# React Native Navigation

## CS571 – Mobile Application Development

**Maharishi International University**

**Department of Computer Science**

**M.S. Thao Huy Vu**

# Maharishi International University - Fairfield, Iowa

# What is a Navigator?

A Navigator is how we move between screens in an application.

Web navigation is usually tied to URLs in the browser.

Mobile apps do not use URLs for navigating within the application.

The native navigation APIs completely different between iOS and Android. But several React Native libraries provide agnostic alternative, we will be using React Navigation.

# React Navigation and alternatives

Two distinct approaches

1. Implement mainly in JavaScript in React Native
2. Implement mostly in native, expose an interface to JavaScript for existing native navigation APIs

# Install React Navigation

Follow the React Navigation Docs for full instructions.

- npm install @react-navigation/native
- npx expo install react-native-screens react-native-safe-area-context

# Navigation Container

```
import * as React from 'react';

import { NavigationContainer } from '@react-navigation/native';

export default function App() {
    return (
        <NavigationContainer>
            {/* Rest of your app code */}
        </NavigationContainer>
    );
}
```

# Navigators and Screen components

A Navigator is a component that implements a navigation pattern (stack, drawer, tabs ..etc)

Each Navigator must have one or more Screens. A Screen is a child of a Navigator. Each **Screen** must have a **name** and a **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.

# Navigator Installation

When you use a navigator (such as stack navigator), you'll need to follow the installation instructions of that navigator for any additional dependencies.

# Stack Navigator

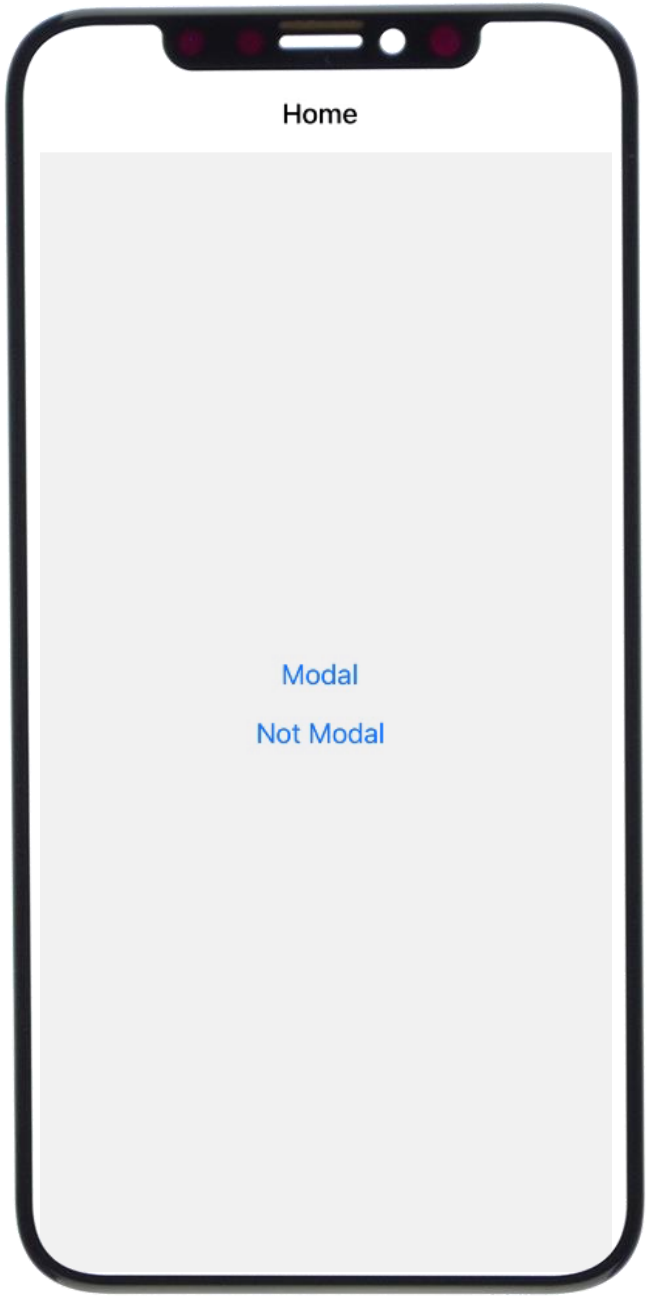
Display one screen at a time.

Screens are stacked on top of each other.

The state of inactive screens is **maintained,** and they remain mounted.

Platform-specific layout, animations, and gestures

Users can **push** and **pop** items from the stack, **replace** the current item.



# Higher order components

**`create*Navigator()`** 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. It is similar to higher order functions, which are functions that either take functions as arguments or return a function as a result.*

# Creating a Stack Navigator

- Install dependency: npm install @react-navigation/native-stack

```
import { createNativeStackNavigator } from '@react-navigation/native-stack';
const Stack = createNativeStackNavigator();

<Stack.Navigator initialRouteName="home">
   <Stack.Screen name="home" component={Home} options={{title: 'Home', headerShown: false }} />
   <Stack.Screen name="about" component={About} options={{title: 'About Screen'}}/>
   <Stack.Screen name="contact" component={Contact} options={{title: 'Contact Screen'}}/>
</Stack.Navigator>
```

# Navigating to Another Route

```
<Button
   title="Go to About"
   onPress={() => props.navigation.navigate('about')} />
```

The **navigation** prop is passed into the screen component for each route.
It is passed via **NavigationContainer** only to the screens tree. If you have a sub-component that is not a screen, it will not receive the prop.

# Returning to the Previous Active Route

```
<Button
   title="Go Back"
   onPress={() => props.navigation.goBack()} />
```

# Navigation prop

**`navigation`**

- **`.navigate()`** - go to another screen
- **`.goBack()`** - close active screen and move back in the stack

https://reactnavigation.org/docs/navigation-prop/

# Stack Specific navigation Actions

`navigation`

- **`.replace()`** - replace the current route with a new one
- **`.push()`** - push a new route onto the stack
- **`.pop()`** - go back in the stack
- **`.popToTop()`** - go to the top of the stack
- **`.reset()`** - wipe the navigator state

# Passing Params to Another Route

```
function HomeScreen({ navigation: { navigate } }) {
  return (
    <View>
       <Button
          onPress={() => navigate('Profile', { names: ['Asaad', 'Mike', 'Mada']})}
          title="Go to Profile"
       />
    </View>
  );
}
```

# Reading Params

```
function ProfileScreen({ route: {params}}) {

    const { names } = params;

}
```

The **`route`** prop is passed into the screen component for each route

# Configuring Screens Options

1. Sharing Common **options** Across Screens

```
<Stack.Navigator screenOptions={{ headerStyle: { backgroundColor: '#f4511e'},
headerTintColor: '#fff',}}> ..Screens.. </Stack.Navigator>
```

2. Screen Specific **options**

```
<Stack.Screen name="home" component={homeScreen} options={{title: 'Home'}}/>
```

3. Updating **options** with **setOptions**

```
const homeScreen = ({ navigation }) => {
   navigation.setOptions({
      title: 'Home',
   });
}
```

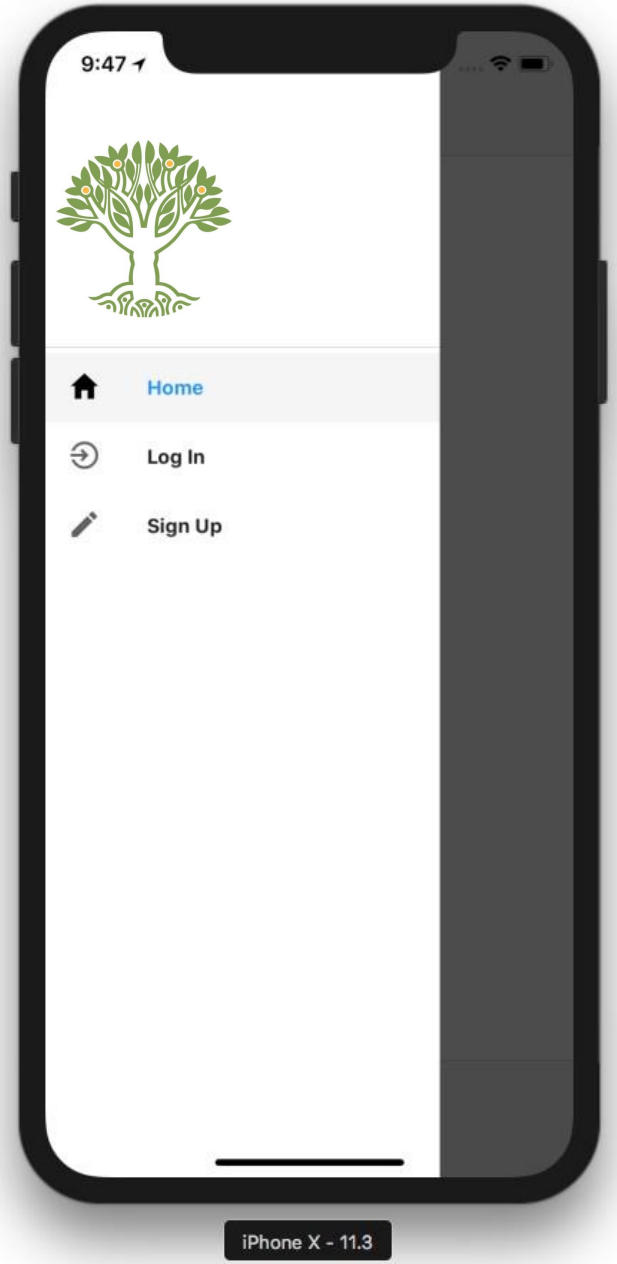
https://reactnavigation.org/docs/headers/

# Drawer Navigator

- `Install dependency:` npm install @react-navigation/drawer
- npx expo install react-native-gesture-handler  react-native-reanimated
- Added plugin to babel.config.js:  plugins: ["react-native-reanimated/plugin"]

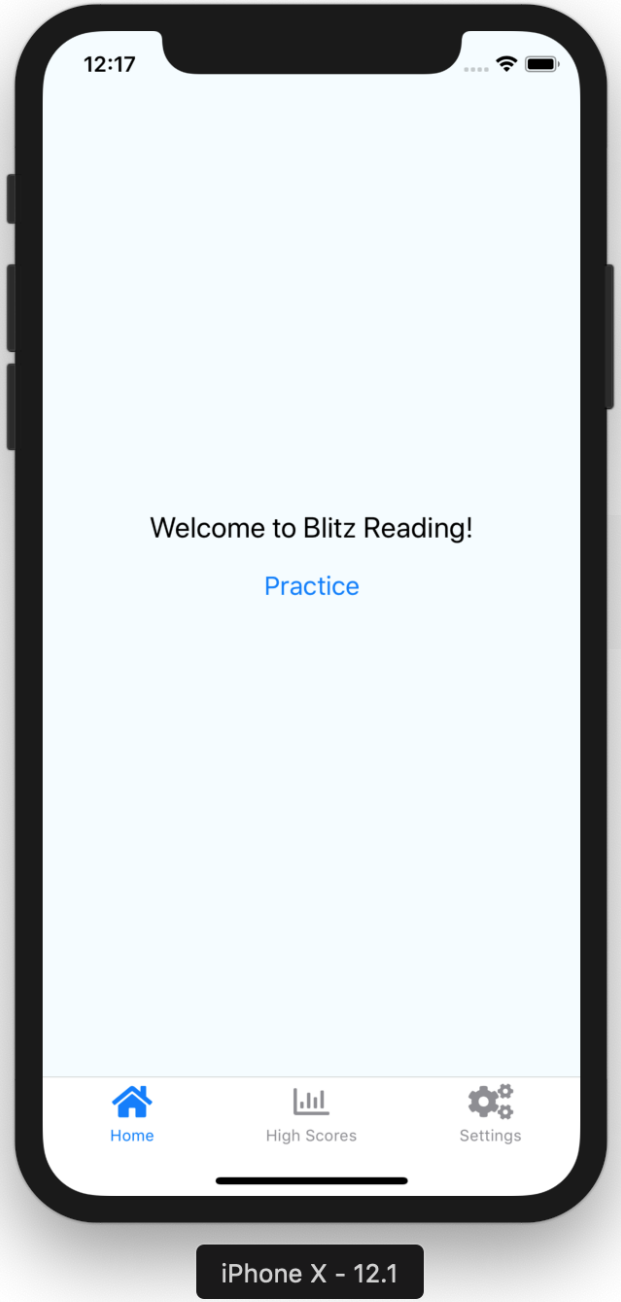
```
Import react-native-gesture-handler;
import {createDrawerNavigator} from '@react-navigation/drawer';
const Drawer = createDrawerNavigator();

<Drawer.Navigator initialRouteName="home">
   <Drawer.Screen name="home" component={Home} />
   <Drawer.Screen name="settings" component={Settings} />
</Drawer.Navigator>
```



# Tab navigators

- Display a simple tab bar to navigate between routes.
- Modern design.
- Install dependency: npm install @react-navigation/bottom-tabs



# Creating a Tab Navigator

```
import { createBottomTabNavigator } from '@react-
navigation/bottom-tabs';

const Tab = createBottomTabNavigator();


function MyTabs() {
  return (
    <Tab.Navigator>
        <Tab.Screen name="Home" component={HomeScreen} />
       <Tab.Screen name="Settings" component={SettingsScreen} />
    </Tab.Navigator>
  );
}
```

# Creating a Material Tab Navigator

```
import { createMaterialBottomTabNavigator } from '@react-navigation/material-bottom-tabs';
const BottomTabs = createMaterialBottomTabNavigator();

<BottomTabs.Navigator initialRouteName={myStackNavigator}
          activeColor="#f0edf6" inactiveColor="#3e2465"  barStyle={{ backgroundColor: '#0066CC' }}>
    <BottomTabs.Screen
        name="homeRoute"
        component={myStackNavigator}
        options={{ title: 'Courses'}} />
</BottomTabs.Navigator>
```

# Configure Tab icons

```
import { MaterialCommunityIcons } from 'react-native-vector-icons';

<BottomTabs.Screen
      name="homeRoute"
      component={myStackNavigator}
      options={{
            title: 'Courses',
            tabBarIcon: ({ color }) => (
                  <MaterialCommunityIcons name="home" color={color} size={26} />
            ),
      }}
/>
```

# Use Common Icon Packs

```
import Ionicons from "react-native-vector-icons/Ionicons";

<Ionicons name="ios-book" size={25} color="#000" />
```

*Resources:*

*https://oblador.github.io/react-native-vector-icons/*

*https://github.com/oblador/react-native-vector-icons*

# Expo Vector Icons

```
import { Ionicons, AntDesign } from '@expo/vector-icons';

<Ionicons name="md-checkmark-circle" size={32} color="green" />
<AntDesign name="book" size={100} color="#0066CC" />
```

*Resources:*

https://expo.github.io/vector-icons/

https://github.com/expo/vector-icons

# 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. Also, `goBack()` works for the whole app.

# Composing Navigators

```
function Home() {
    return (
        <Tab.Navigator>
           <Tab.Screen name="Feed" component={Feed} />
           <Tab.Screen name="Messages" component={Messages} />
        </Tab.Navigator>
   );
}
function App() {
    return (
        <NavigationContainer>
                <Stack.Navigator>
                        <Stack.Screen name="Home" component={Home} />
                        <Stack.Screen name="Profile" component={Profile} />
                        <Stack.Screen name="Settings" component={Settings} />
                </Stack.Navigator>
        </NavigationContainer>
    );
}
```

# Passing params to Nested Navigators

If you have nested navigators, you need to pass params as following:

```
navigation.navigate('Home', { screen: 'Feed',
                              params: { user: 'Asaad' } });
```

# **useNavigation()**

If a component is not part of the **NavigationContainer**, we can grab a reference to the **navigation** object using **useNavigation()** hook.

```
import { useNavigation, useRoute } from '@react-navigation/native';

const Details = (props) => {
    const navigation = useNavigation();
    const route = useRoute();
    ...
}
```

# React Navigation Resources

*React Navigation Documentation* https://reactnavigation.org/