Fragment的使用：

链接地址：<http://developer.android.com/guide/components/fragments.html#Transactions>

You can insert a fragment into your activity layout by declaring the fragment in the activity's layout file, as a <fragment> element, or from your application code by adding it to an existing [ViewGroup](http://developer.android.com/reference/android/view/ViewGroup.html). However, a fragment is not required to be a part of the activity layout; you may also use a fragment without its own UI as an invisible worker for the activity.

ViewGroup ：A ViewGroup is a special view that can contain other views (called children.) The view group is the base class for layouts and views containers.

**Fragment使用时问题：**

链接地址：<http://toughcoder.net/blog/2015/04/30/android-fragment-the-bad-parts/>

fragment 实例：<http://blog.csdn.net/guolin_blog/article/details/13171191>



[onCreate()](http://developer.android.com/reference/android/app/Fragment.html" \l "onCreate(android.os.Bundle))

The system calls this when creating the fragment. Within your implementation, you should initialize（初始化） essential components（组件） of the fragment that you want to retain （保留）when the fragment is paused or stopped, then resumed（恢复）.

[onCreateView()](http://developer.android.com/reference/android/app/Fragment.html" \l "onCreateView(android.view.LayoutInflater, android.view.ViewGroup, android.os.Bundle))

The system calls this when it's time for the fragment to draw its user interface for the first time. To draw a UI for your fragment, you must return a [View](http://developer.android.com/reference/android/view/View.html) from this method that is the root of your fragment's layout. You can return null if the fragment does not provide a UI.

Note

If your fragment is a subclass（子类） of [ListFragment](http://developer.android.com/reference/android/app/ListFragment.html), the default implementation returns a [ListView](http://developer.android.com/reference/android/widget/ListView.html) from[onCreateView()](http://developer.android.com/reference/android/app/Fragment.html" \l "onCreateView(android.view.LayoutInflater, android.view.ViewGroup, android.os.Bundle)), so you don't need to implement it.

[onPause()](http://developer.android.com/reference/android/app/Activity.html" \l "onPause())

The system calls this method as the first indication（指示） that the user is leaving the fragment (though it does not always mean the fragment is being destroyed). This is usually where you should commit any changes that should be persisted beyond the current user session (because the user might not come back).

Most applications should implement at least these three methods for every fragment

To return a layout from [onCreateView()](http://developer.android.com/reference/android/app/Fragment.html" \l "onCreateView(android.view.LayoutInflater, android.view.ViewGroup, android.os.Bundle)), you can inflate it from a [layout resource](http://developer.android.com/guide/topics/resources/layout-resource.html) defined in XML. To help you do so, [onCreateView()](http://developer.android.com/reference/android/app/Fragment.html" \l "onCreateView(android.view.LayoutInflater, android.view.ViewGroup, android.os.Bundle)) provides a [LayoutInflater](http://developer.android.com/reference/android/view/LayoutInflater.html) object.

return inflater.inflate(R.layout.example\_fragment, container, false);

The container parameter passed to [onCreateView()](http://developer.android.com/reference/android/app/Fragment.html" \l "onCreateView(android.view.LayoutInflater, android.view.ViewGroup, android.os.Bundle)) is the parent [ViewGroup](http://developer.android.com/reference/android/view/ViewGroup.html) (from the activity's layout) in which your fragment layout will be inserted.

### **Adding a fragment to an activity**

Declare the fragment inside the activity's layout file.

<fragment android:name="com.example.news.ArticleListFragment"  
            android:id="@+id/list"  
            android:layout\_weight="1"  
            android:layout\_width="0dp"  
            android:layout\_height="match\_parent" />

The android:name attribute in the <fragment> specifies（指定） the [Fragment](http://developer.android.com/reference/android/app/Fragment.html) class to instantiate（实例化） 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" \l "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.

Note:

Each fragment requires a unique identifier that the system can use to restore the fragment if the activity is restarted

There are three ways to provide an ID for a fragment:

Supply the android:id attribute with a unique ID.

Supply the android:tag attribute with a unique string.

If you provide neither of the previous two, the system uses the ID of the container view.

Or, programmatically （以编程方式）add the fragment to an existing [ViewGroup](http://developer.android.com/reference/android/view/ViewGroup.html).

At any time while your activity is running, you can add fragments to your activity layout. You simply need to specify a [ViewGroup](http://developer.android.com/reference/android/view/ViewGroup.html) in which to place the fragment.

To make fragment transactions in your activity (such as add, remove, or replace a fragment), you must use APIs from [FragmentTransaction](http://developer.android.com/reference/android/app/FragmentTransaction.html). You can get an instance of [FragmentTransaction](http://developer.android.com/reference/android/app/FragmentTransaction.html) from your [Activity](http://developer.android.com/reference/android/app/Activity.html)like this:

FragmentManager fragmentManager = [getFragmentManager()](http://developer.android.com/reference/android/app/Activity.html" \l "getFragmentManager());  
FragmentTransaction fragmentTransaction = fragmentManager.[beginTransaction()](http://developer.android.com/reference/android/app/FragmentManager.html" \l "beginTransaction());

You can then add a fragment using the [add()](http://developer.android.com/reference/android/app/FragmentTransaction.html" \l "add(int, android.app.Fragment)) method, specifying the fragment to add and the view in which to insert it. For example:

ExampleFragment fragment = new ExampleFragment();  
fragmentTransaction.add(R.id.fragment\_container, fragment);  
fragmentTransaction.commit();

Each transaction is a set of changes that you want to perform at the same time. You can set up all the changes you want to perform for a given transaction using methods such as [add()](http://developer.android.com/reference/android/app/FragmentTransaction.html" \l "add(android.app.Fragment, java.lang.String)), [remove()](http://developer.android.com/reference/android/app/FragmentTransaction.html" \l "remove(android.app.Fragment)), and [replace()](http://developer.android.com/reference/android/app/FragmentTransaction.html" \l "replace(int, android.app.Fragment)). Then, to apply the transaction to the activity, you must call [commit()](http://developer.android.com/reference/android/app/FragmentTransaction.html" \l "commit()).

 however, you might want to call [addToBackStack()](http://developer.android.com/reference/android/app/FragmentTransaction.html" \l "addToBackStack(java.lang.String)), in order to add the transaction to a back stack of fragment transactions. This back stack is managed by the activity and allows the user to return to the previous fragment state, by pressing the Back button.

### **Communicating with the Activity**

Specifically, the fragment can access（访问） the [Activity](http://developer.android.com/reference/android/app/Activity.html) instance with [getActivity()](http://developer.android.com/reference/android/app/Fragment.html" \l "getActivity()) and easily perform（执行） tasks（任务） such as find a view in the activity layout:

View listView = [getActivity()](http://developer.android.com/reference/android/app/Fragment.html" \l "getActivity()).[findViewById](http://developer.android.com/reference/android/app/Activity.html" \l "findViewById(int))(R.id.list);

Likewise, your activity can call methods in the fragment by acquiring（获取） a reference（参考） to the [Fragment](http://developer.android.com/reference/android/app/Fragment.html) from[FragmentManager](http://developer.android.com/reference/android/app/FragmentManager.html), using [findFragmentById()](http://developer.android.com/reference/android/app/FragmentManager.html" \l "findFragmentById(int)) or [findFragmentByTag()](http://developer.android.com/reference/android/app/FragmentManager.html" \l "findFragmentByTag(java.lang.String)). For example:

ExampleFragment fragment = (ExampleFragment) ExampleFragment fragment = (ExampleFragment) getFragmentManager().findFragmentById(R.id.example\_fragment);

### **Creating event callbacks to the activity**

In some cases, you might need a fragment to share events with the activity. A good way to do that is to define a callback interface inside the fragment and require that the host activity implement it.  When the activity receives a callback through the interface, it can share the information with other fragments in the layout as necessary.