Fragments demo

# Part 1: Targeting one device

Download snippets file named *countries\_fragments\_SNIPPETS.zip* for this script.

Create a new Android Application named *CountriesFragments*. The *MainActivity* needs to have “*Backward Compatibility*” checked. Change *activity\_main.xml* layout to *LinearLayout* with *horizontal* orientation.

Use Theme.AppCompat

Find country images in the snippets file and put them into the *drawable* resource folder.

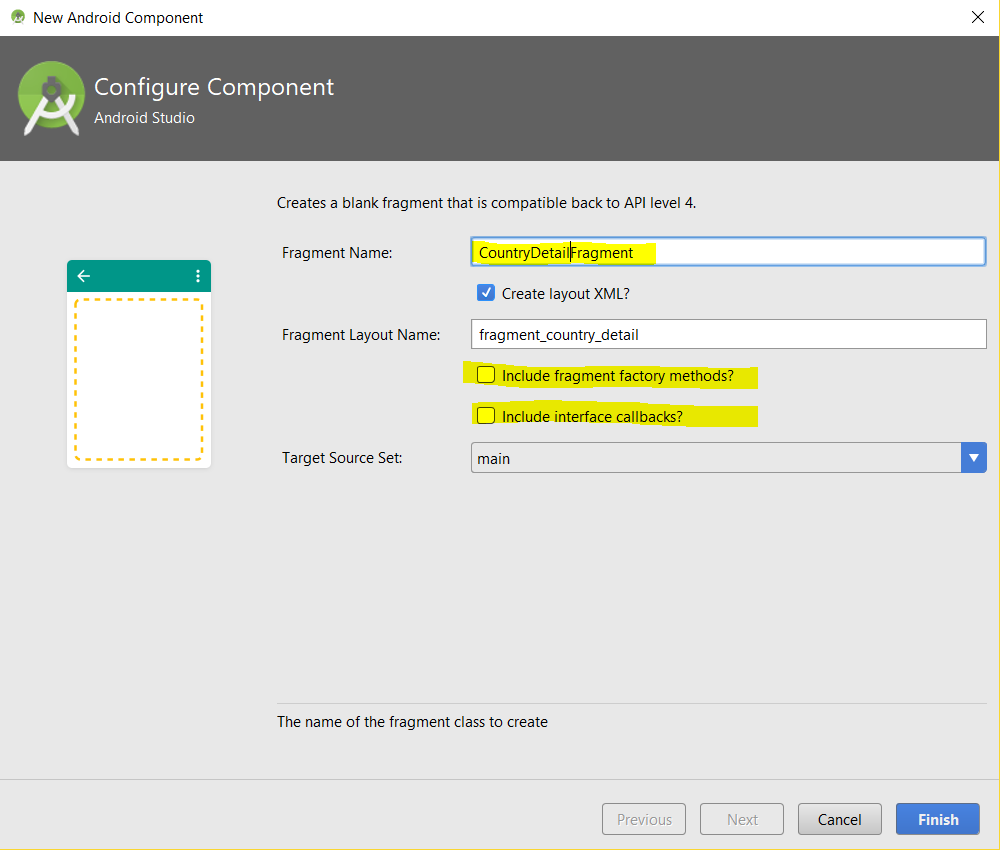
/drawable

Add to your project a mock data source class named *Country.java* found in the snippets.

01. Country class.txt

Add a new fragment:

New >> Fragment >> Fragment (Blank)



Change layout of *fragment\_country\_detail.xml* to *LinearLayout* with *vertical* orientation.

Add the following widgets to *fragment\_country\_detail.xml*:

<TextView

02. fragment\_country\_detail\_widgets.txt

android:id="@+id/name"

android:textAppearance="?android:attr/textAppearanceLarge"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_margin="5dp" />

<TextView

android:id="@+id/description"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_margin="5dp"/>

Notice onCreateView() method in the *CountryDetailFragment.java*:

public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {

// Inflate the layout for this fragment

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

}

Add fragment to the *activity\_main.xml* by replacing the <TextView…> widget with the following:

03. fragment\_in\_main.txt

<fragment

android:name="ca.bcit.countriesfragments.CountryDetailFragment"

android:id="@+id/detail\_frag"

android:layout\_width="match\_parent"

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

For the calling activity to access variables in the Fragment, we will need to create an instance variable in the Fragment with a corresponding *set* method. Add the following Java code to *CountryDetailFragment.java*:

04. countryName.txt

private String countryName;

public void setCountryName(String countryName) {

this.countryName = countryName;

}

For the moment and for testing purposes, we will set the country in the fragment from the main activity with *Mexico*. Add the following code to the bottom of the *onCreate()* method in *MainActivity.java*:

05. setCountryName.txt

CountryDetailFragment frag = (CountryDetailFragment) getSupportFragmentManager().findFragmentById(R.id.detail\_frag);

frag.setCountryName("Mexico");

We need to get *CountryDetailFragment* to update its views with details. Add this *onStart()* method to *CountryDetailFragment*.java:

@Override

06. fragment onStart.txt

public void onStart() {

super.onStart();

View v = getView();

if (v != null) {

Country country = Country.getCountryByContinentAndName("North America", countryName);

TextView name = (TextView)v.findViewById(R.id.name);

name.setText(country.getName());

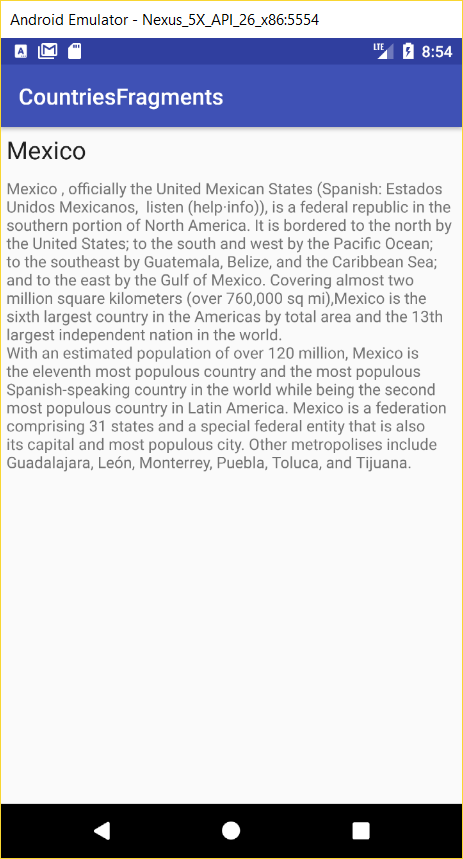
TextView desc = (TextView)v.findViewById(R.id.description);

desc.setText(country.getDescription());

}

}

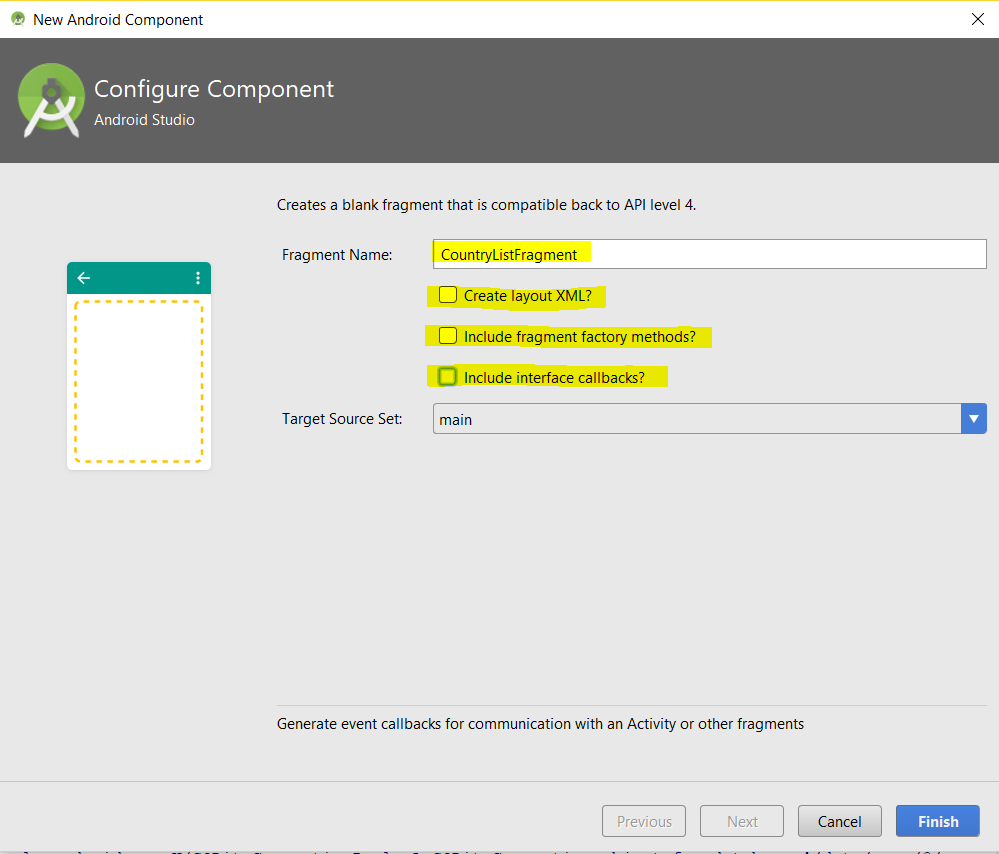
When you run the app it should look like this:



We need another fragment. This time our fragment should list all the countries so that we can choose a country and see details for that country. Just as we used a *ListActivity* before, we can also use a *ListFragment*.

New >> Fragment >> Fragment (Blank)

Name the fragment *CountryListFragment* and uncheck all checkboxes.



Get the fragment class to extend from *android.support.v4.app.ListFragment*.

The only code in the CountryListFragment’s *onCreateView()* method should be:

07. super.onCreateView.txt

String[] names = new String[Country.northAmericaCountries.length];

for (int i=0; i<names.length; i++) {

names[i] = Country.northAmericaCountries[i].getName();

}

ArrayAdapter<String> adapter = new ArrayAdapter<String>(

inflater.getContext(), android.R.layout.simple\_expandable\_list\_item\_1, names

);

setListAdapter(adapter);

return super.onCreateView(inflater, container, savedInstanceState);

Next, let us add CountryListFragment to *activity\_main.xml*, just before the already existing fragment:

08. CountriesListFragment\_in\_main.txt

<fragment

android:name="ca.bcit.countriesfragments.CountryListFragment"

android:id="@+id/list\_frag"

android:layout\_width="0dp"

android:layout\_weight="2"

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

Also, modify the <fragment..> tag for *CountryDetailFragment* in *activity\_main.xml* so it looks like this:

<fragment

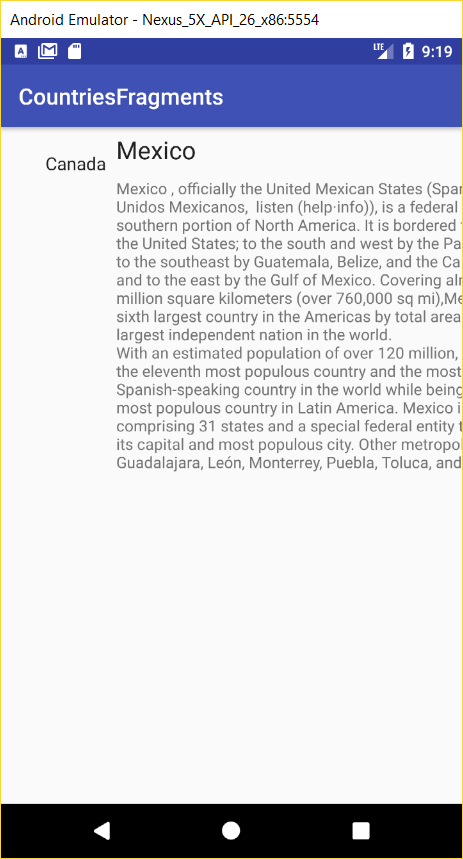
android:name="ca.bcit.countriesfragments.CountryDetailFragment"

android:id="@+id/detail\_frag"

android:layout\_width="match\_parent"

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

This is what it now looks like:



We need a listener for the *ListView* that can be accessible from both the fragment and activity. Create this *CountryListListener* interface:

09. interface.txt

interface CountryListListener {

void itemClicked(String countryName);

}

Add the following instance variable to *CountryListFragment.java*:

private CountryListListener listener;

10. onAttach\_onListItemClick.txt

Add these two methods to *CountryListFragment.java*:

@Override

public void onAttach(Context context) {

super.onAttach(context);

this.listener = (CountryListListener) getActivity();

}

@Override

public void onListItemClick(ListView l, View v, int position, long id) {

if (listener != null) {

listener.itemClicked(Country.northAmericaCountries[(int) id].getName());

}

}

We need to make MainActivity.java implement *CountryListListener*. Make *MainActivity* class implement CountryListListener and add to it the following *itemClicked()* method:

11. itemClicked.txt

@Override

public void itemClicked(String countryName) {

CountryDetailFragment details = new CountryDetailFragment();

FragmentTransaction ft = getSupportFragmentManager().beginTransaction();

details.setCountryName(countryName.trim());

ft.replace(R.id.fragment\_container, details);

ft.addToBackStack(null);

ft.setTransition(FragmentTransaction.TRANSIT\_FRAGMENT\_FADE);

ft.commit();

}

Delete the following code from *onCreate()* in *MainActivity*:

~~CountryDetailFragment frag = (CountryDetailFragment) getSupportFragmentManager().findFragmentById(R.id.detail\_frag);~~

~~frag.setCountryName("Mexico");~~

Replace the second <fragment…> in activity\_main.xml with the following <FrameLayout..> tag:

<FrameLayout

12. FrameLayout.txt

android:id="@+id/fragment\_container"

android:layout\_width="0dp"

android:layout\_weight="3"

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

Rotating the app breaks it. This is because rotation causes the app to restart. We need to save state. Add the following method to *CountryDetailFragment*:

13. onSaveInstanceState.txt

@Override

public void onSaveInstanceState(Bundle outState) {

outState.putString("countryName", countryName);

}

Add this onCreate() method to *CountryDetailFragment:*

@Override

14. onCreate.txt

public void onCreate(@Nullable Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

if (savedInstanceState !=null) {

countryName = savedInstanceState.getString("countryName");

}

}

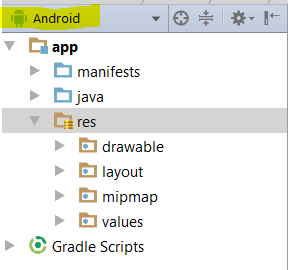
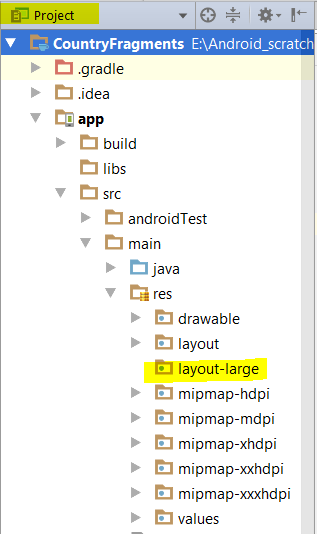
Run the app. It should operate under any orientation.

# Part 2: Targeting Phone and Tablet

The next step is to make the app behave differently depending on whether it is running on a phone or a tablet. We just completed the tablet version of our app that contains both fragments in activity\_main.xml. The phone version of the app will display details in a separate activity, which we will call *DetailAcxtivity.java*.

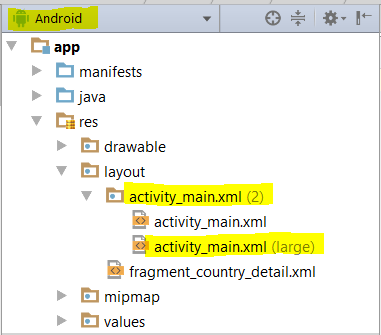
Create a folder under res named layout-large.

You will not see the newly created directory while you are in *Android* view mode. Change to *Project* view mode and you will see it.

Copy *activity\_main.xml* in directory *layout* and paste it in directory *layout-large*.

If you go back in the Android view, *activity\_main.xml* will show in differently:



When the app runs on a phone, we want the *MainActivity* to just display *CountryListFragment*. Go into *res/layout/activity\_main.xml* and delete (or comment out) the second fragment.

<fragment

android:name="ca.bcit.countryfragments.CountryListFragment"

android:id="@+id/list\_frag"

android:layout\_width="0dp"

android:layout\_weight="2"

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

~~<FrameLayout~~

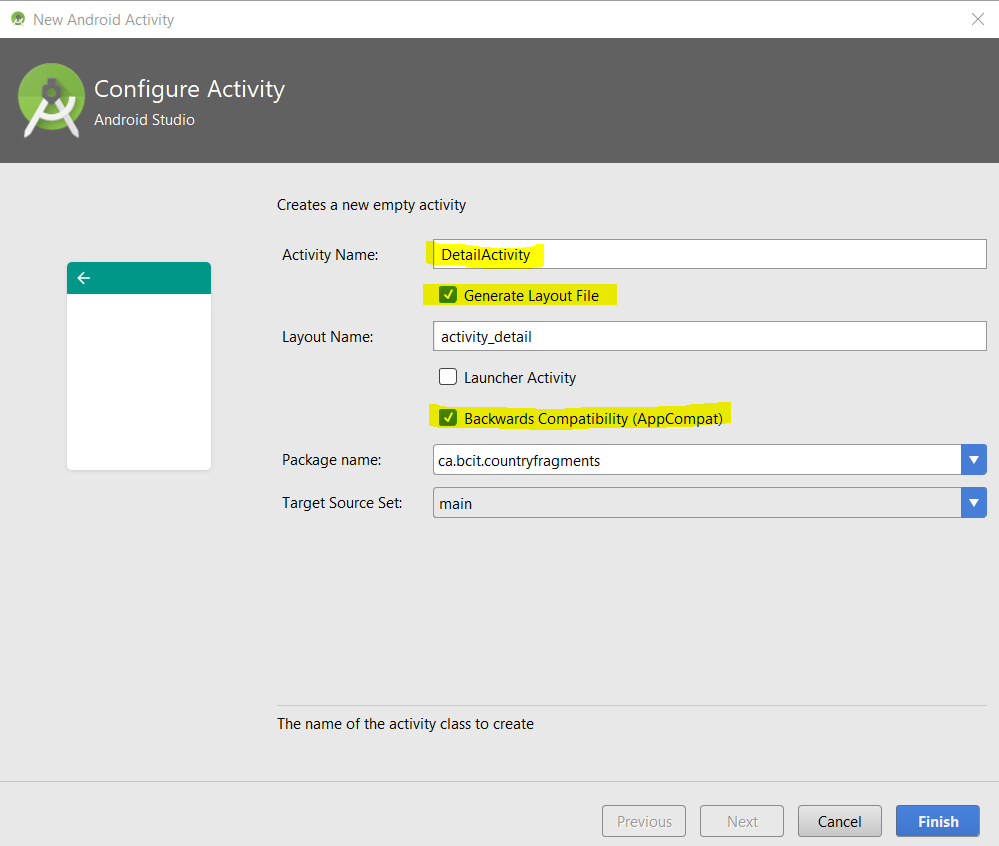
~~android:id="@+id/fragment\_container"~~

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

~~android:layout\_weight="3"~~

~~android:layout\_height="match\_parent" />~~

Next, we need to create a second activity that uses *CountryDetailFragment*. Add an empty activity called *DetailActivity.java*. Do not uncheck “*Backwards Compatibility (AppCompat)*”.



The *activity\_detail.xml* file just needs to contain the fragment *CountryDetailFragment*:

15. activity\_detail\_1.txt

<?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="ca.bcit.countryfragments.DetailActivity">

<fragment

android:name="ca.bcit.countryfragments.CountryDetailFragment"

android:id="@+id/detail\_frag"

android:layout\_width="match\_parent"

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

</LinearLayout>

You can further simplify *activity\_detail.xml* by just having only one <fragment ..> element as follows:

16. activity\_detail\_2.txt

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

<fragment

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

android:name="ca.bcit.countryfragments.CountryDetailFragment"

android:id="@+id/detail\_frag"

android:layout\_width="match\_parent"

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

If the app is running on a phone, *MainActivity* needs to start *DetailActivity* using an *Intent*. This *Intent* then passes the *countryName* to *DetailActivity*. *DetailActivity* subsequently passes *countryName* to *CountryDetailFragment* using the *setCountryName()* method.

The *onCreate()* method in *DetailActivity* should look like this:

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_detail);

String countryName = getIntent().getStringExtra("countryName");

CountryDetailFragment frag = (CountryDetailFragment) getSupportFragmentManager().findFragmentById(R.id.detail\_frag);

frag.setCountryName(countryName);

}

We need *MainActivity* to detect whether the device is a tablet or a phone. This is done by having *MainActivity* search for *fragment\_container*. If *fragment\_container* exists then the device is a tablet, otherwise it is a phone.

Change the *itemClicked()* method in *MainActivity* so that it looks like this:

@Override

public void itemClicked(String countryName) {

View fragmenyContainer = findViewById(R.id.fragment\_container);

if (fragmenyContainer != null) {

CountryDetailFragment details = new CountryDetailFragment();

FragmentTransaction ft = getSupportFragmentManager().beginTransaction();

details.setCountryName(countryName.trim());

ft.replace(R.id.fragment\_container, details);

ft.addToBackStack(null);

ft.setTransition(FragmentTransaction.TRANSIT\_FRAGMENT\_FADE);

ft.commit();

} else {

Intent i = new Intent(this, DetailActivity.class);

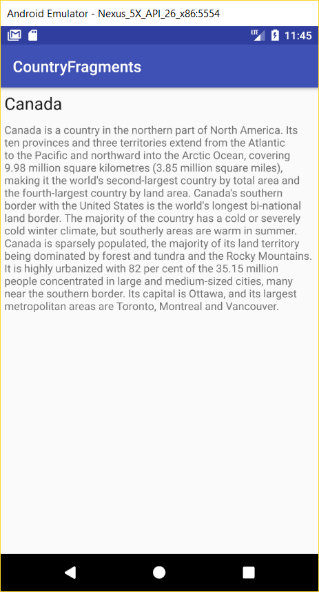
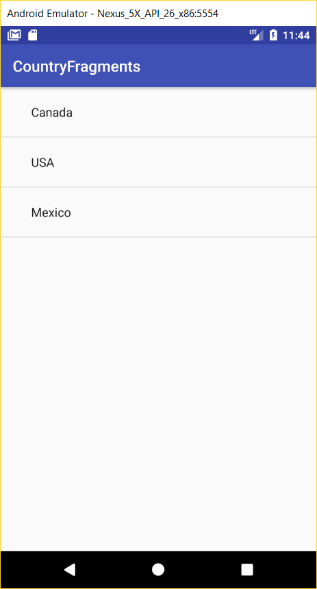
i.putExtra("countryName", countryName.trim());

startActivity(i);

}

}

Run the app in phone emulator:

Run the app in a tablet emulator:

