

Fragment:

All subclasses of Fragment must include a public no-argument constructor

所有的Fragment的子类都必须包括一个不含参数的构造函数。

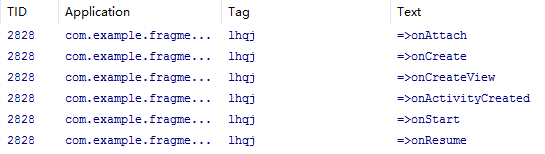
Interaction with fragments is done through [FragmentManager](http://android.xsoftlab.net/reference/android/app/FragmentManager.html), which can be obtained via [Activity.getFragmentManager()](http://android.xsoftlab.net/reference/android/app/Activity.html#getFragmentManager()) and [Fragment.getFragmentManager()](http://android.xsoftlab.net/reference/android/app/Fragment.html#getFragmentManager())

Fragment之间的交互都需要通过FragmentManager,其可以通过Activity.getFragmentManager()或者Fragment.getFragmentManager()获得。

Fragment的有自己的生命周期，但是都是依附于Activity的生命周期，并且Fragment的运行都是需要有Activity为管理者，它们不能单独进行使用。如果Activity停止工作，那么Fragment也会停止。

一、生命周期

（1）加载Fragment



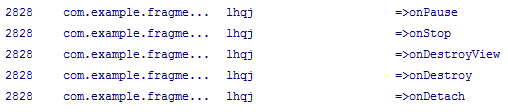
（2）点击home键



（3）重新进入程序



（4）点击back



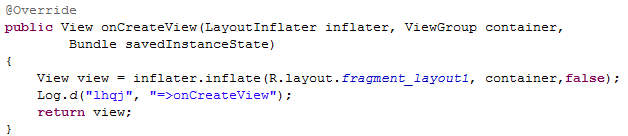
Fragment的生命周期与Activity周期之间的关系：

* onAttach方法：Fragment和Activity建立关联的时候调用。
* onCreateView方法：为Fragment加载布局时调用。
* onActivityCreated方法：当Activity中的onCreate方法执行完后调用。
* onDestroyView方法：Fragment中的布局被移除时调用。
* onDetach方法：Fragment和Activity解除关联的时候调用。

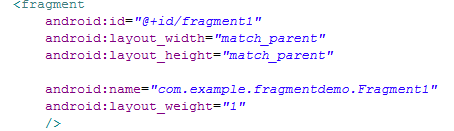
二、Fragment与Activity建立关系

1. 通过静态加载xml的方式

在xml文件中编写布局，创建对应的Fragment类，并继承Fragment，在onCreateView函数中加载xml布局文件。



最后再Activity的主布局中以组件的方式添加Fragment1，



2. 动态加载Fragment

根据屏幕的长宽，判断屏幕是横屏还是竖屏，以此来加载不同的Fragment，注意：要给未添加任何组件的主布局一个id，方便下面的动态加载。

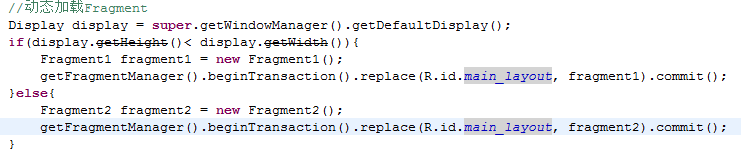
通过Display display = getWindowManager().getDefaultDisplay(); 获取屏幕的长宽。

（1）获取到FragmentManager，在Activity中可以直接通过getFragmentManager得到。

（2）开启一个事务，通过调用beginTransaction方法开启。

（3）向容器内加入Fragment，一般使用replace方法实现，需要传入容器的id和Fragment的实例。

（4）提交事务，调用**commit**方法提交。



当进行屏幕切换时，就会加载不同的fragment布局。

**对Fragment的主要的操作都是FragmentTransaction的方法**

FragmentTransaction transaction = fm.benginTransatcion();//开启一个事务

**transaction.add()**

往Activity中添加一个Fragment

**transaction.remove()**

从Activity中移除一个Fragment，如果被移除的Fragment没有添加到回退栈（回退栈后面会详细说），这个Fragment实例将会被销毁。

**transaction.replace()**

使用另一个Fragment替换当前的，实际上就是remove()然后add()的合体~

**transaction.hide()**

隐藏当前的Fragment，仅仅是设为不可见，并不会销毁

**transaction.show()**

显示之前隐藏的Fragment

**detach()**

会将view从UI中移除,和remove()不同,此时fragment的状态依然由FragmentManager维护。

**attach()**

重建view视图，附加到UI上并显示。

transatcion.commit()//提交一个事务

注意：常用Fragment的哥们，可能会经常遇到这样Activity状态不一致：State loss这样的错误。主要是因为：**commit方法一定要在Activity.onSaveInstance()之前调用**

三、Fragment之间的通信

在上述fragment的基本概述中有提到，fragment之间的通信都是通过activity进行管理的，所有他们之间的通信也是通过Activity进行的。

主要都是通过getActivity这个方法实现的。getActivity方法可以让Fragment获取到关联的Activity，然后再调用Activity的findViewById方法，就可以获取到和这个Activity关联的其它Fragment的视图了。

四、fragment与Activity通信

因为所有的Fragment都是依附于Activity的，所以通信起来并不复杂，大概归纳为：

（1）如果你Activity中包含自己管理的Fragment的引用，可以通过引用直接访问所有的Fragment的public方法

（2）如果Activity中未保存任何Fragment的引用，那么没关系，每个Fragment都有一个唯一的TAG或者ID,可以通过getFragmentManager.findFragmentByTag()或者findFragmentById()获得任何Fragment实例，然后进行操作。

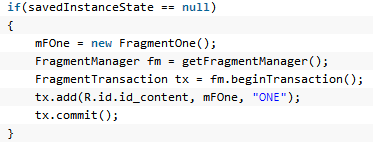
（3）在Fragment中可以通过getActivity得到当前绑定的Activity的实例，然后进行操作。

**注**：如果在Fragment中需要Context，可以通过调用getActivity(),如果该Context需要在Activity被销毁后还存在，则使用getActivity().getApplicationContext()。

五、存在的问题。

（1）当屏幕进行旋转时，Activity会被重新创建，这样回导致产生多个Fragment。

解决办法：通过检查onCreate的参数Bundle savedInstanceState就可以判断，当前是否发生Activity的重新创建：



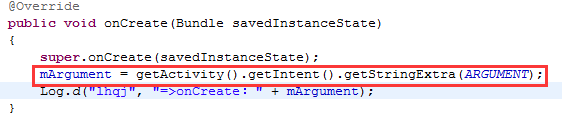
只有当savedInstanceState为空时才会去创建。

（2）还存在一个问题，就是重新绘制时，Fragment发生重建，原本的数据如何保持？

解决方法：其实和Activity类似，Fragment也有onSaveInstanceState的方法，在此方法中进行保存数据，然后在onCreate或者onCreateView或者onActivityCreated进行恢复都可以。

（3）某个按钮触发Activity跳转，需要通过Intent传递参数到目标Activity的Fragment中，那么此Fragment如何获取当前的Intent的值呢？

思路1：直接在Fragment的onCreate中，拿到宿主Activty，宿主Activity中肯定能通过getIntent拿到Intent，然后通过get方法，随意拿参数。



此方法的弊端在于，由于在使用Fragment时很大原因都是为了复用，但是如果这样写，就表明改Fragment和当前宿主Activity绑定在一起了。

思路2：在Fragment中添加一个返回值为当前Fragment对象的方法，如果外界某个Activity想把数据传递到改Fragment，则通过此方法传递进来，设置到bundle中，然后setArguments(bundle)，最后在onCreate中进行获取。这样就可以完成Fragment和Activity之间的解耦合。这里要注意的是：setArguments方法必须在fragment创建以后，添加给Activity前完成。

（4）两个fragment之间要有返回数据，即Fragment1跳转到Fragment2，在Fragment2中会产生一个返回数据，但是由于Fragment没有对应Intent的setResult方法，所有需要从Activity中接收返回结果，但是其自身无法产生返回结果，只有Activity拥有返回结果。

（5）PagerAdapter

FragmentPagerAdapter与FragmentStatePagerAdapter的区别：

主要区别就在与对于fragment是否销毁，下面细说：

FragmentPagerAdapter：对于不再需要的fragment，选择调用detach方法，仅销毁视图，并不会销毁fragment实例。

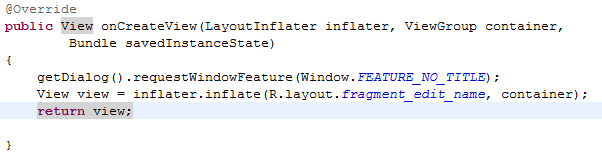
FragmentStatePagerAdapter：会销毁不再需要的fragment，当当前事务提交以后，会彻底的将fragmeng从当前Activity的FragmentManager中移除，state标明，销毁时，会将其onSaveInstanceState(Bundle outState)中的bundle信息保存下来，当用户切换回来，可以通过该bundle恢复生成新的fragment，也就是说，你可以在onSaveInstanceState(Bundle outState)方法中保存一些数据，在onCreate中进行恢复创建。

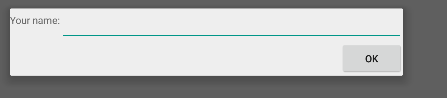
如上所说，使用FragmentStatePagerAdapter当然更省内存，但是销毁新建也是需要时间的。一般情况下，如果你是制作主页面，就3、4个Tab，那么可以选择使用FragmentPagerAdapter，如果你是用于ViewPager展示数量特别多的条目时，那么建议使用FragmentStatePagerAdapter。

六、[Android 官方推荐 : DialogFragment 创建对话框](http://blog.csdn.net/lmj623565791/article/details/37815413)

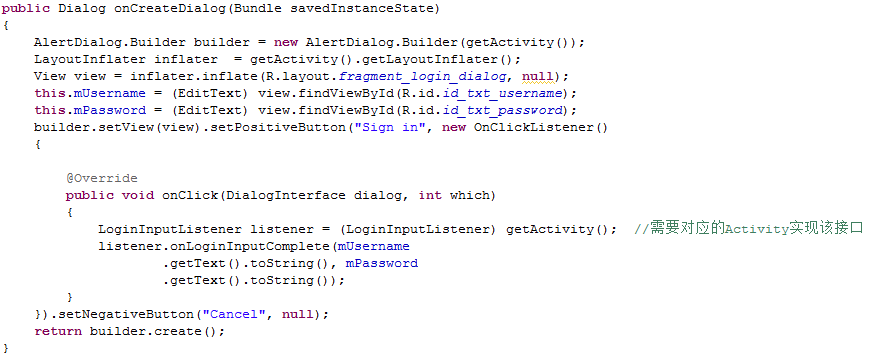
利用DiaglogFragment创建对话框有一下几种类型：

1. 按照自己的布局文件来创建对话框

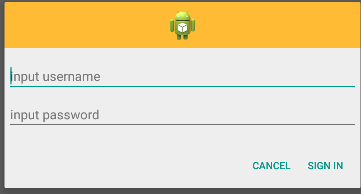




1. 重新onCreateDialog创建Dialog



利用AlertDialog创建对话框。



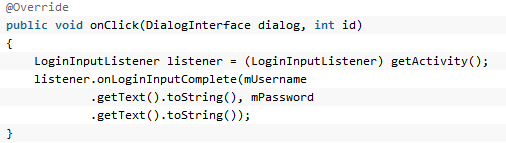
利用DialogFragment来设计一个登录界面的好处在于，当屏幕旋转时，在输入框中的输入数据不会消失掉，而传统的对话框会丢失掉这些数据。

1. 将对话框中的数据传递到Acticvity

在继承DiglogFragment的类中采用“fragment interface pattern”，意思就是添加一个回调接口。



在对话框中保存数据。



让Activity实现该接口的方法，获取数据。

