# 使用 Navigation 管理页面跳转

在今年的 IO 大会上，发布了一套叫 Android Jetpack 的程序库。Android Jetpack 里的组件大部分我们都接触过了，其中也有一些全新的组件，其中一个就是 Navigation。

# 1.使用

我们先来看看 Navigation 的实现过程。

## 1.1添加依赖

首先，需要使用 Android Studio 3.2 以上版本才能使用 Navigation。

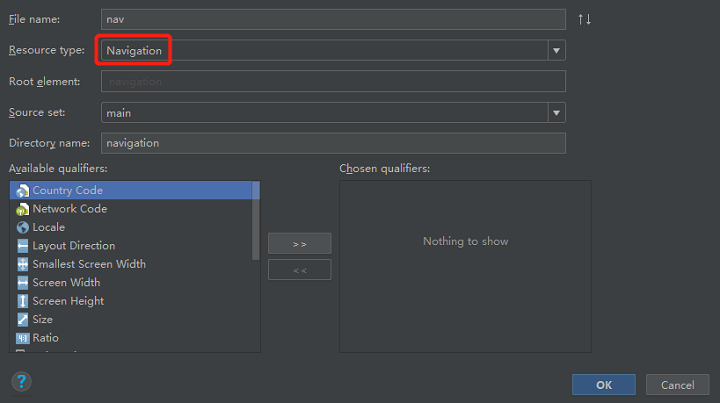
在 build.gradle 中添加依赖：

implementation "android.arch.navigation:navigation-fragment:$nav\_version"

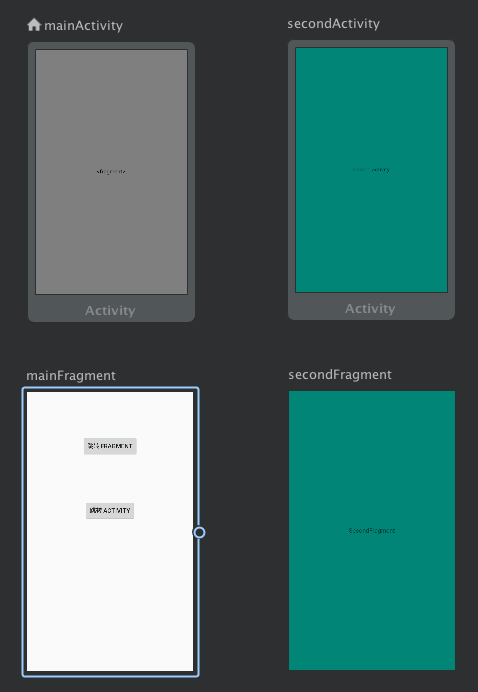
implementation "android.arch.navigation:navigation-ui:$nav\_version"

## 1.2创建 navigation xml 文件

使用 「Android Resource File」创建 xml 文件的时候，可以看到在类型里，多了一个 Navigation 的选项：



创建成功后，就来到了文章开头的那个一个可视化的操作界面。点击左上角的添加小图标，会出现 Activity 和 Fragment，我们这里添加两个 Activity 和两个 Fragment：

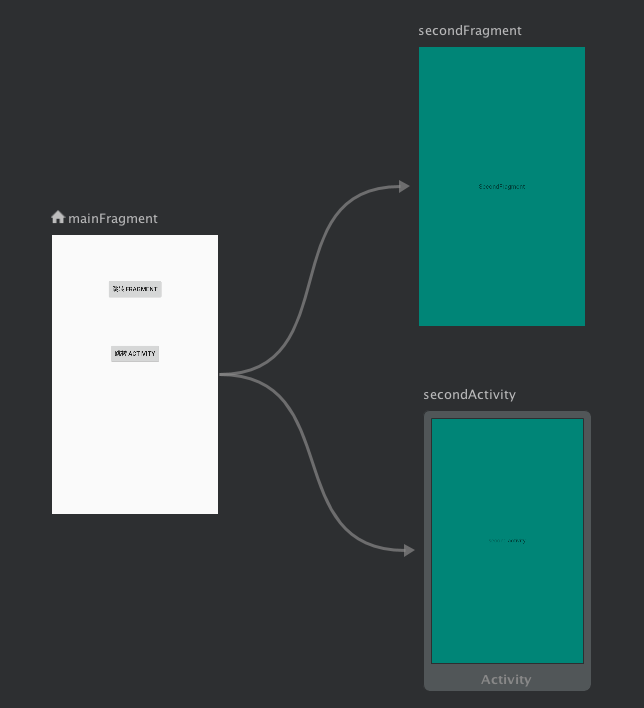


1.3配置 Action

Fragment 的右边有个小圆圈，点击并拖到另一个页面，这样我们就给这个 Fragment 添加了一个跳转行为，也就是 Action。

但是可以发现，Activity 的右边是没有这个小圆圈的，所以 Navigation 并不能处理从 Activity 发起的跳转。

左上角有个小房子的是显示的第一个页面，但由于 Activity 无法发起跳转，所以这里把 MainActivity 删除，把 MainFragment 作为主页面，并给它添加跳转到 SecondFragment 和 SecondActivity 的 Action：



自动生成的 xml 代码是这样的：

<?xml version="1.0" encoding="utf-8"?>

<navigation xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

xmlns:tools="http://schemas.android.com/tools"

app:startDestination="@id/mainFragment">

<fragment

android:id="@+id/mainFragment"

android:name="com.example.navigation.MainFragment"

android:label="fragment\_main"

tools:layout="@layout/fragment\_main">

<action

android:id="@+id/action\_mainFragment\_to\_secondFragment"

app:destination="@id/secondFragment"

app:enterAnim="@anim/slide\_in\_right" />

<action

android:id="@+id/action\_mainFragment\_to\_secondActivity"

app:destination="@id/secondActivity" />

</fragment>

<fragment

android:id="@+id/secondFragment"

android:name="com.example.navigation.SecondFragment"

android:label="fragment\_second"

tools:layout="@layout/fragment\_second" />

<activity

android:id="@+id/secondActivity"

android:name="com.example.navigation.SecondActivity"

android:label="activity\_second"

tools:layout="@layout/activity\_second" />

</navigation>

1.4布局中添加 Fragment

现在，我们第一个页面是 MainFragment，而 Fragment 需要 Activity 作为容器，修改 MainActivity 的布局：

<?xml version="1.0" encoding="utf-8"?>

<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent">

<fragment

android:id="@+id/fragment"

android:name="androidx.navigation.fragment.NavHostFragment"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

app:defaultNavHost="true"

app:navGraph="@navigation/nav" />

</FrameLayout>

其中有三个属性需要注意。使用 android:name 指定 Fragment 的类型为 NavHostFragment，使用 app:navGraph 指定 Navigation 文件。app:defaultNavHost="true" 的作用是，让 Navigation 处理返回事件，点返回按钮时并不是返回上一个 Activity，而是返回上一个「页面」，上一个「页面」有可能是 Activity，也可能是 Fragment。

至此，Navigation 的简单配置就算完成了，接下来看如何使用它。

## 1.5配置跳转

在 Navigation 里，页面的跳转是交给 NavController 来处理的，获取 NavController 的方法有这么三种：

NavHostFragment.findNavController(Fragment)

Navigation.findNavController(Activity, @IdRes int viewId)

Navigation.findNavController(View)

拿到后，通过 navigate 方法，通过传入 Action 的 id，实现跳转，比如：

NavHostFragment

.findNavController(this)

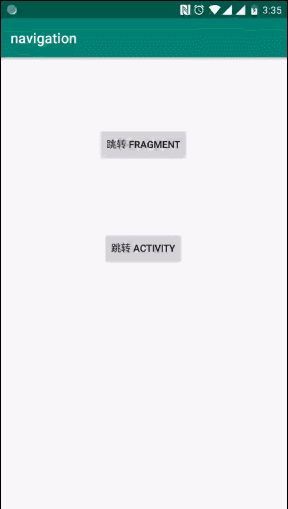
.navigate(R.id.action\_firstFragment\_to\_secondFragment)

NavHostFragment

.findNavController(this)

.navigate(R.id.action\_firstFragment\_to\_secondFragment,bundle)//传参ß

在简单配置了两个跳转后，看一下目前的效果：



## 1.6传参

页面的跳转少不了数据的传递，使用 Navigation，和我们原来的跳转一样，可以通过 Bundle 来传递参数：

val bundle = Bundle()

bundle.putString("name", "SouthernBox")

NavHostFragment

.findNavController(this)

.navigate(R.id.action\_firstFragment\_to\_secondFragment, bundle)

如果跳转到 Activity，可以从 intent.extras 获取到 bundle，如果是 Fragment，则从 arguments 获取到。

此外，还可以在 Navigation 的 xml 文件中配置传参，但这种方式目前支持的数据类型比较少，连 boolean 都不支持，而且我还碰到了 bug，所以目前不建议用。

**1.7转场动画**

如果需要自定义的页面转场动画，使用 Navigation 可以很方便的实现。

这里举个例子，比如我们需要一个从右向左切入的过场动画，先创建这个动画的 xml 文件：

<?xml version="1.0" encoding="utf-8"?>

<set xmlns:android="http://schemas.android.com/apk/res/android">

<translate

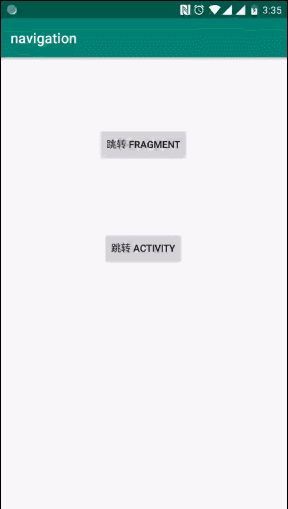
android:duration="600"

android:fromXDelta="100%"

android:toXDelta="0" />

</set>

然后我们回到 Navigation 的可视化编辑页面来，点击跳转的线，右边会出现过场动画的配置选项，将 xxxx 设为刚才创建的动画：



这么简单就搞定了，效果如下：

Navigation 的使用介绍就到这里。

# 3.思考

你可能已经明白，Navigation 主要是用来处理 Fragment 的跳转，所以说它并不是用来代替 startActivity，而是用来代替 FragmentTransaction 的相关操作。

在官方文档里，可以看到一个将传统跳转迁移到 Navigation 的建议。我简单理解为，将原本两个 Activity 之间的跳转，逐渐修改为使用一个 Activity 作为容器，用两个 Fragment 作为页面跳转。

看到这里，我联想到了在去年，Jake Wharton（目前在谷歌）有这么一个有争议的言论：

“一个 APP 只需要一个 Activity。”

在过去，要实现这种方式，就需要去解决复杂的 Fragment 堆栈处理，而且早期的 Fragment 坑比较多，处理不好容易出现页面穿透等问题。现在 Navigation 恰好解决了这些问题。

这一切联系起来，是不是能说明官方间接支持了「少用 Activity 多用 Fragment」的做法？你怎么看？

链接：https://juejin.im/post/5b1938585188257da1245be6