

Flutter 路由：Route

本节具体讲一下 Flutter 路由 Route 的使用，包括两个部分：

- 使用 Route 进行页面跳转
- 使用 Route 在页面间传递参数

Route -- Flutter页面跳转

App 中经常有多个页面，所以需要在多个页面中跳转，例如在 Android 中是有很多页面：Activity，然后使用 `startActivity()` 来跳转页面；在 iOS 中也有很多页面：ViewController，然后使用 `pushViewController` 来跳转页面。

在 Android 和 iOS 中，这种全屏的页面叫 Activity 或 ViewController，在 Flutter 中，这种全屏的页面就是 Route，Navigator 通过堆管理 Route 对象，从而实现页面跳转。

在 Flutter 中跳转页面有两种方式，一种是 Simple Route(简单路由)，一种是 Named Route(命名路由)。

简单路由

1.创建两个Scaffold

首先创建两个页面：FirstPage 和 SecondPage，点击 FirstPage 里的按钮跳转到 SecondPage，点击 SecondPage 里的按钮，在返回到 FirstPage。

下面的代码分别创建了 FirstPage 和 SecondPage：

```
import 'package:flutter/material.dart';
```

```
void main() => runApp(MyApp());

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    // TODO: implement build
    return MaterialApp(
      title: "Flutter Demo",
      theme: ThemeData(
        primaryColor: Colors.blue,
      ),
      home: FirstPage());
  }
}

class FirstPage extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    // TODO: implement build
    return Scaffold(
      appBar: AppBar(
        title: Text("Route -- FirstPage"),
      ),
      body: RaisedButton(
        child: Text("JUMP SecondRoute"),
        onPressed: () {

        },
      ),
    );
  }
}
```

```
class SecondPage extends StatelessWidget {  
  @override  
  Widget build(BuildContext context) {  
    // TODO: implement build  
    return Scaffold(  
      appBar: AppBar(  
        title: Text("Route -- SecondPage"),  
      ),  
      body: RaisedButton(  
        child: Text("Go back!"),  
        onPressed: () {  
  
        },  
      ),  
    );  
  }  
}
```

2.从 FirstPage 跳转到 SecondPage

跳转页面，使用 `Navigator.push()`，`push()` 方法将新的 Route 添加到由 Navigator 管理的 Route 对象的堆顶。那么新的 Route 对象从哪里来呢？可以直接使用 `MaterialPageRoute`，或者你也可以创建自定义的 Route。建议使用 `MaterialPageRoute`，因为 `MaterialPageRoute` 已经封装好了，使用方便，而且自带页面切换动画，并且适配了 Android 和 iOS，如果是 Android 的话，页面进入动画是向上滑动并淡出，退出是相反的，如果是 iOS 的话，页面进入动画是从右侧滑入，退出是相反的。

那么 FirstPage 的 `onPressed` 事件应该这么写：

```
Navigator.push(  
  context, MaterialPageRoute(builder: (context)  
=> SecondPage()));
```

MaterialPageRoute有四个参数：

参数名	类型	是否必填参数	作用
builder	WidgetBuilder	必填	创建Route里要显示的页面
settings	RouteSettings	选填	Route的一些配置参数
maintainState	bool	选填	是否保留之前的Route，如果是true，前面的Route会保留在内存里，如果是false，前面的Route在不需要的时候就会被回收掉（不是立即回收），默认是true
fullscreenDialog	bool	选填	用来标识新的页面是不是dialog，默认是false

可以看到上面的代码实现了 builder，返回了 SecondPage.

3.从 SecondPage 返回到 FirstPage

使用 Navigator.pop() 关闭当前页面，返回上一个页面。pop() 方法将当前的 Route 对象从 Navigator 管理 Route 对象的堆中移除。

所以 SecondPage 的 onPressed 事件应该这么写：

```
Navigator.pop(context);
```

4.完整代码

```
import 'package:flutter/material.dart';

void main() => runApp(MyApp());

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    // TODO: implement build
    return MaterialApp(
      title: "Flutter Demo",
      theme: ThemeData(
        primaryColor: Colors.blue,
      ),
      home: FirstPage());
  }
}

class FirstPage extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    // TODO: implement build
    return Scaffold(
      appBar: AppBar(
        title: Text("Route -- FirstPage"),
      ),
      body: RaisedButton(
        child: Text("JUMP SecondRoute"),
```

```

        onPressed: () {
          Navigator.push(
            context, MaterialPageRoute(builder:
(context) => SecondPage(),maintainState: false));
        },
      ),
    );
  }
}

class SecondPage extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    // TODO: implement build
    return Scaffold(
      appBar: AppBar(
        title: Text("Route -- SecondPage"),
      ),
      body: RaisedButton(
        child: Text("Go back!"),
        onPressed: () {
          Navigator.pop(context);
        },
      ),
    );
  }
}

```

命名路由

如果 App 里有很多地方都要跳转到相同的页面，那么以前面简单路由的方式跳转的话，就得重复写很多代码，这种情况下，为 Route 命名，使用命令路由就会很方便。

命令路由使用这个方法 `Navigator.pushNamed`

1.创建两个 Scaffold

这里和简单路由一样

2.定义路由表

路由表就是将所有Route集合起来，需要使用MaterialAPP的 `routes` 属性和 `initialRoute` 属性。

```
return MaterialApp(  
  title: "Flutter Demo",  
  theme: ThemeData(  
    primaryColor: Colors.blue,  
  ),  
  initialRoute: '/First',  
  routes: {  
    '/First': (context) => FirstPage(),  
    '/Second': (context) => SecondPage()  
  },  
  home: FirstPage());  
}
```

注意：命名的路由不能使用 `'/'`，因为 `'/'` 相当于是根节点，不要用根节点去命名路由。

属性名	类型	作用
<code>routes</code>	<code>Map<String, WidgetBuilder></code>	路由表
<code>initialRoute</code>	<code>String</code>	Flutter APP的初始路由

3.从 FirstPage 跳转到 SecondPage

命令路由的跳转使用 `Navigator.pushNamed`

那么 `FirstPage` 的 `onPressed` 事件应该改写为：

```
Navigator.pushNamed(context, '/Second');
```

4.从 `SecondPage` 返回到 `FirstPage`

命令路由返回上一个页面的使用方法和简单路由一样

5.完整代码

```
import 'package:flutter/material.dart';

void main() => runApp(MyApp());

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    // TODO: implement build
    return MaterialApp(
      title: "Flutter Demo",
      theme: ThemeData(
        primaryColor: Colors.blue,
      ),
      initialRoute: '/First',
      routes: {
        '/First': (context) => FirstPage(),
        "/Second": (context) => SecondPage()
      },
      home: FirstPage());
  }
}
```



```
class FirstPage extends StatelessWidget {  
  @override  
  Widget build(BuildContext context) {  
    // TODO: implement build  
    return Scaffold(  
      appBar: AppBar(  
        title: Text("Route -- FirstPage"),  
      ),  
      body: RaisedButton(  
        child: Text("JUMP SecondRoute"),  
        onPressed: () {  
          Navigator.pushNamed(context,  
'/Second');  
        },  
      ),  
    );  
  }  
}
```

```
class SecondPage extends StatelessWidget {  
  @override  
  Widget build(BuildContext context) {  
    // TODO: implement build  
    return Scaffold(  
      appBar: AppBar(  
        title: Text("Route -- SecondPage"),  
      ),  
      body: RaisedButton(  
        child: Text("Go back!"),  
        onPressed: () {  
          Navigator.pop(context);  
        },  
      ),  
    );  
  }  
}
```

```
    ),  
  );  
}  
}
```

Route -- 页面间的参数传递

前面讲了页面跳转，那么页面跳转的时候，想将数据传递给新页面，或者新页面关闭，要将数据传递给上一个页面，该如何实现呢？

本节就讲一下 Route 间的参数传递。

假设要传递如下的参数：

```
class PassArgumnets{  
  String content;  
  
  PassArgumnets(this.content);  
}
```

跳转页面时传递参数

从一个页面跳转到新的页面，有两种方法传递参数：

- 通过 Navigator.push() 或者 Navigator.pushNamed() 方法传递参数
- 通过 Widget 的构造函数传递参数

通过 Navigator.push() 或者 Navigator.pushNamed() 方法传递参数

1. Navigator.push()

前面讲了 `MaterialPageRoute` 的属性里有 `settings`，使用 `RouteSettings` 里的 `arguments` 来传递参数，`arguments` 是一个 `Object` 类型，可以传递简单类型，如：`String`、`List` 等，也可以自定义一个类，用来传递复杂类型，例如下面的例子，自定义了 `PassArgumnets` 类型用来传递参数：

```
Navigator.push(  
  context,  
  MaterialPageRoute(  
    builder: (context) => SecondPage(),  
    settings: RouteSettings(  
      arguments:  
        PassArgumnets('Data from  
FirstPage Navigator.push()'))),  
);
```

2. `Navigator.pushNamed()`

`Navigator.pushNamed()` 有一个可选参数 `arguments`，`arguments` 是一个 `Object` 类型，可以传递简单类型，如：`String`、`List` 等，也可以自定义一个类，用来传递复杂类型，例如下面的例子，自定义了 `PassArgumnets` 类型用来传递参数：

```
Navigator.pushNamed(context,  
  '/Second',arguments: PassArgumnets('Data from  
FirstPage Navigator.pushNamed()'));
```

3. 在新页面接受参数

接受参数使用 `ModalRoute.of` 方法：

```
final PassArgumnets passArgumnets
=ModalRoute.of(context).settings.arguments;
```

完整代码如下：

```
import 'package:flutter/material.dart';

void main() => runApp(MyApp());

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    // TODO: implement build
    return MaterialApp(
      title: "Flutter Demo",
      theme: ThemeData(
        primaryColor: Colors.blue,
      ),
      initialRoute: '/First',
      routes: {
        '/First': (context) => FirstPage(),
        "/Second": (context) => SecondPage()
      },
      home: FirstPage());
  }
}

class FirstPage extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    // TODO: implement build
    return Scaffold(
```

```

    appBar: AppBar(
      title: Text("Route -- FirstPage"),
    ),
    body: RaisedButton(
      child: Text("JUMP SecondRoute"),
      onPressed: () {
        // Navigator.push(
        //   context,
        //   MaterialPageRoute(
        //     builder: (context) =>
SecondPage(),
        //     settings: RouteSettings(
        //       arguments:
        //         PassArgumnets('Data
from FirstPage Navigator.push()'))),
        // );
        Navigator.pushNamed(context,
'/Second',arguments: PassArgumnets('Data from
FirstPage Navigator.pushNamed()'));
      },
    ),
  );
}
}

```

```

class SecondPage extends StatelessWidget {
  @override
  Widget build(BuildContext context) {

    final PassArgumnets passArgumnets
=ModalRoute.of(context).settings.arguments;

    print(passArgumnets.content);
  }
}

```

```

// TODO: implement build
return Scaffold(
  appBar: AppBar(
    title: Text("Route -- SecondPage"),
  ),
  body: RaisedButton(
    child: Text("Go back!"),
    onPressed: () {
      Navigator.pop(context);
    },
  ),
);
}

class PassArgumnets {
  String content;

  PassArgumnets(this.content);
}

```

通过 Widget 的构造函数传递参数

这个很好理解，直接看代码：

```

class SecondPage extends StatelessWidget {
  PassArgumnets passArgumnets;

  SecondPage(this.passArgumnets);

  ....
}

```

SecondPage 的构造函数里有所需要的参数，那么在跳转的时候可以这么写：

```
Navigator.push(  
    context,  
    MaterialPageRoute(  
        builder: (context) =>  
        SecondPage(PassArgumnets('Data from FirstPage  
Navigator.push()'))),
```

不建议使用这种方式。

页面关闭时返回数据给上一个页面

页面关闭时返回数据给上一个页面,需要用到 `Navigator.pop()`，因为 `pop()` 方法里有一个 `result` 的可选参数，只要给这个 `result` 赋值，就会把数据返回给上一个页面

1. 返回数据

将 SecondPage 里改造如下：

```
Navigator.pop(context,PassArgumnets('Return  
Data from SecondPage'));
```

2. 在上一个页面接受数据：

```
_jumpToSecondPage(BuildContext context) async
{
    var passArgumnets = await
Navigator.pushNamed(context, '/Second',
    arguments: PassArgumnets('Data from
FirstPage Navigator.pushNamed()'));
    print(passArgumnets.content);
}
```

接受的数据就是 Navigator.pushNamed() 返回的结果，为了不阻塞 UI，这里用了 async。

完整代码如下：

```
import 'package:flutter/material.dart';

void main() => runApp(MyApp());

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    // TODO: implement build
    return MaterialApp(
      title: "Flutter Demo",
      theme: ThemeData(
        primaryColor: Colors.blue,
      ),
      initialRoute: '/First',
      routes: {
        '/First': (context) => FirstPage(),
        "/Second": (context) => SecondPage()
      },
      home: FirstPage());
  }
}
```



```

    }
}

class FirstPage extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    // TODO: implement build
    return Scaffold(
      appBar: AppBar(
        title: Text("Route -- FirstPage"),
      ),
      body: RaisedButton(
        child: Text("JUMP SecondRoute"),
        onPressed: () {
          _jumpToSecondPage(context);
        },
      ),
    );
  }

  _jumpToSecondPage(BuildContext context) async {
    var passArgumnets = await
Navigator.pushNamed(context, '/Second',
    arguments: PassArgumnets('Data from
FirstPage Navigator.pushNamed()'));
    print(passArgumnets.content);
  }
}

class SecondPage extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    final PassArgumnets passArgumnets =

```

```

ModalRoute.of(context).settings.arguments;

    print(passArgumnets.content);
    // TODO: implement build
    return Scaffold(
      appBar: AppBar(
        title: Text("Route -- SecondPage"),
      ),
      body: RaisedButton(
        child: Text("Go back!"),
        onPressed: () {
          Navigator.pop(context,
PassArgumnets('Return Data from SecondPage'));
        },
      ),
    );
  }
}

class PassArgumnets {
  String content;

  PassArgumnets(this.content);
}

```