**Navigator**

Navigator trong React Native cũng giống như UINavigationController(giống như một stack để quản lý các views), có thể push, pop,…Chúng ta hoàn toàn có thể tuỳ chỉnh lại nó nhưng sẽ xem ở phần tiếp theo. Import Navigator vào index.ios.js như sau:

'use strict';

**var** React = require('react');

**var** ReactNative = require('react-native');

**var** {

AppRegistry,

StyleSheet,

Navigator,

Text,

View,

} = ReactNative;

import SimpleButton from './SimpleButton';

**class** Test extends React.Component

{

render ()

{

**return** (

<**Navigator**

initialRoute={{name: 'home'}}

renderScene={this.renderScene}/> );

}

}

Navigator nhận một prop bằng cách gọi initialRoute nó truyền vào một đối tượng để làm view đầu tiên trong stack. Cần thêm tên để phân biệt giữa các views, ở đây tôi tạo luôn một function mới có tên renderScene gồm 2 tham số route, nagigator, trong thân hàm sử dụng switch  case để phân biệt giữa các views.

'use strict';

**var** React = require('react');

**var** ReactNative = require('react-native');

**var** {

AppRegistry,

StyleSheet,

Navigator,

Text,

View,

} = ReactNative;

import SimpleButton from './SimpleButton';

**class** Test extends React.Component

{

renderScene (route, navigator) {

**switch** (route.name) {

**case** 'home':

**return** (

<**View** style={styles.container}>

<**SimpleButton**

onPress={() => console.log('Pressed!')}

customText='Create Note'

/>

</**View**> );

case 'createNote':

}

}

render ()

{

return (

<**Navigator**

initialRoute={{name: 'home'}}

renderScene={this.renderScene}/> );

}

}

Ngoài view home, bạn có thể thêm một số các view khác, cũng giống như ở phần trước chúng ta tạo một file mới có tên NoteScreen.js và bắt sự kiện khi onPress thì nó gọi đến view NoteScreen.js

**Mã của file NoteScreen.js:**

'use strict';

**var** React = require('react');

**var** ReactNative = require('react-native');

**var** {

StyleSheet,

Navigator,

Text,

View,

} = ReactNative;

export **default** **class** NoteScreen extends React.Component

{

render ()

{

**return** (

<**View** style={styles.container}>

<**Text**>Create Note Screen!</**Text**>

</**View**>

);

}

}

const styles = StyleSheet.create({

container: {

flex: 1,

justifyContent: 'center',

alignItems: 'center',

backgroundColor: '#F5FCFF',

},

});

**Bây giờ quay lại file index.ios.js**. Khi ấn vào nút Create Note thì nó sẽ push NoteScreen vào stack bằng cách gọi đến .push:

renderScene (route, navigator){

**switch** (route.name) {

**case** 'home':

**return** (

<**View** style={styles.container}>

<**SimpleButton** onPress={() => {

navigator.push({

name: 'createNote'

});

}}

customText='Create Note'/>

</**View**> );

case 'createNote':

return (

<**NoteScreen** />

);

}

}

Chú ý là cần tên để khi nó chạy vào hàm renderScene thì nói sẽ return về NoteScreen.

**Navigator.NavigationBar**

Bạn có thể thưởng tượng nó là một thanh ở top các màn hình của ứng dụng gồm 2 nút trái, phải, title(hoàn toàn có thể chỉnh sửa được nhưng về phần customize này chúng ta sẽ xem ở phần tiếp theo) ở phần này đơn giản sẽ có 2 nút để chúng ta có thể push và pop qua lại các views. Ở index.ios.js các bạn thay đổi phần khởi tạo Navigator:

render ()

{

**return** (

<**Navigator**

initialRoute={{name: 'home'}}

renderScene={this.renderScene}

navigationBar={

<Navigator.NavigationBar routeMapper={NavigationBarRouteMapper}

/>

}

/>

);

}

Prop routeMapper chó phép truyền vào 1 biến bao gồm nút bên trái, phải, tiêu đề. Chèn đoạn mã khai báo biến NavigationBarRouteMapper chữa 2 nút và 1 tiêu đề.

**var** NavigationBarRouteMapper = {

LeftButton: **function**(route, navigator, index, navState) {

//Các đoạn mã nút trái

},

RightButton: **function**(route, navigator, index, navState) {

//Các đoạn mã nút phải

},

Title: **function**(route, navigator, index, navState) {

//Các đoạn mã phần nhãn

} };

Ở nút bên phải sẽ push sang view createNote, và với title của nó là ‘CreateNote’:

RightButton: **function**(route, navigator, index, navState) {

**switch** (route.name) {

**case** 'home':

**return** (

<**SimpleButton**

onPress={() => {

navigator.push({

name: 'createNote'

});

}}

customText='Create Note'

/>

);

default:

return null;

}

}

Nút bên trái dĩ nhiên sẽ là pop lại, với title là: “Back”

LeftButton: **function**(route, navigator, index, navState) {

**switch** (route.name) {

**case** 'createNote':

**return** (

<**SimpleButton**

onPress={() => navigator.pop()}

customText='Back'

/>

);

default:

return null;

}

}

Với phương thức naivator.pop() nó sẽ remove view ở trên top, như vậy thì màn hình hiện tại của chúng ta sẽ quay lại view cũ.

Cuối cùng thêm các đoạn mã để khi sang 1 view mới nó sẽ hiển thị đúng nhãn của view đó:

Title: **function**(route, navigator, index, navState) {

**switch** (route.name) {

**case** 'home':

**return** (

<**Text**>Home Screen</**Text**>

);

case 'createNote':

return (

<**Text**>Create Note</**Text**>

);

}

}

Bây giờ tạo thì tạo screen HomeScreen đặt tên là HomeScreen.js và viết đoạn mã sau:

'use strict';

**var** React = require('react');

**var** ReactNative = require('react-native');

**var** {

StyleSheet,

Navigator,

Text,

View,

} = ReactNative;

export **default** **class** HomeScreen extends React.Component

{

render ()

{

**return** (

<**View** style={styles.container}>

<**Text**>Home</**Text**>

</**View**>

);

}

}

const styles = StyleSheet.create({

container: {

flex: 1,

justifyContent: 'center',

alignItems: 'center',

backgroundColor: '#F5FCFF',

},

});

Tiếp theo ở index.ios.js thêm đoạn mã sau:

import HomeScreen from './NoteScreen';

Và sửa lại hàm renderScene như sau:

renderScene (route, navigator)

{

**switch** (route.name) {

**case** 'home':

**return** (

<**View** style={styles.container}>

<**HomeScreen** />

</**View**> );

case 'createNote':

return (

<**NoteScreen** />

);

}

}