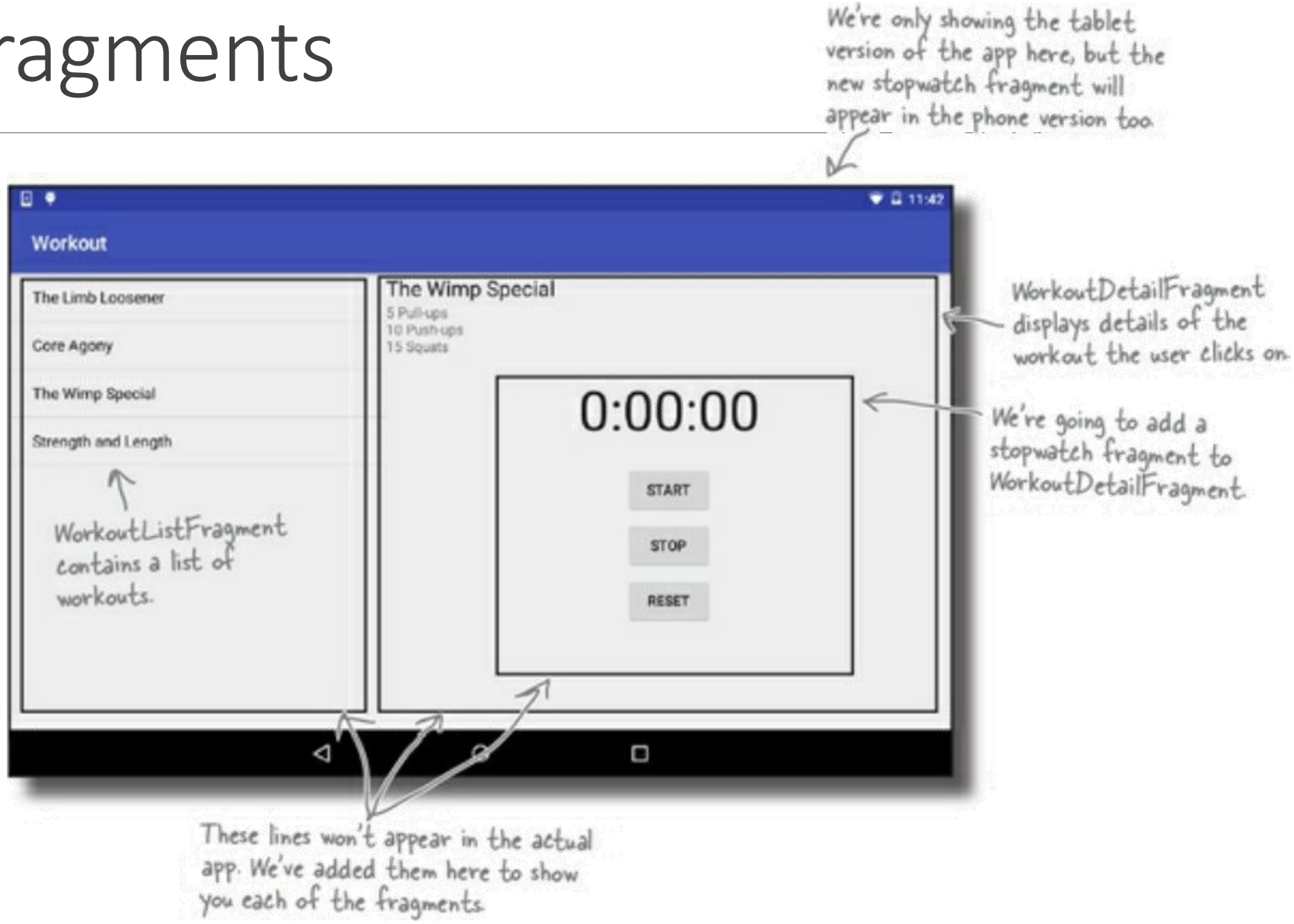
Lab 9 – Dynamic Fragments

KUAN-TING LAI

2020/11/30

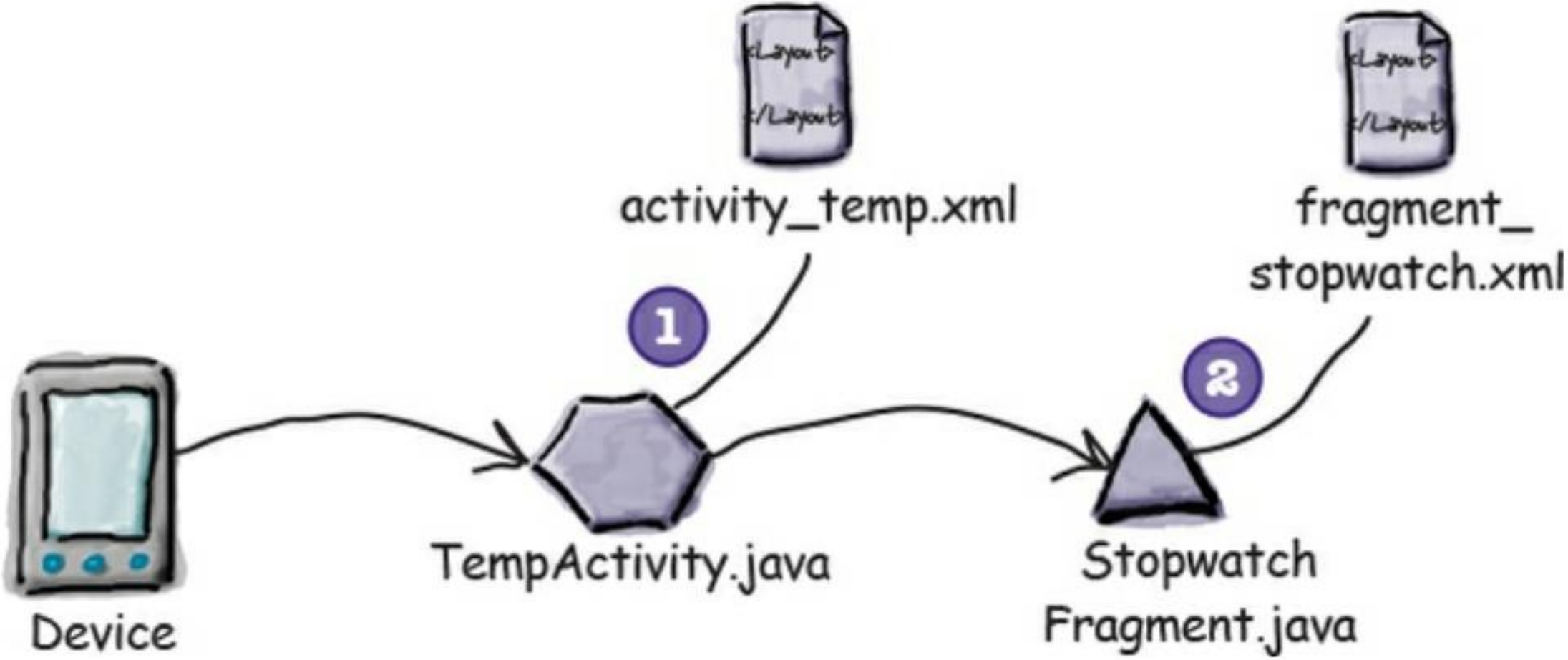
Dynamic Fragments

- Work App:
  - Workout List + Stopwatch
- This work is based on Lab 8: Fragments



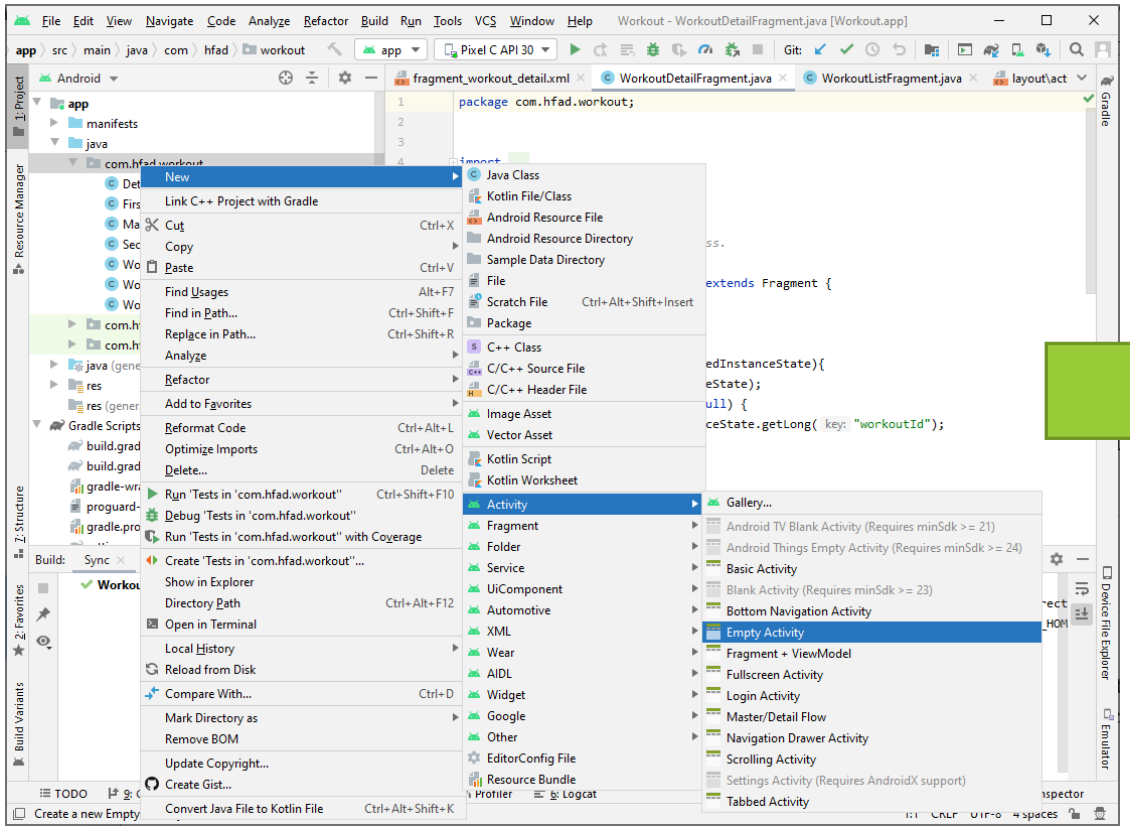
Create StopwatchFragment.java

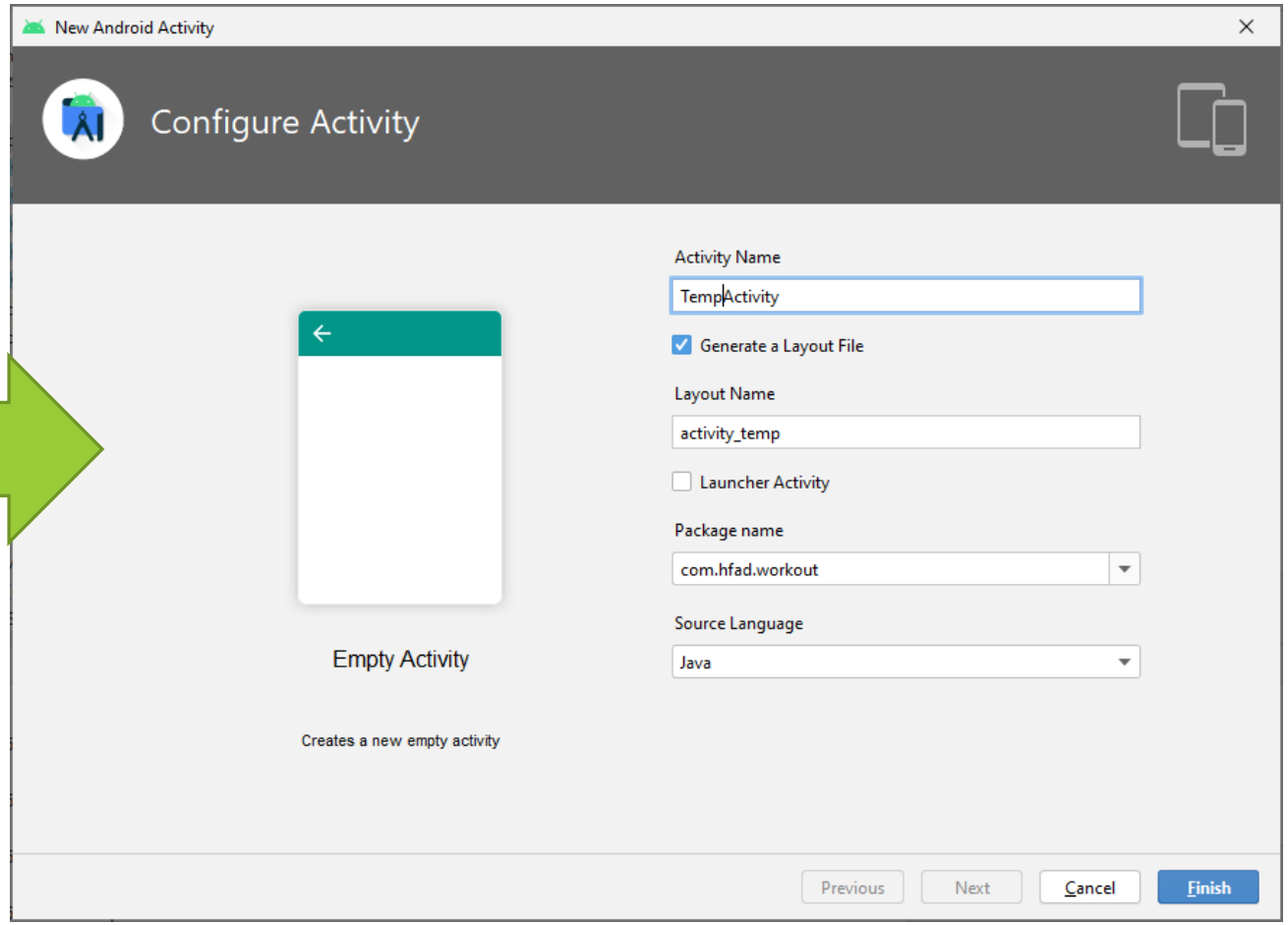
- Create TempActivity.java to test StopwatchFragment



Create TempActivity.java

- New a Empty Activity

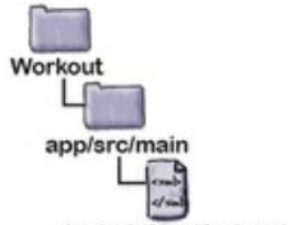




Make TempActivity as Start Activity

- Modify AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.hfad.workout">

    <application
        ...
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity android:name=".DetailActivity" />
        <activity android:name=".TempActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```



This bit specifies that it's the main activity of the app.

This says the activity can be used to launch the app.

Create StartWatchFragment

- Modify code to fit Fragment lifecycles

Lifecycle method	Activity	Fragment
onAttach()		✓
onCreate()	✓	✓
onCreateView()		✓
onActivityCreated()		✓
onStart()	✓	✓
onPause()	✓	✓
onResume()	✓	✓
onStop()	✓	✓
onDestroyView()		✓
onRestart()	✓	
onDestroy()	✓	✓
onDetach()		✓

StopWatch Activity Code (2-1)

```
public class StopwatchActivity extends Activity {
    //Number of seconds displayed on the stopwatch.
    private int seconds = 0;
    //Is the stopwatch running?
    private boolean running;
    private boolean wasRunning;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_stopwatch);
        if (savedInstanceState != null) {
            seconds = savedInstanceState.getInt("seconds");
            running = savedInstanceState.getBoolean("running");
            wasRunning = savedInstanceState.getBoolean("wasRunning");
        }
        runTimer();
    }

    @Override
    protected void onPause() {
        super.onPause();
        wasRunning = running;
        running = false;
    }
```

The number of seconds that have passed

running says whether the stopwatch is running.
wasRunning says whether the stopwatch was running before the stopwatch was paused.

If the activity was destroyed and recreated, restore the state of the variables from the savedInstanceState Bundle.

Start the runTimer() method.

Stop the stopwatch if the activity is paused.

StopWatch Activity Code (2-2)

```
@Override
protected void onResume() {
    super.onResume();
    if (wasRunning) {
        running = true;
    }
}

@Override
public void onSaveInstanceState(Bundle savedInstanceState) {
    savedInstanceState.putInt("seconds", seconds);
    savedInstanceState.putBoolean("running", running);
    savedInstanceState.putBoolean("wasRunning", wasRunning);
}

public void onClickStart(View view) {
    running = true;
}

public void onClickStop(View view) {
    running = false;
}

public void onClickReset(View view) {
    running = false;
    seconds = 0;
}

private void runTimer() {
    final TextView timeView = (TextView)findViewById(R.id.time_view);
    final Handler handler = new Handler();
    handler.post(new Runnable() {
        @Override
        public void run() {
            int hours = seconds/3600;
            int minutes = (seconds%3600)/60;
            int secs = seconds%60;
            String time = String.format(Locale.getDefault(),
                    "%d:%02d:%02d", hours, minutes, secs);
            timeView.setText(time);
            if (running) {
                seconds++;
            }
            handler.postDelayed(this, 1000);
        }
    });
}
}
```

Start the stopwatch if the activity is resumed.

Save the activity's state before the activity is destroyed.

Start, stop, or reset the stopwatch depending on which button is clicked.

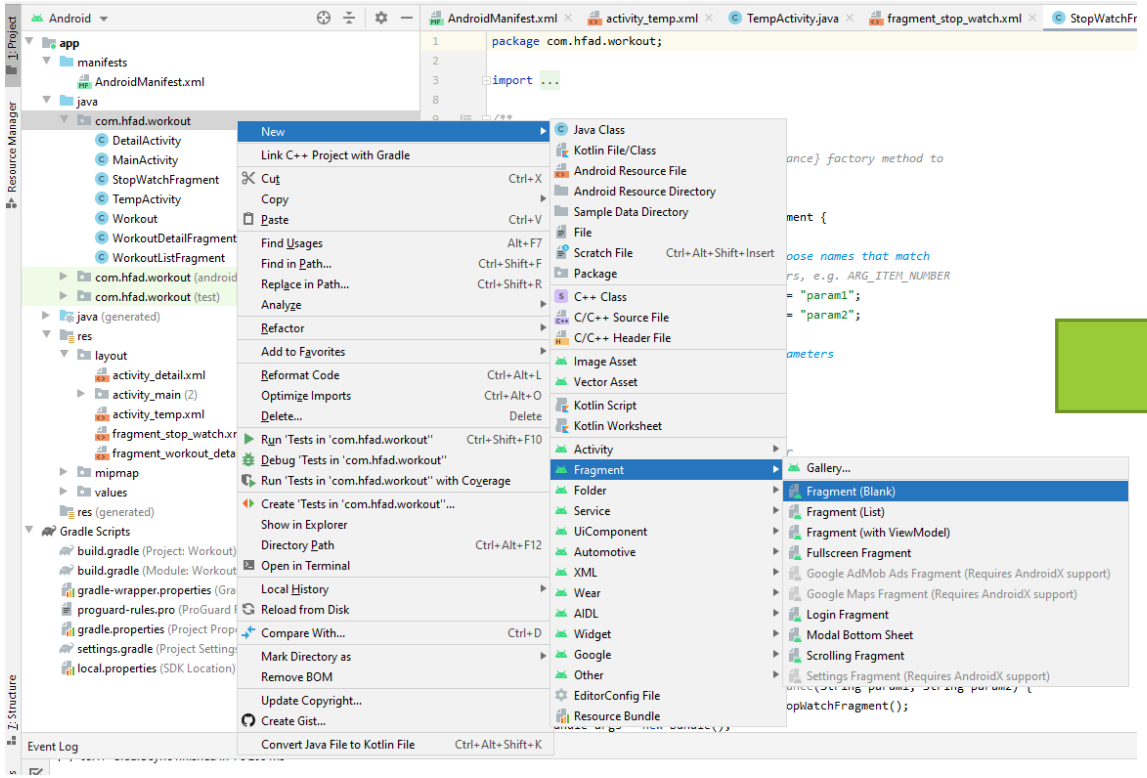
Use a Handler to post code to increment the number of seconds and update the text view every second

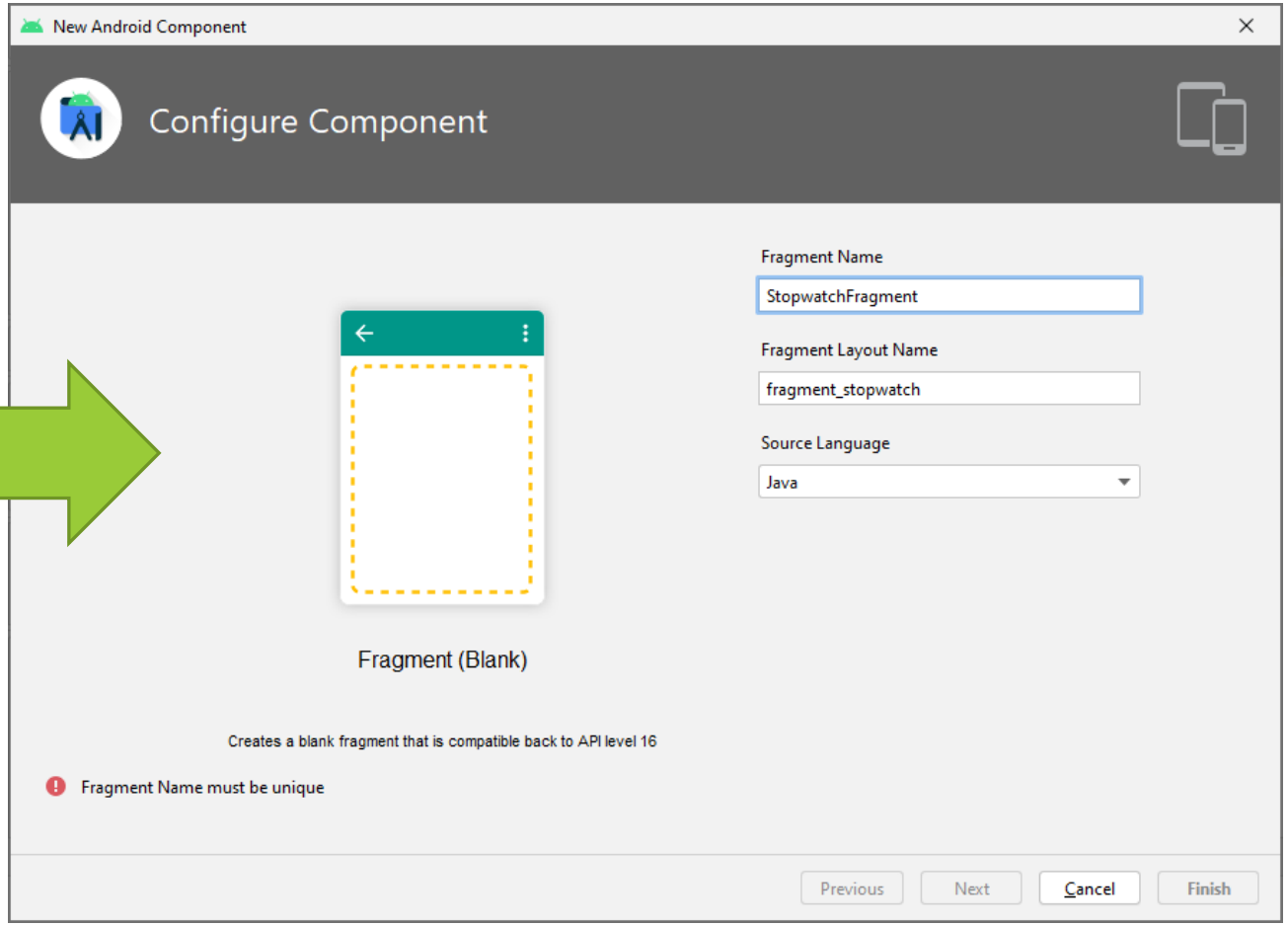
Change From Activity to Fragment

```
public class StopwatchFragment extends Fragment {
    //Number of seconds displayed on the stopwatch.
    private int seconds = 0;
    //Is the stopwatch running?
    private boolean running;
    private boolean wasRunning;

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        if (savedInstanceState != null) {
            seconds = savedInstanceState.getInt("seconds");
            running = savedInstanceState.getBoolean("running");
            wasRunning = savedInstanceState.getBoolean("wasRunning");
        }
    }

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

    @Override
    public void onPause() {
        super.onPause();
        wasRunning = running;
        running = false;
    }
```

```
    @Override
    public void onResume() {
        super.onResume();
        if (wasRunning) {
            running = true;
        }
    }

    @Override
    public void onSaveInstanceState(Bundle savedInstanceState) {
        savedInstanceState.putInt("seconds", seconds);
        savedInstanceState.putBoolean("running", running);
        savedInstanceState.putBoolean("wasRunning", wasRunning);
    }

    public void onClickStart(View view) {
        running = true;
    }

    public void onClickStop(View view) {
        running = false;
    }

    public void onClickReset(View view) {
        running = false;
        seconds = 0;
    }

    private void runTimer(View view) {
        final TextView timeView = (TextView) view.findViewById(R.id.time_view);
        final Handler handler = new Handler();
        handler.post(new Runnable() {
            @Override
            public void run() {
                int hours = seconds/3600;
                int minutes = (seconds%3600)/60;
                int secs = seconds%60;
                String time = String.format(Locale.getDefault(),
                        "%d:%02d:%02d", hours, minutes, secs);
                timeView.setText(time);
                if (running) {
                    seconds++;
                }
                handler.postDelayed(this, 1000);
            }
        });
    }
```

New StopwatchFragment





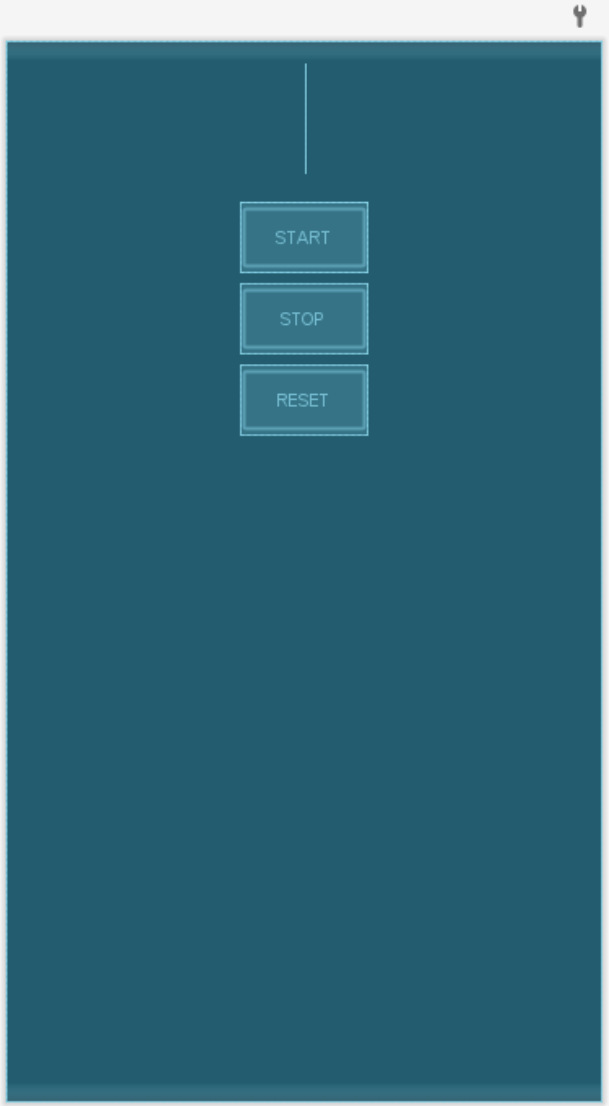
Code Split Design

```xml
<?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:orientation="vertical"
    android:padding="16dp"
    >
    <TextView
        android:id="@+id/time_view"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center_horizontal"
        android:textAppearance="@android:style/TextAppearance.Large"
        android:textSize="56sp"
        />
    <Button
        android:id="@+id/start_button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center_horizontal"
        android:layout_marginTop="20dp"
        android:text="Start"
        />
    <Button
        android:id="@+id/stop_button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center_horizontal"
        android:layout_marginTop="8dp"
        android:text="Stop"
        />
    <Button
        android:id="@+id/reset_button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center_horizontal"
        android:layout_marginTop="8dp"
        android:text="Reset"
        />
</LinearLayout>
```

Palette

Pixel 29 AppTheme Default (en-us)

Attributes



Component Tree

1:1

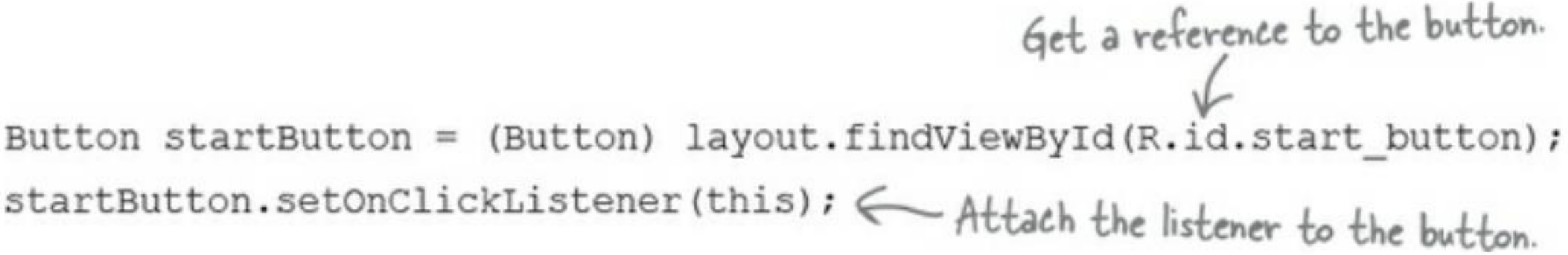
fragment_stopwatch.xml

```
<?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:orientation="vertical"
    android:padding="16dp"
    >
    <TextView
        android:id="@+id/time_view"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center_horizontal"
        android:textAppearance="@android:style/TextAppearance.Large"
        android:textSize="56sp"
        />
    <Button
        android:id="@+id/start_button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center_horizontal"
        android:layout_marginTop="20dp"
        android:text="Start"
        />
    <Button
        android:id="@+id/stop_button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center_horizontal"
        android:layout_marginTop="8dp"
        android:text="Stop"
        />
    <Button
        android:id="@+id/reset_button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center_horizontal"
        android:layout_marginTop="8dp"
        android:text="Reset"
        />
</LinearLayout>
```

Stopwatch.java (3-1)

```
public class StopeatchFragment extends Fragment implements View.OnClickListener {
    //Number of seconds displayed on the stopwatch.
    private int seconds = 0;
    //Is the stopwatch running?
    private boolean running;
    private boolean wasRunning;
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        if (savedInstanceState != null) {
            seconds = savedInstanceState.getInt("seconds");
            running = savedInstanceState.getBoolean("running");
            wasRunning = savedInstanceState.getBoolean("wasRunning");
        }
    }
    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {
        View layout = inflater.inflate(R.layout.fragment_stopwatch, container, false);
        runTimer(layout);
        Button startButton = (Button)layout.findViewById(R.id.start_button);
        startButton.setOnClickListener(this);
        Button stopButton = (Button)layout.findViewById(R.id.stop_button);
        stopButton.setOnClickListener(this);
        Button resetButton = (Button)layout.findViewById(R.id.reset_button);
        resetButton.setOnClickListener(this);
        return layout;
    }
……
}
```

View.findViewById()



Stopwatch.java (3-2)

```
……
@Override
public void onClick(View v) {
    switch (v.getId()) {
        case R.id.start_button:
            onClickStart();
            break;
        case R.id.stop_button:
            onClickStop();
            break;
        case R.id.reset_button:
            onClickReset();
            break;
    }
}
@Override
public void onPause() {
    super.onPause();
    wasRunning = running;
    running = false;
}
@Override
public void onResume() {
    super.onResume();
    if (wasRunning) {
        running = true;
    }
}
……
```

Difference Between Activity & Fragment

Our methods no longer need to be public, so we can make them private.

```
private void onClickStart() {
}
```

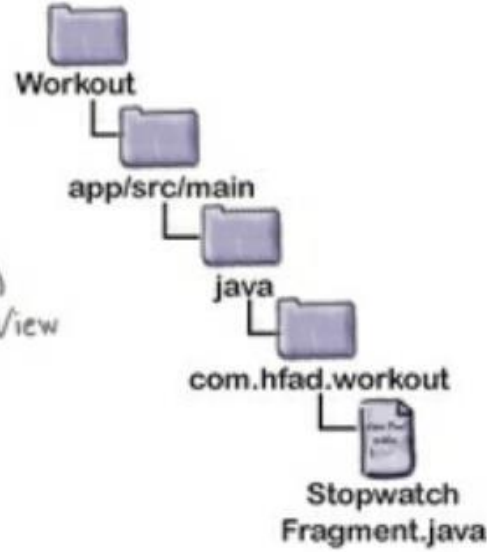
We no longer need the View parameter.

So let's update our fragment code. Change the onClickStart(), onClickStop(), and onClickReset() methods in *StopwatchFragment.java* to match ours:

Change the methods to private.

```
...
    private void onClickStart(View view) {
        running = true;
    }

    private void onClickStop(View view) {
        running = false;
    }

    private void onClickReset(View view) {
        running = false;
        seconds = 0;
    }
...
```

Remove the View parameters.



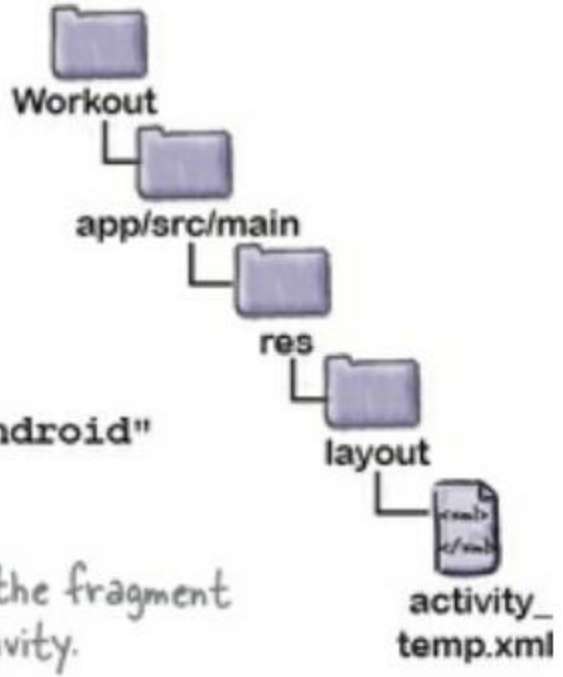
Stopwatch.java (3-3)

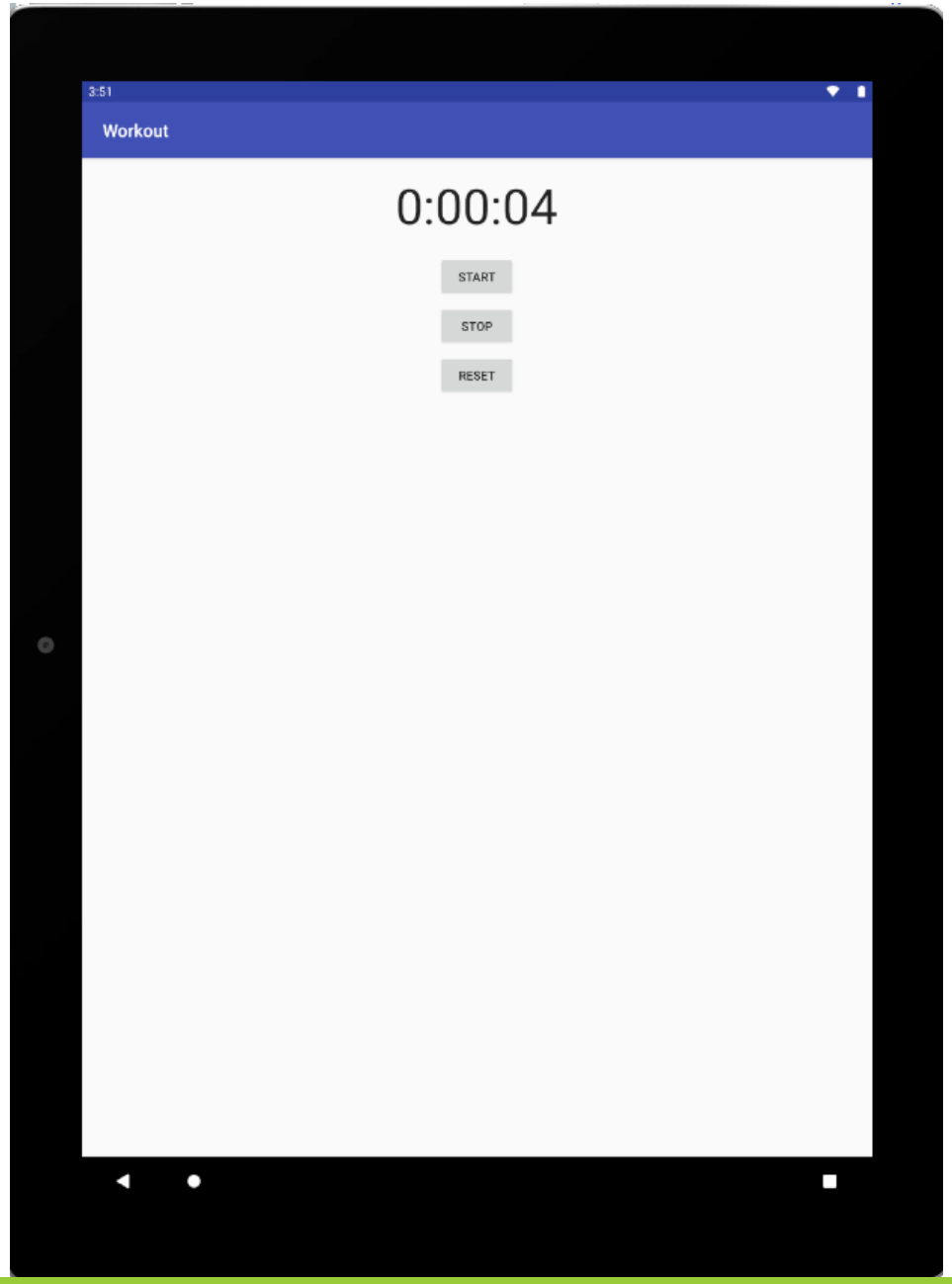
```
@Override
public void onSaveInstanceState(Bundle savedInstanceState) {
    savedInstanceState.putInt("seconds", seconds);
    savedInstanceState.putBoolean("running", running);
    savedInstanceState.putBoolean("wasRunning", wasRunning);
}
private void onClickStart() {
    running = true;
}
private void onClickStop() {
    running = false;
}
private void onClickReset() {
    running = false;
    seconds = 0;
}
private void runTimer(View view) {
    final TextView timeView = (TextView) view.findViewById(R.id.time_view);
    final Handler handler = new Handler();
    handler.post(new Runnable() {
        @Override
        public void run() {
            int hours = seconds/3600;
            int minutes = (seconds%3600)/60;
            int secs = seconds%60;
            String time = String.format(Locale.getDefault(),
                    "%d:%02d:%02d", hours, minutes, secs); timeView.setText(time);
            if (running) {
                seconds++;
            }
            handler.postDelayed(this, 1000);
        }
    });
}
```

Add StopwatchFragment to TempActivity

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

This adds the fragment to the activity.

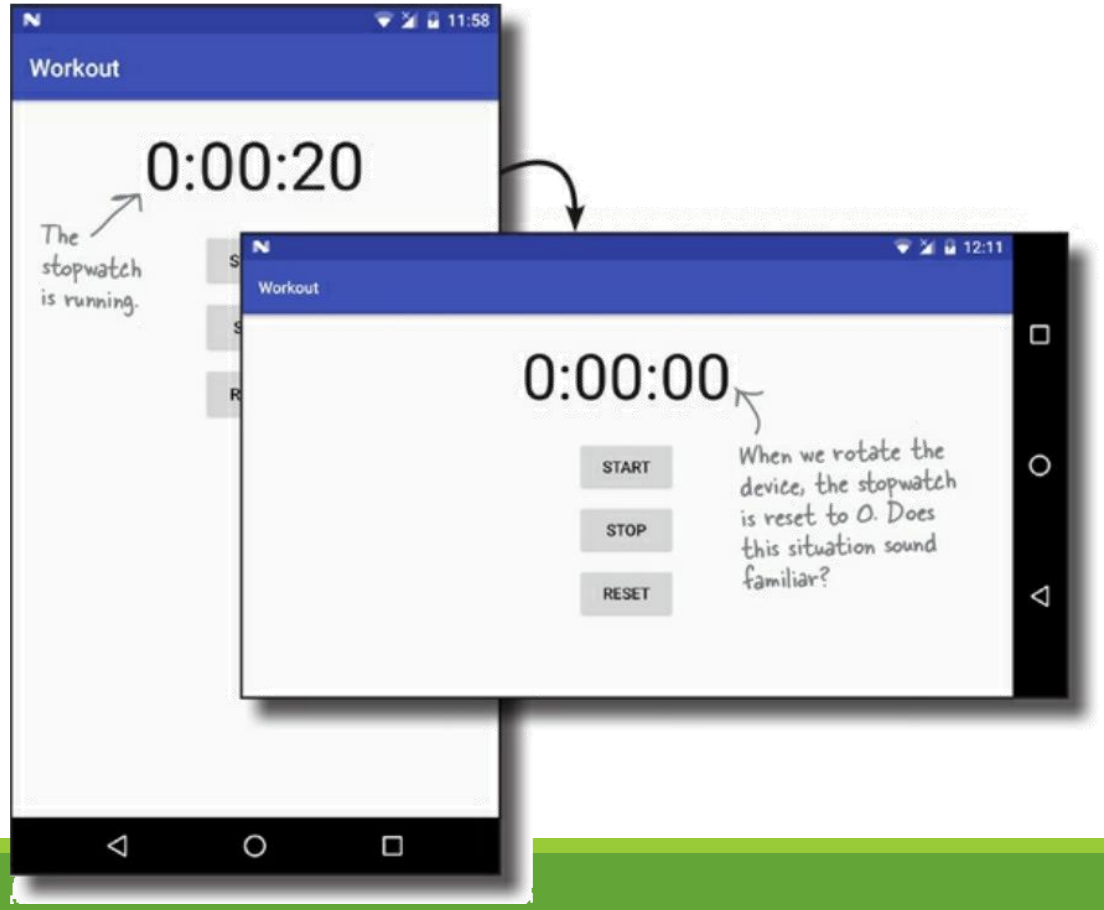




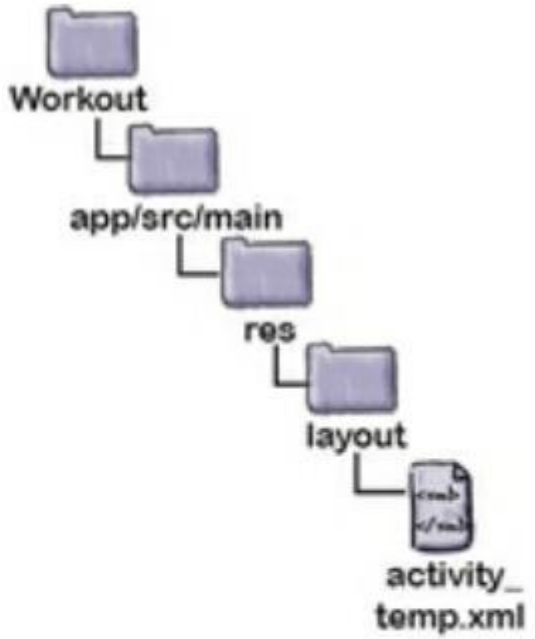
Test StopwatchFragment

Rotation Time-lost Problem

- Lost the time state when rotated
- This time, it’s caused by adding StopwatchFragment to TempActivity



Use FrameLayout

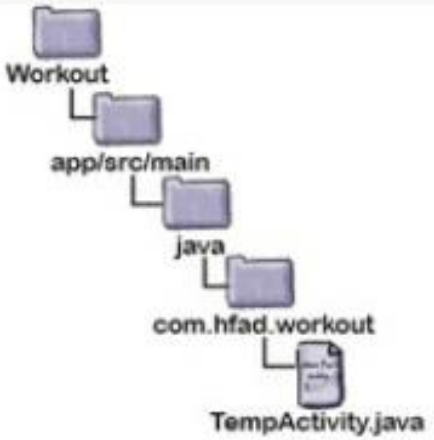



```
<?xml version="1.0" encoding="utf-8"?>
<fragment FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:name="com.hfad.workout.StopwatchFragment"
    android:id="@+id/stopwatch_container"
    android:layout_width="match_parent"
    android:layout_height="match_parent"/>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

Replace the fragment with a FrameLayout.

Delete this line.

Add code to onCreate function




```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_temp);
    if (savedInstanceState == null) {
        StopwatchFragment stopwatch = new StopwatchFragment();
        FragmentTransaction ft = getSupportFragmentManager().beginTransaction();
        ft.add(R.id.stopwatch_container, stopwatch);
        ft.addToBackStack(null);
        ft.setTransition(FragmentTransaction.TRANSIT_FRAGMENT_FADE);
        ft.commit();
    }
}
```

This begins the fragment transaction. We need to use getSupportFragmentManager(), not getFragmentManager(), as we're using fragments from the Support Library.

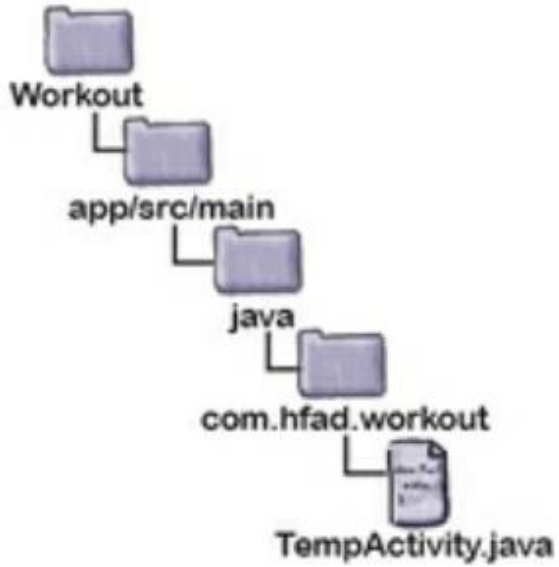
Add an instance of StopwatchFragment to TempActivity's layout

Add the transaction to the back stack

Commit the transaction

You need to import the FragmentTransaction class from the Support Library.

```
package com.hfad.workout;

import android.support.v4.app.FragmentTransaction;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;

public class TempActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_temp);
        if (savedInstanceState == null) {
            StopwatchFragment stopwatch = new StopwatchFragment();
            FragmentTransaction ft = getSupportFragmentManager().beginTransaction();
            ft.add(R.id.stopwatch_container, stopwatch);
            ft.addToBackStack(null);
            ft.setTransition(FragmentTransaction.TRANSIT_FRAGMENT_FADE);
            ft.commit();
        }
    }
}
```



We only want to add the fragment if the activity isn't being recreated after having been destroyed.

Begin the fragment transaction.

Add the stopwatch, and add the transaction to the back stack.

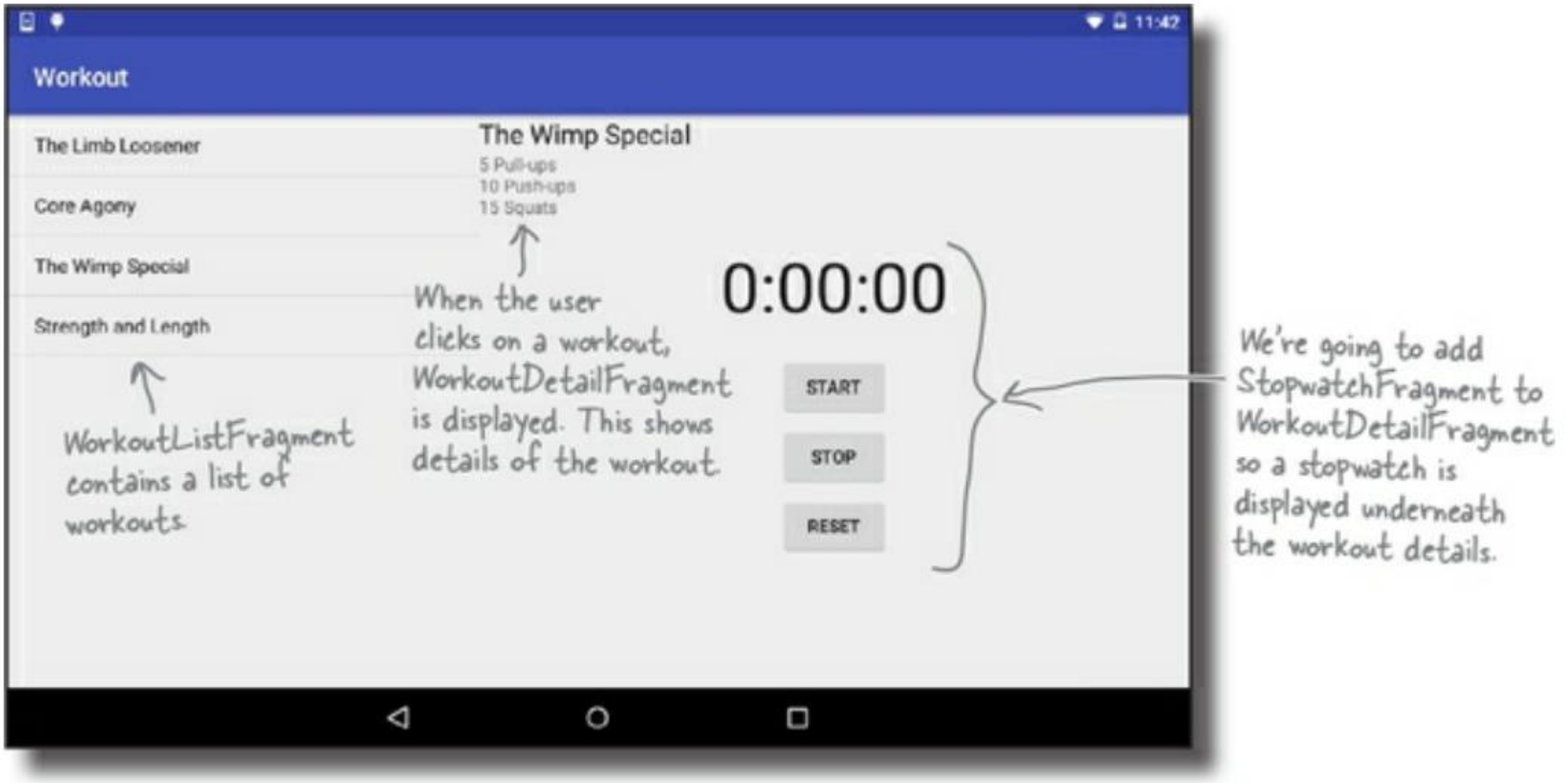
Set the fragment transition to fade in and out.

Commit the transaction. This applies the changes.

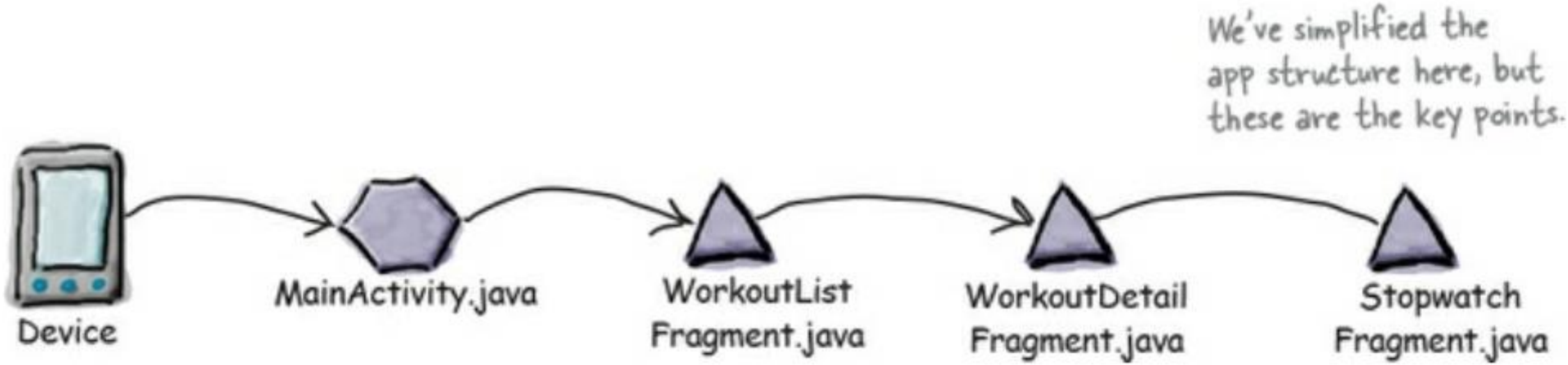
TempActivity.java

```
import android.support.v4.app.FragmentTransaction;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;

public class TempActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_temp);
        if (savedInstanceState == null) {
            StopwatchFragment stopwatch = new StopwatchFragment();
            FragmentTransaction ft = getSupportFragmentManager().beginTransaction();
            ft.add(R.id.stopwatch_container, stopwatch);
            ft.addToBackStack(null);
            ft.setTransition(FragmentTransaction.TRANSIT_FRAGMENT_FADE);
            ft.commit();
        }
    }
}
```

Add StopwatchFragment to Workout App



Workout App Flow



AndroidManifest.xml

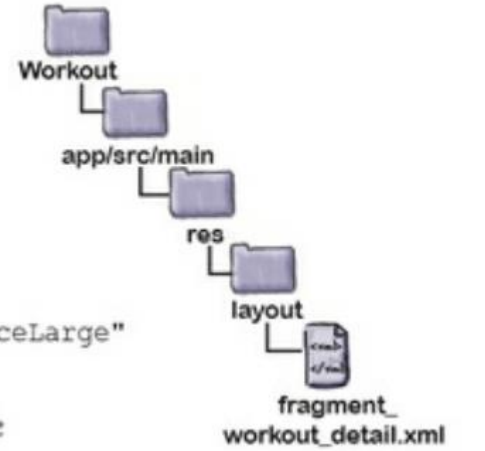

```
...
    <application
        ...
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity android:name=".DetailActivity" />
        <activity android:name=".TempActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
...
```

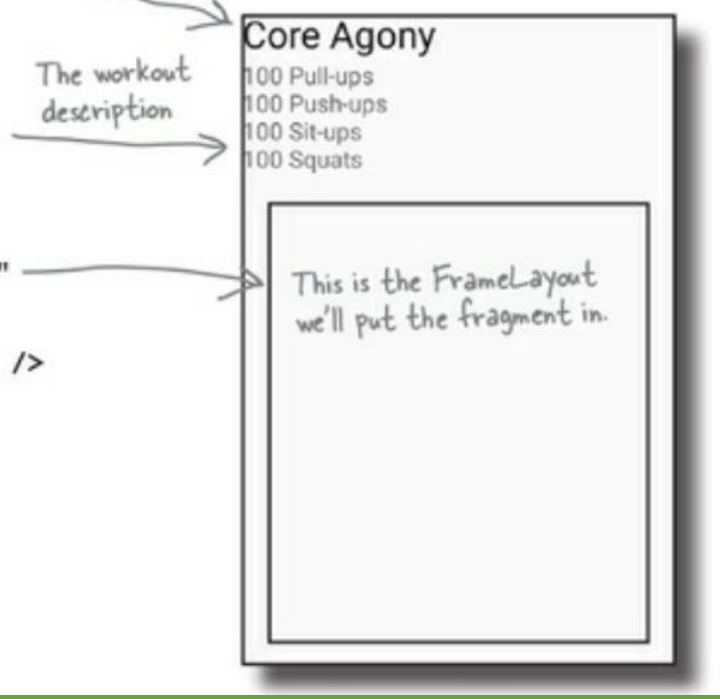
Add an intent filter to start MainActivity when the app is launched.

Remove the intent filter from TempActivity.

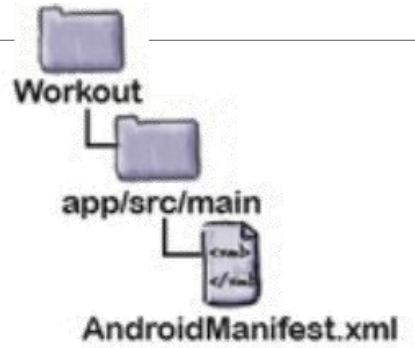
Add FrameLayout in fragment_workout_detail.xml

```
<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" />

    <FrameLayout
        android:id="@+id/stopwatch_container"
        android:layout_width="match_parent"
        android:layout_height="match_parent" />
</LinearLayout>
```





Set MainActivity.xml as Launcher

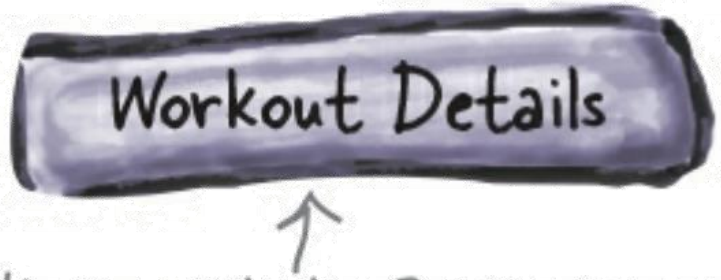



```
...
    <application
        ...
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity android:name=".DetailActivity" />
        <activity android:name=".TempActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
...
```

Add an intent filter to start MainActivity when the app is launched.

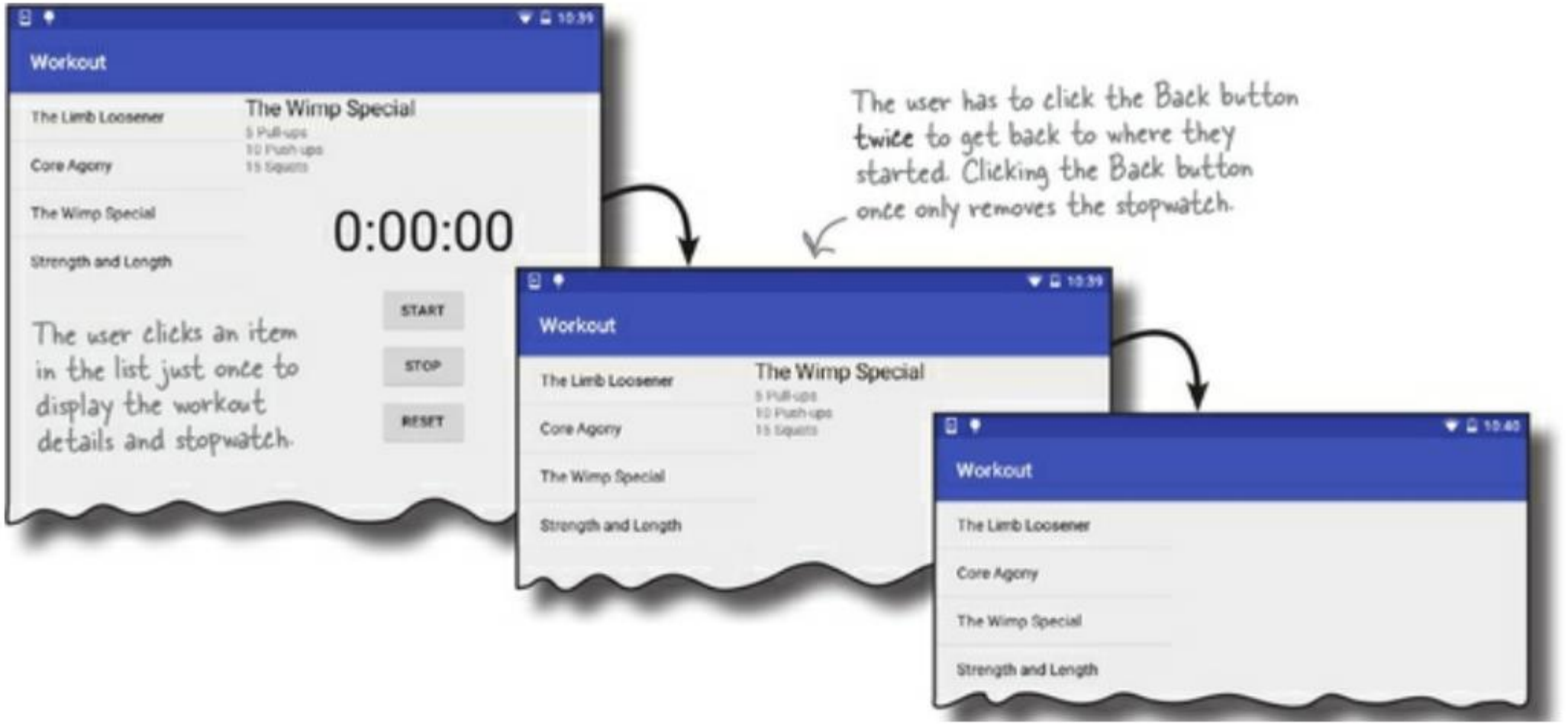
Remove the intent filter from TempActivity.

A transaction for WorkoutDetailFragment is added to the back stack, followed by a separate transaction for StopwatchFragment.



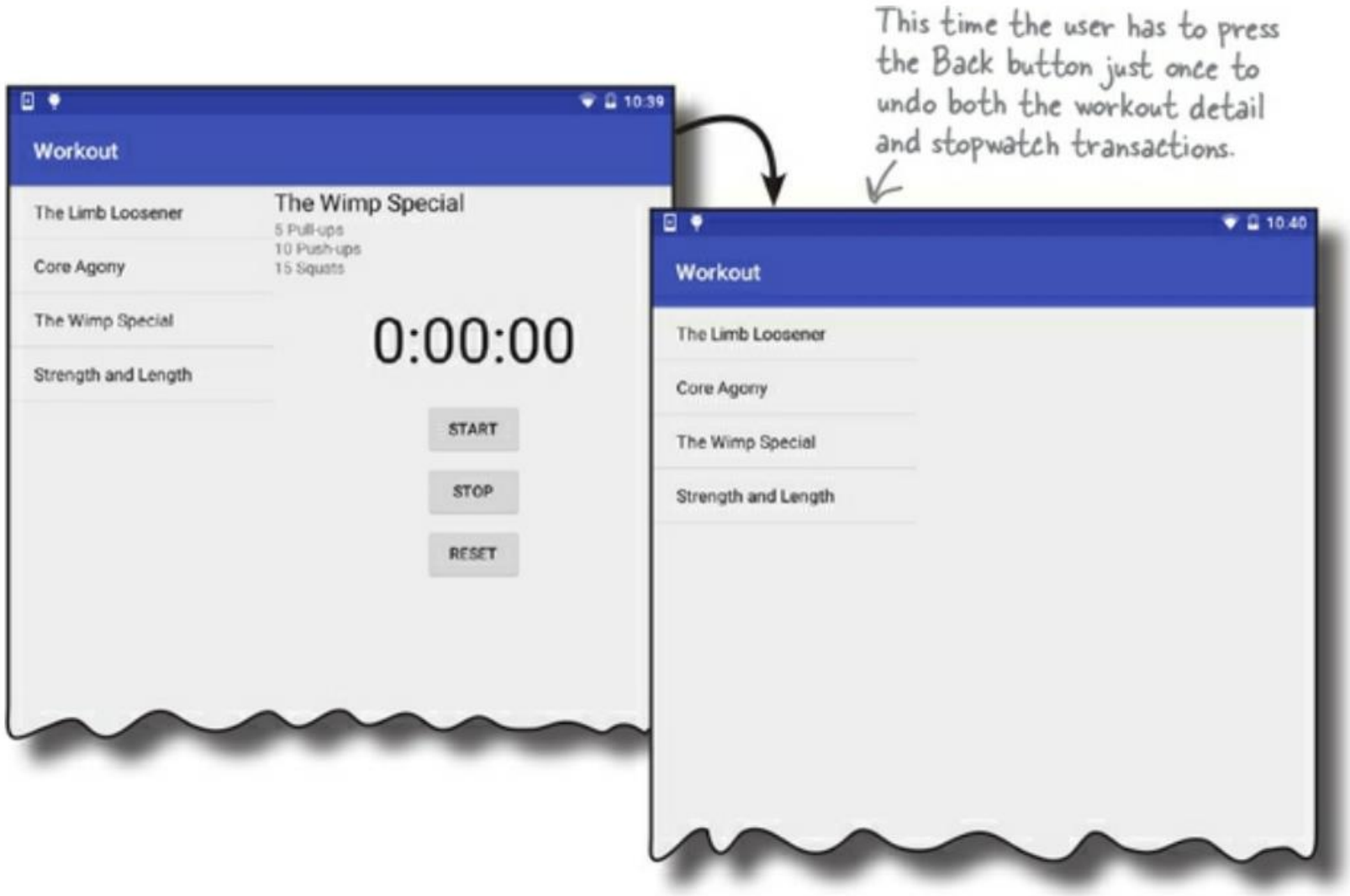
When the user hits the Back button, the StopwatchFragment transaction is popped off the back stack. The transaction for WorkoutDetailFragment stays on the back stack.

Back Button Clicking Issue

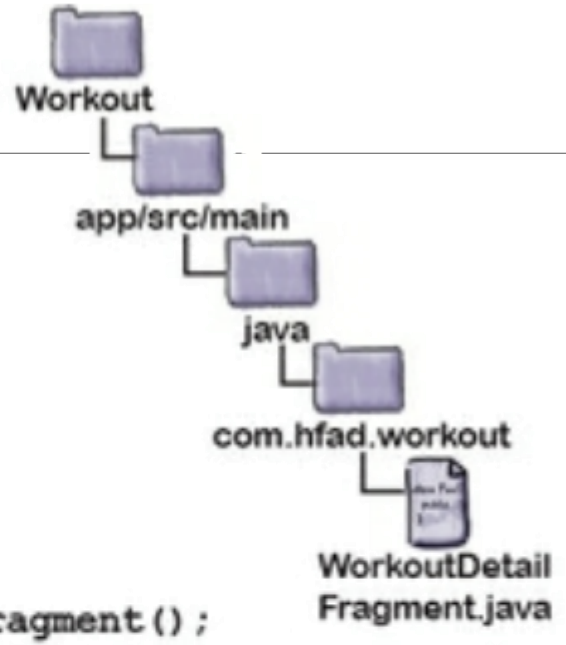


Nested Fragments

Click Back Button Once to Undo Two Fragments



WorkoutDetailFragment.java

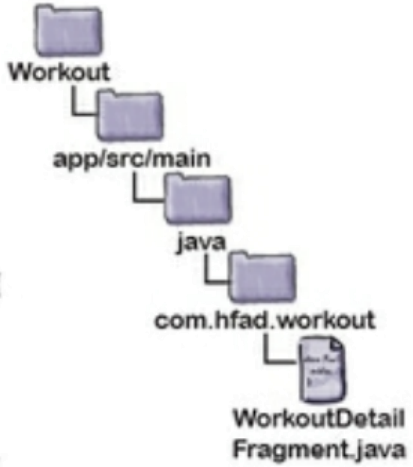



```
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    if (savedInstanceState == null) {
        StopwatchFragment stopwatch = new StopwatchFragment();
        FragmentTransaction ft = getChildFragmentManager().beginTransaction();
        ft.add(R.id.stopwatch_container, stopwatch);
        ft.addToBackStack(null);
        ft.setTransition(FragmentTransaction.TRANSIT_FRAGMENT_FADE);
        ft.commit();
    } else {
        workoutId = savedInstanceState.getLong("workoutId");
    }
}
```

We're using getChildFragmentManager() instead of getSupportFragmentManager(). Apart from that, the code is the same as we had earlier.

```
package com.hfad.workout;

import android.support.v4.app.Fragment;
import android.support.v4.app.FragmentTransaction;
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 void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        if (savedInstanceState != null) {
        if (savedInstanceState == null) {
            StopwatchFragment stopwatch = new StopwatchFragment();
            FragmentTransaction ft = getChildFragmentManager().beginTransaction();
            ft.add(R.id.stopwatch_container, stopwatch);
            ft.addToBackStack(null);
            ft.setTransition(FragmentTransaction.TRANSIT_FRAGMENT_FADE);
            ft.commit();
        } else {
            workoutId = savedInstanceState.getLong("workoutId");
        }
    }
```

You need to import the FragmentTransaction class from the Support Library.



Delete this line.

We only want to add the fragment if the activity isn't being recreated after having been destroyed.

Begin the fragment transaction.

Add the stopwatch, and add the transaction to the back stack.

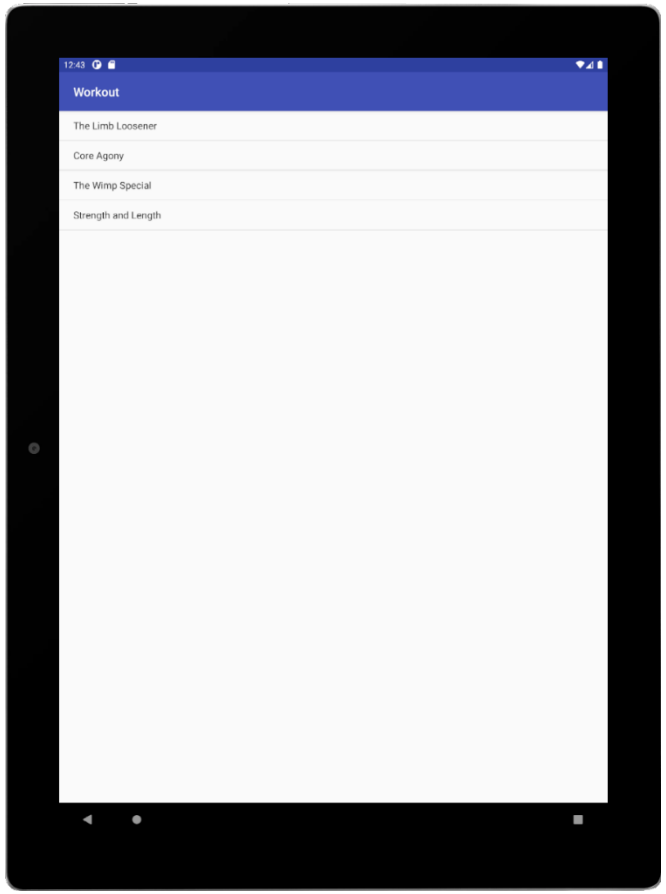
Commit the transaction.

Set the fragment transition to fade in and out.

WorkoutDetailFragment.java

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

    @Override
    public void onCreate(Bundle savedInstanceState){
        super.onCreate(savedInstanceState);
        if (savedInstanceState == null) {
            StopwatchFragment stopwatch = new StopwatchFragment();
            FragmentTransaction ft = getChildFragmentManager().beginTransaction();
            ft.add(R.id.stopwatch_container, stopwatch);
            ft.addToBackStack(null);
            ft.setTransition(FragmentTransaction.TRANSIT_FRAGMENT_FADE);
            ft.commit();
        } else {
            workoutId = savedInstanceState.getLong("workoutId");
        }
    }
……
……
}
```

Final Result



Click