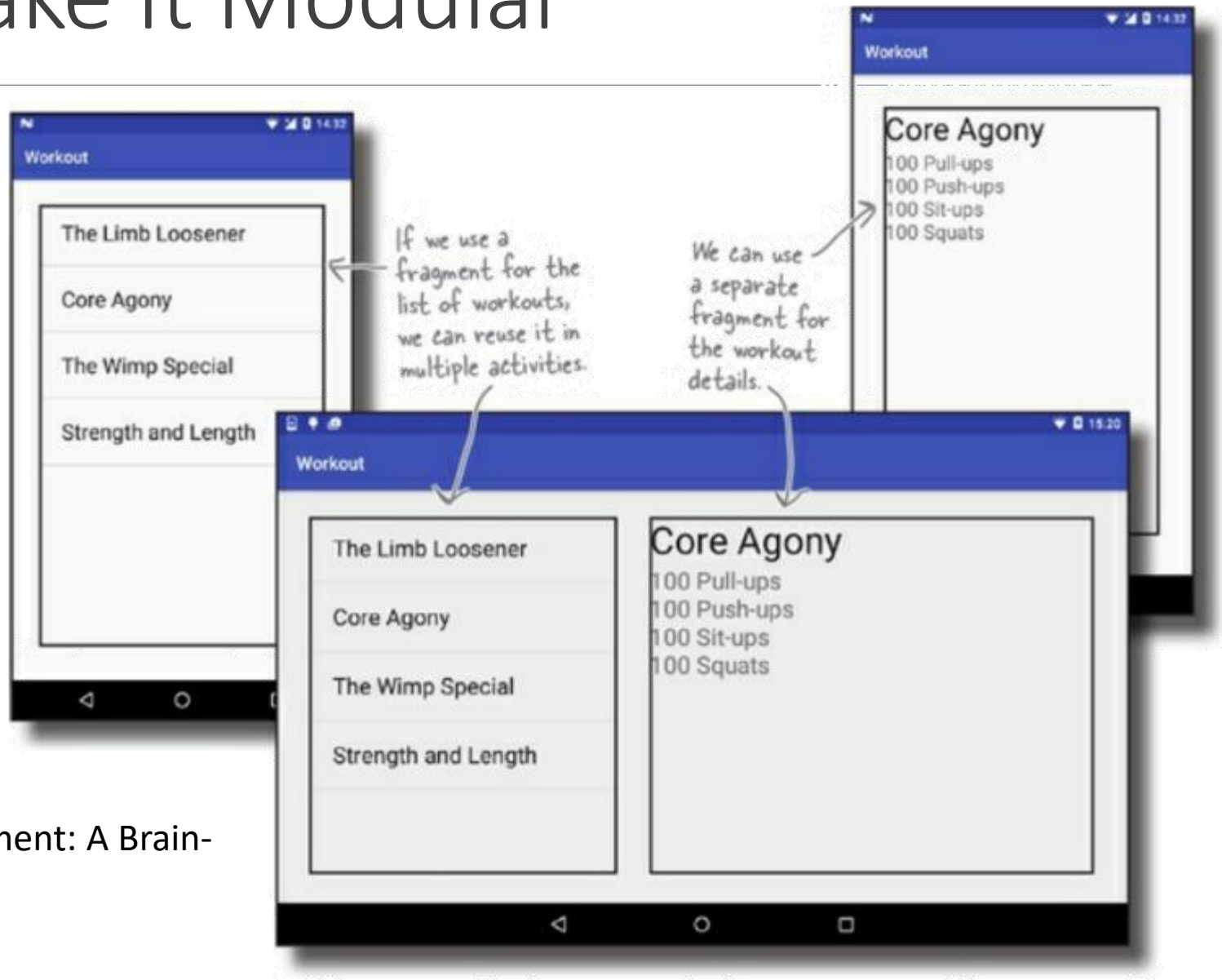
Lab 8 – Fragments

KUAN-TING LAI

2020/10/8

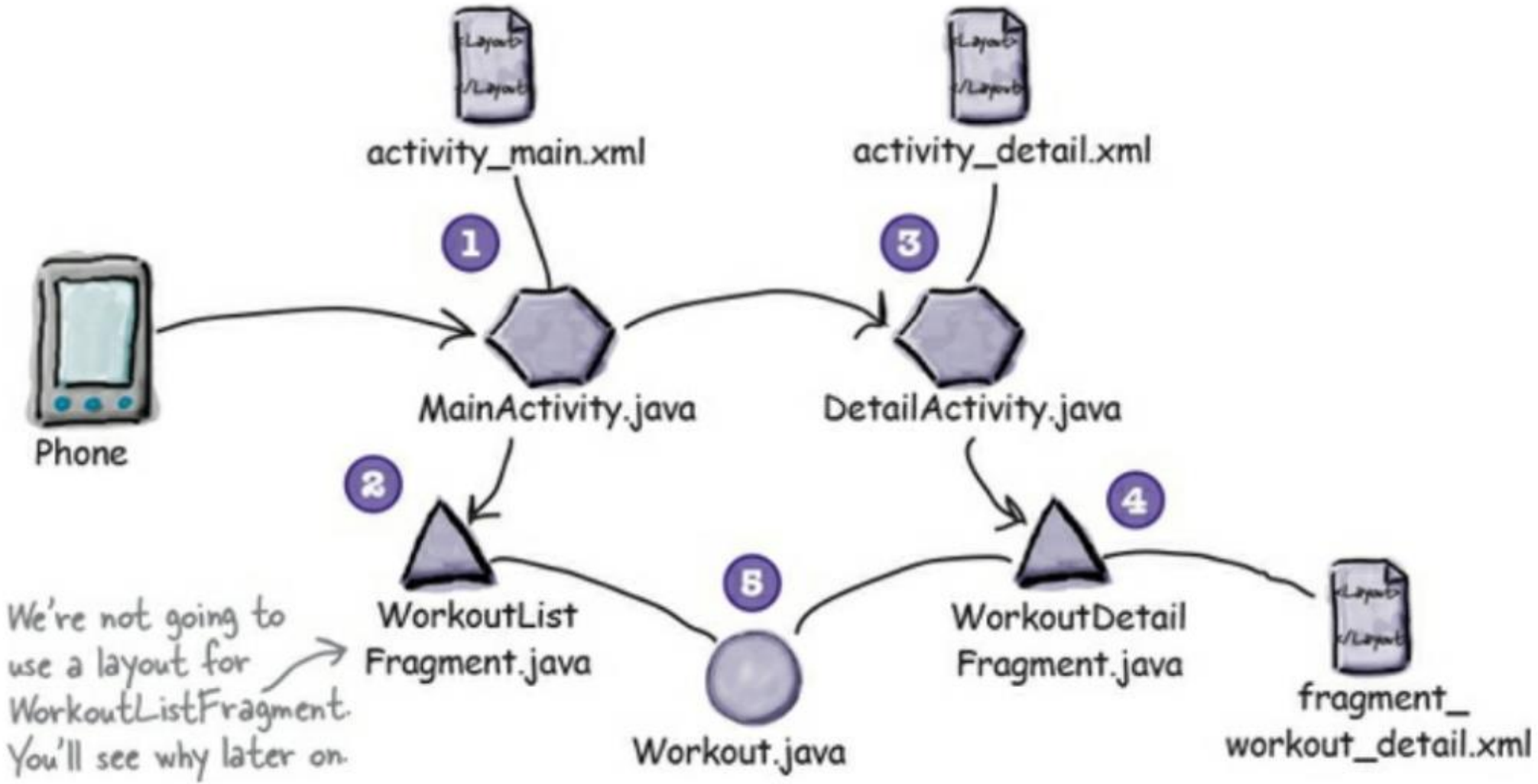
Fragments: Make It Modular

- Fragments:
  - Reusable Components
- Develop a workout APP



Griffiths et al. “Head First Android Development: A Brain-Friendly Guide,” O'Reilly Media, Chapter 9.

APP Workflow



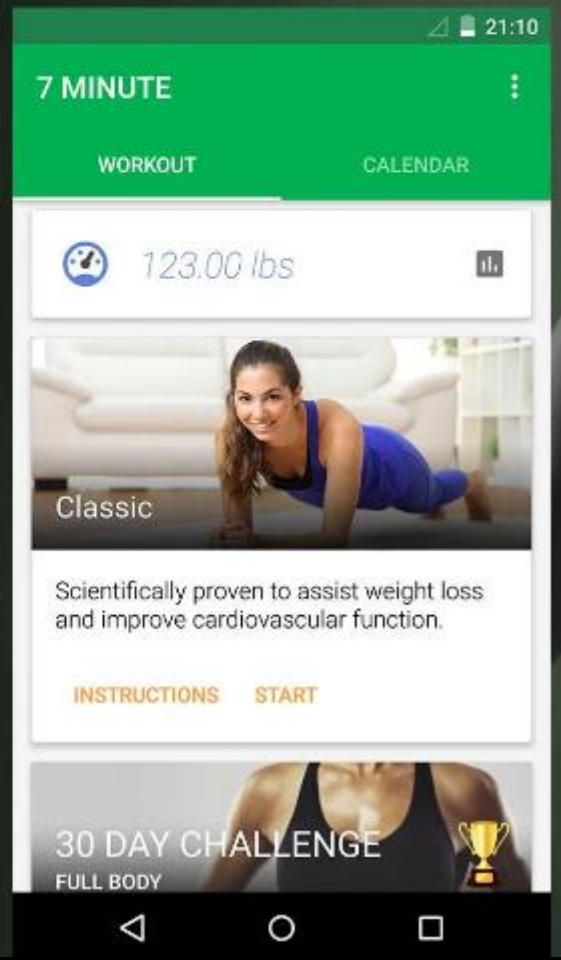
7 MINUTE WORKOUT

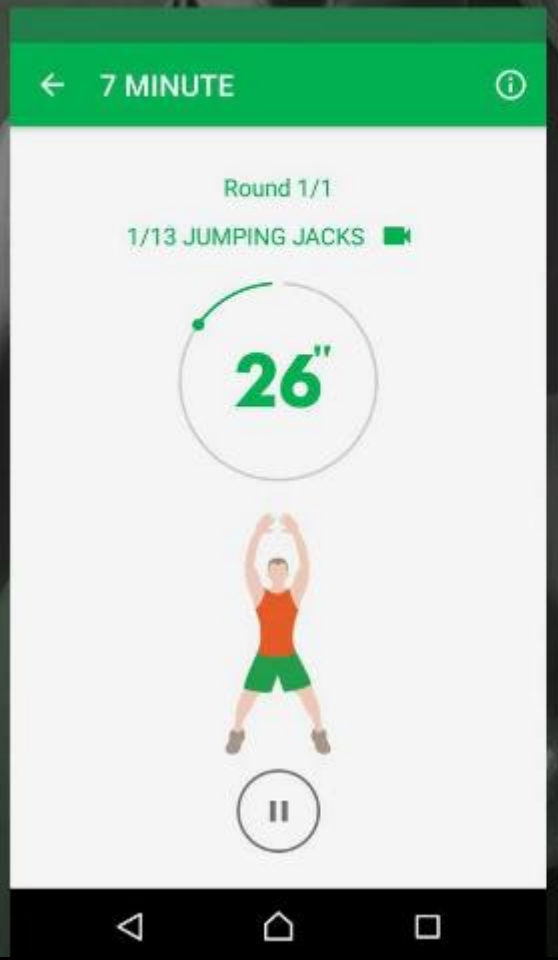
10 MILLION+ HAVE JOINED US

BEST OF 2016 APPS

Google Play

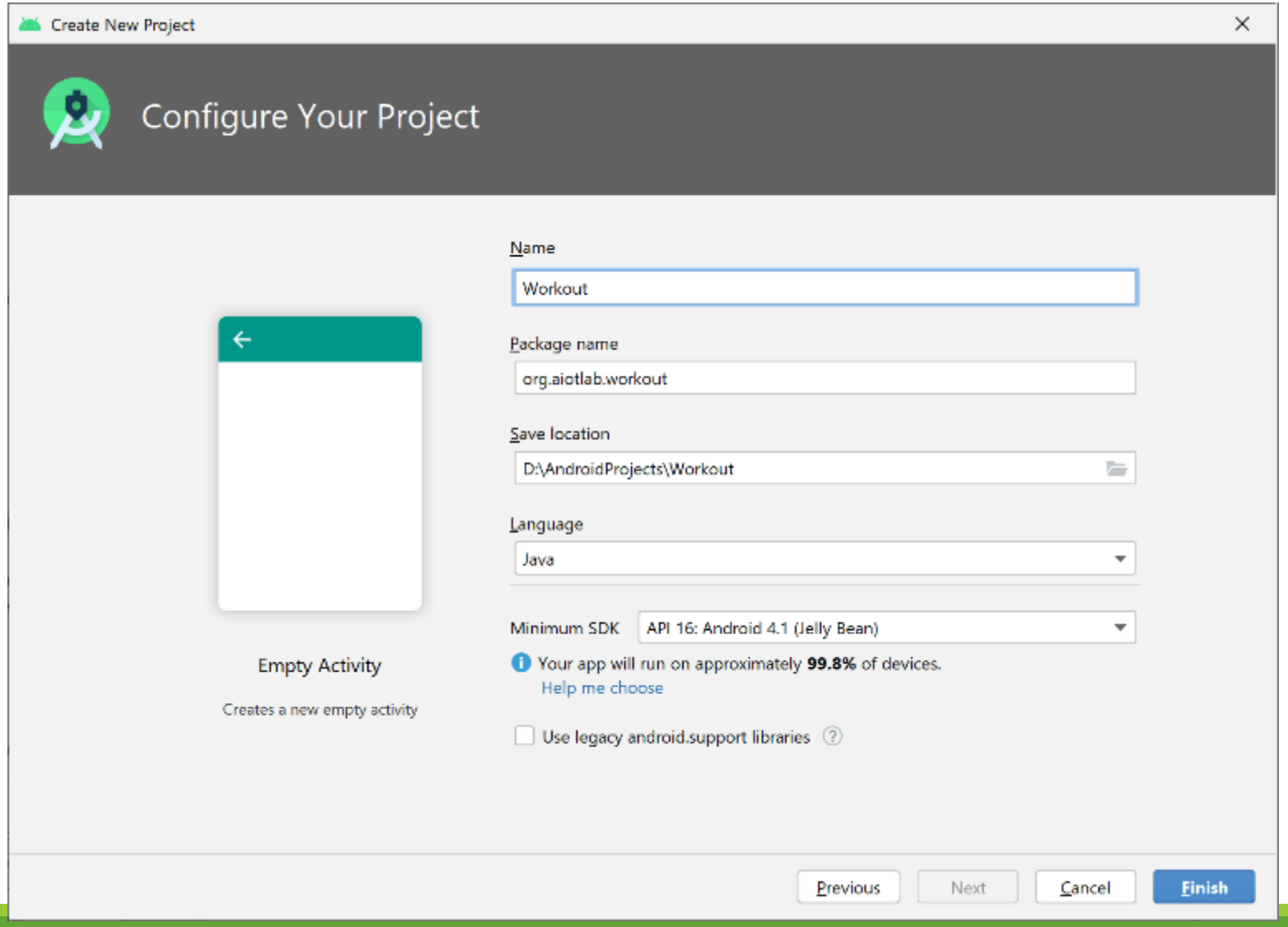
Best Android App of 2016



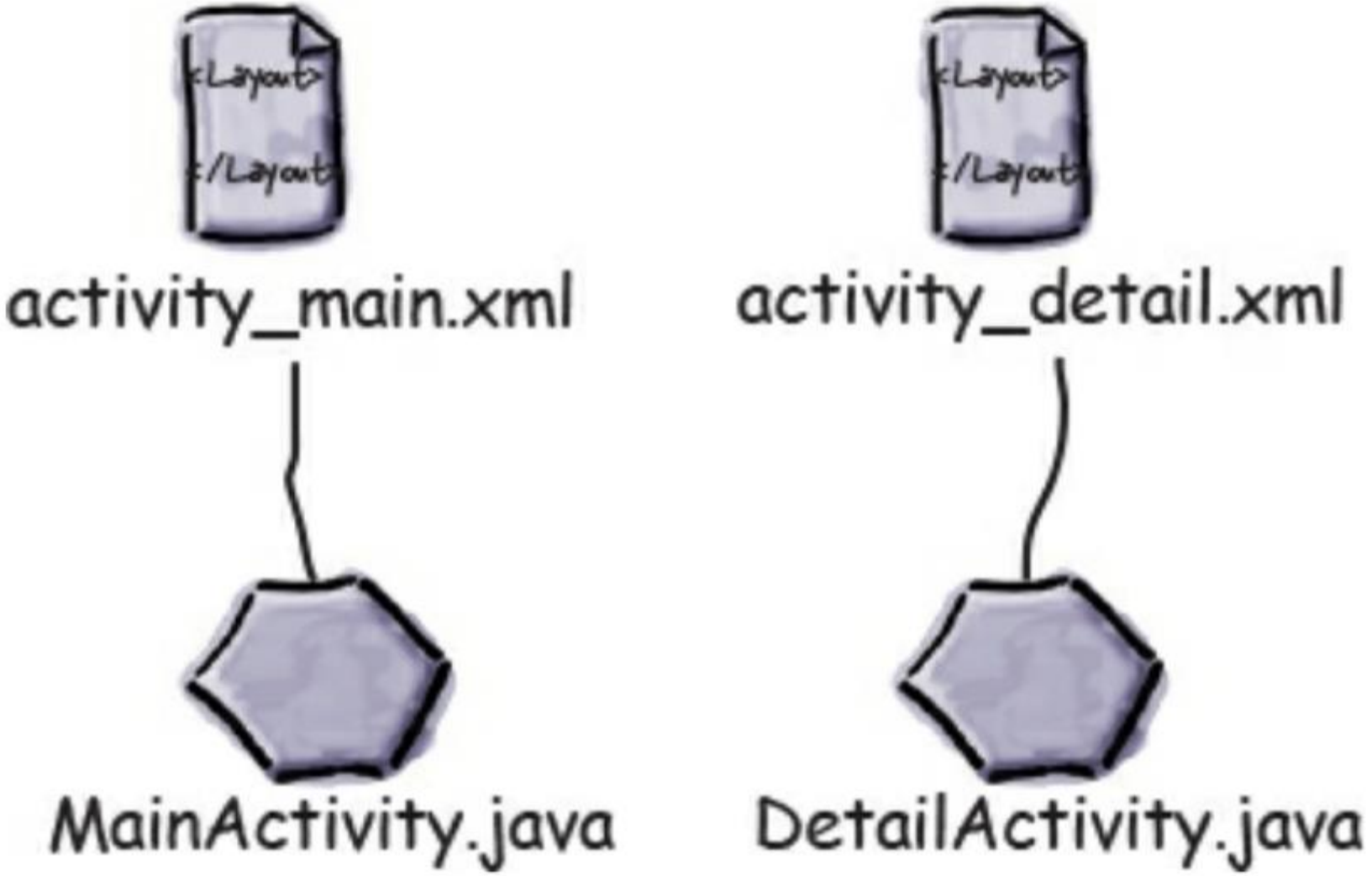


https://play.google.com/store/apps/details?hl=en&id=com.popularapp.sevenmins

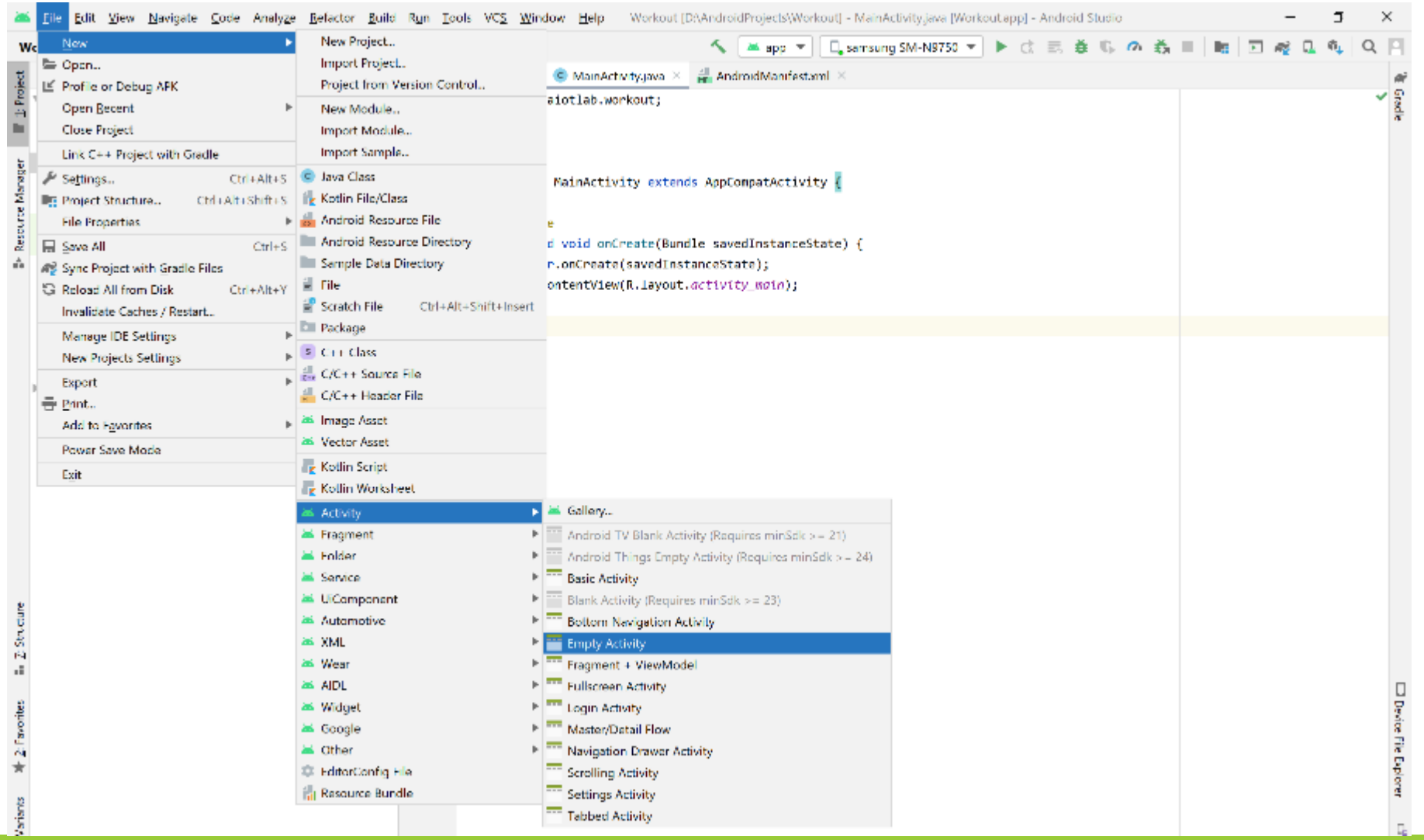
Create workout App



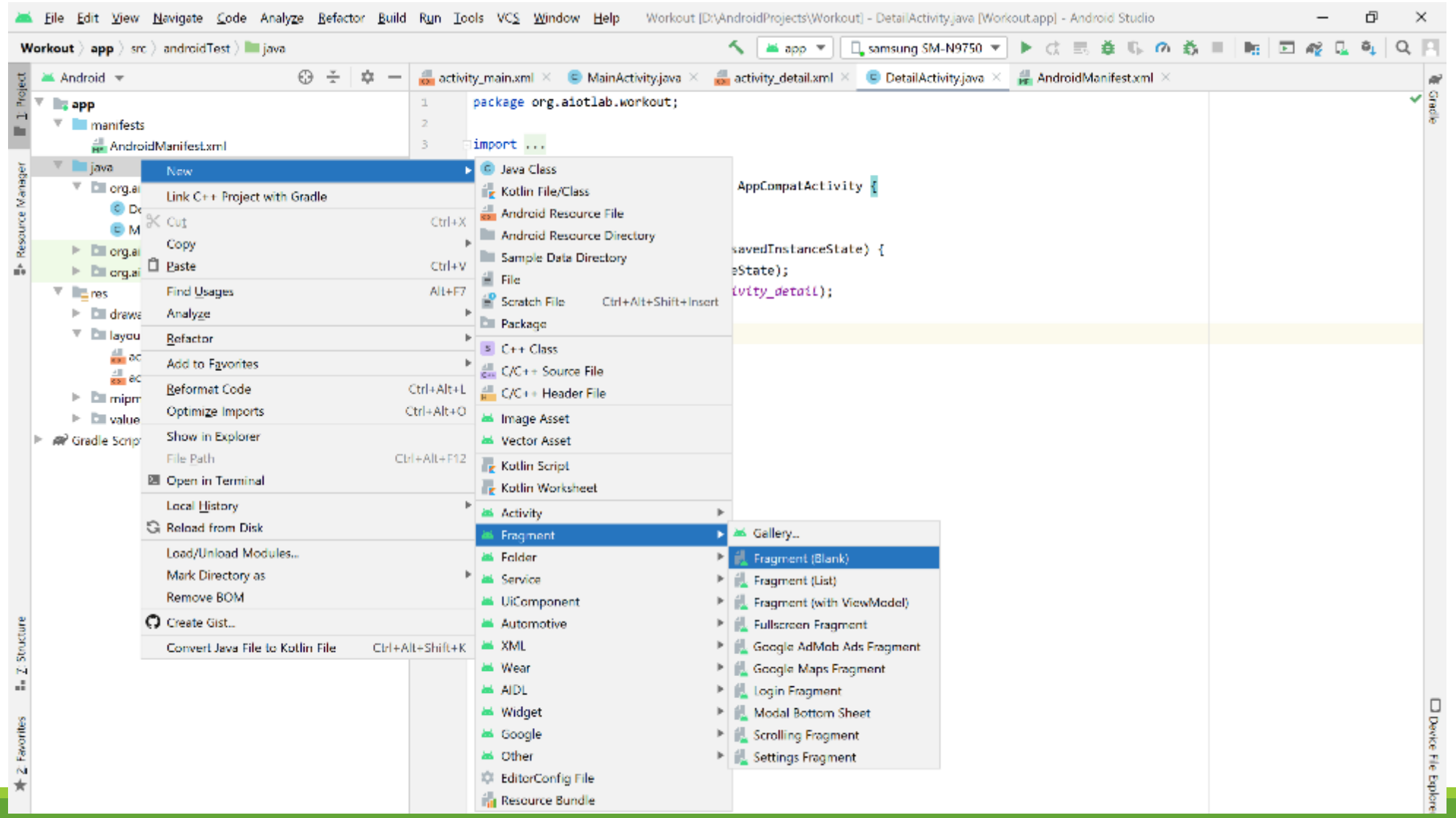
Create 2 Activities: MainActivity & DetailActivity



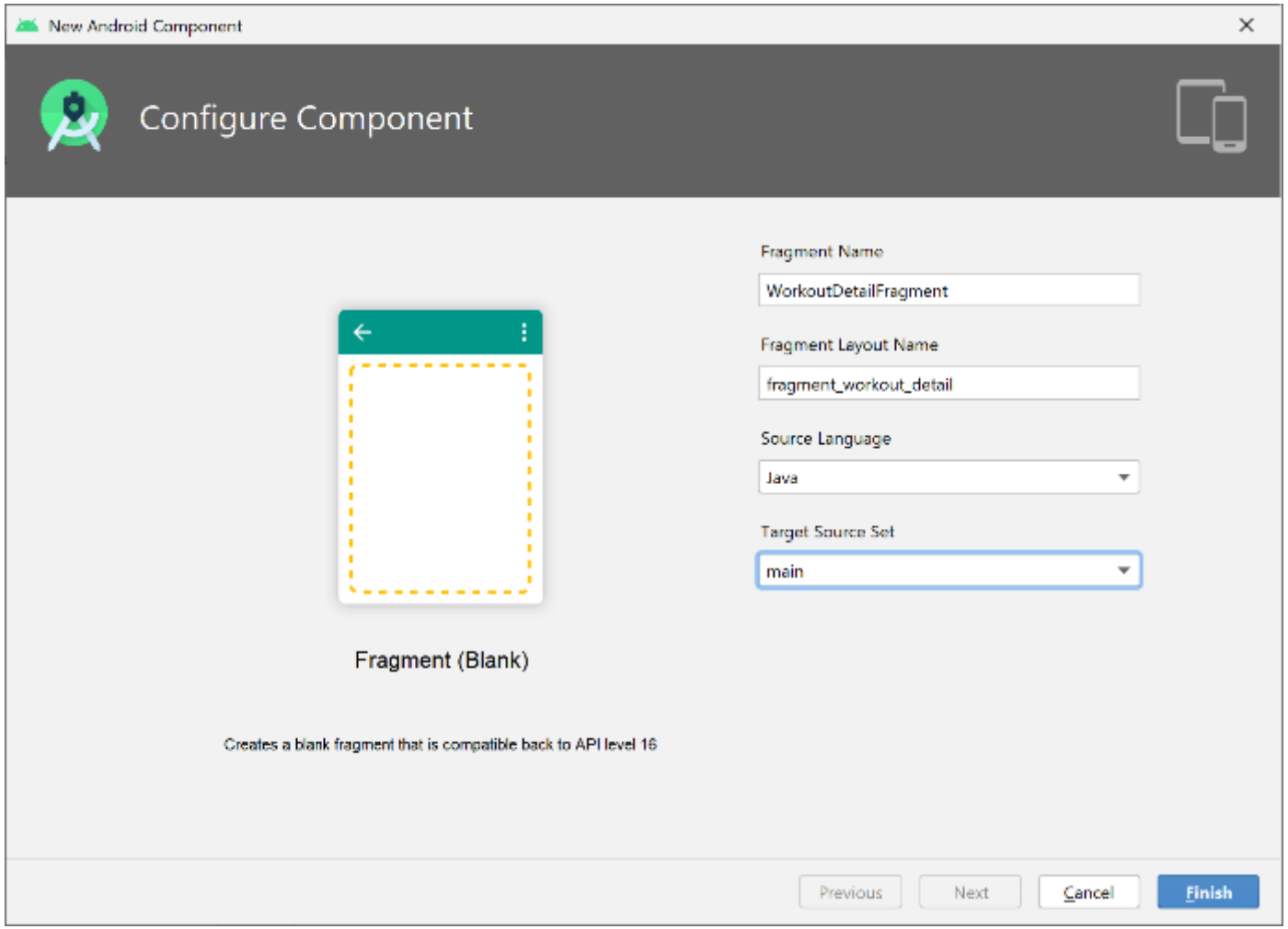
New DetailActivity

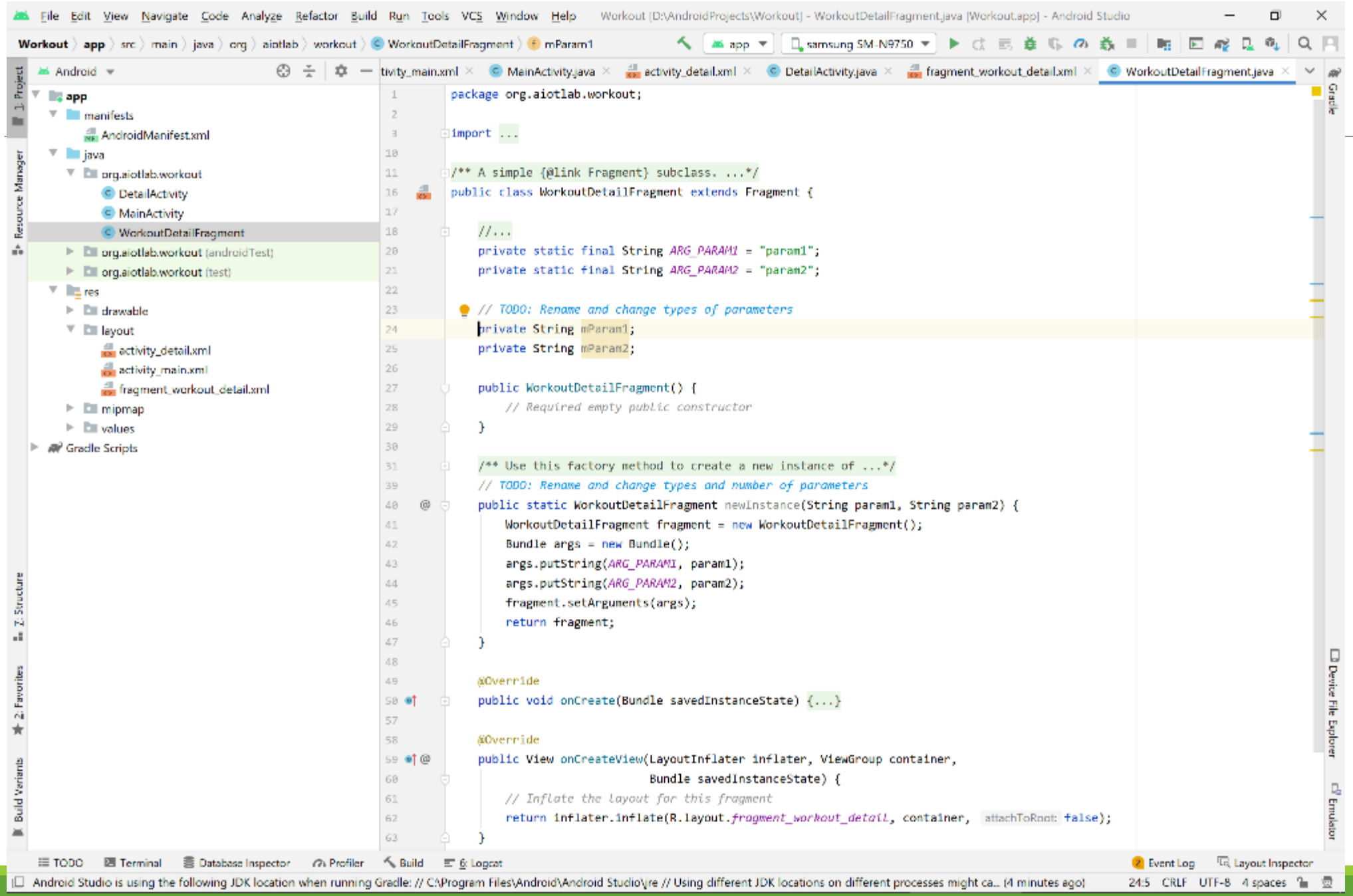


New WorkoutDetailFragment



New WorkoutDetailFragment




File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
Workout (D:\AndroidProjects\Workout) - WorkoutDetailFragment.java [Workout.app] - Android Studio
Workout app src main java org aiotlab workout WorkoutDetailFragment mParam1
app
samsung SM-N9750
Android
1: Project
Resource Manager
manifests
AndroidManifest.xml
java
org.aiotlab.workout
DetailActivity
MainActivity
WorkoutDetailFragment
org.aiotlab.workout (androidTest)
org.aiotlab.workout (test)
res
drawable
layout
activity_detail.xml
activity_main.xml
fragment_workout_detail.xml
mipmap
values
Gradle Scripts
activity_main.xml
MainActivity.java
activity_detail.xml
DetailActivity.java
fragment_workout_detail.xml
WorkoutDetailFragment.java
package org.aiotlab.workout;
import ...
/** A simple {@link Fragment} subclass. ...*/
public class WorkoutDetailFragment extends Fragment {
//...
private static final String ARG_PARAM1 = "param1";
private static final String ARG_PARAM2 = "param2";
// TODO: Rename and change types of parameters
private String mParam1;
private String mParam2;
public WorkoutDetailFragment() {
// Required empty public constructor
}
/** Use this factory method to create a new instance of ...*/
// TODO: Rename and change types and number of parameters
public static WorkoutDetailFragment newInstance(String param1, String param2) {
WorkoutDetailFragment fragment = new WorkoutDetailFragment();
Bundle args = new Bundle();
args.putString(ARG_PARAM1, param1);
args.putString(ARG_PARAM2, param2);
fragment.setArguments(args);
return fragment;
}
@Override
public void onCreate(Bundle savedInstanceState) {...}
@Override
public View onCreateView(LayoutInflater inflater, ViewGroup container,
Bundle savedInstanceState) {
// Inflate the layout for this fragment
return inflater.inflate(R.layout.fragment_workout_detail, container, attachToRoot: false);
}
Gradle
Device File Explorer
Emulator
2: Favorites
Z: Structure
Build Variants
TODO
Terminal
Database Inspector
Profiler
Build
6: Logcat
Event Log
Layout Inspector
Android Studio is using the following JDK location when running Gradle: // C:\Program Files\Android\Android Studio\jre // Using different JDK locations on different processes might ca... (4 minutes ago)
24:5 CRLF UTF-8 4 spaces

Open WorkoutDetailFragment.java

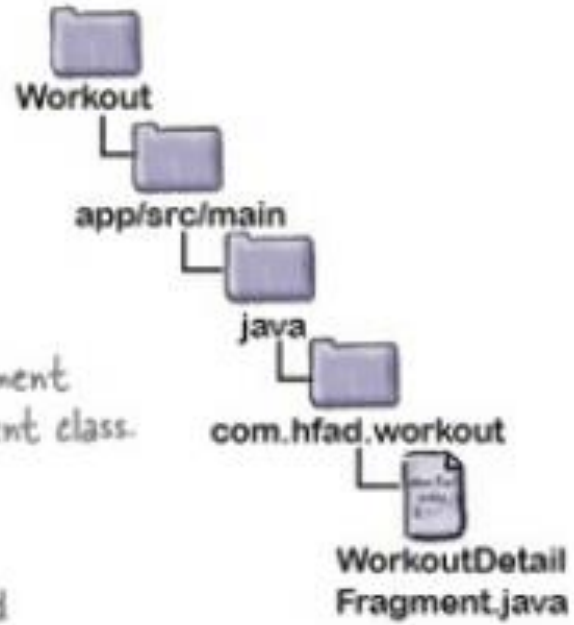
```
package com.hfad.workout;

import android.support.v4.app.Fragment;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;

public class WorkoutDetailFragment extends Fragment {

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

We're using the Fragment class from the Android Support Library.

WorkoutDetailFragment extends the Fragment class.

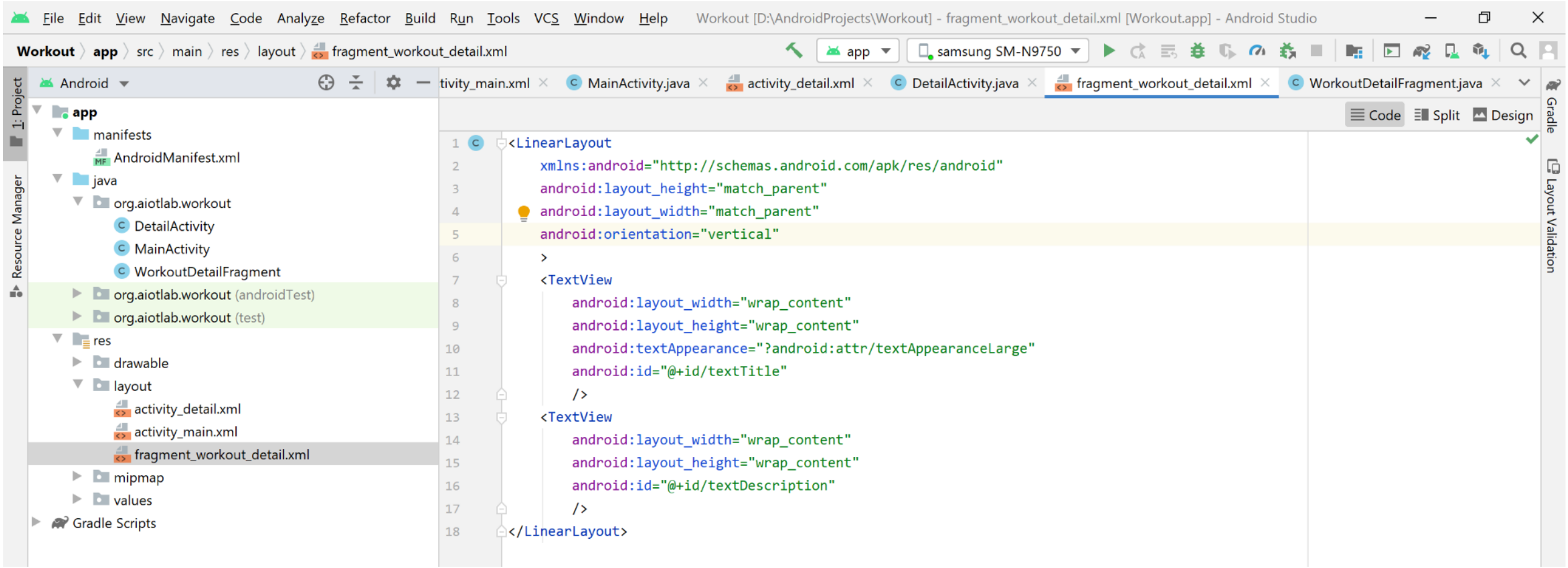
This is the onCreateView() method. It's called when Android needs the fragment's layout

This tells Android which layout the fragment uses (in this case, it's fragment_workout_detail).



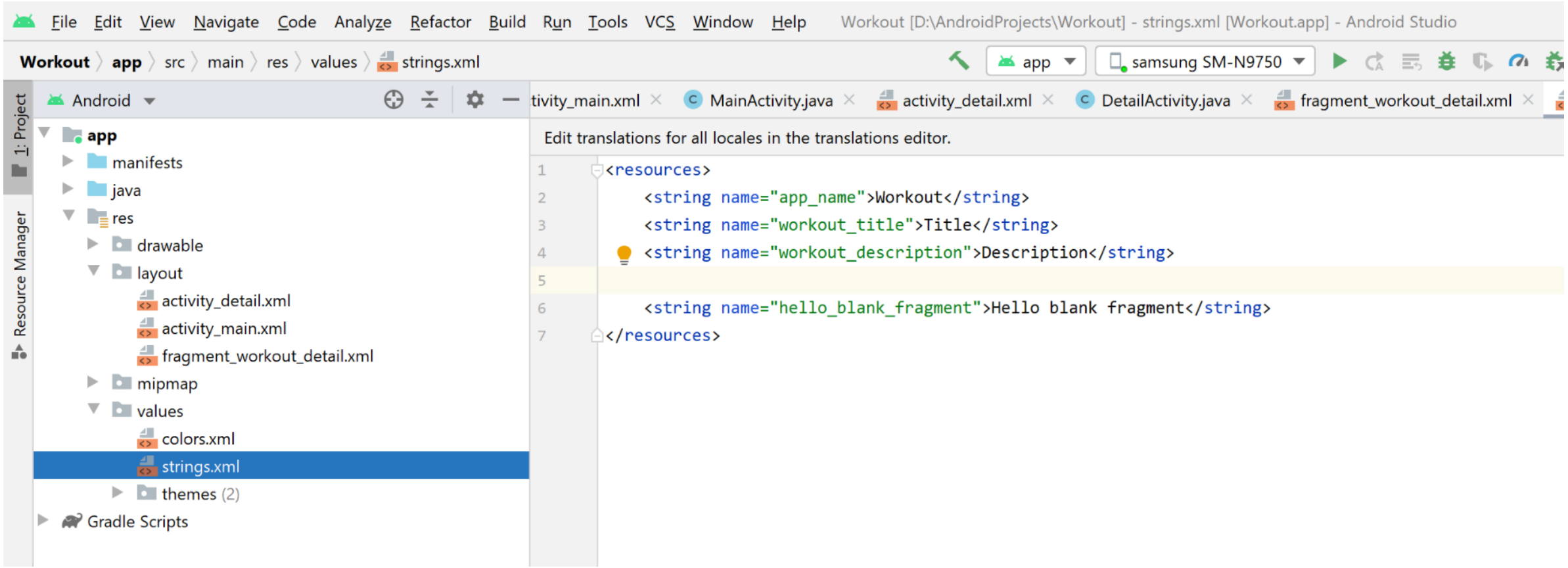
Add TextViews to Fragment Layout

- Edit fragment_workout_detail.xml



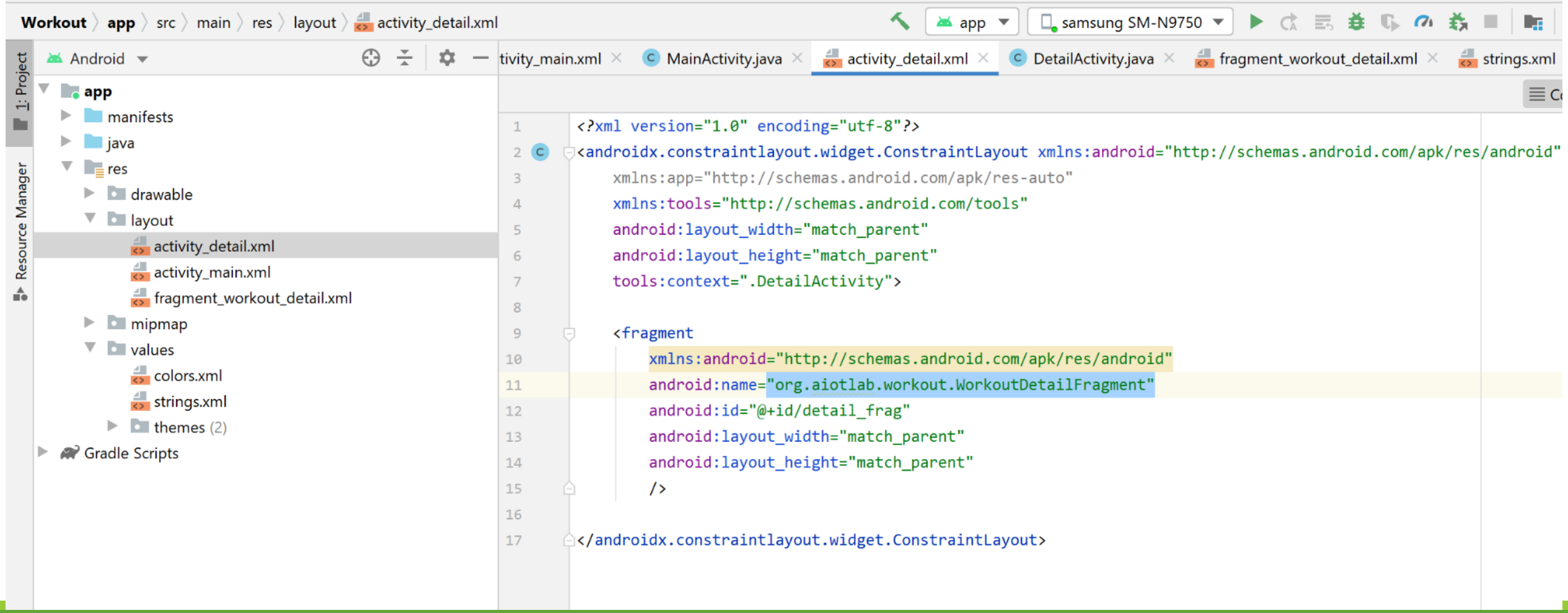
Add String Resource

- Open strings.xml, add “workout_title” and “workout_description”



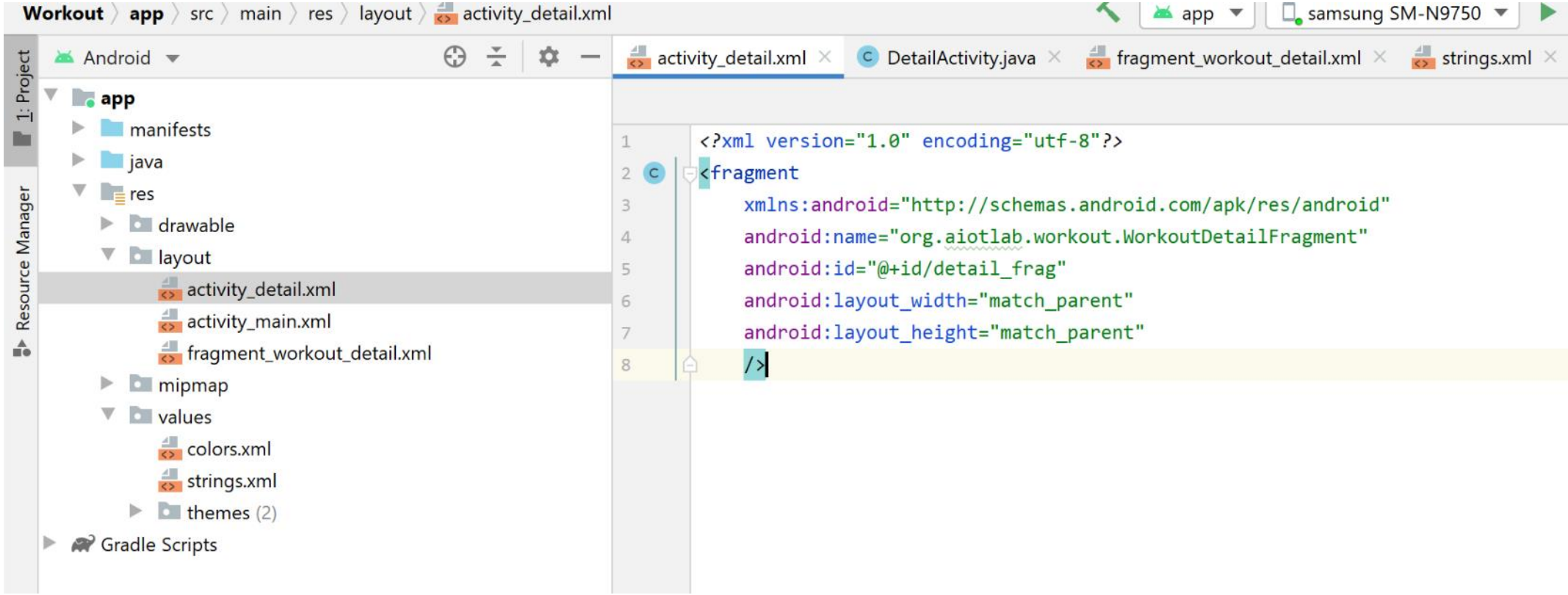
Add Fragment to Activity

- Add <fragment … /> to “activity_detail.xml”



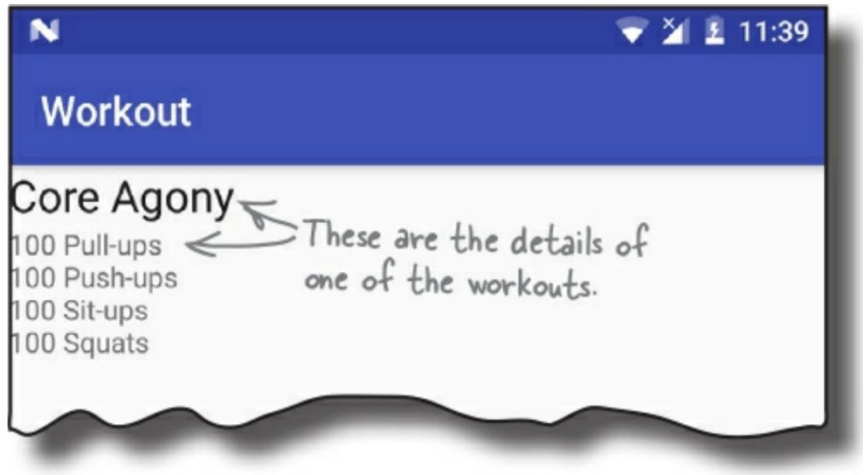
Simplify the Layout

- If your layout contains a single fragment, the <fragment> element can be the layout file's root.



Add Details of Each Workout

- Edit “fragment_workout_detail.xml”



fragment_workout_detail.xml

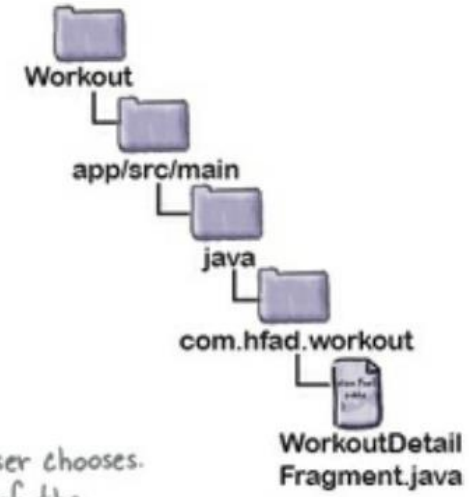
```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_height="match_parent"
    android:layout_width="match_parent"
    android:orientation="vertical"
    >
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceLarge"
        android:id="@+id/textTitle"
        />
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/textDescription"
        />
</LinearLayout>
```

Pass workout ID to the fragment

```
package com.hfad.workout;

import android.support.v4.app.Fragment;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;

public class WorkoutDetailFragment extends Fragment {
    private long workoutId;

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

    public void setWorkout(long id) {
        this.workoutId = id;
    }
}
```

This is the ID of the workout the user chooses. Later, we'll use it to set the values of the fragment's views with the workout details.

This is a setter method for the workout ID. The activity will use this method to set the value of the workout ID.



Fragment Manager

- findFragmentById()

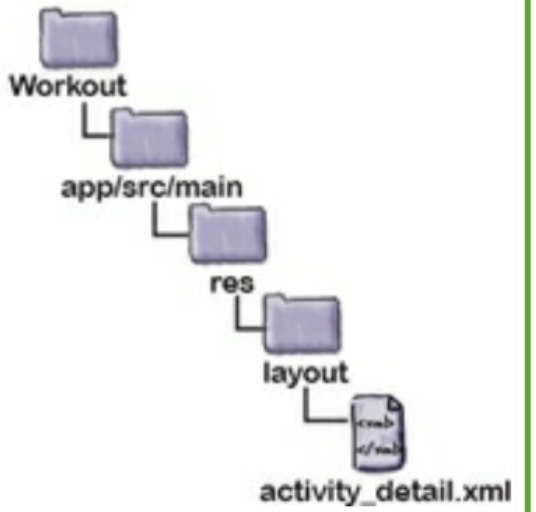
```
getSupportFragmentManager().findFragmentById(R.id.fragment_id)
```

This is the ID of the fragment in the activity's layout.

findFragmentById() is a bit like findViewById() except you use it to get a reference to a fragment.

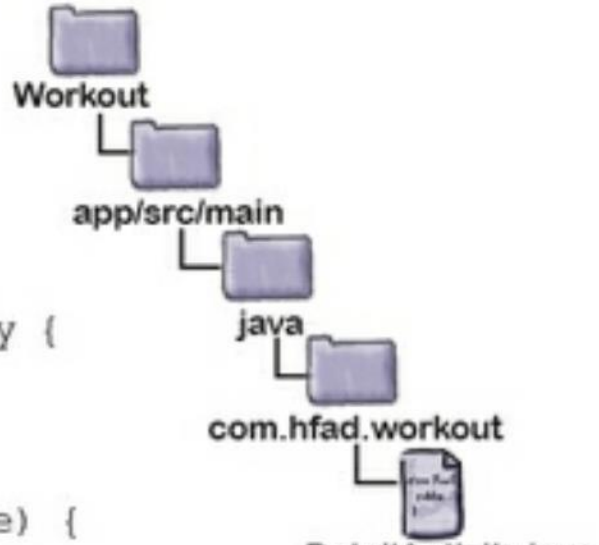
```
<?xml version="1.0" encoding="utf-8"?>
<fragment xmlns:android="http://schemas.android.com/apk/res/android"
    android:name="com.hfad.workout.WorkoutDetailFragment"
    android:id="@+id/detail_frag"
    android:layout_width="match_parent"
    android:layout_height="match_parent" />
```

Add an ID to the fragment.



Set Workout ID in DetailActivity.java

```
package com.hfad.workout;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;

public class DetailActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_detail);
        WorkoutDetailFragment frag = (WorkoutDetailFragment)
                getSupportFragmentManager().findFragmentById(R.id.detail_frag);
        frag.setWorkout(1);
    }
}
```

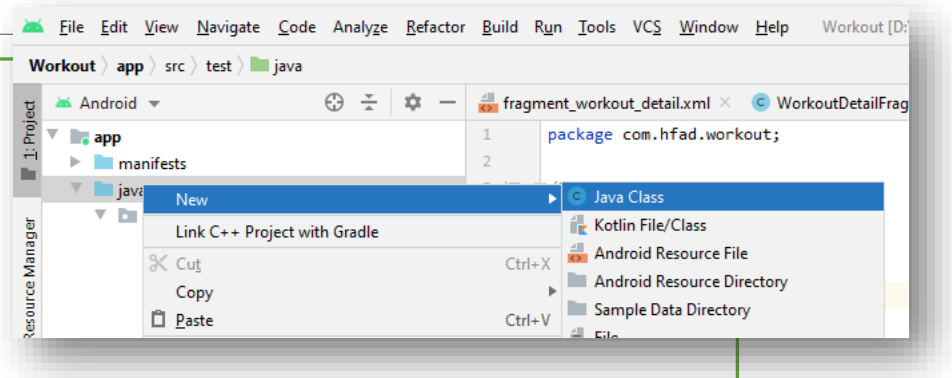


We're going to get WorkoutDetailFragment to display details of a workout here to check that it's working.

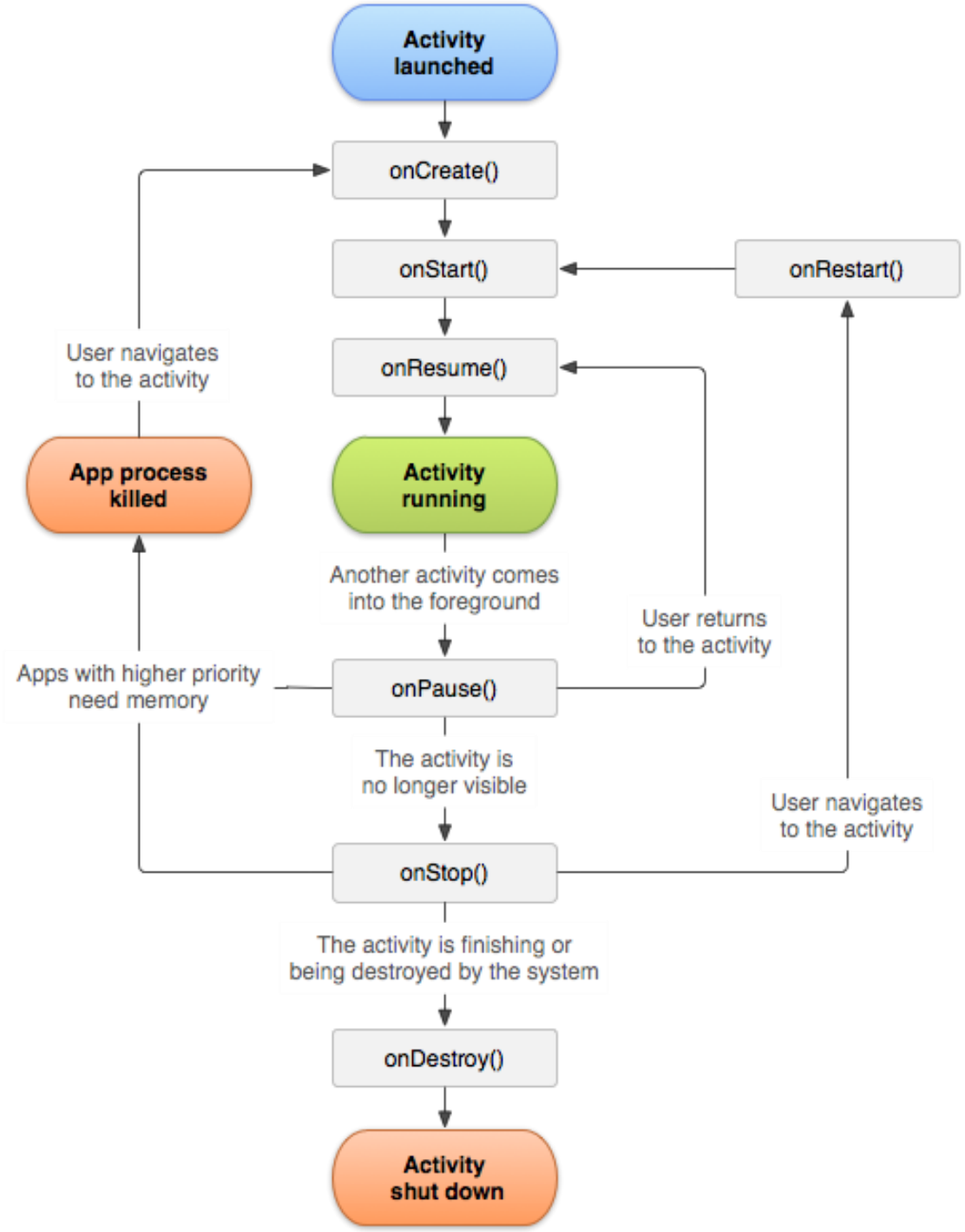
This gets us a reference to WorkoutDetailFragment. Its id in the activity's layout is detail_frag.

Create class Workout

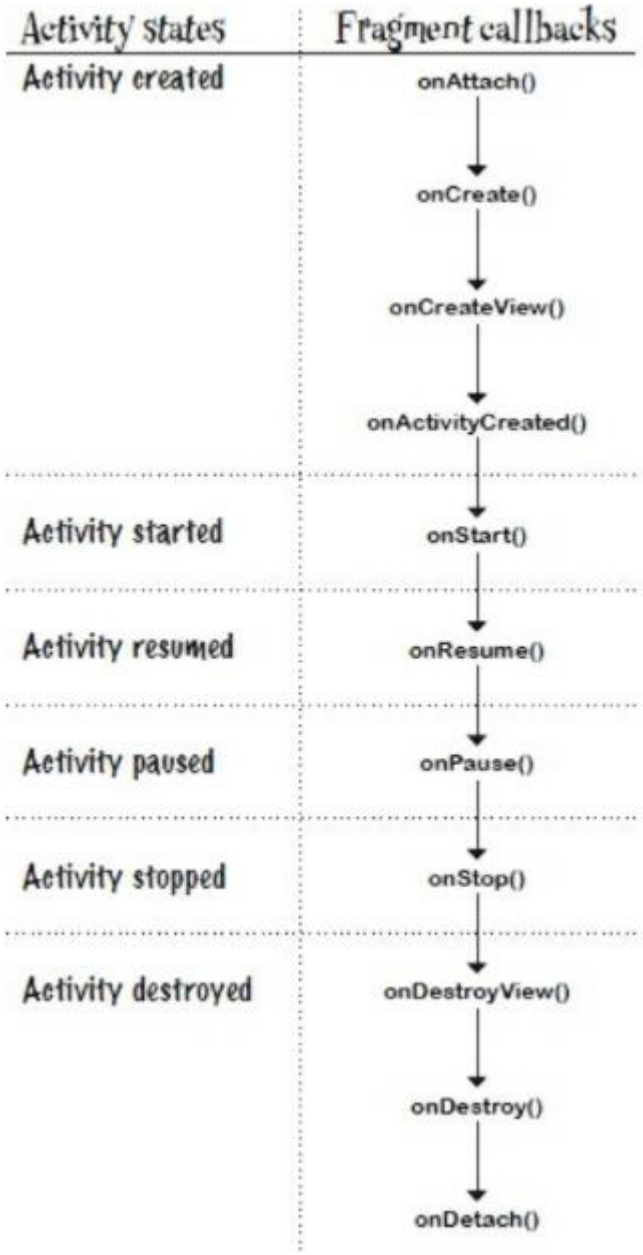
```
public class Workout {
    private String name;
    private String description;

    public static final Workout[] workouts = {
            new Workout("The Limb Loosener",
                    "5 Handstand push-ups\n10 1-legged squats\n15 Pull-ups"),
            new Workout("Core Agony",
                    "100 Pull-ups\n100 Push-ups\n100 Sit-ups\n100 Squats"),
            new Workout("The Wimp Special",
                    "5 Pull-ups\n10 Push-ups\n15 Squats"),
            new Workout("Strength and Length",
                    "500 meter run\n21 x 1.5 pood kettleball swing\n21 x pull-ups")
    };
    //Each Workout has a name and description
    private Workout(String name, String description) {
        this.name = name;
        this.description = description;
    }
    public String getDescription() {
        return description;
    }
    public String getName() {
        return name;
    }
    public String toString() {
        return this.name;
    }
}
```



Activity Life Cycle Revisit



Fragment Life Cycle



onAttach(Context)
This happens when the fragment is associated with a context, in this case an activity.

onCreate(Bundle)
This is very similar to the activity's `onCreate()` method. It can be used to do the initial setup of the fragment.

onCreateView(LayoutInflater, ViewGroup, Bundle)
Fragments use a layout inflater to create their view at this stage.

onActivityCreated(Bundle)
Called when the `onCreate()` method of the activity has completed.

onStart()
Called when the fragment is about to become visible.

onResume()
Called when the fragment is visible and actively running.

onPause()
Called when the fragment is no longer interacting with the user.

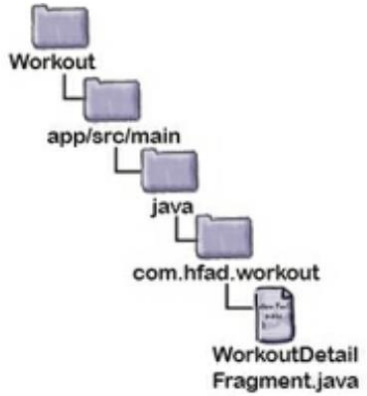
onStop()
Called when the fragment is no longer visible to the user.

onDestroyView()
Gives the fragment the chance to clear away any resources that were associated with its view.

onDestroy()
In this method, the fragment can clear away any other resources it created.

onDetach()
Called when the fragment finally loses contact with the activity.

Set view's values in the fragment's onStart()




```
package com.hfad.workout;

import android.support.v4.app.Fragment;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.TextView;

public class WorkoutDetailFragment extends Fragment {
    private long workoutId;

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

    @Override
    public void onStart() {
        super.onStart();
        View view = getView();
        if (view != null) {
            TextView title = (TextView) view.findViewById(R.id.textTitle);
            Workout workout = Workout.workouts[(int) workoutId];
            title.setText(workout.getName());
            TextView description = (TextView) view.findViewById(R.id.textDescription);
            description.setText(workout.getDescription());
        }
    }

    public void setWorkout(long id) {
        this.workoutId = id;
    }
}
```

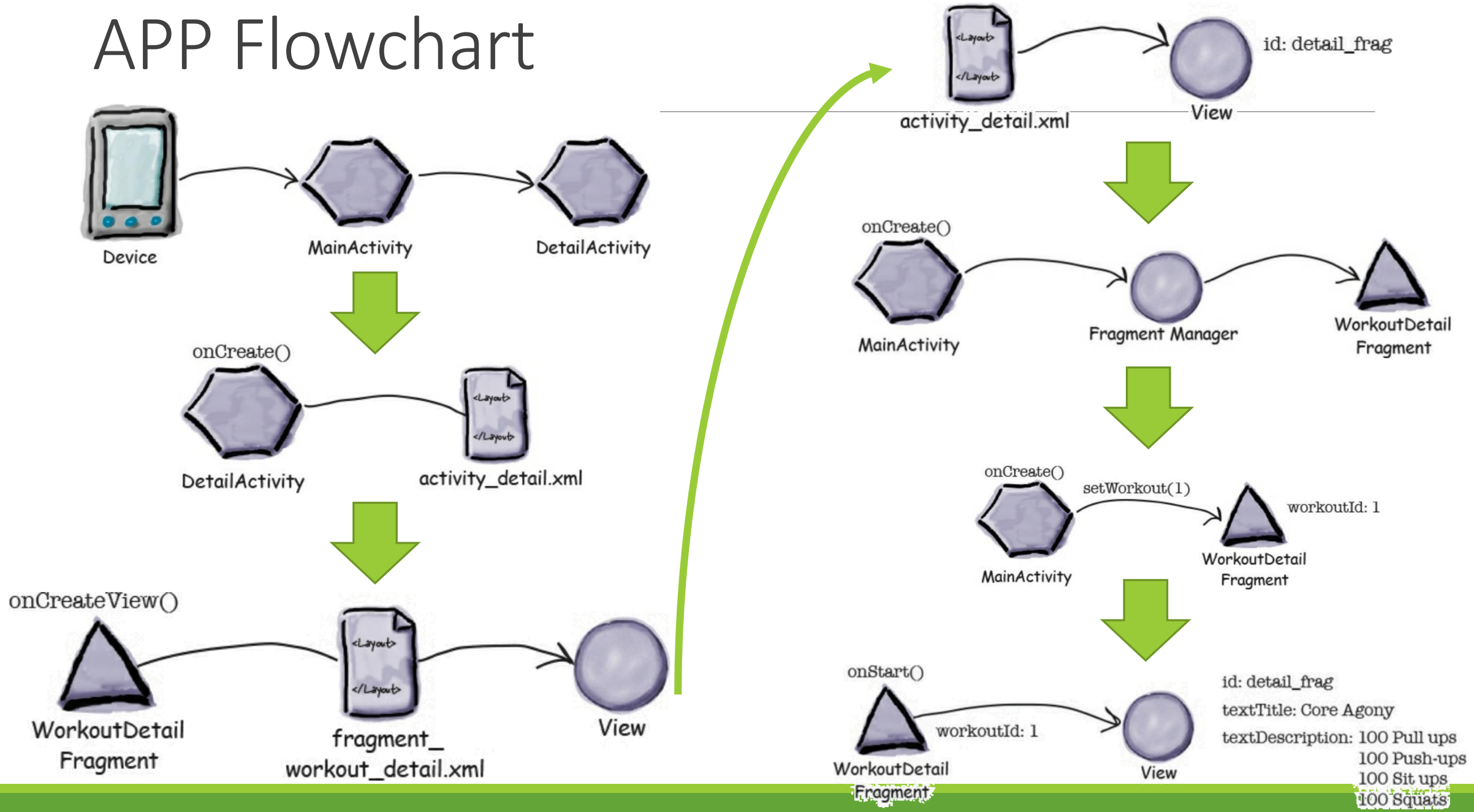
We're using this class in the onStart() method.

The getView() method gets the fragment's root View. We can then use this to get references to the workout title and description text views.

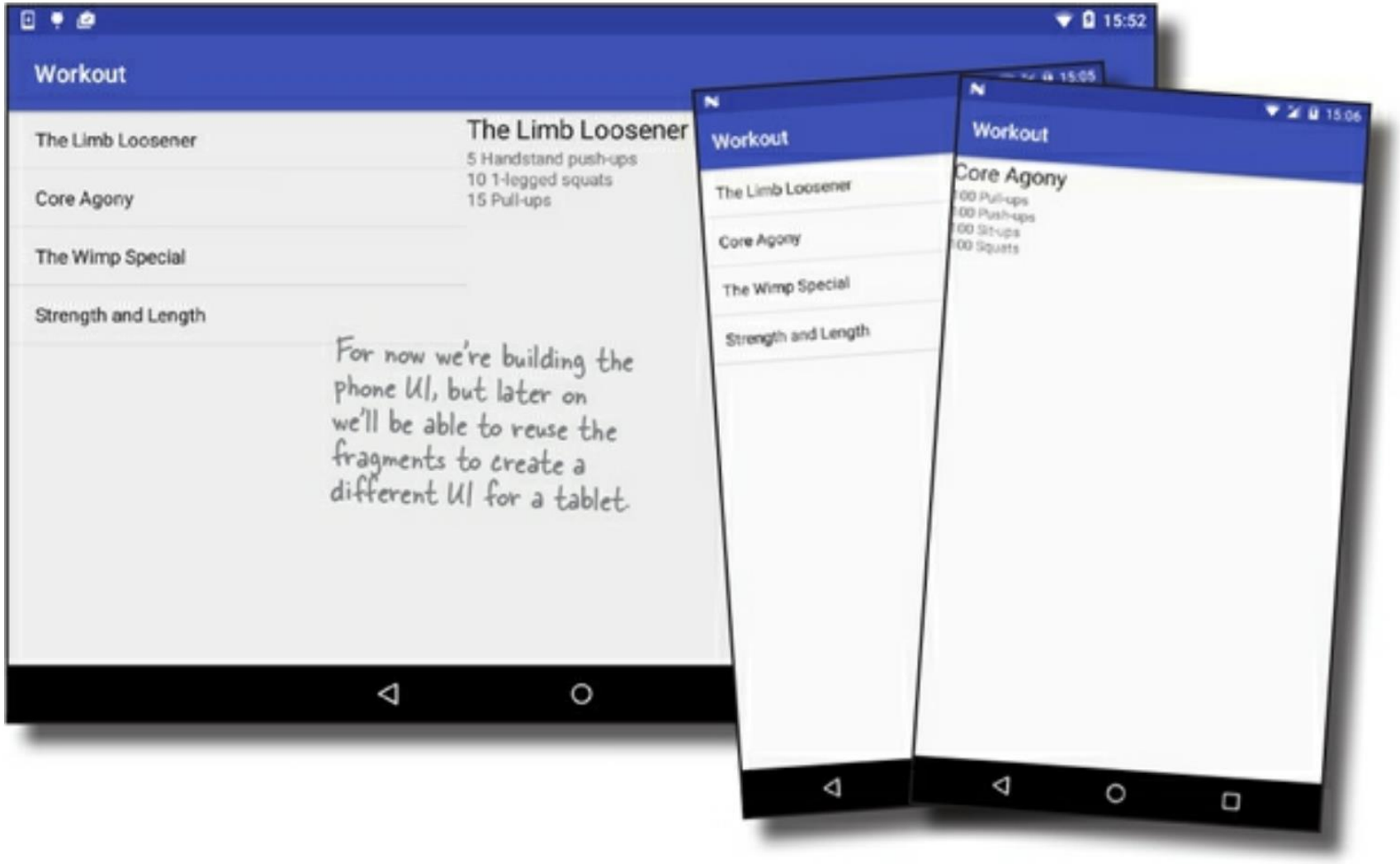
WorkoutDetailFragment.java

```
public class WorkoutDetailFragment extends Fragment {
    private long workoutId;

    @Override
    public void onCreate(Bundle savedInstanceState){
        super.onCreate(savedInstanceState);
        if (savedInstanceState != null) {
            workoutId = savedInstanceState.getLong("workoutId");
        }
    }
    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {
        return inflater.inflate(R.layout.fragment_workout_detail, container, false);
    }
    @Override
    public void onStart() {
        super.onStart();
        View view = getView();
        if (view != null) {
            TextView title = (TextView) view.findViewById(R.id.textTitle);
            Workout workout = Workout.workouts[(int) workoutId];
            title.setText(workout.getName());
            TextView description = (TextView) view.findViewById(R.id.textDescription);
            description.setText(workout.getDescription());
        }
    }
    @Override
    public void onSaveInstanceState(Bundle savedInstanceState) {
        savedInstanceState.putLong("workoutId", workoutId);
    }
    public void setWorkout(long id) {
        this.workoutId = id;
    }
```

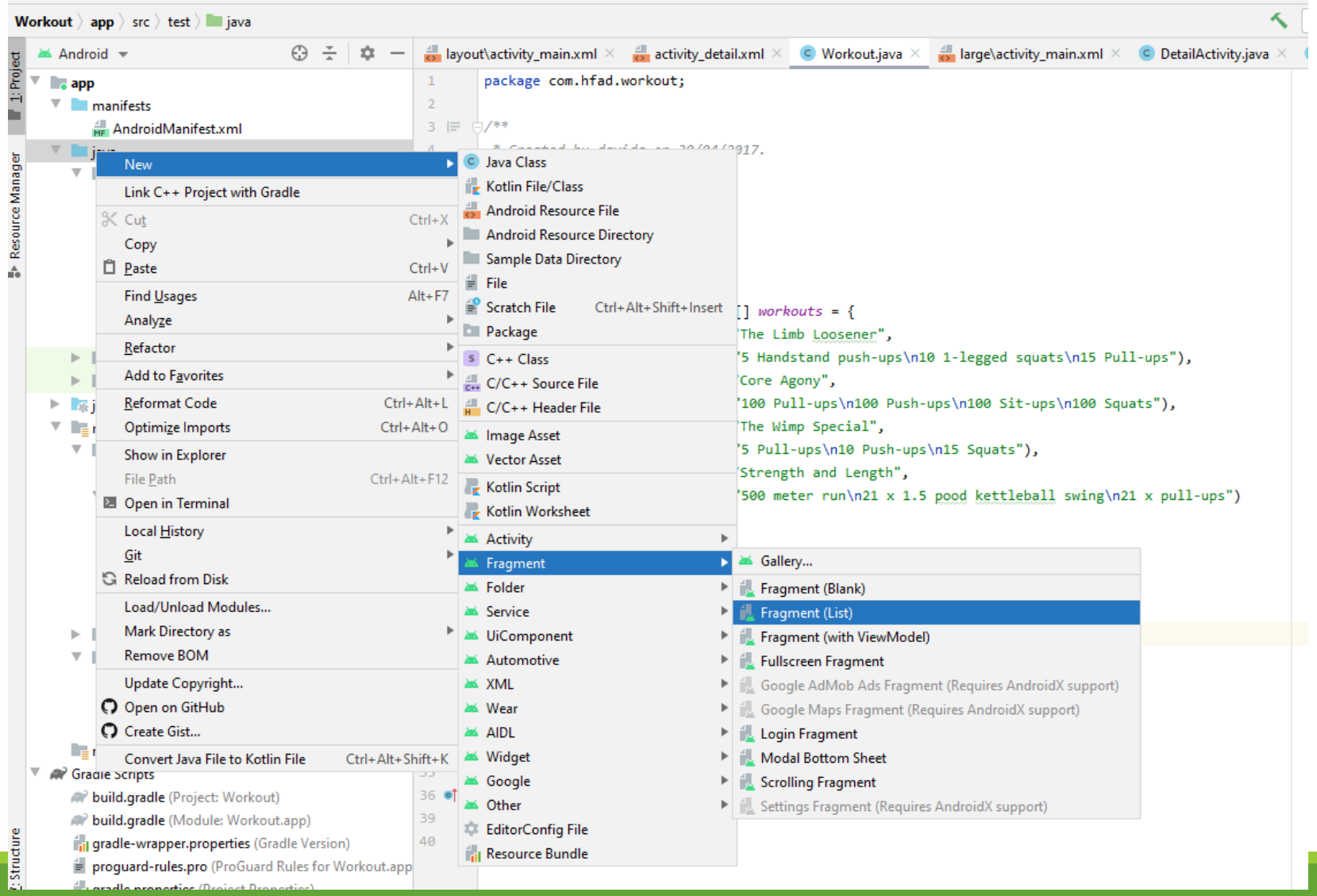
APP Flowchart



Create ListFragment



New WorkoutListFragment



WorkoutListFragment.java

- ArrayAdapter

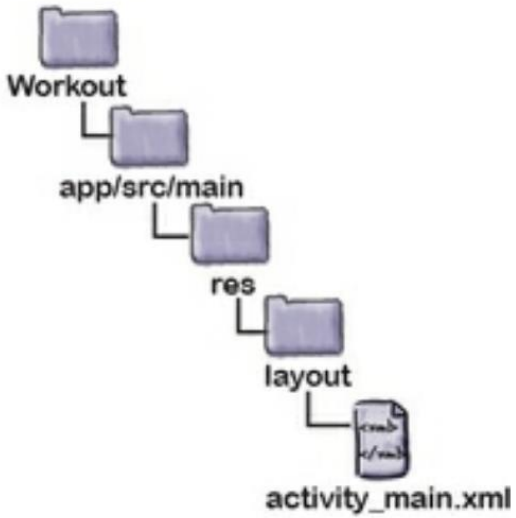
```
public class WorkoutListFragment extends ListFragment {
@Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {
        String[] names = new String[Workout.workouts.length];

        for (int i = 0; i < names.length; i++) {
            names[i] = Workout.workouts[i].getName();
        }

        ArrayAdapter<String> adapter = new ArrayAdapter<>(inflater.getContext(),
                android.R.layout.simple_list_item_1, names);
        setListAdapter(adapter);

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

Use WorkoutListFragment in activity_main.xml

Our layout only contains a single fragment, so we can get rid of the LinearLayout.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:padding="16dp"
    android:orientation="vertical"
    tools:context="com.hfad.workout.MainActivity">

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:onClick="onShowDetails"
        android:text="@string/details_button" />
</LinearLayout>
```

We no longer need this button.



Here's the fragment.

```
<fragment xmlns:android="http://schemas.android.com/apk/res/android"
    android:name="com.hfad.workout.WorkoutListFragment"
    android:layout_width="match_parent"
    android:layout_height="match_parent"/>
```

Click on Fragments

1. **Define the interface.**
   We'll define the listener's interface in `WorkoutListFragment`. We're defining the interface here, as its purpose is to allow `WorkoutListFragment` to communicate with any activity.

2. **Register the listener (in this case MainActivity) when WorkoutListFragment gets attached to it.**
   This will give `WorkoutListFragment` a reference to `MainActivity`.

3. **Tell the listener when an item gets clicked.**
   `MainActivity` will then be able to respond to the click.

You need to go through similar steps to these whenever you have a fragment that needs to communicate with the activity it's attached to.

WorkoutListFragment.java

```
public class WorkoutListFragment extends ListFragment {
    static interface Listener {
        void itemClicked(long id);
    };
    private Listener listener;
    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {
        String[] names = new String[Workout.workouts.length];
        for (int i = 0; i < names.length; i++) {
            names[i] = Workout.workouts[i].getName();
        }
        ArrayAdapter<String> adapter = new ArrayAdapter<>(inflater.getContext(),
                android.R.layout.simple_list_item_1, names);
        setListAdapter(adapter);

        return super.onCreateView(inflater, container, savedInstanceState);
    }
    @Override
    public void onAttach(Context context) {
        super.onAttach(context);
        this.listener = (Listener)context;
    }
    @Override
    public void onListItemClick(ListView listView, View itemView, int position, long id) {
        if (listener != null) {
            listener.itemClicked(id);
        }
    }
}
```

Add FrameLayout in “activity_main.xml”

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    >
    <fragment
        android:name= “org.aiotlab.workout.WorkoutListFragment"
        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"
        />
</LinearLayout>
```

MainActivity Needs to Implement the Interface

```
public class MainActivity extends AppCompatActivity implements WorkoutListFragment.Listener {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
    @Override
    public void itemClicked(long id) {
        View fragmentContainer = findViewById(R.id.fragment_container);
        if (fragmentContainer != null) {
            WorkoutDetailFragment details = new WorkoutDetailFragment();
            FragmentTransaction ft = getSupportFragmentManager().beginTransaction();
            details.setWorkout(id);
            ft.replace(R.id.fragment_container, details);
            ft.setTransition(FragmentTransaction.TRANSIT_FRAGMENT_FADE);
            ft.addToBackStack(null);
            ft.commit();
        }
        else {
            Intent intent = new Intent(this, DetailActivity.class);
            intent.putExtra(DetailActivity.EXTRA_WORKOUT_ID, (int)id);
            startActivity(intent);
        }
    }
}
```

DetailActivity.java

```
public class DetailActivity extends AppCompatActivity {
    public static final String EXTRA_WORKOUT_ID = "id";

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_detail);
        WorkoutDetailFragment frag = (WorkoutDetailFragment)
                getSupportFragmentManager().findFragmentById(R.id.detail_frag);
        int workoutId = (int) getIntent().getExtras().get(EXTRA_WORKOUT_ID);
        frag.setWorkout(workoutId);
    }
}
```

Final Result