**FLAG QUIZ**

[I) TIÊU ĐIỂM: 2](#_Toc464024871)

[II) CÀI ĐẶT: 2](#_Toc464024872)

[1. Cấu hình SE7 3](#_Toc464024873)

[2. Thêm Flag Images vào Project 3](#_Toc464024874)

[3. Thêm giá trị file string.xml 4](#_Toc464024875)

[4. Tạo file arrays.xml 5](#_Toc464024876)

[5. Sửa file colors.xml 6](#_Toc464024877)

[6. Tạo button\_text\_color.xml 7](#_Toc464024878)

[7. Chỉnh sửa menu\_main.xml 8](#_Toc464024879)

[8. Tạo the Flag Shake Animation 9](#_Toc464024880)

[9. Tạo Preferences.xml cho trang cài đặt App 10](#_Toc464024881)

[10. Thêm Classes SettingsActivity và SettingsActivityFragment 11](#_Toc464024882)

[III) THIẾT KẾ ỨNG DỤNG 12](#_Toc464024883)

[1. Thiết kế fragment\_main.xml Layout 13](#_Toc464024884)

[2. MainActivity Class 19](#_Toc464024885)

[3. MainActivityFragment Class 23](#_Toc464024886)

[4. SettingsActivity Class 32](#_Toc464024887)

[5. SettingsActivityFragment Class 32](#_Toc464024888)

[6. AndroidManifest.xml 33](#_Toc464024889)

# **TIÊU ĐIỂM:**

\_**Fragment**

\_**Preferences**

\_Animation

\_AlertDialog

# **CÀI ĐẶT:**

\_Tạo 1 project Basic Activity

\_API chọn 23 – Android 6.0

## Cấu hình SE7

**1.** Chuột phải vào ***app*** folder và chọn ***Open Module Settings*** để mở cửa sổ  
***Project Structure***.

**2.** Chọn tab ***Properties*.**

**3.** Tại 2 trường ***Source Compatibility*** và ***Target Compatibility***, Chọn  
1.7, và click ***OK****.*

## Thêm Flag Images vào Project

**1.** Chuột phải ***app*** folder, chọn ***New* > *Folder* > *Assets Folder***. Cửa sổ ***Customize the Activity*** hiện lên, nhấn **Finish**.

**2.** Sau đó, ta sẽ kéo toàn bộ file trong images/FlagQuiz vào folder Asset.

## Thêm giá trị file string.xml

**1.** Mở ***strings.xml*** trong ***res/values***.

**2.** Phía trên bên phải, nhấp mở ***Open Editor*,** cửa sổ ***Translations Editor*** hiện lên.

**3.** Nhấp vào  phía trên góc trái.

**4.** Tại trường ***Key***: enter number\_of\_choices,   
trường ***Default Value:*** Number of Choices for the,   
sau đó nhấn OK.

**5.** Lặp lại 4 bước trên với bảng sau:

|  |  |
| --- | --- |
| ***Key*** | ***Value*** |
| number\_of\_choices\_description | Display 2, 4, 6 or 8 guess buttons |
| world\_regions | Regions |
| world\_regions\_description | Regions to include in the quiz |
| guess\_country | Guess the Country |
| results | %1$d guesses, %2$.02f%% correct |
| incorrect\_answer | Incorrect! |
| default\_region\_message | One region must be selected. Setting North America as the default region. |
| restarting\_quiz | Quiz will restart with your new settings |
| question | Question %1$d of %2$d |
| reset\_quiz | Reset Quiz |
| image\_description | Image of the current flag in the quiz |
| default\_region | North\_America |

## Tạo file arrays.xml

**1.** Chuột phải ***values*** folder, chọn select ***New* > *Values resource file***

**2.** Điền ***arrays.xml*** tại trường ***File name*** và nhấp ***OK*.**

**3.** Mở file ***arrays.xml*** lên, chọn qua text và nhập code sau:

*<?***xml version="1.0" encoding="utf-8"***?>*<**resources**>  
 <**string-array name="regions\_list"**>  
 <**item**>Africa</**item**>  
 <**item**>Asia</**item**>  
 <**item**>Europe</**item**>  
 <**item**>North\_America</**item**>  
 <**item**>Oceania</**item**>  
 <**item**>South\_America</**item**>  
 </**string-array**>  
  
 <**string-array name="regions\_list\_for\_settings"**>  
 <**item**>Africa</**item**>  
 <**item**>Asia</**item**>  
 <**item**>Europe</**item**>  
 <**item**>North America</**item**>  
 <**item**>Oceania</**item**>  
 <**item**>South America</**item**>  
 </**string-array**>  
  
 <**string-array name="guesses\_list"**>  
 <**item**>2</**item**>  
 <**item**>4</**item**>  
 <**item**>6</**item**>  
 <**item**>8</**item**>  
 </**string-array**>  
  
</**resources**>

## Tạo button\_text\_color.xml

**1.** Chuột phải ***res*** folder, chọn ***New* > *Andorid resource file***

**2.** Điền ***button\_text\_color.xml*** tại trường ***File name***.

**3.** Tại ***Resource type*,** chọn ***Color***.

**4.** Nhấp ***OK.***

**5.** Mở file ***button\_text\_color.xml,*** chọn ***text:***

*<?***xml version="1.0" encoding="utf-8"***?>*<**selector xmlns:android="http://schemas.android.com/apk/res/android"**>  
 <**item  
 android:color="@android:color/primary\_text\_dark"  
 android:state\_enabled="true"**/>  
  
 <**item  
 android:color="@android:color/darker\_gray"  
 android:state\_enabled="false"**/>  
  
</**selector**>

## Chỉnh sửa menu\_main.xml

Thêm nút cài đặt vào màn hình:

1. Chọn ***File* > *New* > *Vector Asset***
2. Click chọn hình Icon: 
3. Bảng ***Select Icon*** hiện lên, chọn hình , nhấn ***Next*** -> ***Finish***
4. Trong thư mục ***res/drawable*** , các bạn sẽ thấy file ***ic\_settings\_black\_24dp***
5. Vì thanh bar là màu xanh, nếu để màu icon màu đen sẽ rất xấu, nên ta sẽ đổi màu icon đó thành màu trắng. Click đúp chuột vào file ic\_settings\_black\_24dp, và sửa dòng android:fillColor như sau:

android:fillColor="@android:color/white"

Tiếp theo, ta sẽ chèn icon vào Menu:

1. Trong thử mục ***res/menu***, mở ***menu\_main.xml,*** Ta chọn qua phần ***Text***  để code
2. Trong thẻ ***item,***  thêm dòng sau:

android:icon="@drawable/ic\_settings\_24dp"

và sửa dòng **app:showAsAction:**

app:showAsAction="always"

Hoàn tất, quay lại *design,*  ta sẽ thấy nút *Settings*, ở góc phải trên thanh bar.

## Tạo the Flag Shake Animation

Phần này, ta sẽ tạo hiệu ứng rung lá cờ khi chọn đáp án sai:

1. Chuột phải thư mục ***res***, chọn ***New* > *Android resource file***
2. Điền “incorrect\_shake.xml” tại mục ***File name***
3. Tại mục ***Resource type***, chọn ***Animation***. Và mục ***Directory name***, điền ***anim***.
4. Nhấp ***OK.***

Mở ***incorrect\_shake.xml,*** sửa lại như sau:

*<?***xml version="1.0" encoding="utf-8"***?>*<**set xmlns:android="http://schemas.android.com/apk/res/android"  
 android:interpolator="@android:anim/decelerate\_interpolator"**>  
  
 <**translate android:duration="100" android:fromXDelta="0"  
 android:toXDelta="-5%p"** />  
  
 <**translate android:duration="100" android:fromXDelta="-5%p"  
 android:toXDelta="5%p" android:startOffset="100"** />  
  
 <**translate android:duration="100" android:fromXDelta="5%p"  
 android:toXDelta="-5%p" android:startOffset="200"** />  
</**set**>

## Sửa file colors.xml

1. Mở file ***colors.xml:***

*<?***xml version="1.0" encoding="utf-8"***?>*<**resources**>  
 <**color name="colorPrimary"**>#3F51B5</**color**>  
 <**color name="colorPrimaryDark"**>#303F9F</**color**>  
 <**color name="colorAccent"**>#448AFF</**color**>  
 <**color name="correct\_answer"**>#00CC00</**color**>  
 <**color name="incorrect\_answer"**>#FF0000</**color**>  
</**resources**>

## Tạo Preferences.xml cho trang cài đặt App

Phần này, ta sẽ tạo file liên kết cài đặt app:

1. Chuột phải thư mục ***res***, chọn ***New* > *Android resource file*.**
2. Điền “preferences.xml” tại mục ***File name.***
3. Tại mục ***Resource type***, chọn ***XML***. Và mục ***Directory name***, điền ***xml***.
4. Nhấp ***OK.***

Mở ***preferences.xml***, vào Text và viết code sau:

*<?***xml version="1.0" encoding="utf-8"***?>*<**PreferenceScreen xmlns:android="http://schemas.android.com/apk/res/android"**>  
 <**ListPreference  
 android:entries="@array/guesses\_list"  
 android:entryValues="@array/guesses\_list"  
 android:key="pref\_numberOfChoices"  
 android:title="@string/number\_of\_choices"  
 android:summary="@string/number\_of\_choices\_description"  
 android:persistent="true"  
 android:defaultValue="4"** />  
  
 <**MultiSelectListPreference  
 android:entries="@array/regions\_list\_for\_settings"  
 android:entryValues="@array/regions\_list"  
 android:key="pref\_regionsToInclude"  
 android:title="@string/world\_regions"  
 android:summary="@string/world\_regions\_description"  
 android:persistent="true"  
 android:defaultValue="@array/regions\_list"** />  
  
</**PreferenceScreen**>

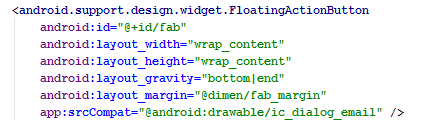
## Thêm Classes SettingsActivity và SettingsActivityFragment

Phần này, ta sẽ tạo 2 lớp là SettingsActivity và SettingsActivityFragment

1. Chuột phải thư mục ***app***, chọn ***New*** > ***Activity*** > ***BasicActivity*.**
2. Điền “SettingsActivity” tại mục ***Activity Name.***
3. Dưới mục ***Title,*** chọn ***Use a Fragment.***
4. Tại mục ***Hierarchical Parent,*** nhấp vào biểu tượng “…” phía cuối, chọn ***MainActivity***, nhấp OK.
5. Nhấp Finish.

Như vậy, project ta sẽ tạo ra các file: ***activity\_settings.xml***, ***content\_settings.xml*** and ***fragment\_settings.xml*** trong thư mục ***res/layout,***  ***SettingsActivity.java*** và ***SettingsActivityFragment.java*** trong thư mục Java.

Mở ***activity\_settings.xml,*** và XÓA đoạn code sau:



# **THIẾT KẾ ỨNG DỤNG**



Nhằm dễ dàng quản lí các fragment, ta sẽ đổi id như sau:

**1.** Mở ***content\_main.xml***, tại tab ***Design***

**2.** Tại cửa sổ ***Component Tree,***  ta nhấp chọn ***fragment***

**3.** Trong cửa sổ **Properties**, tại mục ***ID,*** ta đổi thành “quizFragment”

**4.**  Lưu lại content\_main.xml.

## Thiết kế fragment\_main.xml Layout

***Bước 1: Đổi RelativeLayout thành LinearLayout***

**1.** Mở ***fragment\_main.xml***, chọn tab ***Text***.

**2.** Đổi ***RelativeLayout*** to ***LinearLayout***.

**3.** Chuyển về tab ***Design***.

**4.** Tại cửa sổ ***Component Tree*,** chọn ***LinearLayout***.

**5.** Trong cửa sổ **Properties**, tại trường ***orientation*,** chọn ***vertical***.

**6.** Chắc chắn rằng trường **layout:width** và **layout:height** đều là ***match\_parent***.

**7.** Đổi ID thành quizLinearLayout.

***Bước 2: Thêm questionNumberTextView vào LinearLayout***

Tại cửa sổ ***Component Tree*,** chọn ***TextView,*** sửa các thuộc tính sau:

• **ID**: questionNumberTextView.

• **textAppearance**: Material.Medium – Cỡ chữ trung bình.

• **layout:gravity center**: center\_horizontal – Căn giữa theo chiều ngang.

• **layout:margin> layout:marginB**: @dimen/spacing – Để thiết lập thuộc tính này, chọn trường **layout:margin**, click chọn biểu tượng (**…**) ở cuối trường. Cửa sổ **Resources** hiện lên, Chọn ***add new resource > New dimen Value.* Resource name**: spacing - **Resource value**: 8dp. Nhấn ***OK***.

• **text**: @string/question – Để thiết lập thuộc tính này, chọn trường ***Text***, click chọn biểu tượng (**…**) ở cuối trường. Cửa sổ **Resources** hiện lên, tại tab **Project,** chọn ***question***, sau đó nhấn ***OK***.

***Bước 3: Thêm flagImageView vào LinearLayout***

Kéo ***ImageView*** vào ***Component Tree***  sau ***TextView,*** sửa các thuộc tính sau:

• **ID**: flagImageView.

• **layout:width**: match\_parent

• **layout:height**: 0dp

• **layout:gravity:** center

• **layout:margin> layout:marginB:** @dimen/spacing

• **layout:marginL** và **layout:marginR**: @dimen/activity\_horizontal\_margin

• **layout:weight**: 1 – chọn thành phần flagImageView quan trọng số 1 so với các thành phần khác trong quizLinearLayout.

• **adjustViewBounds**: true—Setting the ImageView’s **Adjust View Bounds** property to  
true (by checking its checkbox) indicates that the ImageView maintains its image’s aspect ratio.

• **contentDescription**: @string/image\_description

• **scaleType**: fitCenter—This indicates that the ImageView should scale the image  
to fill either the ImageView’s width or height while maintaining the original image’s aspect ratio. If the image’s width is less than the ImageView’s, the image is  
centered horizontally. Similarly, if the image’s height is less than the ImageView’s,  
the image is centered vertically.

***Bước 4: Thêm guessCountryTextView vào LinearLayout***

Kéo ***TextView*** vào ***Component Tree***  sau ***ImageView,*** sửa các thuộc tính sau:

• **ID**: guessCountryTextView.

• **textAppearance**: Material.Medium – Cỡ chữ trung bình.

• **layout:gravity center**: horizontal

• **text**: @string/guess\_country

***Bước 5: Thêm 8 Buttons vào LinearLayout:***

**1.** Kéo **LinearLayout (Horizontal)** vào **Component Tree** và đặt **id** là row1LinearLayout, chọn **layout:height:** wrap\_content.

**2.** Kéo **Button** vào **row1LinearLayout** trong **Component Tree**. Chúng ta ko cần đặt ID.

**3.** Kéo tiếp 1 **Button** vào **row1LinearLayout** như bước 2.

**4.** Lặp lại bước *1–3* sao cho ta có 4 dòng, 8 button. Lưu ý: ID các dòng là row1LinearLayout, row2LinearLayout, row3LinearLayout, row4LinearLayout.

***Bước 6: Thêm answerTextView vào LinearLayout***

Kéo ***TextView*** vào ***Component Tree***  sau ***ImageView,*** sửa các thuộc tính sau:

• **ID**: answerTextView.

• **textAppearance**: Material.Medium – Cỡ chữ trung bình.

• **layout:gravity**: Check **bottom** và **center\_horizontal**.

• **gravity**: center\_horizontal

• **textSize**: @dimen/answer\_size, ta tạo dimen như các phần ở trên, **Resource Name**: answer\_size, **Resource Value**: 36sp.

• **textStyle**: bold.

***Bước 7: Cài đặt thuộc tính cho 8 cái button của chúng ta:***

Tại ***Component Tree*** , ta chọn tất cả 8 cái button (ctrl + nhấp 8 button):

• **layout:width:** 0dp

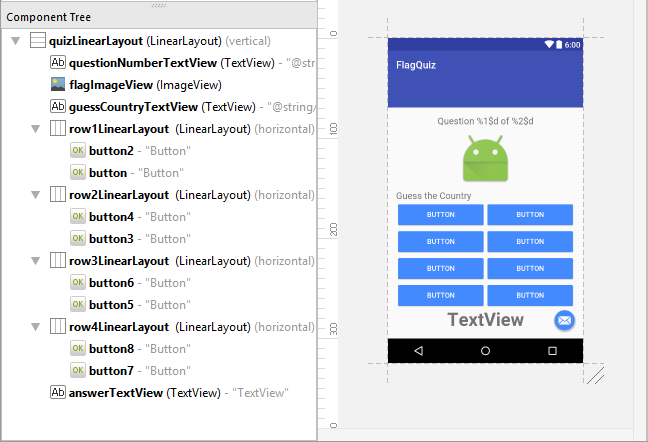
• **layout:height**: match\_parent

• **layout:weight**:1

• **Lines: 2**

• **Style:** @android:style/Widget.Material.Button.Colored cho Android 6 và @android:style/Widget.AppCompat.Button.Colored cho Android nhỏ hơn

• **textColor:** @color/button\_text\_color



Tiếp theo, ta sẽ thiết kế cho màn hình ngang:

***Tạo content\_main.xml cho màn hình ngang***

**1.** Chuột phải ***res/layout***, chọn **New > Layout resource file**.

**2.** Nhập “content\_main.xml” trong **File name.**

**3.** Chắc chắn rằng trường **Root element** là LinearLayout.

**4.** Tại danh sách **Available qualifiers**, chọn **Smallest Screen Width**, nhấp nút **>>** và điền tại trường **Smallest Screen Width** là 700.

**5.** Tương tự bước 4, chọn **Orientation**, nhấp nút **>>** và chọn tại trường **Screen Orientation** là Landscape.

**6.** Nhấp **OK**.



***Tạo giao diện cho màn hình ngang***

**1.** Mở chọn ***content\_main*** mới tạo phía trên. Chọn LinearLayout (vertical) ở **Component Tree** window, tại trường **orientation** chọn horizontal.

**2.** kéo **<fragment>** trong cửa sổ **Palette**’s **Custom** vào **Component Tree**. Cửa sổ **Fragments** hiện lên, Chọn SettingsActivityFragment, nhấp **OK**. Chọn fragment vừa tạo đặt **id** là settingsActivityFragment.

**3.** Lặp lại bước trên, nhưng lần này chọn MainActivityFragment. Và đặt ID là: quizFragment

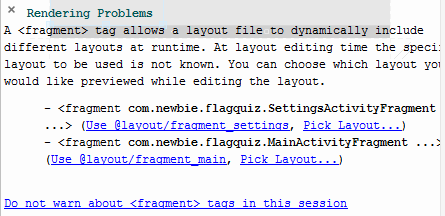
**4.** Chọn settingsActivityFragment trong **Component Tree** window. Đặt  
**layout:width:** 0dp, **layout:height:** match\_parent, **layout:weight:** 1.

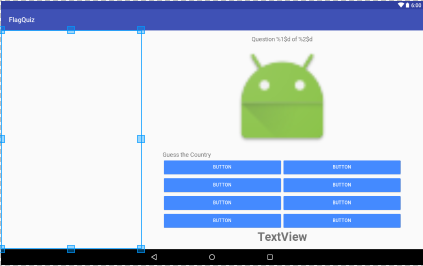
**5.** Chọn quizFragment trong **Component Tree** window. Đặt **layout:width**:  
0dp, **layout:height:**  match\_parent, **layout:weight**: 2.

**6.** Chuyển qua tab **Text,** thêm 2 dòng dưới vào tag <LinearLayout … > ở đầu:

xmlns:app="http://schemas.android.com/apk/res-auto"  
app:layout\_behavior="@string/appbar\_scrolling\_view\_behavior"

**7.**  Chuyển qua tab Design, sẽ có thông báo lỗi, tại ô thông báo lỗi, chọn



Như vậy, giao diện cho màn hình ngang đã xong:

## MainActivity Class

**public class** MainActivity **extends** AppCompatActivity {  
  
 **public static final** String ***CHOICES*** = **"pref\_numberOfChoices"**;  
 **public static final** String ***REGIONS*** = **"pref\_regionsToInclude"**;  
  
 **private boolean phoneDevice** = **true**; *// used to force portrait mode* **private boolean preferencesChanged** = **true**; *// did preferences change?*

**import** android.content.Intent;  
**import** android.content.SharedPreferences;  
**import** android.content.pm.ActivityInfo;  
**import** android.content.res.Configuration;  
**import** android.os.Bundle;  
**import** android.preference.PreferenceManager;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.support.v7.widget.Toolbar;  
**import** android.view.Menu;  
**import** android.view.MenuItem;  
**import** android.widget.Toast;  
  
**import** java.util.Set;

*// called after onCreate completes execution*@Override  
**protected void** onStart() {  
  
 **super**.onStart();  
  
 **if** (**preferencesChanged**) {  
 *// now that the default preferences have been set,  
 // initialize MainActivityFragment and start the quiz* MainActivityFragment quizFragment = (MainActivityFragment)  
 getSupportFragmentManager().findFragmentById(R.id.***quizFragment***);  
 quizFragment.updateGuessRows(PreferenceManager.*getDefaultSharedPreferences*(**this**));  
 quizFragment.updateRegions(PreferenceManager.*getDefaultSharedPreferences*(**this**));  
 quizFragment.resetQuiz();  
 **preferencesChanged** = **false**;  
 }  
}

**protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_main***);  
  
 Toolbar toolbar = (Toolbar) findViewById(R.id.***toolbar***);  
 setSupportActionBar(toolbar);  
  
 *// set default values in the app's SharedPreferences* PreferenceManager.*setDefaultValues*(**this**, R.xml.***preferences***, **false**);  
  
 *// register listener for SharedPreferences changes* PreferenceManager.*getDefaultSharedPreferences*(**this**).registerOnSharedPreferenceChangeListener(**preferencesChangeListener**);  
  
 *// determine screen size* **int** screenSize = getResources().getConfiguration().**screenLayout** & Configuration.***SCREENLAYOUT\_SIZE\_MASK***;  
  
 *// if device is a tablet, set phoneDevice to false* **if** (screenSize == Configuration.***SCREENLAYOUT\_SIZE\_LARGE*** || screenSize == Configuration.***SCREENLAYOUT\_SIZE\_XLARGE***)  
 **phoneDevice** = **false**; *// not a phone-sized device  
  
 // if running on phone-sized device, allow only portrait orientation* **if** (**phoneDevice**)  
 setRequestedOrientation( ActivityInfo.***SCREEN\_ORIENTATION\_PORTRAIT***);  
  
}

@Override  
**public boolean** onOptionsItemSelected(MenuItem item) {  
 Intent preferencesIntent = **new** Intent(**this**, SettingsActivity.**class**);  
 startActivity(preferencesIntent);  
 **return super**.onOptionsItemSelected(item);  
}

@Override  
**public boolean** onCreateOptionsMenu(Menu menu) {  
 *// get the device's current orientation* **int** orientation = getResources().getConfiguration().**orientation**;  
  
 *// display the app's menu only in portrait orientation* **if** (orientation == Configuration.***ORIENTATION\_PORTRAIT***){  
 *// inflate the menu* getMenuInflater().inflate(R.menu.***menu\_main***, menu);  
 **return true**;  
 }  
 **else  
 return false**;  
}

*// listener for changes to the app's SharedPreferences* **private** SharedPreferences.OnSharedPreferenceChangeListener **preferencesChangeListener** =  
 **new** SharedPreferences.OnSharedPreferenceChangeListener() {  
 *// called when the user changes the app's preferences* @Override  
 **public void** onSharedPreferenceChanged(SharedPreferences sharedPreferences, String key)  
 {  
 **preferencesChanged** = **true**; *// user changed app settings* MainActivityFragment quizFragment = (MainActivityFragment)  
 getSupportFragmentManager().findFragmentById(  
 R.id.***quizFragment***);  
  
 **if** (key.equals(***CHOICES***)) { *// # of choices to display changed* quizFragment.updateGuessRows(sharedPreferences);  
 quizFragment.resetQuiz();  
 } **else if** (key.equals(***REGIONS***)) { *// regions to include changed* Set<String> regions =  
 sharedPreferences.getStringSet(***REGIONS***, **null**);  
  
 **if** (regions != **null** && regions.size() > 0) {  
 quizFragment.updateRegions(sharedPreferences);  
 quizFragment.resetQuiz();  
 } **else** {  
 *// must select one region--set North America as default* SharedPreferences.Editor editor =  
 sharedPreferences.edit();  
 regions.add(getString(R.string.***default\_region***));  
 editor.putStringSet(***REGIONS***, regions);  
 editor.apply();  
 Toast.*makeText*(MainActivity.**this**,  
 R.string.***default\_region\_message***,  
 Toast.***LENGTH\_SHORT***).show();  
 }  
  
 Toast.*makeText*(MainActivity.**this**,  
 R.string.***restarting\_quiz***,  
 Toast.***LENGTH\_SHORT***).show();  
 }  
  
 }  
 };  
}

## MainActivityFragment Class

**import** android.animation.Animator;  
**import** android.animation.AnimatorListenerAdapter;  
**import** android.app.AlertDialog;  
**import** android.app.Dialog;  
**import** android.content.DialogInterface;  
**import** android.content.SharedPreferences;  
**import** android.content.res.AssetManager;  
**import** android.graphics.drawable.Drawable;  
**import** android.os.Build;  
**import** android.os.Bundle;  
**import** android.os.Handler;  
**import** android.support.annotation.RequiresApi;  
**import** android.support.v4.app.DialogFragment;  
**import** android.support.v4.app.Fragment;  
**import** android.util.Log;  
**import** android.view.LayoutInflater;  
**import** android.view.View;  
**import** android.view.ViewAnimationUtils;  
**import** android.view.ViewGroup;  
**import** android.view.animation.Animation;  
**import** android.view.animation.AnimationUtils;  
**import** android.widget.Button;  
**import** android.widget.ImageView;  
**import** android.widget.LinearLayout;  
**import** android.widget.TextView;  
  
**import** java.io.IOException;  
**import** java.io.InputStream;  
**import** java.security.SecureRandom;  
**import** java.util.ArrayList;  
**import** java.util.Collections;  
**import** java.util.List;  
**import** java.util.Set;

**public class** MainActivityFragment **extends** Fragment {  
  
 *// String used when logging error messages* **private static final** String ***TAG*** = **"FlagQuiz Activity"**;  
  
 **private static final int *FLAGS\_IN\_QUIZ*** = 10;  
  
 **private** List<String> **fileNameList**; *// flag file names* **private** List<String> **quizCountriesList**; *// countries in current quiz* **private** Set<String> **regionsSet**; *// world regions in current quiz* **private** String **correctAnswer**; *// correct country for the current flag* **private int totalGuesses**; *// number of guesses made* **private int correctAnswers**; *// number of correct guesses* **private int guessRows**; *// number of rows displaying guess Buttons* **private** SecureRandom **random**; *// used to randomize the quiz* **private** Handler **handler**; *// used to delay loading next flag* **private** Animation **shakeAnimation**; *// animation for incorrect guess* **private** LinearLayout **quizLinearLayout**; *// layout that contains the quiz* **private** TextView **questionNumberTextView**; *// shows current question #* **private** ImageView **flagImageView**; *// displays a flag* **private** LinearLayout[] **guessLinearLayouts**; *// rows of answer Buttons* **private** TextView **answerTextView**; *// displays correct answer*

**public** View onCreateView(LayoutInflater inflater, ViewGroup container,  
 Bundle savedInstanceState) {  
  
 **super**.onCreateView(inflater, container, savedInstanceState);  
 View view = inflater.inflate(R.layout.***fragment\_main***, container, **false**);  
  
 **fileNameList** = **new** ArrayList<>();  
 **random** = **new** SecureRandom();  
 **quizCountriesList** = **new** ArrayList<>();  
 **handler** = **new** Handler();  
  
 *// load the shake animation that's used for incorrect answers* **shakeAnimation** = AnimationUtils.*loadAnimation*(getActivity(), R.anim.***incorrect\_shake***);  
 **shakeAnimation**.setRepeatCount(3); *// animation repeats 3 times  
  
 // get references to GUI components* **quizLinearLayout** = (LinearLayout) view.findViewById(R.id.***quizLinearLayout***);  
 **questionNumberTextView** = (TextView) view.findViewById(R.id.***questionNumberTextView***);  
 **flagImageView** = (ImageView) view.findViewById(R.id.***flagImageView***);  
 **guessLinearLayouts** = **new** LinearLayout[4];  
 **guessLinearLayouts**[0] = (LinearLayout) view.findViewById(R.id.***row1LinearLayout***);  
 **guessLinearLayouts**[1] = (LinearLayout) view.findViewById(R.id.***row2LinearLayout***);  
 **guessLinearLayouts**[2] = (LinearLayout) view.findViewById(R.id.***row3LinearLayout***);  
 **guessLinearLayouts**[3] = (LinearLayout) view.findViewById(R.id.***row4LinearLayout***);  
 **answerTextView** = (TextView) view.findViewById(R.id.***answerTextView***);  
  
 *// configure listeners for the guess Buttons* **for** (LinearLayout row : **guessLinearLayouts**) {  
 **for** (**int** column = 0; column < row.getChildCount(); column++) {  
 Button button = (Button) row.getChildAt(column);  
 button.setOnClickListener(**guessButtonListener**);  
 }  
 }  
  
 *// set questionNumberTextView's text* **questionNumberTextView**.setText(getString(R.string.***question***, 1, ***FLAGS\_IN\_QUIZ***));  
  
 **return** view; *// return the fragment's view for display*}

*// update world regions for quiz based on values in SharedPreferences***public void** updateRegions(SharedPreferences sharedPreferences) {  
  
 **regionsSet** = sharedPreferences.getStringSet(MainActivity.***REGIONS***, **null**);  
  
}

*// update guessRows based on value in SharedPreferences*

**public void** updateGuessRows(SharedPreferences sharedPreferences) {  
 *// get the number of guess buttons that should be displayed* String choices = sharedPreferences.getString(MainActivity.***CHOICES***, **null**);  
  
 **guessRows** = Integer.*parseInt*(choices) / 2;  
 *// hide all quess button LinearLayouts* **for** (LinearLayout layout : **guessLinearLayouts**)  
 layout.setVisibility(View.***GONE***);  
  
 *// display appropriate guess button LinearLayouts* **for** (**int** row = 0; row < **guessRows**; row++)  
 **guessLinearLayouts**[row].setVisibility(View.***VISIBLE***);  
}

*// parses the country flag file name and returns the country name***private** String getCountryName(String name) {  
 **return** name.substring(name.indexOf(**'-'**) + 1).replace(**'\_'**, **' '**);  
}

*// set up and start the next quiz***public void** resetQuiz() {  
 *// use AssetManager to get image file names for enabled regions* AssetManager assets = getActivity().getAssets();  
 **fileNameList**.clear(); *// empty list of image file names* **try** {  
 *// loop through each region* **for** (String region : **regionsSet**) {  
 *// get a list of all flag image files in this region* String[] paths = assets.list(region);  
 **for** (String path : paths)  
 **fileNameList**.add(path.replace(**".png"**, **""**));  
 }  
 } **catch** (IOException exception) {  
 Log.*e*(***TAG***, **"Error loading image file names"**, exception);  
 }  
 **correctAnswers** = 0; *// reset the number of correct answers made* **totalGuesses** = 0; *// reset the total number of guesses the user made* **quizCountriesList**.clear(); *// clear prior list of quiz countries* **int** flagCounter = 1;  
 **int** numberOfFlags = **fileNameList**.size();  
 *// add FLAGS\_IN\_QUIZ random file names to the quizCountriesList* **while** (flagCounter <= ***FLAGS\_IN\_QUIZ***) {  
 **int** randomIndex = **random**.nextInt(numberOfFlags);  
 *// get the random file name* String filename = **fileNameList**.get(randomIndex);  
 *// if the region is enabled and it hasn't already been chosen* **if** (!**quizCountriesList**.contains(filename)) {  
 **quizCountriesList**.add(filename); *// add the file to the list* ++flagCounter;  
 }  
 }  
 **try** {  
 loadNextFlag(); *// start the quiz by loading the first flag* } **catch** (IOException e) {  
 e.printStackTrace();  
 }  
}

*// after the user guesses a correct flag, load the next flag***private void** loadNextFlag() **throws** IOException {  
 *// get file name of the next flag and remove it from the list* String nextImage = **quizCountriesList**.remove(0);  
 **correctAnswer** = nextImage; *// update the correct answer* **answerTextView**.setText(**""**); *// clear answerTextView  
  
 // display current question number* **questionNumberTextView**.setText(getString(R.string.***question***, (**correctAnswers** + 1), ***FLAGS\_IN\_QUIZ***));  
  
 *// extract the region from the next image's name* String region = nextImage.substring(0, nextImage.indexOf(**'-'**));  
  
 *// use AssetManager to load next image from assets folder* AssetManager assets = getActivity().getAssets();  
  
 *// get an InputStream to the asset representing the next flag  
 // and try to use the InputStream* **try** (InputStream stream = assets.open(region + **"/"** + nextImage + **".png"**)) {  
  
 *// load the asset as a Drawable and display on the flagImageView* Drawable flag = Drawable.*createFromStream*(stream, nextImage);  
 **flagImageView**.setImageDrawable(flag);  
 animate(**false**); *// animate the flag onto the screen* } **catch** (IOException exception) {  
 Log.*e*(***TAG***, **"Error loading "** + nextImage, exception);  
 }  
  
 Collections.*shuffle*(**fileNameList**); *// shuffle file names  
 // put the correct answer at the end of fileNameList* **int** correct = **fileNameList**.indexOf(**correctAnswer**);  
 **fileNameList**.add(**fileNameList**.remove(correct));  
  
 *// add 2, 4, 6 or 8 guess Buttons based on the value of guessRows* **for** (**int** row = 0; row < **guessRows**; row++) {  
 *// place Buttons in currentTableRow* **for** (**int** column = 0;  
 column < **guessLinearLayouts**[row].getChildCount();  
 column++) {  
 *// get reference to Button to configure* Button newGuessButton = (Button) **guessLinearLayouts**[row].getChildAt(column);  
 newGuessButton.setEnabled(**true**);  
 *// get country name and set it as newGuessButton's text* String filename = **fileNameList**.get((row \* 2) + column);  
 newGuessButton.setText(getCountryName(filename));  
 }  
 }  
 *// randomly replace one Button with the correct answer* **int** row = **random**.nextInt(**guessRows**); *// pick random row* **int** column = **random**.nextInt(2); *// pick random column* LinearLayout randomRow = **guessLinearLayouts**[row]; *// get the row* String countryName = getCountryName(**correctAnswer**);  
 ((Button) randomRow.getChildAt(column)).setText(countryName);  
}

*// animates the entire quizLinearLayout on or off screen***private void** animate(**boolean** animateOut) {  
 *// prevent animation into the the UI for the first flag* **if** (**correctAnswers** == 0)  
 **return**;  
 *// calculate center x and center y* **int** centerX = (**quizLinearLayout**.getLeft() +  
 **quizLinearLayout**.getRight()) / 2;  
 **int** centerY = (**quizLinearLayout**.getTop() +  
 **quizLinearLayout**.getBottom()) / 2;  
 *// calculate animation radius* **int** radius = Math.*max*(**quizLinearLayout**.getWidth(),  
 **quizLinearLayout**.getHeight());  
  
 Animator animator;  
  
 *// if the quizLinearLayout should animate out rather than in* **if** (animateOut) {  
 *// create circular reveal animation* animator = ViewAnimationUtils.*createCircularReveal*(  
 **quizLinearLayout**, centerX, centerY, radius, 0);  
 animator.addListener(  
 **new** AnimatorListenerAdapter() {  
 *// called when the animation finishes* @Override  
 **public void** onAnimationEnd(Animator animation) {  
 **try** {  
 loadNextFlag();  
 } **catch** (IOException e) {  
 e.printStackTrace();  
 }  
 }  
 }  
 );  
 } **else** { *// if the quizLinearLayout should animate in* animator = ViewAnimationUtils.*createCircularReveal*(  
 **quizLinearLayout**, centerX, centerY, 0, radius);  
 }  
  
 animator.setDuration(500); *// set animation duration to 500 ms* animator.start(); *// start the animation*}

*// if the user has correctly identified FLAGS\_IN\_QUIZ flags***if** (**correctAnswers** == ***FLAGS\_IN\_QUIZ***) {  
 *// DialogFragment to display quiz stats and start new quiz* DialogFragment quizResults = **new** DialogFragment() {  
 *// create an AlertDialog and return it* @Override  
 **public** Dialog onCreateDialog(Bundle bundle) {  
  
 AlertDialog.Builder builder = **new** AlertDialog.Builder(getActivity());  
 builder.setMessage(getString(R.string.***results***, **totalGuesses**, (1000 / (**double**) **totalGuesses**)));  
  
  
 *// "Reset Quiz" Button* builder.setPositiveButton(R.string.***reset\_quiz***, **new** DialogInterface.OnClickListener() {  
 **public void** onClick(DialogInterface dialog, **int** id) {  
 resetQuiz();  
 }  
 }  
 );  
 **return** builder.create(); *// return the AlertDialog* }  
 };  
  
 *// use FragmentManager to display the DialogFragment* quizResults.setCancelable(**false**);  
 quizResults.show(getFragmentManager(), **"quiz results"**);  
  
}

*// called when a guess Button is touched***private** View.OnClickListener **guessButtonListener** = **new** View.OnClickListener() {  
 @RequiresApi(api = Build.VERSION\_CODES.***M***)  
 @Override  
 **public void** onClick(View v) {  
 Button guessButton = ((Button) v);  
 String guess = guessButton.getText().toString();  
 String answer = getCountryName(**correctAnswer**);  
 ++**totalGuesses**; *// increment number of guesses the user has made* **if** (guess.equals(answer)) { *// if the guess is correct* ++**correctAnswers**; *// increment the number of correct answers  
  
 // display correct answer in green text* **answerTextView**.setText(answer + **"!"**);  
 **answerTextView**.setTextColor(getResources().getColor(R.color.***correct\_answer***, getContext().getTheme()));  
  
 disableButtons(); *// disable all guess Buttons*

*// utility method that disables all answer Buttons***private void** disableButtons() {  
 **for** (**int** row = 0; row < **guessRows**; row++) {  
 LinearLayout guessRow = **guessLinearLayouts**[row];  
 **for** (**int** i = 0; i < guessRow.getChildCount(); i++)  
 guessRow.getChildAt(i).setEnabled(**false**);  
 }  
}

**else** { *// answer is correct but quiz is not over  
 // load the next flag after a 2-second delay* **handler**.postDelayed(**new** Runnable() {  
 @Override  
 **public void** run() {  
 animate(**true**); *// animate the flag off the screen* }  
 }, 2000); *// 2000 milliseconds for 2-second delay* }  
 } **else** { *// answer was incorrect* **flagImageView**.startAnimation(**shakeAnimation**); *// play shake  
 // display "Incorrect!" in red* **answerTextView**.setText(R.string.***incorrect\_answer***);  
 **answerTextView**.setTextColor(getResources().getColor(R.color.***incorrect\_answer***, getContext().getTheme()));  
 guessButton.setEnabled(**false**); *// disable incorrect answer* }  
 }  
};

## SettingsActivity Class

**import** android.os.Bundle;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.support.v7.widget.Toolbar;  
  
**public class** SettingsActivity **extends** AppCompatActivity {  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_settings***);  
 Toolbar toolbar = (Toolbar) findViewById(R.id.***toolbar***);  
 setSupportActionBar(toolbar);  
 getSupportActionBar().setDisplayHomeAsUpEnabled(**true**);  
 }  
  
}

## SettingsActivityFragment Class

**package** com.newbie.flagquiz;  
  
**import** android.os.Bundle;  
**import** android.preference.PreferenceFragment;  
  
*/\*\*  
 \* A placeholder fragment containing a simple view.  
 \*/***public class** SettingsActivityFragment **extends** PreferenceFragment {  
 *// creates preferences GUI from preferences.xml file in res/xml* @Override  
 **public void** onCreate(Bundle bundle) {  
 **super**.onCreate(bundle);  
 addPreferencesFromResource(R.xml.***preferences***); *// load from XML}* }  
}

## AndroidManifest.xml

<**application  
 android:allowBackup="true"  
 android:icon="@mipmap/ic\_launcher"  
 android:label="@string/app\_name"  
 android:supportsRtl="true"  
 android:theme="@style/AppTheme"**>  
 <**activity  
 android:name=".MainActivity"  
 android:label="@string/app\_name"  
 android:launchMode="singleTop"  
 android:theme="@style/AppTheme.NoActionBar"**>  
 <**intent-filter**>  
 <**action android:name="android.intent.action.MAIN"** />  
  
 <**category android:name="android.intent.category.LAUNCHER"** />  
 </**intent-filter**>  
 </**activity**>  
 <**activity  
 android:name=".SettingsActivity"  
 android:label="@string/title\_activity\_settings"  
 android:parentActivityName=".MainActivity"  
 android:theme="@style/AppTheme.NoActionBar"**>  
 <**meta-data  
 android:name="android.support.PARENT\_ACTIVITY"  
 android:value="com.newbie.flagquiz.MainActivity"** />  
 </**activity**>  
</**application**>