

# react-navigation

一： react-navigation包含三类组件： □

- (1) **StackNavigator**: 用来跳转页面和传递参数
- (2) **TabNavigator**: 类似底部导航栏，用来在同一屏幕下切换不同页面
- (3) **DrawerNavigator**: 侧滑菜单导航栏，用来设置带轴导航的屏幕

二： 功能详解

- (1) **react-navigation库属性介绍**
- (2) **StackNavigator, TabNavigator实现界面间跳转, Tab切换**
- (3) **StackNavigator界面之间的跳转, 传值, 取值**
- (4) **DrawerNavigator实现抽屉导航菜单**
- (5) **DrawerNavigator扩展功能**

(1) **StackNavigator属性介绍**

**navigationOptions**: 配置StackNavigator的一些属性。

**title**: 标题，如果设置了这个导航栏和标签栏的**title**就会变成一样的，不推荐使用  
**header**: 可以设置一些导航的属性，如果隐藏顶部导航栏只要将这个属性设置为**null**  
**headerTitle**: 设置导航栏标题，推荐  
**headerBackTitle**: 设置跳转页面左侧返回箭头后面的文字，默认是上一个页面的标题。可以自定义，也可以设置为**null**  
**headerTruncatedBackTitle**: 设置当上个页面标题不符合返回箭头后的文字时，默认改成“返回”  
**headerRight**: 设置导航条右侧。可以是按钮或者其他视图控件  
**headerLeft**: 设置导航条左侧。可以是按钮或者其他视图控件  
**headerStyle**: 设置导航条的样式。背景色，宽高等  
**headerTitleStyle**: 设置导航栏文字样式  
**headerBackTitleStyle**: 设置导航栏‘返回’文字样式  
**headerTintColor**: 设置导航栏颜色  
**headerPressColorAndroid**: 安卓独有的设置颜色纹理，需要安卓版本大于5.0  
**gesturesEnabled**: 是否支持滑动返回手势，iOS默认支持，安卓默认关闭

**screen**: 对应界面名称，需要填入import之后的页面

**mode**: 定义跳转风格

**card**: 使用iOS和安卓默认的风格

**modal**: iOS独有的使屏幕从底部画出。类似iOS的**present**效果

**headerMode**: 返回上级页面时动画效果

**float**: iOS默认的效果

**screen**: 滑动过程中，整个页面都会返回

**none**: 无动画

**cardStyle**: 自定义设置跳转效果

**transitionConfig**: 自定义设置滑动返回的配置

**onTransitionStart**: 当转换动画即将开始时被调用的功能

**onTransitionEnd**: 当转换动画完成，将被调用的功能

**path**: 路由中设置的路径的覆盖映射配置

**initialRouteName**: 设置默认的页面组件, 必须是上面已注册的页面组件

**initialRouteParams**: 初始路由参数

**path**属性适用于其他app或浏览器使用url打开本app并进入指定页面。**path**属性用于声明一个界面路径, 例如:  
【/pages/Home】。此时我们可以在手机浏览器中输入: app名称://pages/Home来启动该App, 并进入Home界面。

## (2) TabNavigator属性介绍

**screen**: 和导航的功能是一样的, 对应界面名称, 可以在其他页面通过这个**screen**传值和跳转。

**navigationOptions**: 配置TabNavigator的一些属性

**title**: 标题, 会同时设置导航条和标签栏的**title**

**tabBarVisible**: 是否隐藏标签栏。默认不隐藏(true)

**tabBarIcon**: 设置标签栏的图标。需要给每个都设置

**tabBarLabel**: 设置标签栏的**title**。推荐

导航栏配置

**tabBarPosition**: 设置tabbar的位置, iOS默认在底部, 安卓默认在顶部。 (属性值: 'top', 'bottom')

**swipeEnabled**: 是否允许在标签之间进行滑动

**animationEnabled**: 是否在更改标签时显示动画

**lazy**: 是否根据需要懒惰呈现标签, 而不是提前, 意思是在app打开的时候将底部标签栏全部加载, 默认false, 推荐为true

**trueinitialRouteName**: 设置默认的页面组件

**backBehavior**: 按 back 键是否跳转到第一个Tab(首页), none 为不跳转

**tabBarOptions**: 配置标签栏的一些属性iOS属性

**activeTintColor**: label和icon的前景色 活跃状态下

**activeBackgroundColor**: label和icon的背景色 活跃状态下

**inactiveTintColor**: label和icon的前景色 不活跃状态下

**inactiveBackgroundColor**: label和icon的背景色 不活跃状态下

**showLabel**: 是否显示label, 默认开启 style: tabBar的样式

**labelStyle**: label的样式安卓属性

**activeTintColor**: label和icon的前景色 活跃状态下

**inactiveTintColor**: label和icon的前景色 不活跃状态下

**showIcon**: 是否显示图标, 默认关闭

**showLabel**: 是否显示label, 默认开启 style: tabBar的样式

**labelStyle**: label的样式 **upperCaseLabel**: 是否使标签大写, 默认为true

**pressColor**: material涟漪效果的颜色 (安卓版本需要大于5.0)

`pressOpacity`: 按压标签的透明度变化（安卓版本需要小于5.0）

`scrollEnabled`: 是否启用可滚动选项卡 `tabStyle`: tab的样式

`indicatorStyle`: 标签指示器的样式对象（选项卡底部的行）。安卓底部会多出一条线，可以将`height`设置为0来暂时解决这个问题

`labelStyle`: label的样式

`iconStyle`: 图标样式

### (3) DrawerNavigator属性介绍

#### DrawerNavigatorConfig

`drawerWidth` – 抽屉的宽度

`drawerPosition` – 选项是左或右。默认为左侧位置

`contentComponent` – 用于呈现抽屉内容的组件，例如导航项。接收抽屉的导航。默认为`DrawerItems`

`contentOptions` – 配置抽屉内容

`initialRouteName` – 初始路由的`routeName`

`order` – 定义抽屉项目顺序的`routeNames`数组。

`路径` – 提供`routeName`到路径配置的映射，它覆盖`routeConfigs`中设置的路径。

`backBehavior` – 后退按钮是否会切换到初始路由？如果是，设置为`initialRoute`，否则为`none`。默认为`initialRoute`行为

#### DrawerItems的contentOptions属性

`activeTintColor` – 活动标签的标签和图标颜色

`activeBackgroundColor` – 活动标签的背景颜色

`inactiveTintColor` – 非活动标签的标签和图标颜色

`inactiveBackgroundColor` – 非活动标签的背景颜色

内容部分的样式对象

`labelStyle` – 当您的标签是字符串时，要覆盖内容部分中的文本样式的样式对象

### (4) 使用StackNavigator + TabNavigator实现Tab界面切换，界面间导航，多级跳转

从头开始一个导航的小案例：

(1) \$ react-native init FirstApp // 新建一个项目

\$ cd FirstApp

\$ react-native run-ios // 启动项目 // 启动的时候可能会报错，这是因为babel-preset-react-native版本的问题，打开package.json

(2) 在根目录下新建 `src/screen/TabNavigation` 目录（目录可以随便建，只要引用的时候保证路径是对的即可）在`TabNavigation`

(3) `TabNavigation/HomePage.js`文件内容

```
import React from 'react';
import {
  StyleSheet,
  View,
  Text,
  Button,
  Image
} from 'react-native';
```

```
import {
  StackNavigator,
  TabNavigator
} from 'react-navigation';
```

```
import ChatScreen from './ChatScreen';
import NewScreen from "./NewPage";
import LoginScreen from "./LoginPage";
```

```

class HomePage extends React.Component {
  static navigationOptions = {
    title: '首页',//设置标题内容
  };
  onClickNew() {
    // 跳到New页面，并且传参数
    this.props.navigation.navigate('New', {name: 'zhangsan'});
  }
  render() {
    const { navigate } = this.props.navigation;
    return (
      <View style={styles.container}>
        <Text style={{ padding: 10 }}>Hello, Navigation!</Text>
        <Button
          onPress={() => navigate('Chat', { user: 'Sybil' })}
          title="点击跳转" />
        <Button
          onPress={this.onClickNew.bind(this)}
          title="New" />
      </View>
    )
  }
}
const MainScreen = StackNavigator({
  Tab: { screen: HomePage },
  Chat: { screen: ChatScreen },
  New: { screen: NewScreen },
})
const styles = StyleSheet.create({
  container: {
    flex: 1,
    backgroundColor: '#fff'
  },
  icon: {
    height: 22,
    width: 22,
    resizeMode: 'contain'
  }
});
// SimpleApp是要暴露出去的方法，也是我们运行项目时进入到的第一个页面，首先我们会进入到login页面，因为在login页面中
const SimpleApp = StackNavigator({
  Login: { screen: LoginScreen },
  Home: { screen: MainScreen },
}, {
  headerMode: 'none',
});
export default SimpleApp;

```

(4) 在TabNavigation目录下新建Loginjs文件

```

TabNavigation/Loginjs 文件内容
import React,{Component} from 'react';
import {
  Button,
  View,
  Text,
  StyleSheet
} from 'react-native';

class LoginScreen extends Component{
  render(){
    const { navigate } = this.props.navigation;
    return(
      <View style={{backgroundColor:'#fff',flex:1}}>
        <Text style={{padding:20}}>this is Login page</Text>
        <Button
          onPress={() => navigate('Home', { user: 'Sybil' })}
          title="登录" />
        <Button
          onPress={() => navigate('Chat', { user: 'haha' })}
          title="Chat" />
      </View>
    );
  }
}
const styles = StyleSheet.create({
  icon: {
    width: 24,
    height: 24,
  },
});
export default LoginScreen;

```

(5) 在TabNavigation目录下新建ChatScreen.js文件

TabNavigation/ChatScreen.js 文件内容

```

import React from 'react';
import {
  Button,
  View,
  Text
} from 'react-native';
class ChatScreen extends React.Component {
  static navigationOptions = {
    title:'聊天',
  };
  render() {
    // 通过this.props.navigation.state属性可以拿到跳过来的页面传的参数
    const {params} = this.props.navigation.state;
    // 通过this.props.navigation属性可以跳到某个页面，并给该页面传参
    const { navigate } = this.props.navigation;
    return (
      <View style={{backgroundColor:'#fff',flex:1}}>
        <Text style={{padding:20}}>Chat with {params.user}</Text>
        <Button onPress={() => navigate('New', {name: '张三'})} title='点击跳到new页' />
      </View>
    );
  }
}
export default ChatScreen;

```

(6) 在TabNavigation目录下新建NewPage.js文件

```

TabNavigation/newPage.js 文件内容
import React,{Component} from 'react';
import {
  View,
  Text,
  StyleSheet
} from 'react-native';
class NewScreen extends Component{
  static navigationOptions = {
    title: '新闻',
  };
  render(){
    const { params } = this.props.navigation.state;
    return(
      <View style={{backgroundColor:'#fff',flex:1}}>
        <Text style={{padding:20}}>this is new page</Text>
      </View>
    );
  }
  const styles = StyleSheet.create({
    icon: {
      width: 24,
      height: 24,
    },
  });
}
export default NewScreen;

```

(7) 修改index.js文件中的内容

```

import { AppRegistry } from 'react-native';
import SimpleApp from './src/screen/TabNavigation/HomePage';
AppRegistry.registerComponent('FirstApp', () => SimpleApp);

```

此时运行代码就会发现首页是login页，从login页可以进到home页，从home页可以进到New页和Chat页，从Chat页可以进入到New

上面的案例只用到了StackNavigator，即只是页面之间的跳转，并没有用到TabNavigator，下面我们把TabNavigator也加入到页面中

(1) \$ yarn add react-native-elements // 安装react-native-elements，我们下面要用到里面的icon。如果报错的话，把该包删除

(2) 在TabNavigation目录下新建MinePage.js文件

```

TabNavigation/MinePage.js 文件内容
import React,{Component} from 'react';
import {
  Button,
  Image,
  View,
  Text,
  StyleSheet
} from 'react-native';

class MinePage extends Component{

```

```

static navigationOptions = {
  title:'我的',
  drawerLabel: '我的',
  drawerIcon: ({ tintColor }) => (
    ),
};
render(){
  return(
    <View style={{backgroundColor:'#fff',flex:1}>
      <Text style={{padding:20}}>Sybil</Text>
      <Button
        style={{padding:20}}
        onPress={() => this.props.navigation.navigate('DrawerOpen')}
        title="点击打开侧滑菜单"
      />
    </View>
  );
}
const styles = StyleSheet.create({
  icon: {
    width: 24,
    height: 24,
  },
});
export default MinePage;

```

(2) 修改TabNavigation/HomePage.js中的内容

```

import React from 'react';
import {
  StyleSheet,
  View,
  Text,
  Button,
} from 'react-native';

import {
  StackNavigator,
  TabNavigator
} from 'react-navigation';
import { Icon } from 'react-native-elements';

import ChatScreen from './ChatScreen';
import MinePage from './MinePage';
import NewScreen from "./NewPage";
import LoginScreen from "./LoginPage";

class HomePage extends React.Component {

  static navigationOptions = {
    title: '首页',//设置标题内容
  };

  onClickNew() {
    this.props.navigation.navigate('New', {name: '张三'});
  }

  render() {
    const { navigate } = this.props.navigation;
    return (
      <View style={styles.container}>
        <Text style={{ padding: 10 }}>Hello, Navigation!</Text>
        <Button
          onPress={() => navigate('Chat', { user: 'Sybil' })}
          title="点击跳转" />
        <Button
          onPress={this.onClickNew.bind(this)}
          title="New" />
      </View>
    )
  }
}

// 添加TabNavigator底部导航按钮
const TabScreen = TabNavigator({
  Home: {
    screen: HomePage, // 要跳转到的页面
    navigationOptions: {
      tabBarLabel: '首页',
      tabBarIcon: ({ tintColor }) => , // 图标
    }
  }
});

```

```

    },
    },
},
Certificate: {
  screen: MinePage,
  navigationOptions: {
    tabBarLabel: '我的',
    tabBarIcon: ({ tintColor }) => ,
  },
  },
},
);
};

const MainScreen = StackNavigator({
  // Tab: {screen: TabScreen},
  Tab: {screen: HomePage},
  Chat: { screen: ChatScreen},
  New: { screen: NewScreen},
})
;

const styles = StyleSheet.create({
  container: {
    flex: 1,
    backgroundColor: '#fff'
  },
  icon: {
    height: 22,
    width: 22,
    resizeMode: 'contain'
  }
});
;

const SimpleApp = StackNavigator({
  Login: { screen: LoginScreen },
  Home: { screen: MainScreen },
},
{
  headerMode: 'none',
});
;

export default SimpleApp;

```

此时再运行react-native run-ios 会发现在屏幕底部多了两个导航，这两个导航分别可以跳到home页和mine页

## (5) DrawerNavigator扩展功能

上面的案例只用到了StackNavigator和Tabnavigator，下面继续使用DrawerNavigator

(1) 我们要从MinePage中打开侧导航，所以要先修改MinePage.js中的内容

```

import React,{Component} from 'react';
import {
  Button,
  View,
  Text,
  StyleSheet
} from 'react-native';
import { DrawerNavigator } from 'react-navigation';
import { Icon } from 'react-native-elements';

import DrawerScreen from './DrawerPage';
import NewScreen from './NewPage';

class MinePage extends Component{
  static navigationOptions = {
    title:'我的',
    drawerIcon: ({ tintColor }) => <Icon name="perm-identity" size={25} color={tintColor} />,
  };
  render(){
    return(
      <View style={{backgroundColor:'#fff',flex:1}}>
        <Text style={{padding:20}}>Sybil</Text>
        <Button
          style={{padding:20}}
          onPress={() => this.props.navigation.navigate('DrawerOpen')}
          title="点击打开侧滑菜单"
        />
      </View>
    );
  }
}
const styles = StyleSheet.create({

```

```

icon: {
  width: 24,
  height: 24,
},
);
const MyChatNavigator = DrawerNavigator({
  MyChat: {
    screen: MinePage,
  },
  Notifications: {
    screen: DrawerScreen,
  },
  New: {
    screen: NewScreen,
  }
},{
  drawerWidth: 220, // 抽屉宽
  drawerPosition: 'left', // 抽屉在左边还是右边
  contentOptions: {
    initialRouteName: DrawerScreen, // 默认页面组件
    activeTintColor: '#008AC9', // 选中文字颜色
    activeBackgroundColor: '#f5f5f5', // 选中背景颜色
    inactiveTintColor: '#000', // 未选中文字颜色
    inactiveBackgroundColor: '#fff', // 未选中背景颜色
    style: {} // 样式
  }
});
export default MyChatNavigator;

```

(2) 在TabNavigation目录下新建DrawerPage.js文件

```

TabNavigation/DrawerPage.js 文件内容
import React from 'react';
import {
  StyleSheet,
  View,
  Button,
} from 'react-native';
import { Icon } from 'react-native-elements';
class DrawerScreen extends React.Component {
  static navigationOptions = {
    title:'通知',
    drawerIcon: ({ tintColor }) => <Icon name="message" size={25} color={tintColor} />,
  };
  render() {
    return (
      <View style={{backgroundColor:'#fff'}>
        <Button
          style={{padding:20}}
          onPress={() => this.props.navigation.navigate('DrawerOpen')}
          title="点击打开侧滑菜单"
        />
        <Button
          onPress={() => this.props.navigation.goBack()}
          title="返回我的界面"
        />
      </View>
    );
  }
}
const styles = StyleSheet.create({
  tabIcon: {
    width: 24,
    height: 24,
  },
});
export default DrawerScreen;

```



