

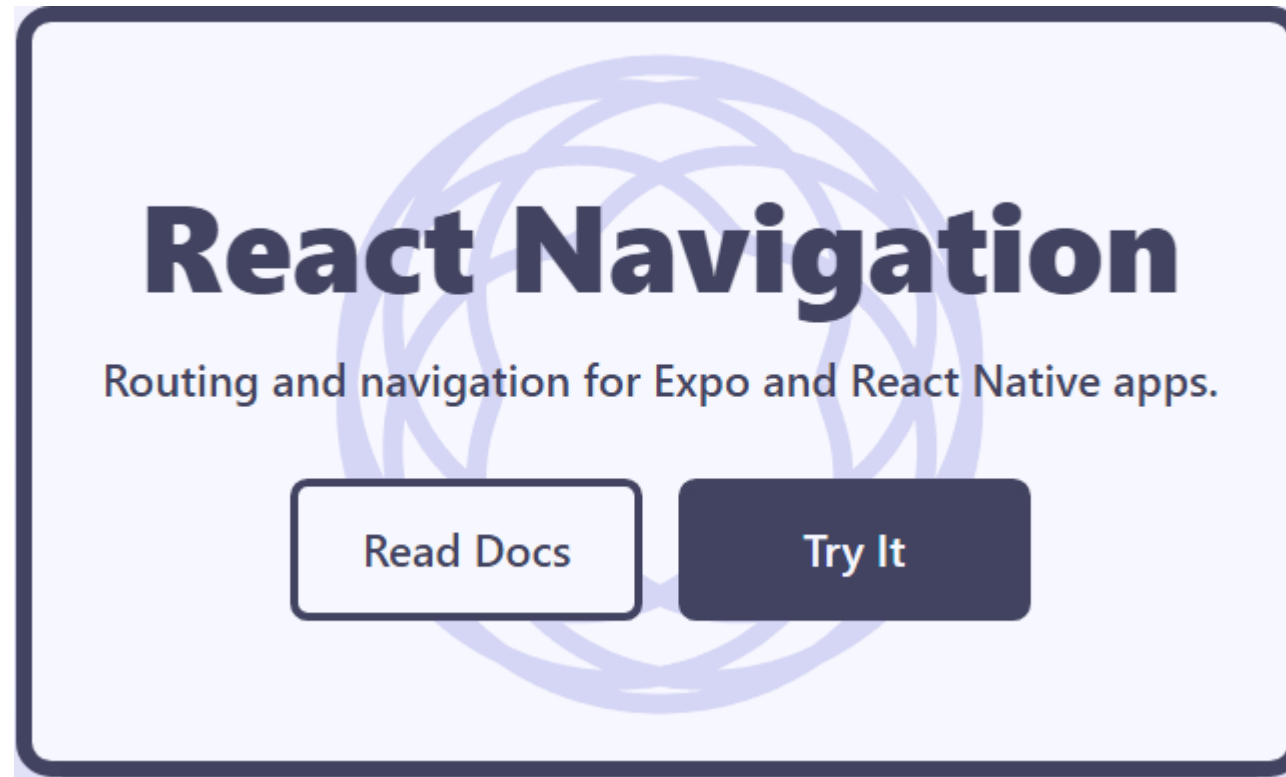
NAVIGATION



NGUYEN TRONG TIEN



<https://reactnavigation.org/>





npm

```
npm install @react-navigation/native
```

```
npx expo install react-native-screens react-native-safe-area-  
context
```

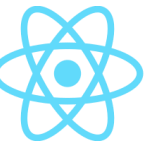


Navigation

Stack Navigation

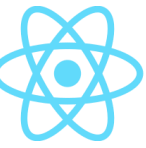
Tab Navigation

Drawer Navigation



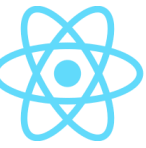
Wrapping your app in NavigationContainer

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



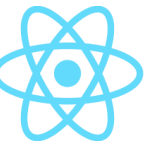
Hello React Navigation

```
npm install @react-navigation/native-stack
```



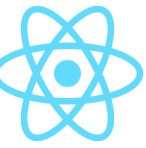
Creating a native stack navigator

```
function HomeScreen() {  
  return (  
    <View style={{ flex: 1, alignItems: 'center', justifyContent: 'center' }}>  
      <Text>Home Screen</Text>  
    </View>  
  );  
}
```



Creating a native stack navigator

```
import * as React from 'react';  
import { View, Text } from 'react-native';  
import { NavigationContainer } from '@react-navigation/native';  
import { createNativeStackNavigator } from '@react-navigation/native-stack';
```


Creating a native stack navigator

```
const Stack = createNativeStackNavigator();  
  
function App() {  
  return (  
    <NavigationContainer>  
      <Stack.Navigator>  
        <Stack.Screen name="Home" component={HomeScreen} />  
      </Stack.Navigator>  
    </NavigationContainer>  
  );  
}  
  
export default App;
```



Creating a native stack navigator

```
function DetailsScreen() {  
  return (  
    <View style={{ flex: 1, alignItems: 'center', justifyContent: 'center' }}>  
      <Text>Details Screen</Text>  
    </View>  
  );  
}
```



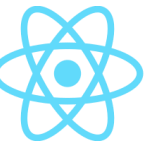
```
const Stack = createNativeStackNavigator();

function App() {
  return (
    <NavigationContainer>
      <Stack.Navigator>
        <Stack.Screen name="Home" component={HomeScreen} />
        <Stack.Screen name="DetailsScreen"
component={DetailsScreen} />
      </Stack.Navigator>
    </NavigationContainer>
  );
}
```



Specifying options

```
function DetailsScreen() {  
  return (  
    <View style={{ flex: 1, alignItems: 'center', justifyContent: 'center' }}>  
      <Text>Details Screen</Text>  
    </View>  
  );  
}
```



Passing additional props

We can do that with 2 approaches:

- Use React context and wrap the navigator with a context provider to pass data to the screens (recommended).
- Use a render callback for the screen instead of specifying a component prop:

```
<Stack.Screen name="Home">  
  {(props) => <HomeScreen {...props}  
    extraData={someData} />}  
</Stack.Screen>
```

Navigating to a new screen

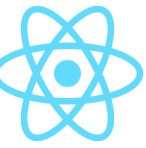
```
function HomeScreen({ navigation }) {  
  return (  
    <View style={{ flex: 1, alignItems: 'center',  
justifyContent: 'center' }}>  
      <Text>Home Screen</Text>  
      <Button  
        title="Go to Details"  
        onPress={() => navigation.navigate('Details')}  
      />  
    </View>  
  );  
}
```

Passing parameters to routes

```
function HomeScreen({ navigation }) {  
  return (  
    <View style={{ flex: 1, alignItems: 'center',  
justifyContent: 'center' }}>  
      <Text>Home Screen</Text>  
      <Button  
        title="Go to Details"  
        onPress={() => {  
          /* 1. Navigate to the Details route with  
params */  
          navigation.navigate('Details', {  
            itemId: 86,  
            otherParam: 'anything you want here',  
          });  
        }}  
      />  
    </View>  
  );  
}
```

Passing parameters to routes

```
function DetailsScreen({ route, navigation }) {  
  /* 2. Get the param */  
  const { itemId, otherParam } = route.params;  
  return (  
    <View style={{ flex: 1, alignItems: 'center', justifyContent: 'center' }}>  
      <Text>Details Screen</Text>  
      <Text>itemId: {JSON.stringify(itemId)}</Text>  
      <Text>otherParam: {JSON.stringify(otherParam)}</Text>  
      <Button  
        title="Go to Details... again"  
        onPress={() =>  
          navigation.push('Details', {  
            itemId: Math.floor(Math.random() * 100),  
          })  
        } />  
      <Button title="Go to Home" onPress={() => navigation.navigate('Home')} />  
      <Button title="Go back" onPress={() => navigation.goBack()} />  
    </View>  
  );  
}
```

Initial params

```
<Stack.Screen  
  name="Details"  
  component={DetailsScreen}  
  initialParams={{ itemId: 42 }}  
>
```



Updating params

```
<Button
  title="Update param"
  onPress={() =>
    navigation.setParams({
      itemId: Math.floor(Math.random() * 100),
    })
  }
/>
```

Passing
params
to a
previous
screen

```
function HomeScreen({ navigation, route }) {  
  React.useEffect(() => {  
    if (route.params?.post) {  
      // Post updated, do something with `route.params.post`  
      // For example, send the post to the server  
    }  
  }, [route.params?.post]);  
  
  return (  
    <View style={{ flex: 1, alignItems: 'center', justifyContent: 'center' }}>  
      <Button ="Create post"  
        onPress={() => navigation.navigate('CreatePost')}  
      />  
      <Text style={{ margin: 10 }}>Post: {route.params?.post}</Text>  
    </View>  
  );  
}
```

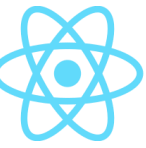
Passing
params
to a
previous
screen

```
function CreatePostScreen({ navigation, route }) {  
  const [postText, setPostText] = React.useState('');  
  return (  
    <TextInput      multiline  
      placeholder="What's on your mind?"  
      style={{ height: 200, padding: 10, backgroundColor: 'white' }}  
      value={postText}  
      onChangeText={setPostText}/>  
    <Button      title="Done"  
      onPress={() => {  
        // Pass and merge params back to home screen  
        navigation.navigate({  
          name: 'Home',  
          params: { post: postText },  
          merge: true,  
        });  
      }}  
    </Button>  
  )  
}
```

Passing params to nested navigators

If you have nested navigators, you need to pass params a bit differently. For example, say you have a navigator inside the Account screen, and want to pass params to the Settings screen inside that navigator. Then you can pass params as following:

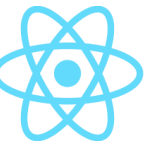
```
navigation.navigate('Account', {  
  screen: 'Settings',  
  params: { user: 'jane' },  
});
```



What should be in params?



Configuring the header bar



Setting the header title

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


Using params in the title

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



Updating options with setOptions

```
<Button  
  title="Update the title"  
  onPress={() => navigation.setOptions({ title:  
    'Updated!' })}  
>
```

Adjusting header styles

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

Sharing common options across screens

```
function StackScreen() {  
  return (  
    <Stack.Navigator  
      screenOptions={{  
        headerStyle: {  
          backgroundColor: '#f4511e',  
        },  
        headerTintColor: '#fff',  
        headerTitleStyle: {  
          fontWeight: 'bold',  
        },  
      }} >  
      <Stack.Screen name="Home" component={HomeScreen}  
        options={{ title: 'My home' }}  
      />  
    </Stack.Navigator>  
  );  
}
```

Replacing
the title with a
custom
component

```
function LogoTitle() {  
  return ( <Image style={{ width: 50, height: 50 }}  
source={require('@expo/snack-static/react-native-logo.png')}  
/> );  
  
function StackScreen() {  
  return (  
    <Stack.Navigator>  
      <Stack.Screen  
        name="Home"  
        component={HomeScreen}  
        options={{ headerTitle: (props) => <LogoTitle  
{...props} /> }}  
      />  
    </Stack.Navigator>  
  );  
}
```

Header buttons



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"  
            />  
          ) ,  
        }}  
      />  
    </Stack.Navigator>  
  );  
}
```

Header
interaction
with its
screen
component

In some cases, components in the header need to interact with the screen component. For this use case, we need to use `navigation.setOptions` to update our options. By using `navigation.setOptions` inside the screen component, we get access to screen's props, state, context etc.

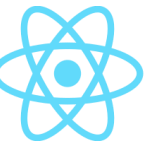
Nesting navigators



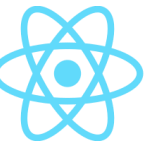
Nesting navigators

Nesting navigators means rendering a navigator inside a screen of another navigator

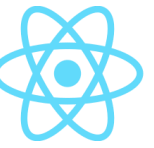
- `Stack.Navigator`
 - `Home (Tab.Navigator)`
 - `Feed (Screen)`
 - `Messages (Screen)`
 - `Profile (Screen)`
 - `Settings (Screen)`



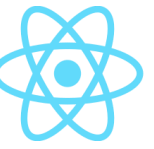
The next of Navigator in Slide Advanced...



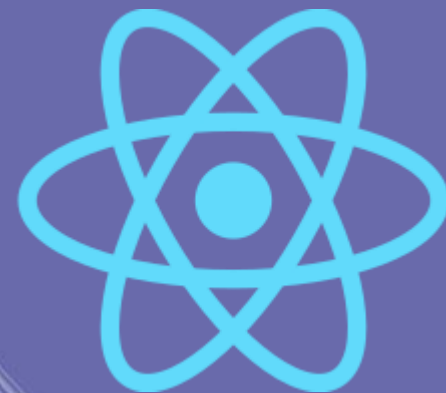
Tab Navigation



Tab Navigation



Drawer Navigation



THANK YOU!