Android 部分 Fragment 篇

1.Fragment 为什么会被称为第五大组件？

Android 中的 4 大组件为：Activity，Broadcast，Service，ContentProvider，那么为什么 Fragment 可以称之为第 5 大组件呢？你可以这么回答：

因为 Fragment 有生命周期，使用频率不输于 4 大组件，可灵活加载到 Activity 中。然后谈论谈论 Fragment 如何加载到 Activity 中：

1.1 Fragment 加载到 Activity 的 2 种方式

Fragment 加载到 Activity 的方式分为静态加载和动态加载，下面我们看看这两种加载方式：

静态加载:直接在 Activity 布局文件中指定 Fragment。使用指定属性 name 即可，代码如下所示：

```
<fragment
    android:name="com.example.myfragment.MyFragment"
    android:id="@+id/myfragment_1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"/>
```

动态加载: 动态加载需要使用到 FragmentManager，这种动态加载的方式在开发中是非常常见的，示例代码如下：

```
FragmentTransaction fragmentTransaction = getSupportFragmentManager().beginTransaction();
//将 FragmentA 从容器中移除掉，减少内存的消耗
fragmentTransaction.remove(fragmentA);
fragmentTransaction.add(R.id.fragment_layout,new FragmentB());
fragmentTransaction.commit();
```

1.2 Fragment 与 ViewPager 的搭配使用

通常情况下我们开发应用最常见的使用情况是 TabLayout+ViewPager+Fragment 的使用方式，这就涉及到两个常用的适配器的使用，一个是 FragmentPagerAdapter，另外一个是 FragmentStatePagerAdapter,那么它们之间有什么区别呢？其实很简单，FragmentPagerAdapter 适用于页面较少的情况，而 FragmentStatePagerAdapter 适用于页面较多的情况。

2.Fragment 的生命周期

Fragment 的生命周期面试的时候概率还是蛮大的，不过问的时候并不是单纯说整个生命周期的回调函数等等，而是问些特殊情况下的 Fragment 的生命周期情况，为的是检验你的开发经验，不过不用方，看看笔者推荐的一个链接，你会搞懂各种情况下的 Fragment 的生命周期，这样面试就不用担心这种问题了：

Fragment

1.界面打开
onCreate() 方法执行！
onCreateView() 方法执行！
onActivityCreated() 方法执行！
onStart() 方法执行！
onResume() 方法执行！

2.按下主屏幕键/锁屏
onPause() 方法执行！
onStop() 方法执行！

3.重新打开
onStart() 方法执行！
onResume() 方法执行！

4.按下后退键
onPause() 方法执行！
onStop() 方法执行！
onDestroyView() 方法执行！
onDestroy() 方法执行！
onDetach() 方法执行！

Activity

```
1.   1. 打开应用
2.  onCreate() 方法执行！
3.  onStart() 方法执行！
4.  onResume() 方法执行！
5.
6.  2. 按下主屏幕键/锁屏
7.  onPause() 方法执行！
8.  onStop() 方法执行！
9.
10. 3. 重新打开应用
11. onRestart() 方法执行！
```

```
onStart() 方法执行!
onResume() 方法执行!

4. 按下后退键
onPause() 方法执行!
onStop() 方法执行!
onDestroy() 方法执行!
```

开启一个带 Fragment 的 Activity 的生命周期情况如下：

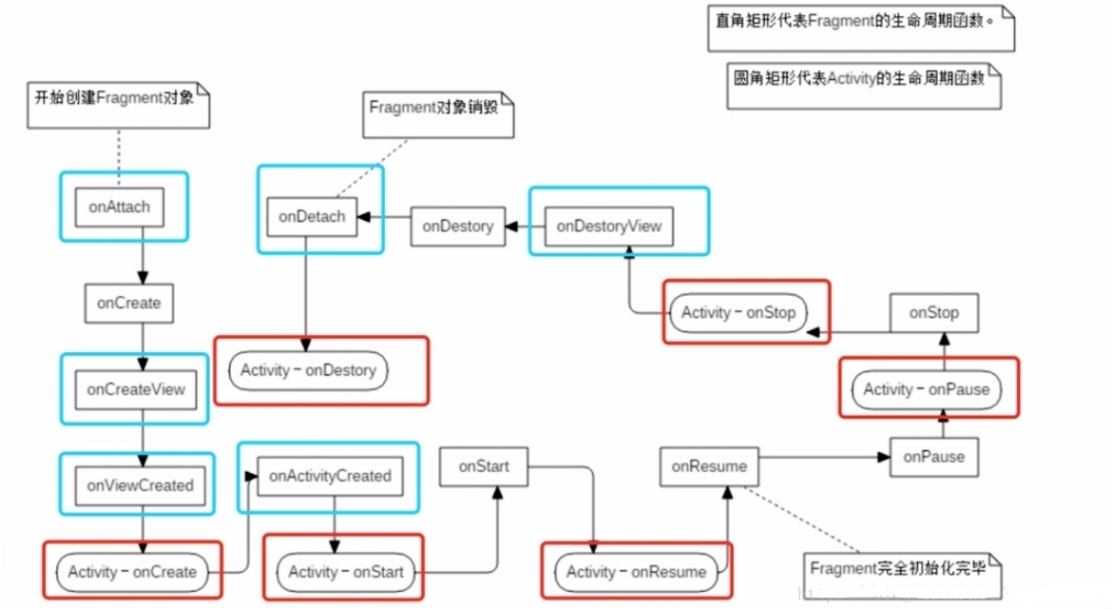
1.打开
Fragment onAttach()方法执行
Fragment onCreate() 方法执行!
Fragment onCreateView() 方法执行!
Fragment onViewCreated()方法执行
Activity onCreate() 方法执行!
Fragment onActivityCreated() 方法执行!
Activity onStart() 方法执行!
Fragment onStart() 方法执行!
Activity onResume() 方法执行!
Fragment onResume() 方法执行!

2.按下主屏幕键/锁屏
Fragment onPause() 方法执行!
Activity onPause() 方法执行!
Fragment onStop() 方法执行!
Activity onStop() 方法执行!

3.再次打开
Activity onRestart() 方法执行!
Activity onStart() 方法执行!
Fragment onStart() 方法执行!
Activity onResume() 方法执行!
Fragment onResume() 方法执行!

4.按下后退键
Fragment onPause() 方法执行!
Activity onPause() 方法执行!
Fragment onStop() 方法执行!
Activity onStop() 方法执行!
Fragment onDestroyView() 方法执行!
Fragment onDestroy() 方法执行!
Fragment onDetach() 方法执行!

Activity onDestroy() 方法执行！



3.Fragment 的通信

3.1 在 Fragment 中调用 Activity 中的方法

在 Fragment 中调用 Activity 的方法很简单，Fragment 有个 getActivity()的方法，比如，在 MainActivity 中的一个 Fragment 中获取 MainActivity 的引用，并调用 MainActivity 的某个方法 methodA()方法你可以这么写：

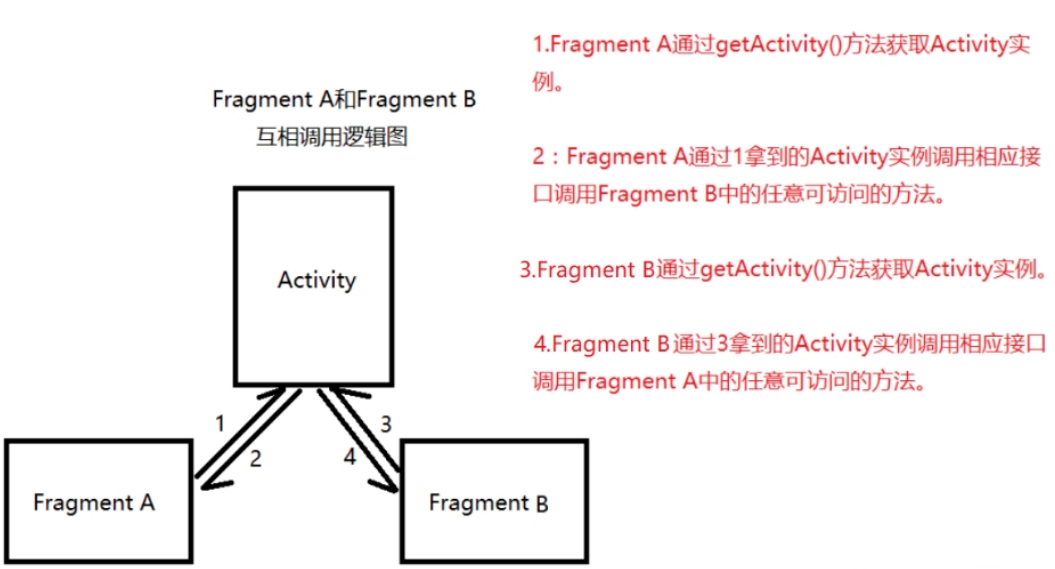
```
1.  MainActivity mainActivity = (MainActivity) getActivity();
2.  ainActivity.methodA();
```

3.2 在 Activity 中调用 Fragment 的方法

在 Activity 中调用 Fragment 中的方法是最简单的，我想这里我不用多说吧！直接接口回调即可调用 Fragment 的任何可访问的方法。

3.3 在 Fragment 中调用另外一个 Fragment 的方法

这个可就需要一定的思维性了，首先要想调用 Fragment A 的方法，除了这个 Fragment A 自身可以调用外，这个 Fragment A 所属的 Activity 也可以调用，要想另外一个 Fragment B 调用此 Fragment A 的方法，Fragment B 可以间接通过 Activity 来进行调用，也就是 3.1 和 3.2 的结合。看看下图你自然就会明白了：



4.Fragment 的切换方式

Fragment 的切换方式分为 add()&remove(),hide()&show(),detach()&attach(),也许你会问不是还有 replace()方法吗？其实 replace()其实是先调用了 remove()然后再调用 add()方法，所以不算那三种其实之一。那么这三种方式有什么区别呢？add()&remove()就是添加和移除，因此 replace()这个方法只是在上一个 Fragment 不再需要时采用的简便方法。而 hide()&show()则是指隐藏和显示，这种方式防止 Fragment 多次创建实例对象，所以正确的切换方式是 add()，切换时 hide()，add()另一个 Fragment；再次切换时，只需 hide()当前，show()另一个，这样就能做到多个 Fragment 切换不重新实例化。那么 detach()&attach()呢？使用 detach()会将 view 从 ViewTree 中删除,和 remove()不同,此时 Fragment 的状态依然保持着,在使用 attach()时会再次调用 onCreateView()来重绘视图,注意使用 detach()后 Fragment.isAdded()方法将返回 false,在使用 attach()还原 Fragment 后 isAdded()会依然返回 false(需要再次确认)执行 detach()和 replace()后要还原视图的话， 可以在相应的 Fragment 中保持相应的 view,并在 onCreateView()方法中通过 view 的 parent 的 removeView()方法将 view 和 parent 的关联删除后返回,这种方式极少使用，但是面试官文问的话，你如果答得很全，那么面试官会眼前一亮的感觉。