

Navigation with Back-stack in Android App Development

Android Fragment Backstack.

**Fragments** are useful when we want to **support multiple screen size**. To manage**fragments** we need a FragmentManager that help us to handle **trasaction** between fragments. **With transaction we mean a sequence of steps to add, replace or remove fragments**.

To load Fragment dynamic way

We have to use FrameLayout. With FrameLayout we can handle fragments as we need at runtime, but to do it we need a manager, in other words a component that can handle fragments.This is FragmentManager.

This component can add, replace and remove fragments at runtime.

[Link to Example With Navigation Drawer](https://drive.google.com/file/d/0B-p7lFlJim9YMThuaGVVdWdqa0k/edit?usp=sharing)

[Calander Tutorial](https://drive.google.com/file/d/0B-p7lFlJim9YSmdrRW5udnlEcHc/edit?usp=sharing)

<android.support.v4.widget.DrawerLayout

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

android:id=*"@+id/drawer\_layout"*

android:layout\_width=*"match\_parent"*

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

<!-- Displaying Fragments -->

<FrameLayout

android:id=*"@+id/frame\_container"*

android:layout\_width=*"match\_parent"*

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

/>

<!-- Displaying Drawer -->

<ListView

android:id=*"@+id/list\_slidermenu"*

android:layout\_width=*"240dp"*

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

android:layout\_gravity=*"start"*

android:choiceMode=*"singleChoice"*

android:divider=*"@color/list\_divider"*

android:dividerHeight=*"1dp"*

android:listSelector=*"@drawable/drawer\_list\_selection"*

android:background=*"@color/list\_background"*/>

</android.support.v4.widget.DrawerLayout>

## FragmentManager

As we said before FragmentManager is the key component. Using FragmentManager we can discover (**find**) fragment inside our layout using findFragmentById or findFragmentByTag.While the first method is very simple and we use the common android id to discover the component, the second method (that uses tag) is unsual. A tag in a fragment is simply a “name” we give to the fragment so that we can find it later using that name.

All the operation that are made by the FragmentManager happens inside a “transaction” like in a database operation. First, we can get the FragmentManger using the Activity method getFragmentManager(). Once we have the reference to this component we have to start the transaction in this way:

// get referance

FragmentManager fragmentManager1 = getFragmentManager();

// begin transaction

fragmentManager1.beginTransaction();

// Transaction commint

fragmentManager1.commit();

At the end when we finished and we are ready to show our fragment we have to call the commit method that marks the end of the transaction.

Fragment fragment1 = **new** Fragment1Nasdaq();

FragmentManager fragmentManager1 = getFragmentManager();

fragmentManager1.beginTransaction().add(R.id.*frame\_container*, fragment1).commit();

We can perform following operations:

* add a new fragment
* replace an existing fragment
* remove a fragment

## Fragment Backstack

The back button acts at activity level not at the fragment level. So our activity is the same while we replace fragments as the user interacts with the app. In this way when we tap on the back button we select the first activity on the activity stack, in our case the home. We don’t want this behaviour but we want that when we click on the back button we go back in the fragments stack. We can achieve it adding the fragment to the backstack. We do it in this way:

FragmentManager fragmentManager = getFragmentManager();

fragmentManager.beginTransaction().replace(R.id.*frame\_container*, fragment).addToBackStack("").commit();

We use the addToBackStack method of the FragmentTrasaction and we add every fragment to the backstack. Call [addToBackStack()](http://developer.android.com/reference/android/app/FragmentTransaction.html#addToBackStack(java.lang.String)) before you commit the transaction. In this way when we tap on the back button we have the correct behaviour.

When there are [FragmentTransaction](http://developer.android.com/reference/android/app/FragmentTransaction.html) objects on the back stack and the user presses the Back button the[FragmentManager](http://developer.android.com/reference/android/app/FragmentManager.html) pops the most recent transaction off the back stack and performs the reverse action(such as removing a fragment if the transaction added it).

[Check Example of Fragment Back Stack](https://docs.google.com/file/d/0B-p7lFlJim9YMThuaGVVdWdqa0k/edit)