University of Stirling
Computing Science
Mobile App Development

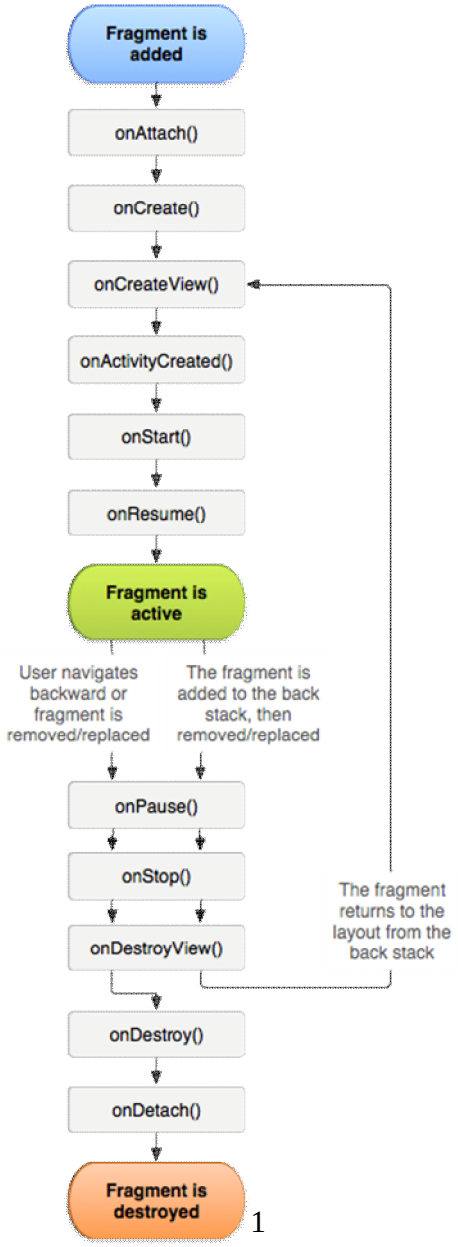
Android Project: Fragments

Checkpoint at the End

In this session we will start the project off and focus on Android Fragments. The aim of this lab is to create an Android application which has two tabs, and two fragments.

Fragments

Fragments allow for a more flexibile UI. By dividing the layout of an activity into fragments, you can modify the activity's appearance at runtime. For example one fragment will show input controls for users to select units, whereas the other will show a input panel for numeric user input. Users will be able to toggle between the two fragments by using Tabs in the application.



Each fragment has its own set of lifecycle callback methods and handle their own user input events. You can see them a bit like a subactivity. Fragments define their own layout and have their own lifecycle (albeit linked to the one of the activity). Fragments can belong to different activities. For instance on a tablet, two fragments belonging to an activity can be shown by the same activity, whereas on a phone, one fragment is shown by one activity and the second fragment by a second activity.

To create a fragment, you create a subclass of Fragment. The Fragment class has code that looks a lot like an Activity. You should implement the following methods for a fragment:
onCreate() The system calls this when creating the fragment.
onCreateView() 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 from this method that is the root of your fragment's layout.
onPause() The system calls this method as the first indication that the user is leaving the fragment

Let's start the project. Create a new Android Studio Project. This time, use the AppCompatActivity baseclass (i.e. do not untick the option in the New Project Dialogue). AppCompatActivity supports a number of useful libraries for this project. It inherits from Activity but adds other support classes/libraries.

In the Gradle file for the module:app change the compileSdkVersion and targetSDKVersion to 27 (or whatever SDK version you have installed on your machine. We will start with the layout. As we will use some newer components, you will need to add a dependency to the Gradle script for your module:

```
implementation 'com.android.support:design:27.1.0'
```

Again, your version on your machine may differ. In any case you will need to have installed "Android Support Repository" (in the SDK Manager under SDK Tools). You can make Android Studio to select the version by changing this to:

```
implementation 'com.android.support:appcompat-v7:+'
```

Then in the main layout we will add three components: a Toolbar, TabLayout with tabs, and a ViewPager. The toolbar will eventually host a menu. The tabs will be used to toggle between the two fragments of the app, and the pager will host the fragement output.

Add a Toolbar, TabLayout (don't confuse with TableLayout!) and ViewPager element (all in Containers) to your layout (The TextView which is there by default is best removed). As before, this step is quite fiddly. Don't worry about the error message about missing constraints. We will add these in the next step. We also do not need the 3 TabItems automatically generated with the TabLayout. You can delete them. Try and arrange these components starting at the top of the screen (using the grid). Now we can edit the look and feel of these components as follows:

Toolbar:

```
<android.support.v7.widget.Toolbar
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:id="@+id/toolbar"
    android:layout_alignParentTop="true"
    android:layout_alignParentStart="true"
    android:background="@color/colorPrimary"
    android:elevation="6dp" />
```

TabLayout:

```
<android.support.design.widget.TabLayout
    android:id="@+id/tab_layout"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_below="@+id/toolbar"
    android:background="@color/colorPrimary"
    android:elevation="6dp"
    android:minHeight="?attr/actionBarSize"
    android:theme="@style/ThemeOverlay.AppCompat.Dark.ActionBar"
    android:layout_alignParentStart="true" />
```

ViewPager:

```
<android.support.v4.view.ViewPager
    android:layout_width="match_parent"
    android:layout_height="fill_parent"
    android:id="@+id/pager"
    android:layout_below="@+id/tab_layout"
    android:layout_alignParentStart="true" />
```

Next we need a layout for each of the two fragments. Create two new layouts named page1_fragment and page2_fragment (or similar). Besides the default layout format (it might suggest ConstraintLayout) add a textView element with a placeholder text. You do not need the absolute coordinates added. Remove these. So your TextView elements should look similar to:

```
<TextView
    android:id="@+id/textView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Fragment 1" />
```

Finally, as we have defined our own Toolbar, in the styles.xml file (under values) change the style of the app to

```
<style name="AppTheme" parent="Theme.AppCompat.Light.NoActionBar">
```

This defines the required layout for the basic app skeleton. Next we move to the code.

Firstly, lets create a class for the two fragments. The class for each Fragement will extend the base class Fragement. Within these classes, you will need to overload the standard method onCreateView(). This gets executed when the Fragement comes into view. The method should load the layout for the fragement and return the View. Below is sample code for such a class:

```
public class Page1Fragment extends Fragment {

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle
savedInstanceState) {
        return inflater.inflate(R.layout.page1_fragment, container, false);
    }
}
```

Make sure you import `import android.support.v4.app.Fragment;` for the Fragment class. Android Studio will suggest some other imports as well (Bundle, View, ViewGroup and LayoutInflater).

Next, you will need to create a class for the PagerAdaptor http://developer.android.com/reference/android/support/v4/app/FragmentStatePagerAdapter.html. This class will manage the two fragments and their status. Below is some sample code for the PageAdaptor. Besides the constructor, there is a method getItem() which returns an instance of the right fragement based on an integer input indicating the tab position. Similarly, there is a method which returns the number of fragements managed by this class. The code requires some three imports for classes in the Support V4 package.

```
class PagerAdapter extends FragmentStatePagerAdapter {

    private final int mNumOfTabs;

    public PagerAdapter(FragmentManager fragmentManager, int numOfTabs) {
        super(fragmentManager);
        this.mNumOfTabs = numOfTabs;
    }

    @Override
    public Fragment getItem(int position) {
        switch (position) {
            case 0: return new Page1Fragment();
            case 1: return new Page2Fragment();
            default: return null;
        }
    }

    @Override
    public int getCount() {
        return mNumOfTabs;
    }
}
```

Android Studio will suggest a number of imports for the Fragment, FragmentManager, and FragmentStatePagerAdapter. Make sure you select the imports based on

```
android.support.v4.app.*
```

Finally, we will need to put it all together in the main app activity class. Here we will need to overwrite the onCreate() method (as you have done in previous practicals. We will need to set the View (main activity layout with the toolbar and tabs etc). Then we need to initialise the toolbar by loading its reasource and passing it into a Toolbar object. This toolbar object then needs to be linked with the main activity.

```
public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main_activity);

        Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);
        setSupportActionBar(toolbar);

        ... //include code below
    }
```

Next we will need setup the two tabs we need for the fragments. For this we need to load the TabLayout resource and define the two tabs. The code below uses two String resources which contain strings to identify the tabs. You may want to create these two String resources also, or alternatively, simply use a Java string.

```
TabLayout tabLayout = (TabLayout) findViewById(R.id.tab_layout);
tabLayout.addTab(tabLayout.newTab().setText(R.string.tab_page1));
tabLayout.addTab(tabLayout.newTab().setText(R.string.tab_page2));
tabLayout.setTabGravity(TabLayout.GRAVITY_FILL);
```

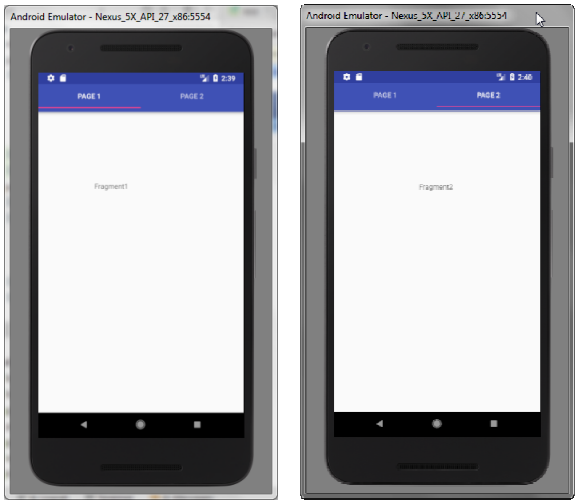
Then we need to link the ViewPager area in the layout with an instance of the PageAdaptor class defined above.

```
final ViewPager viewPager = (ViewPager) findViewById(R.id.pager);
final PagerAdapter adapter = new PagerAdapter(getSupportFragmentManager(),
                                                    tabLayout.getTabCount());
viewPager.setAdapter(adapter);
viewPager.addOnPageChangeListener(new
                                  TabLayout.TabLayoutOnPageChangeListener(tabLayout));

tabLayout.addOnTabSelectedListener(new TabLayout.OnTabSelectedListener() {
    @Override
    public void onTabSelected(TabLayout.Tab tab) {
        viewPager.setCurrentItem(tab.getPosition());
    }

    @Override
    public void onTabUnselected(TabLayout.Tab tab) {}
    @Override
    public void onTabReselected(TabLayout.Tab tab) {}
});
```

Finally, include the required imports for TabLayout, ViewPager and Toolbar, and define the two Strings tab_page1 and tab_page2 in the resources. They contain the strings for the Tabs. Try your app.



You have now reached the checkpoint.

If you wish, you can find more information on Fragments at
http://developer.android.com/guide/components/fragments.html#Creating

You may wish to extend your application to have an icon which you may also want to add to the toolbar, or have an additional tab with fragement. Indeed you may want to have icons for each of the tabs.