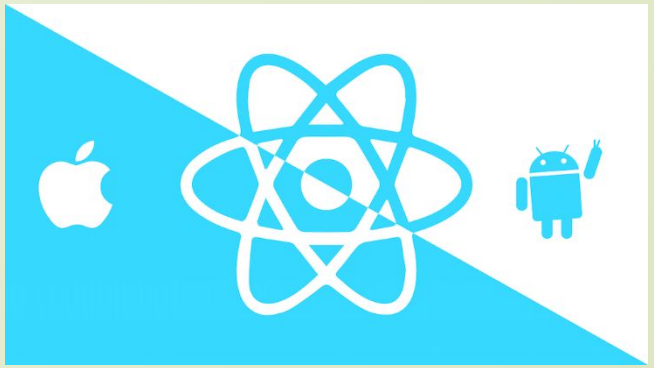
React Native Navigation between screens

- DEV {Education}
- Преподаватель –Эльмар Гусейнов

React Native

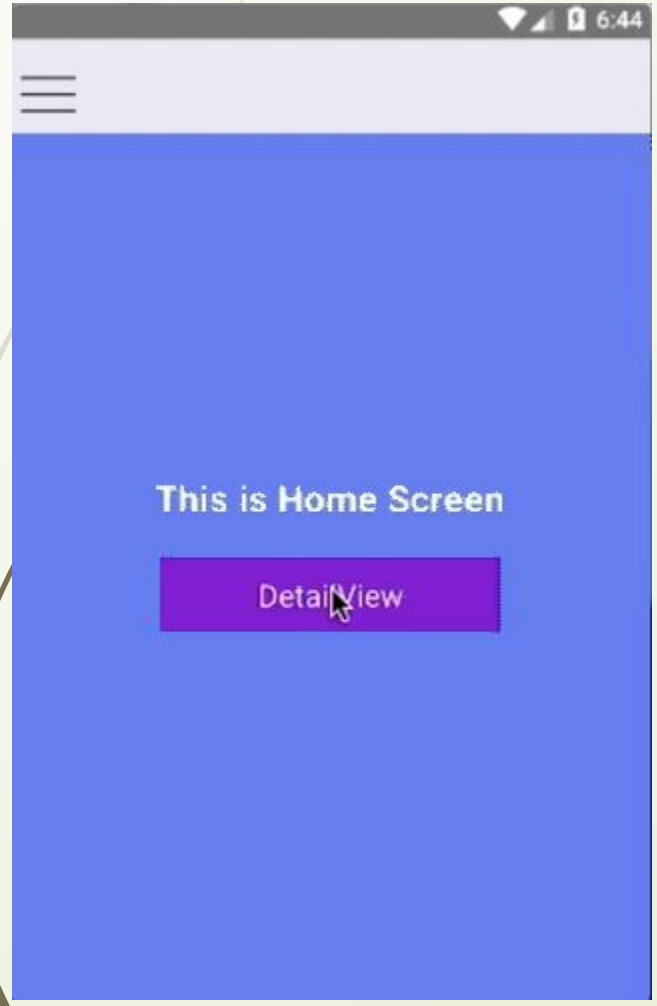
- **3 types of navigation menu**

- Stack navigator
- Tab navigator
- Drawer navigator

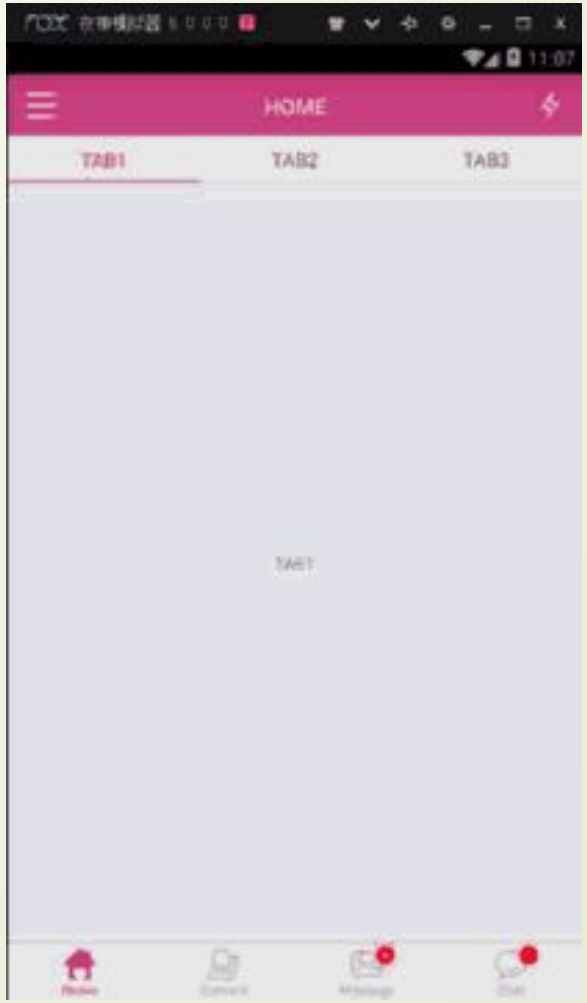


React Native

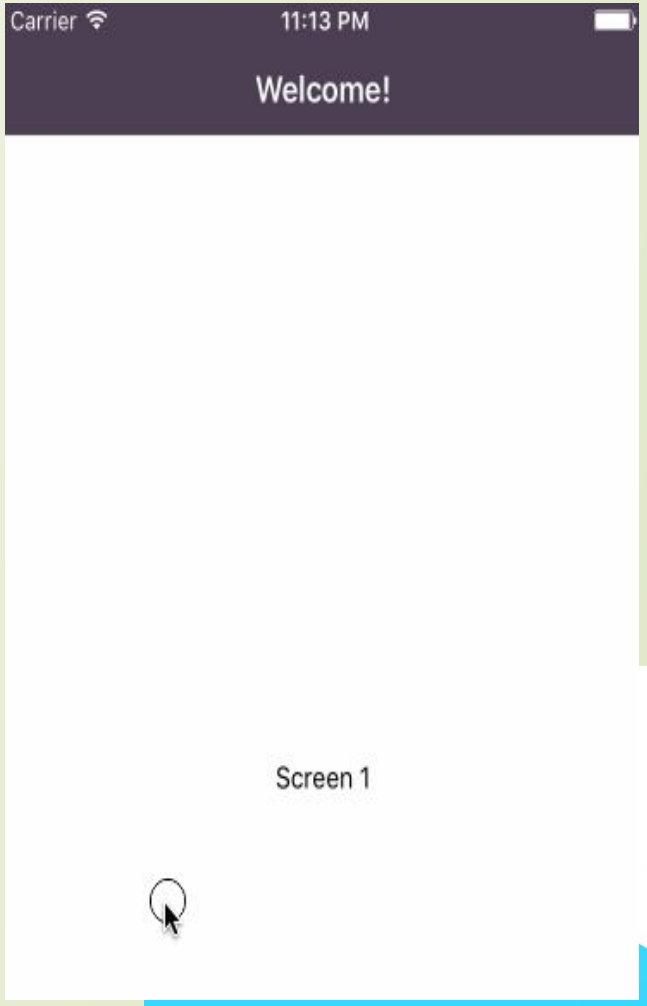
- Stack navigator



- Tab navigator



- Drawer navigator

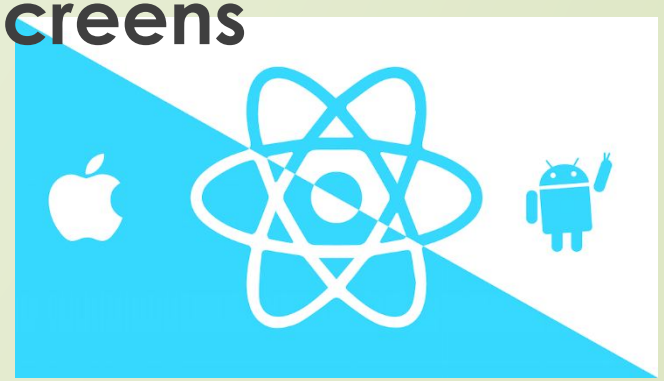


React Native

First Install to new or exist project

- 1) install react-navigation: **npm install --save @react-navigation/native**
- 2) install dependencies:

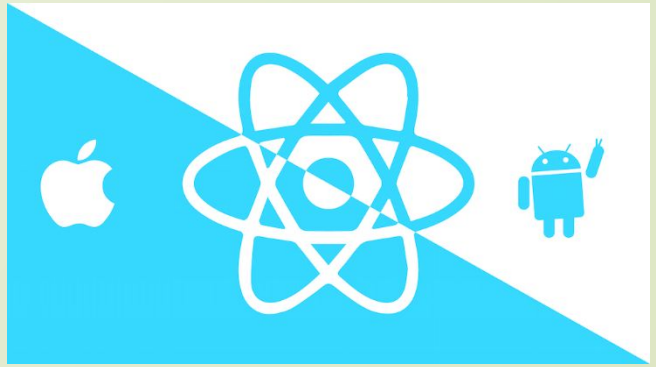
npm install --save react-native-reanimated react-native-gesture-handler react-native-screens react-native-safe-area-context @react-native-community/masked-view



React Native

- For Stack Navigator

- npm install -- save @react-navigation/stack

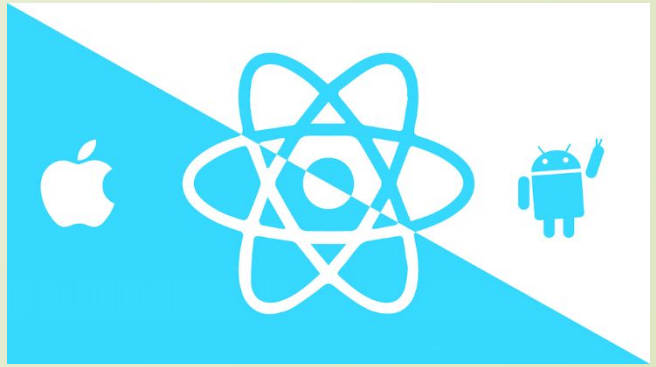


React Native

Stack Navigator

- 1) import { NavigationContainer } from '@react-navigation/native';
- 2) import { createStackNavigator } from '@react-navigation/stack';
- 3) include NavigationContainer into App.js or index.js:

```
import { NavigationContainer } from '@react-navigation/native';
export default function App() {
return (
<NavigationContainer>{/* Rest of your app code */}</NavigationContainer>
);
}
```

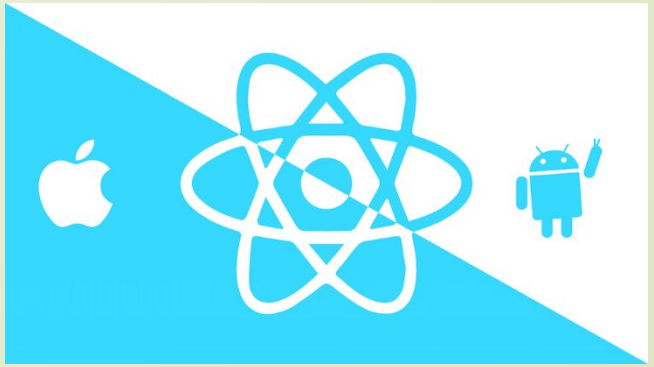


React Native

Example Stack Navigator (MyStack.js file)

```
.....
const MyRootStackName = createStackNavigator ();
function MyStack() {
return (
<MyRootStackName.Navigator>
<MyRootStackName.Screen name="Home" component={Home} />
<MyRootStackName.Screen name="Notifications"
component={Notifications} />
</Stack.Navigator>);}
```

WHERE Name – ROUTE name!!!

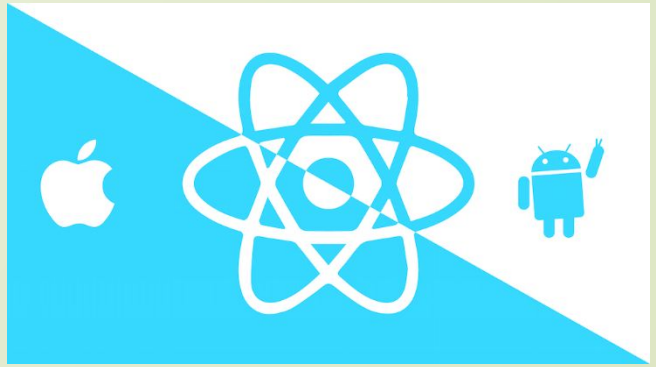


React Native

Add to App.js (MyStack.js)

```
import React, {Component} from 'react';

import {NavigationContainer} from '@react-navigation/native';
import {MyStack} from './navigation/myStack';
export default class App extends Component {
  render() {
    return (
      <NavigationContainer>
        <MyStack />
      </NavigationContainer>
    );
  }
}
```

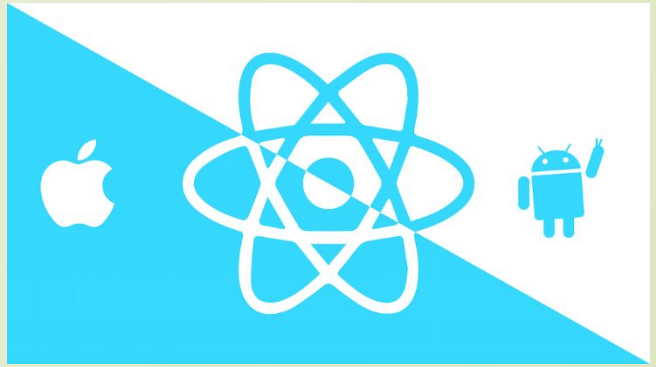


React Native

Navigate between screens

Command: navigation.navigate (‘Routename’) (if you are make more tames to same screen –nothing happend)

```
import * as React from 'react';
import { Button, View, Text } from 'react-native';
import { NavigationContainer } from '@react-navigation/native';
import { createStackNavigator } from '@react-navigation/stack';
function HomeScreen({ navigation }) {
return (
<View style={{ flex: 1, alignItems: 'center', justifyContent: 'center' }}>
<Text>Home Screen</Text>
<Button
title="Go to Details"
onPress={() => navigation.navigate('Details')}
/>
</View>
);
}
```



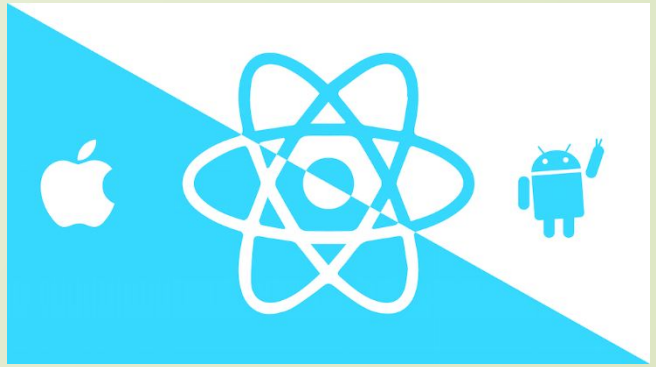
React Native

Stack Navigator

- **Navigate to same screen multiple times**

- **We need to change navigation.navigate to push**

```
<Button title="Go to Details... again"
onPress={() => navigation.push('Details')} />
```



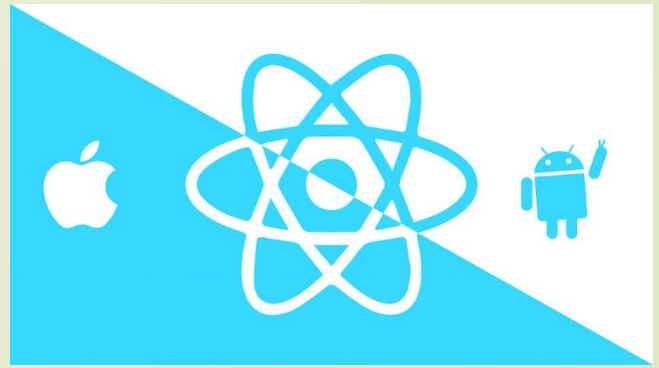
React Native

Stack Navigator

Passing parameters to routes

- **1.send parameters:** navigate('RouteName', { /* params go here */ })
- **2. get parameters:** route.params – it is object, need to be extract
- **Example 1:** onPress={() => {navigation.navigate('Details', {itemId: 86, otherParam: 'anything you want here',});}}
- **Example 2:** function DetailsScreen({ route, navigation }) {/* 2. *Get the param* */

```
const { itemId } = route.params;
const { otherParam } = route.params;
return (…….your View and etc…..
<Text>itemId: {JSON.stringify(itemId)}</Text>
<Text>otherParam: {JSON.stringify(otherParam)}</Text>
……..……..
```



Example

function HomeScreen({ navigation }}

React Native

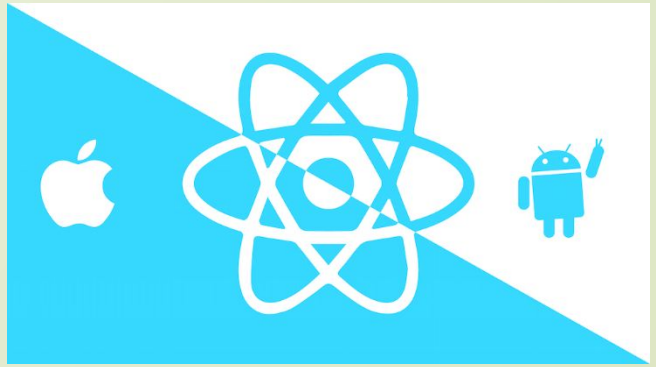
Stack Navigator

Setting the header title

```
function StackScreen() {
return (
<Stack.Navigator>
<Stack.Screen
name="Home"
component={HomeScreen}
options={{ title: 'My home' }}
/>
</Stack.Navigator>
);
}
```

Using parameters in title

```
function StackScreen() {
return (
<Stack.Navigator>
<Stack.Screen
```



React Native

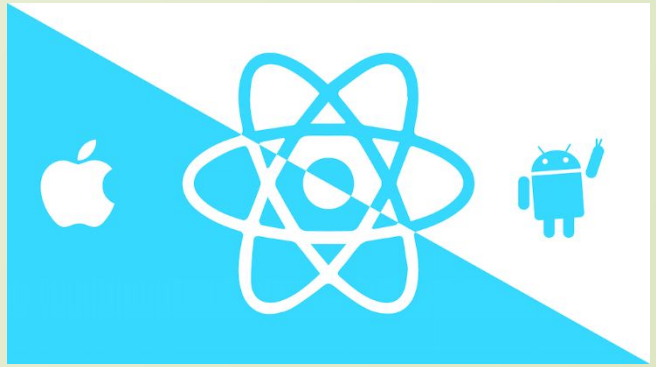
Stack Navigator

Updating header by setOptions

Updating options with setOptions

It's often necessary to update the options configuration for the active screen from the mounted screen component itself. We can do this using navigation.setOptions

```
/* Inside of render() of React class */
<Button
  title="Update the title"
  onPress={() => navigation.setOptions({ title: 'Updated!' })}
/>
```



React Native

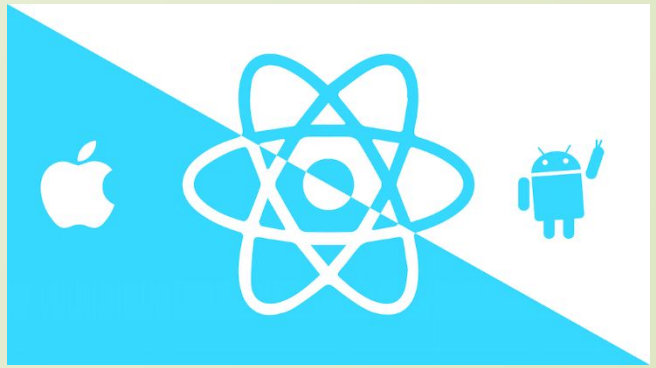
Stack Navigator

Styles for Stack navigator

See documentation.

Example

```
function StackScreen() {
return (
<Stack.Navigator>
<Stack.Screen
name="Home"
component={HomeScreen}
options={{
title: 'My home',
headerStyle: {
backgroundColor: '#f4511e',
```

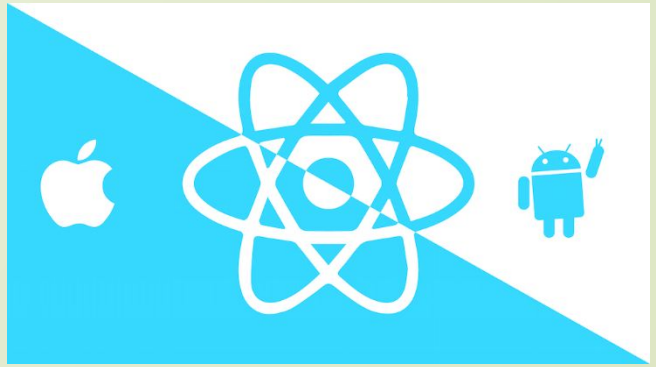


React Native

Stack Navigator

Sharing common options across screens

```
function StackScreen() {
  return (
    <Stack.Navigator
      screenOptions={{
        headerStyle: {
          backgroundColor: '#f4511e',
        },
        headerTintColor: '#fff',
        headerTitleStyle: {
          fontWeight: 'bold',
        },
      }}
    >
```

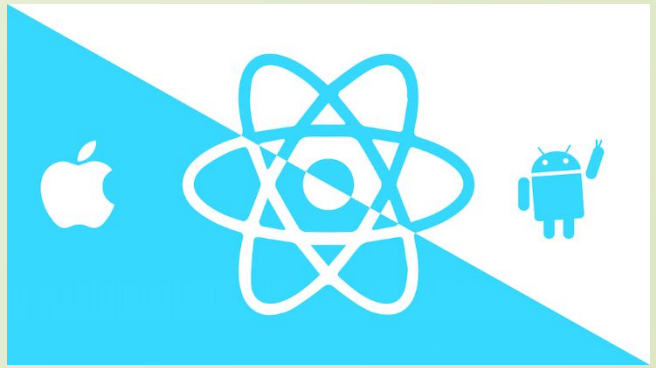


React Native

Stack Navigator

- Header button

```
<Stack.Screen
        name="Home"
        component={HomeScreen}
options={{
         headerTitle: props => <LogoTitle {...props} />,
         headerRight: () => (
           <Button
             onPress={() => alert('This is a button!')}
             title="Info"
             color="#fff"
           />
         ),
        }}
```

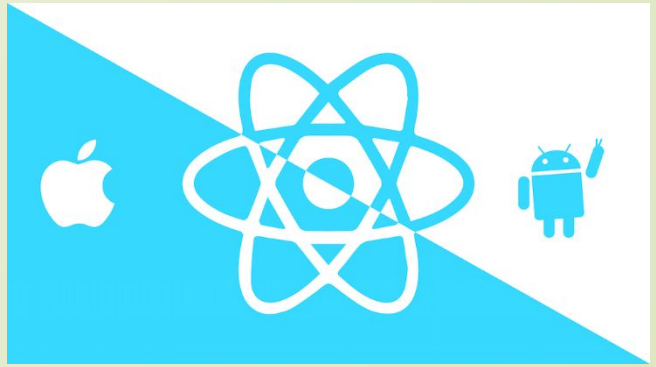


React Native

Stack Navigator

Adding a button to the header

```
function StackScreen() {
  return (
    <Stack.Navigator>
      <Stack.Screen
        name="Home"
        component={HomeScreen}
        options={{
          headerTitle: props => <LogoTitle {...props} />,
          headerRight: () => (
            <Button
              onPress={() => alert('This is a button!')}
              title="Info"
              color="#fff"
```

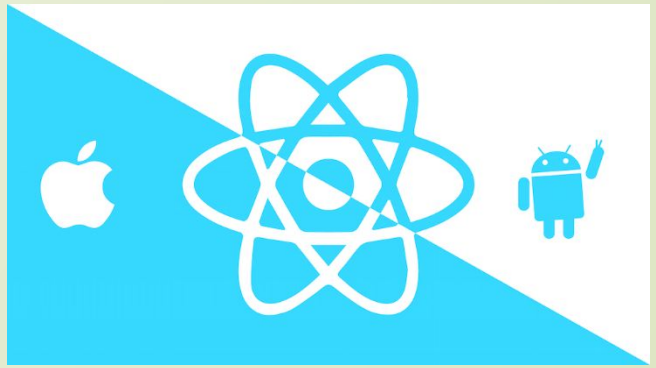


React Native

Stack Navigator

- **Other props**

https://reactnavigation.org/docs/stack-navigator#props

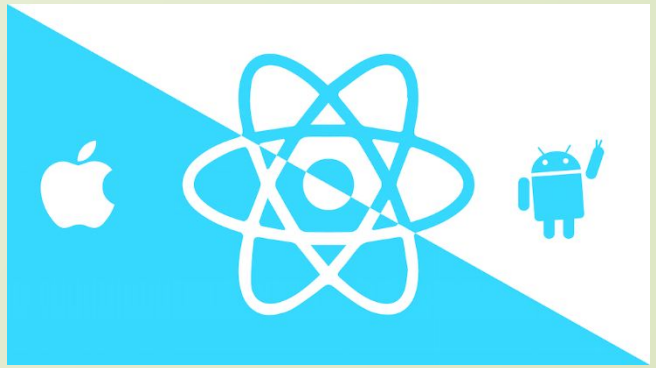


React Native
Tab navigator

Installation Tab navigator

3 types of Tab navigator:

- **BottomTabNavigator**
- **MaterialBottomTabNavigator**
- **MaterialTopTabNavigator**



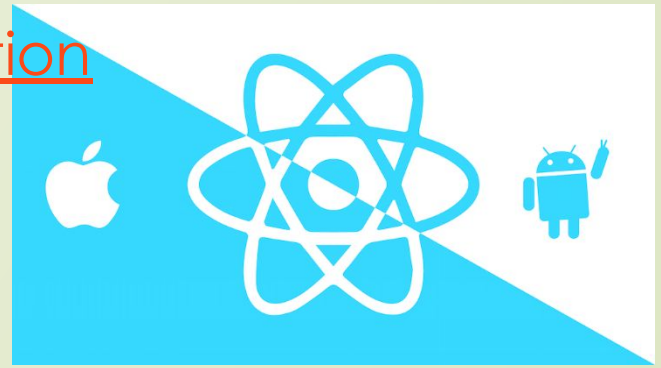
React Native
Tab navigator

Installation Tab navigator

- 1) install react-navigation: **npm install –save @react-navigation/native**
- 2) install dependencies:

npm install –save react-native-reanimated react-native-gesture-handler react-native-screens react-native-safe-area-context @react-native-community/masked-view

- 3) install tab nav: **npm install - - save @react-navigation/bottom-tabs**
- https://reactnavigation.org/docs/tab-based-navigation

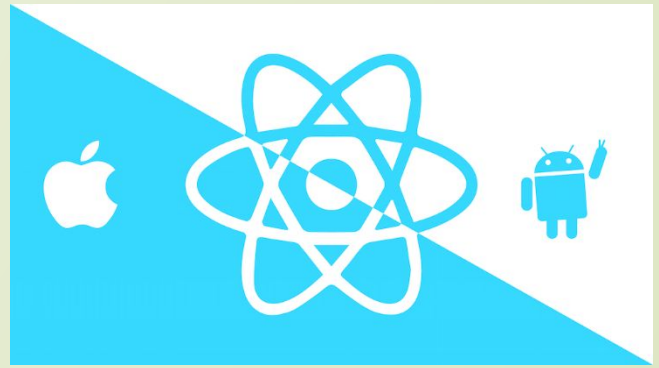


React Native

Tab Navigator

1) import { NavigationContainer } from '@react-navigation/native';
2) import { createTabNavigator } from '@react-navigation/bottom-tabs';
3) include NavigationContainer into App.js or index.js:

```
import { NavigationContainer } from '@react-navigation/native';
export default function App() {
return (
<NavigationContainer>{/* Rest of your app code */}</NavigationContainer>
);
}
```

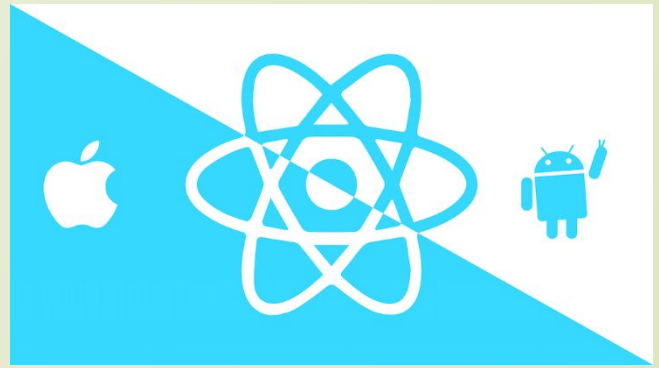


React Native

Tab Navigator

Jumping between tabs

- navigation.navigate same as in Stack navigator



React Native

- **Tab Navigator(Tab.Navigator accept props)**

- **initialRouteName** - The name of the route to render on first load of the navigator.
- **screenOptions**- Default options to use for the screens in the navigator.
- **backBehavior-** Behavior of back button handling.

  initialRoute - to return to initial tab

  order - to return to previous tab

  history to return to last visited tab

  none to not handle back button

- **Lazy** Defaults to true. If false, all tabs are rendered immediately. When true, tabs are rendered only when they are made active for the first time. Note: tabs are not re-rendered upon subsequent visits.

React Native

Tab Navigator(Tab.Navigator accept props)

- **tabBar -** Function that returns a React element to display as the tab bar.
- **tabBarOptions:** - An object containing the props for the tab bar component. It can contain the following properties:

✔ activeTintColor - Label and icon color of the active tab.

✔ activeBackgroundColor - Background color of the active tab.

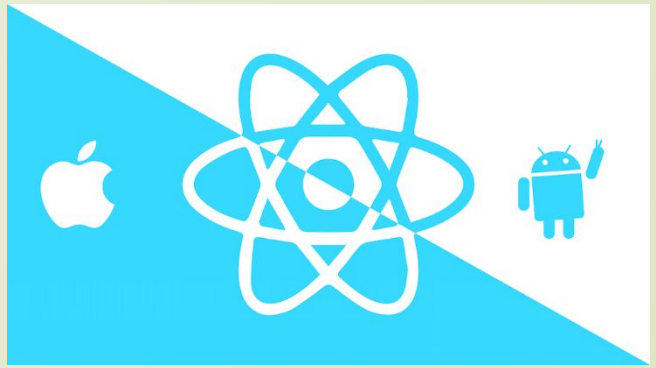
✔ inactiveTintColor - Label and icon color of the inactive tab.

✔ inactiveBackgroundColor - Background color of the inactive tab.

✔ showLabel - Whether to show label for tab, default is true.

✔ showIcon - Whether to show icon for tab, default is true.

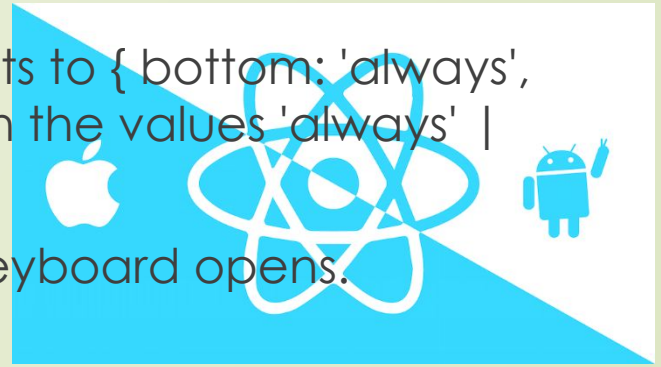
✔ style - Style object for the tab bar.



React Native

Tab Navigator(Tab.Navigator accept props)

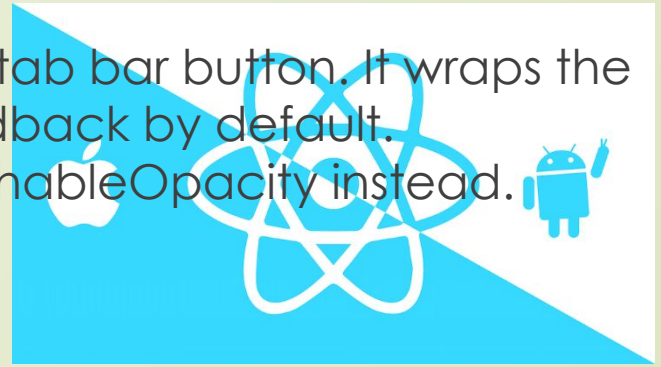
- labelStyle - Style object for the tab label.
- labelPosition - Where to show the tab label in relation to the tab icon. Available values are beside-icon and below-icon. Defaults to beside-icon.
- tabStyle - Style object for the tab.
- allowFontScaling - Whether label font should scale to respect Text Size accessibility settings, default is true.
- adaptive - Should the tab icons and labels alignment change based on screen size? Defaults to true for iOS 11. If false, tab icons and labels align vertically all the time. When true, tab icons and labels align horizontally on tablet.
- safeAreaInset - Override the forceInset prop for <SafeAreaView>. Defaults to { bottom: 'always', top: 'never' }. Available keys are top | bottom | left | right provided with the values 'always' | 'never'.
- keyboardHidesTabBar - Defaults to false. If true hide the tab bar when keyboard opens.



React Native

Tab Navigator(Configure individual screens)

- Options: The options prop can be used to configure individual screens inside the navigator. Supported options are:
- tittle -Generic title that can be used as a fallback for headerTitle and tabBarLabel.
- tabBarVisible true or false to show or hide the tab bar, if not set then defaults to true.
- tabBarIcon - Function that given { focused: boolean, color: string, size: number } returns a React.Node, to display in the tab bar.
- tabBarLabel - Title string of a tab displayed in the tab bar or a function that given { focused: boolean, color: string } returns a React.Node, to display in tab bar. When undefined, scene title is used. To hide, see tabBarOptions.showLabel in the previous section.
- tabBarButton - Function which returns a React element to render as the tab bar button. It wraps the icon and label and implements onPress. Renders TouchableWithoutFeedback by default. tabBarButton: props => <TouchableOpacity {...props} /> would use TouchableOpacity instead.

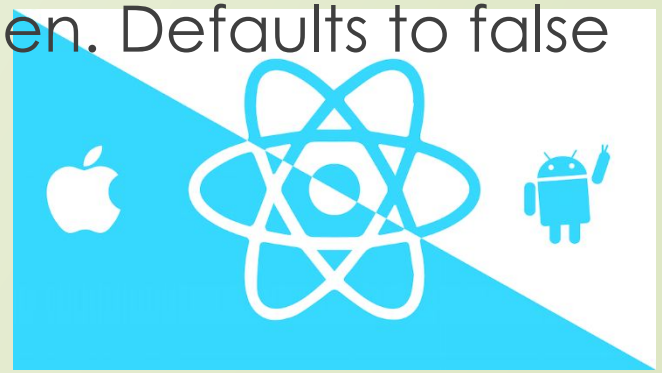


React Native

- **Tab Navigator(Configure individual screens)**

Options:

- tabBarAccessibilityLabel - Accessibility label for the tab button. This is read by the screen reader when the user taps the tab. It's recommended to set this if you don't have a label for the tab.
- tabBarTestID - ID to locate this tab button in tests.
- unmountOnBlur - Whether this screen should be unmounted when navigating away from it. Unmounting a screen resets any local state in the screen as well as state of nested navigators in the screen. Defaults to false



React Native

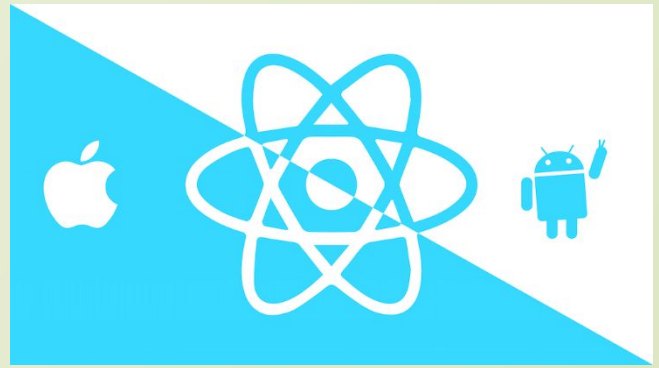
Tab Navigator

- **Full info all tab navigators**

https://reactnavigation.org/docs/bottom-tab-navigator

https://reactnavigation.org/docs/material-bottom-tab-navigator

https://reactnavigation.org/docs/material-top-tab-navigator



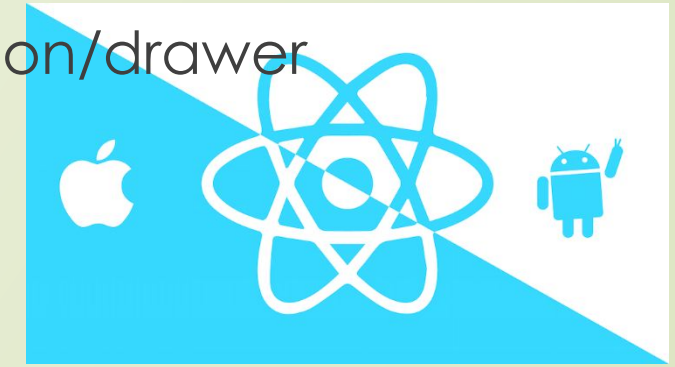
Example
import React from 'react';

React Native Drawer navigator

Installation Drawer navigator

3 steps to implement navigation into your project:

- 1) install react-navigation: npm install –save @react-navigation/native
- 2) install dependencies:
- npm install –save react-native-reanimated react-native-gesture-handler react-native-screens react-native-safe-area-context @react-native-community/masked-view
- 3) install tab nav: npm install - - save @react-navigation/drawer

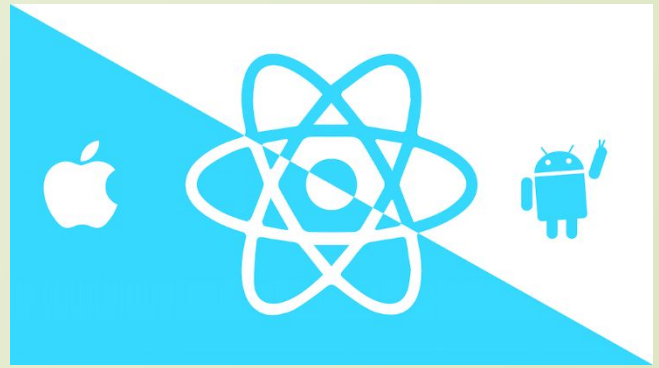


React Native

- **Drawer Navigator**

1) import { NavigationContainer } from '@react-navigation/native';
2) import { createDrawerNavigator } from '@react-navigation/drawer';
3) include NavigationContainer into App.js or index.js:

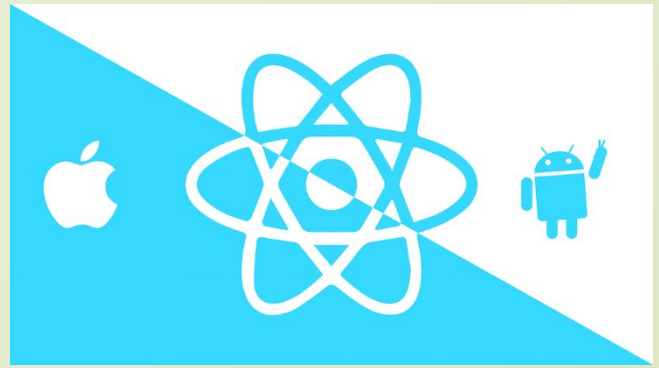
```
import { NavigationContainer } from '@react-navigation/native';
export default function App() {
return (
<NavigationContainer>{/* Rest of your app code */}</NavigationContainer>
);
}
```



React Native Drawer navigator

Open, Close, Toggle

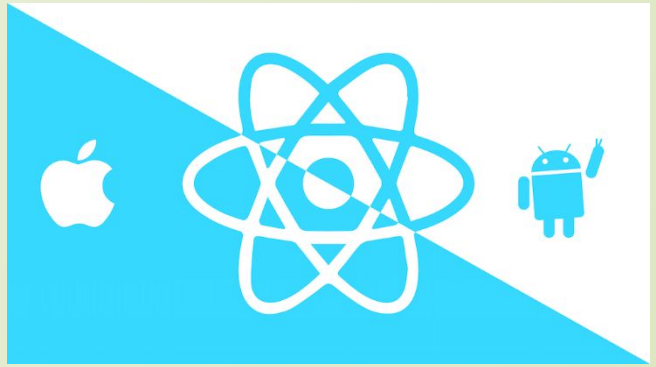
- navigation.**openDrawer**();
- navigation.**closeDrawer**();
- navigation.**toggleDrawer**();



React Native Drawer navigator

TO check Drawer navigation

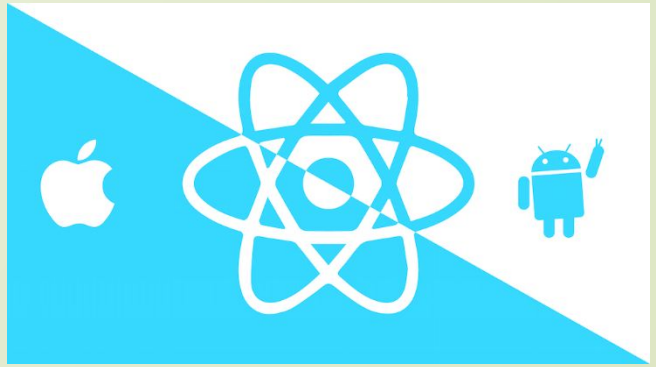
- const isDrawerOpen = useIsDrawerOpen();



React Native Drawer navigator

Drawer navigation Props

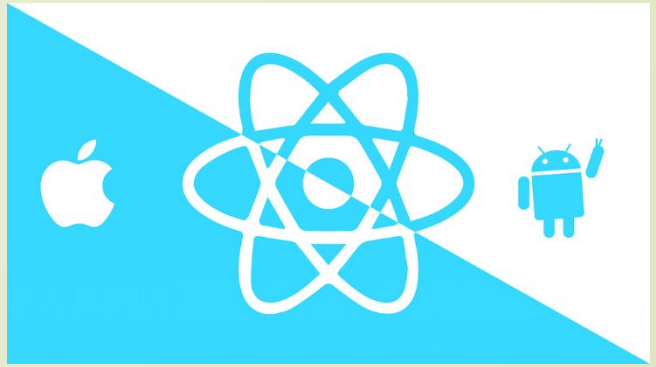
- https://reactnavigation.org/docs/drawer-navigator#props



React Native Drawer navigator

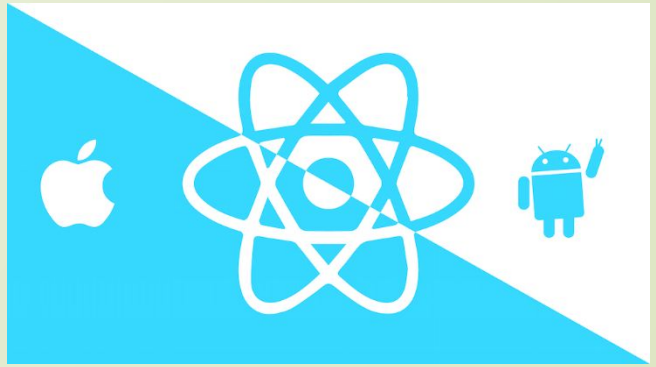
Drawer navigation Options

- https://reactnavigation.org/docs/drawer-navigator#options



React Native Drawer navigator

- **Full Documentation here:**
  - https://reactnavigation.org/docs/en/getting-started.html



React Native Drawer navigator

Git link

- https://github.com/elmardeveducation/MynavigationV5example.git