* 未看：

Adding items to the Action Bar

* 问题：

当fragment是通过layout.xml的方式加入，remove后，它依然显示在屏幕上

环境：4.2.2, galaxy nexus

* 参见：

my/fragment

my/fragmentTab

my/viewpager

* 是什么

提供一部分UI逻辑（主要为UI逻辑，非UI逻辑为UI逻辑服务），就像完整的UI逻辑的一部分碎片

each frgment defines

its own layout and

its own behavior

with its own lifecycle callbacks

设计它，是为了：

封装一些彼此相关的UI逻辑

使得这部分逻辑可以复用

一个例子：

you can reuse your fragments in different layout configurations to optimize the user experience based on the available screen spac

to continue with the news application example—the application can embed two fragments inActivity A, when running on a tablet-sized device. However, on a handset-sized screen, there's not enough room for both fragments, so Activity A includes only the fragment for the list of articles, and when the user selects an article, it starts Activity B, which includes the second fragment to read the article. Thus, the application supports both tablets and handsets by reusing fragments in different combinations

它要attach到一个activity上

时机：Once the activity reaches the resumed state, you can freely add and remove fragments to the activity

它被关联到一个container（该activity的viewgroup）上（fragment.view成为container的孩子，该container还可以有其他的孩子，还可能关联其他的fragment），且，一旦关联上，在整个生命周期内（create-destroy），不能改变这种关联

改变关联，修改fragment.container.parent

Vs activity：

Activity对象并不允许用户自行构建，且其生命周期由android来控制

Fragment对象要用户自行构建，其还定义了生命周期方法，其由android控制

vs 被封装的view提供器：

我们可以写自己的viewmaker，所以，为什么需要fragment呢

因为：

Android控制了Fragment的生命（状态转换，生命周期回调），帮你考虑更多的情况，如：

Activity的状态迁移，将直接影响，fragment的状态迁移

when the activity receives [onPause()](http://developer.android.com/reference/android/app/Activity.html#onPause()), each fragment in the activity receives [onPause()](http://developer.android.com/reference/android/app/Fragment.html#onPause()).

标识：

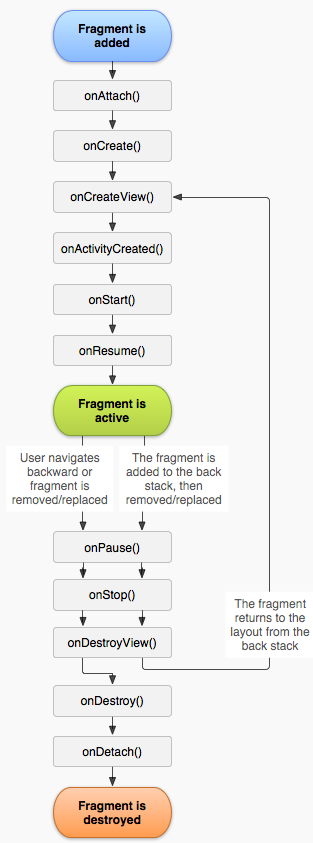
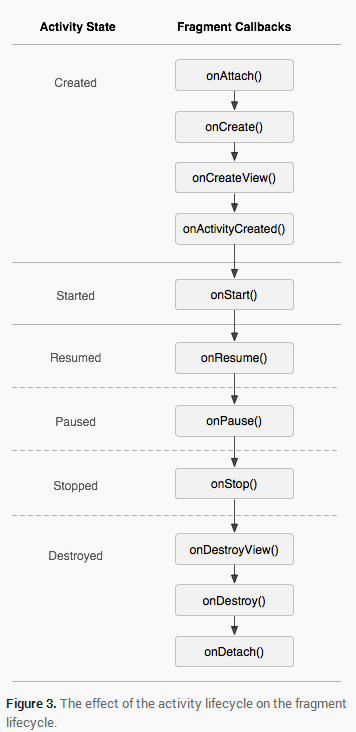
Ui类型的fragment可由id,tag标识

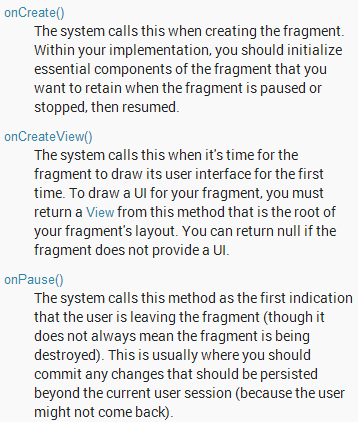
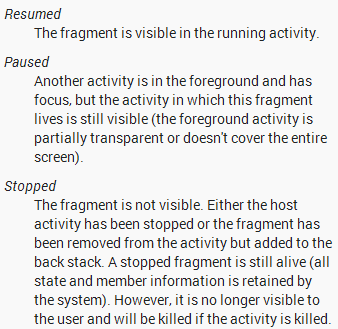
对于layout类型，你可以直接设置android:id

对于非layout类型，你可以通过container.id来找到fragment

非Ui类型的fragment可由tag标识

* 生命周期

当一个fragment attach到一个activity上

当activity进入stop状态，则，fragment进入stop状态（ondestroyview以及之后的回调 不会被调用）

当activity又恢复到resume状态，fragment进入start状态

onSaveInstanceState：

you can retain the state of a fragment using a [Bundle](http://developer.android.com/reference/android/os/Bundle.html), in case the activity's process is killed and you need to restore the fragment state when the activity is recreated. You can save the state during the fragment's[onSaveInstanceState()](http://developer.android.com/reference/android/app/Fragment.html#onSaveInstanceState(android.os.Bundle)) callback and restore it during either [onCreate()](http://developer.android.com/reference/android/app/Fragment.html#onCreate(android.os.Bundle)), [onCreateView()](http://developer.android.com/reference/android/app/Fragment.html#onCreateView(android.view.LayoutInflater, android.view.ViewGroup, android.os.Bundle)), or [onActivityCreated()](http://developer.android.com/reference/android/app/Fragment.html#onActivityCreated(android.os.Bundle)).

* Managing Fragments：
* Fragmentmanager
* fragmentTransacton

是什么：Each set of changes that you commit to the activity is called a transaction

能做什么：add, remove, or replace a fragment, .etc.

--

Add/remove vs attach/detach：

* Add()

当你要添加一个fragment时，你需要使用add来表示：该fragment要被attach到activity上

虽然fragment对象由用户构建，但是标准的生命周期回调由android控制。此时，oncreate…会被调用，就好像，该fragment是刚出生的一样

* Remove()

当你认为activity的那个fragment不需要再存在了，你会调用remove，fragment会被android认为是“垃圾”，并试图“回收”（ondestroyxxx），（但是，这仅是概念上给人的感觉，该fragment对象还依然存在，并未被当作垃圾回收，因为fragment对象是由我们手动产生，而不是交由android自动）

之后，android将不再控制该fragment的生命（任何回调）

* Detach()

fragment要被attach到一个activity上，那么detach意味着，该fragment不再“attach”到activity上，那么意味着什么：fragment定义的view将不再被认为有用，那么 构建view所涉及的资源 将被释放（ondestroyview）

不要被搞乱了：

真实的detach为：fragment脱离了activity，即是：不再有用，要被销毁

但是，android并不认为detach()之后，fragment要成为垃圾，要被destroy（也意味着：Android依然会控制该fragment的生命）

所以，只能理解，detach()仅仅是为了表达：fragment的view应该被回收（这在逻辑上，也是通的）

* Attach()

对比于detach()，attach()意味着：fragment的view再次有用：需要再次构建（oncreateview），并呈现出来

--

Backstack：

You can save each transaction to a back stack managed by the activity

Vs activity

An activity is placed into a back stack of activities that's managed by the system when it's stopped, by default (so that the user can navigate back to it with the Back button, as discussed in [Tasks and Back Stack](http://developer.android.com/guide/components/tasks-and-back-stack.html)). However, a fragment is placed into a back stack managed by the host activity only when you explicitly request that the instance be saved by calling [addToBackStack()](http://developer.android.com/reference/android/app/FragmentTransaction.html#addToBackStack(java.lang.String)) during a transaction that removes the fragment.

猜想：一个transact被放入到backstack，那么点击back时，就让transact做一个相反的操作，如：之前做的是detach，那么back时，做的就是attach

* 通信：
* Fragment和activity间的通信

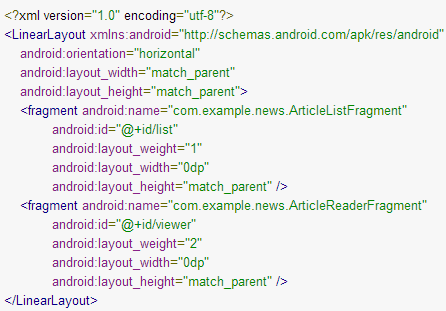
Fragment.getactivity()得到activity

Fragmentmanager.findfragmentby…()得到fragment

* Fragment之间

请使用设计模式中的 观察者模式（设置好依赖关系）

* 样式：
* 包含Ui的fragment



The android:name attribute in the <fragment> specifies the [Fragment](http://developer.android.com/reference/android/app/Fragment.html) class to instantiate in the layout.

When the system creates this activity layout, it instantiates each fragment specified in the layout and calls the [onCreateView()](http://developer.android.com/reference/android/app/Fragment.html#onCreateView(android.view.LayoutInflater, android.view.ViewGroup, android.os.Bundle)) method for each one, to retrieve each fragment's layout.

The system inserts the [View](http://developer.android.com/reference/android/view/View.html)returned by the fragment directly in place of the <fragment> element.

* 无UI的fragment

you can also use a fragment to provide a background behavior for the activity without presenting additional UI.