**PRACTICAL # 08**

**OBJECT:**

Working with Fragments

**THEORY:**

A Fragment is a modular section of an activity with its own user interface. It has its own lifecycle, receives its own events, and it can be added or removed while the activity is running. You can combine multiple fragments in a single activity to build a multi-pane UI and reuse a fragment in multiple activities. A fragment is hosted in an activity. Fragment's lifecycle depends on host activity's lifecycle.

There is more space on large screens to combine and interchange UI components. Fragments allow such designs without need to manage complex changes to the view hierarchy as shown in figure 1. By dividing the layout of an activity into fragments, you can modify the activity's appearance at runtime and preserve those changes in a back stack.



Figure 1: Fragments on Tablet and Phone screen

For example, a news application can use one fragment to show a list of articles on the left and another fragment to display an article on the right.

We can create Fragments by extending Fragment class or by inserting a Fragment into our Activity layout by declaring the Fragment in the activity’s layout file, as a <fragment> element. We can manipulate each Fragment independently, such as add or remove them.

While performing Fragment Transaction we can add a Fragment into back stack managed by the Activity. Back stack allows to reverse a Fragment transaction on pressing Back button of device. For Example if we replace a Fragment and add it in back stack then on pressing the Back button on device it display the previous Fragment.

**Creating a Fragment**

To create a fragment, create a subclass of Fragment. The Fragment class has callback methods similar to an activity. To convert an existing application to use fragments, you might simply move code from your

activity's callback methods into respective callback methods of fragment.

**Important lifecycle methods:**

onCreate()

Called when creating the fragment. Here initialize essential components of the fragment that you want to retain when the fragment is paused or stopped, then resumed.

onCreateView()

Called when the fragment draws its user interface for the first time. Return a View from this method that is the root of your fragment's layout. Return null if the fragment does not provide a UI.

onPause()

Called when user is leaving the fragment. Commit any changes that should be persisted beyond the current user session.

**Adding user interface:**

To provide a layout for fragment, implement onCreateView() callback method. Return a View that is the root of your fragment's layout, using inflate from an XML layout resource. Example:

*public class FragmentA extends Fragment {*

*@Override*

*public View onCreateView(LayoutInflater inflater, @Nullable*

*ViewGroup container, @Nullable Bundle savedInstanceState) {*

*return inflater.inflate(R.layout.fragment\_a, container, false);*

*}*

*}*

The container parameter passed to onCreateView() is the parent ViewGroup (from the activity's layout) where fragment layout is inserted. The savedInstanceState is a Bundle that provides data about the previous instance of the fragment, if the fragment is being resumed.

The inflate() method takes three arguments:

1. Resource ID: of the layout to inflate.

2. ViewGroup: the parent of the inflated layout.

3. A boolean: indicating whether the inflated layout should be attached to the ViewGroup during inflation. (Here this is false because the system is already inserting the inflated layout into the container—passing true would create a redundant view group in the final layout.)

**Adding fragment to an activity**

There are two ways to add a fragment to activity layout:

1. Declare fragment in the activity's layout:

Here is the layout file example for an activity with two fragments:

*<?xml version="1.0" encoding="utf-8"?>*

*<LinearLayout*

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

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

*xmlns:tools="http://schemas.android.com/tools"*

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

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

*tools:context=".MainActivity">*

*<fragment*

*android:id="@+id/fragmentA"*

*android:name="com.example.shan.fragdemo.FragmentA"*

*android:layout\_width="0dp"*

*android:layout\_weight="1"*

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

*/>*

*<fragment*

*android:id="@+id/fragmentB"*

*android:name="com.example.shan.fragdemo.FragmentB"*

*android:layout\_width="0dp"*

*android:layout\_weight="1"*

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

*/>*

*</LinearLayout>*

The android:name attribute in the <fragment> specifies the Fragment 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() method for each one. The system inserts the View returned by the fragment in place of the <fragment> element.

You will also need corresponding Fragment classes and their layouts.

FragmentA class:

public class FragmentA extends Fragment {

@Override

public void onCreate(@Nullable Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

}

@Override

public View onCreateView(@NonNull LayoutInflater inflater,

@Nullable ViewGroup container, @Nullable Bundle savedInstanceState) {

return inflater.inflate(R.layout.fragment\_a, container, false);

}

}

FragmentB class:

*public class FragmentB extends Fragment {*

*@Override*

*public void onCreate(@Nullable Bundle savedInstanceState) {*

*super.onCreate(savedInstanceState);*

*}*

*@Nullable*

*@Override*

*public View onCreateView(@NonNull LayoutInflater inflater,*

*@Nullable ViewGroup container, @Nullable Bundle savedInstanceState) {*

*return inflater.inflate(R.layout.fragment\_b, container, false);*

*}*

*}*

FragmentA layout fragment\_a.xml:

*<?xml version="1.0" encoding="utf-8"?>*

*<FrameLayout*

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

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

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

*android:background="#FF0022">*

*<TextView*

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

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

*android:text="Fragment List"/>*

*</FrameLayout>*

FragmentB layout fragment\_b.xml”

*<?xml version="1.0" encoding="utf-8"?>*

*<FrameLayout*

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

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

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

*android:background="#00AA22">*

*<TextView*

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

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

*android:text="Fragment Detail"/>*

*</FrameLayout>*

**ACTIVITIES**

**Activity 1**

Create a dictionary that uses two fragments. One fragment shows list of words, the other shows meaning of a word selected from the list. If the phone is portrait mode, only list fragment should be visible. When the phone goes in landscape mode, both the fragments should be visible.

**REVIEW QUESTIONS**

1. What is a fragment?
2. Are fragment lifecycle events dependent on activity lifecycle events ?
3. How do you add a layout to fragment?
4. How is a fragment attached to an activity?