ViewPager

In this Lecture, you will learn:

- ViewPager
- Sliding Fragments with ViewPager

<u>ViewPager</u>

ViewPager in the UI lets users navigate between list items by swiping across the screen to page forward or backward. ViewPager is often used with Fragment. There are standard adapters for using fragments with the ViewPager. These standard adapters inclyde FragmentPagerAdapter and FragmentStatePagerAdapter.

Screen slides are transitions between one screen to another. This lecture discusses screen slides with a ViewPager from the support library. ViewPager objects can animate screen slides automatically.

ViewPager is part of AndroidX. Like the Support Library, libraries in androidx namespace ship separately from the Android platform and provide backward compatibility across Android releases. AndroidX is a major improvement to the original Android Support Library, which is no longer maintained.

Sliding Fragments with ViewPager

We will create a todo tasks app that will use Fragments and ViewPager to swipe between tasks.

Start with a simple Model class called task:

Listing 1: Task class
```java
public class Task {
   private int mTaskId;
   private String mTaskDesc;
   public Task(int taskId, String taskDesc) {
      mTaskId = taskId;
      mTaskDesc = taskDesc;
   }
   public void setTaskId(int taskId) {
      mTaskId = taskId;
   }
   public String getTaskDesc() {
      return mTaskDesc;
   }
   public void setTaskDesc(String taskDesc) {
      mTaskDesc = taskDesc;
   }
   public int getTaskId(){
      return this.mTaskId;
   }
}
```

This model is really simple containing only task id and task description.

Next we will create a Fragment class to display the task.

Listing 2: TaskFragment
```java
import android.os.Bundle;
import android.support.annotation.NonNull;
import android.support.annotation.Nullable;
import android.support.v4.app.Fragment;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.TextView;
public class TaskFragment extends Fragment {
   private TextView mTextViewId, mTextViewDesc;
   @Nullable
   @Override
   public View onCreateView(@NonNull LayoutInflater inflater,
@Nullable ViewGroup container, @Nullable Bundle
savedInstanceState) {
      super.onCreateView(inflater, container, savedInstanceState);
      View view = inflater.inflate(R.layout.fragment_task,
container, false);
      Bundle bundle = this.getArguments();
      mTextViewId = view.findViewById(R.id.textViewTaskId);
      mTextViewDesc = view.findViewById(R.id.textViewDesc);
      mTextViewId.setText( String.valueOf(bundle.getInt("tid")) );
      mTextViewDesc.setText( bundle.getString("tdesc") );
      return view;
   }
}
```

This class overrides the onCreateView to initialize the two TextViews containing the task id and task description. These details are passed from the host activity to the Fragment using Bundle object.

The layout file corresponding to the task is listed below.

Listing 3: fragment_task.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
   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">
   <TextView
      android:id="@+id/textViewTaskId"
      android:layout_width="wrap_content"
      android:layout_height="28dp"
      android:layout_marginStart="27dp"
      android:layout_marginTop="46dp"
      android:layout_marginEnd="87dp"
      android:layout_marginBottom="446dp"
      android:text="TextView"
      app:layout_constraintBottom_toBottomOf="parent"
      app:layout_constraintEnd_toStartOf="@+id/textViewDesc"
```

```
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.954" />
    <TextView
        android:id="@+id/textViewDesc"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="87dp"
        android:layout_marginTop="46dp"
        android:layout_marginEnd="154dp"
        android:layout_marginBottom="446dp"
        android:text="TextView"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toEndOf="@+id/textViewTaskId"
        app:layout_constraintTop_toTopOf="parent" />
</android.support.constraint.ConstraintLayout>
```

Layout contains two TextView elements to display task id and description.

ViewPager and PagerAdapter:

Like a RecyclerView requires an Adapter to provide views, a ViewPager requires a PagerAdapter. However, the conversation between ViewPager and PagerAdapter is much more involved than the conversation between RecyclerView and Adapter. But you can use FragmentStatePagerAdapter, a subclass of PagerAdapter, to take care of many of the details.

FragmentStatePagerAdapter has two simple methods: getCount() and getItem(int). When getItem(int) method is called for a position in your array of tasks, it will return a TaskFragment to display the task at that position. In TaskPagerActivity, set the ViewPager's pager adapter and implement its getCount() and getItem(int) methods.
After having the task model class and task fragment in place, create the host activity class to populate the tasks list and initialize the fragment with ViewPager.

Listing 4: TaskPagerActivity

```
import android.support.v4.app.Fragment;
import android.support.v4.app.FragmentManager;
import android.support.v4.app.FragmentStatePagerAdapter;
import android.support.v4.view.ViewPager;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import java.util.ArrayList;
import java.util.List;
public class TaskPagerActivity extends AppCompatActivity {
    private ViewPager mViewPager;
    private List<Task> mTasks;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_task_pager);
        mTasks = new ArrayList<Task>();
        mTasks.add( new Task(1, "Submit Project on 3rd Feb") ) ;
        mTasks.add( new Task(2, "Visit Doctor") ) ;
        mTasks.add( new Task(3, "Collect Passport") ) ;
        mViewPager = (ViewPager) findViewById(R.id.task_view_pager);
        FragmentManager fragmentManager = getSupportFragmentManager();
        mViewPager.setAdapter(new
          FragmentStatePagerAdapter(fragmentManager) {
            @Override
            public Fragment getItem(int position) {
                Task task = mTasks.get(position);
                Bundle bundle= new Bundle();
                bundle.putInt("tid", task.getTaskId());
                bundle.putString("tdesc", task.getTaskDesc());
                Fragment fragment = new TaskFragment();
                fragment.setArguments(bundle);
                return fragment;
            }
            @Override
            public int getCount() {
                return mTasks.size();
            }
          });
    }
}
```

Use ViewPager’s full package name (android.support.v4.view.ViewPager) because the ViewPager class is from the support library. Unlike Fragment, ViewPager is only available in the support library.

Finally, the activity layout file contains the ViewPager.

Listing 5: activity_task_pager

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.v4.view.ViewPager
xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/task_view_pager"
    android:layout_width="match_parent"
android:layout_height="match_parent">
</android.support.v4.view.ViewPager>
```