ITMD 455/555 *Intelligent Device Applications* Lab 5

#### Quiz App- 50 points

**Introduction**. This lab will have you create a quiz via a question bank from the URL <http://www.papademas.net:81/sample.txt>. Check it.

**Objective:** For this lab you will use file processing techniques to open and read into an arraylist, the

**sample.txt** file which has questions for the user to answer. This is only a true or false

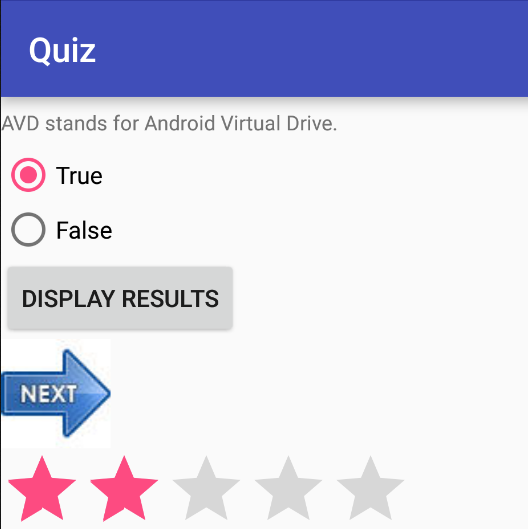
quiz. You will add in radio buttons to allow for a user to choose True or False as a response to each

question displayed via a TextView. An image will be clickable to allow the user to go to the next

question. After the user goes thru 5 questions you will rate them by showing them how many

questions they have correct – which will be shown via **x** number of stars from Android’s rating bar!

Here is a sample snapshot to get an idea of what you will be working with:

****

As you will see when you run your app, the user will press the **Display Results** button to see if their answer is correct depending if they clicked on the True vs. False radio option.

An answer of Right! or Wrong! will then appear briefly in a Toast message. The user will

then press the **Next** image and then the next question will appear at the top in a TextView

to try another question. At the very end a rating bar will appear showing the number of

questions they have correct denoted by Red Stars.

**STEP 1 Creating a New Android Project**

Create a new project called Quiz.

Create your first Empty Activity and call it MainActivity. Use the default layout name.

First things first. Make sure your manifest file has this line of code (after opening manifest tag ending is fine for placement) to receive data from the web...

**<uses-permission android:name=*"android.permission.INTERNET"*/>**

Also after your opening application tag add in the following to recevie data as well

<**application  
 android:usesCleartextTraffic="true"**

**STEP 2 Obtain an image from the web that will serve as your “clickable” image.**

Download an image icon from the web that says Next or depicts a right arrow or a

combination of the two and copy it into your drawable folder. The image will serve as a

guide to assist the user to grab another question, as you will see by just clicking on the

image itself!

\*Make sure to name your image **next**.

**STEP 3 Open up your activity\_main.xml file and include the following code:**

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

<**LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical"** >  
  
 <**TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:textSize="12sp"  
 android:id="@+id/textView1"  
 android:paddingTop="20px"  
 android:paddingBottom="20px"  
 android:width="400dp"** />  
  
 <**RadioGroup  
 android:id="@+id/radioQuestions"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"** >  
  
 <**RadioButton  
 android:id="@+id/radioTrue"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="@string/radio\_true"  
 android:checked="true"** />  
  
 <**RadioButton  
 android:id="@+id/radioFalse"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="@string/radio\_false"** />  
  
 </**RadioGroup**>  
  
 <**Button  
 android:id="@+id/btnDisplay"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="@string/btn\_display"** />  
  
 <**ImageView  
 android:id="@+id/imageView1"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:src="@drawable/next"  
 android:clickable="true"**/>  
  
 <**RatingBar  
 android:id="@+id/ratingBar"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:numStars="5"  
 android:stepSize="1.0"  
 android:rating="2.0"  
 android:isIndicator="true"** />  
  
</**LinearLayout**>

Also open up your strings.xml file and add in the following resources:

<resources>

<string name=*"app\_name"*>Quiz</string>

<string name=*"radio\_true"*>True</string>

<string name=*"radio\_false"*>False</string>

<string name=*"btn\_display"*>Display Results</string>

</resources>

**STEP 4 Open up your MainActivity.java file and add in the following code and import**

**statements:**

**First add the following code into your Activity:**

**public class** MainActivity **extends** AppCompatActivity {  
 Activity **context**;  
 TextView **txtView**;  
 ProgressDialog **pd**;  
  
 ArrayList<String> **stringList** = **new** ArrayList<String>();  
  
 **static int** *questionNum* = 0;  
  
 **private** RadioGroup **radioQuestions**;  
 **private** RadioButton **radioButton**;  
  
 ImageView **image**;  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_main***);  
 **context** = **this**;  
  
 BackgroundTask bt = **new** BackgroundTask();  
 bt.execute(**"http://www.papademas.net:81/sample.txt"**); *//grab url* }*//end onCreate  
  
 //background process to download the file from internet* **public class** BackgroundTask **extends** AsyncTask<String, Integer, Void> {  
  
 **protected void** onPreExecute() {  
 **super**.onPreExecute();  
 *//display progress dialog* **pd** = **new** ProgressDialog(**context**);  
 **pd**.setTitle(**"Reading the text file"**);  
 **pd**.setMessage(**"Please wait."**);  
 **pd**.setCancelable(**true**);  
 **pd**.setIndeterminate(**false**);  
 **pd**.show();  
  
 }  
  
 **protected** Void doInBackground(String... params) {  
 URL url;  
 String StringBuffer = **null**;  
 **try** {  
 *//create url object to point to the file location on internet* url = **new** URL(params[0]);  
 *//make a request to server* HttpURLConnection con = (HttpURLConnection) url.openConnection();  
 *//get InputStream instance* InputStream is = con.getInputStream();  
 *//create BufferedReader object* BufferedReader br = **new** BufferedReader(**new** InputStreamReader(is));  
  
 *//read content of the file line by line & add it to Stringbuffer* **while** ((StringBuffer = br.readLine()) != **null**) {  
 **stringList**.add(StringBuffer); *//add to Arraylist* }  
  
 br.close();  
  
 } **catch** (Exception e) {  
 e.printStackTrace();  
 *//close dialog if error occurs* **if** (**pd** != **null**) **pd**.dismiss();  
 }  
  
 **return null**;  
  
 }  
  
 **protected void** onPostExecute(Void result) {  
 *//close dialog* **if** (**pd** != **null**)  
 **pd**.dismiss();  
 **txtView** = findViewById(R.id.***textView1***);  
 *//display read text in TextVeiw* **txtView**.setText(**stringList**.get(0));  
 startQuiz();  
  
 }  
 }*//end BackgroundTask class* **public void** startQuiz() {  
 buttonListener();  
 }  
  
 **public void** buttonListener() {  
  
 Button btnDisplay;  
  
 **radioQuestions** = findViewById(R.id.***radioQuestions***);  
 btnDisplay = findViewById(R.id.***btnDisplay***);  
  
 btnDisplay.setOnClickListener(**new** OnClickListener() {  
  
 @Override  
 **public void** onClick(View v) {  
  
 *// get selected radio button from radioGroup* **int** selectedId = **radioQuestions**.getCheckedRadioButtonId();  
  
 *// find the radiobutton by returned id* **radioButton** = findViewById(selectedId);  
  
 **switch** (*questionNum*) {  
 **case** 0:  
 *//verify if result matches the right button selection  
 //i.e., (True or false!)* **if** (**radioButton**.getText().equals(**"True"**))  
 Toast.*makeText*(MainActivity.**this**,  
 **" Right!"**, Toast.***LENGTH\_SHORT***).show();  
 **else** Toast.*makeText*(MainActivity.**this**,  
 **" Wrong!"**, Toast.***LENGTH\_SHORT***).show();  
 **break**;  
 **case** 1:  
 *//verify if result matches the right button selection  
 //i.e., (True or false!)* **if** (**radioButton**.getText().equals(**"False"**))  
 Toast.*makeText*(MainActivity.**this**,  
 **" Right!"**, Toast.***LENGTH\_SHORT***).show();  
 **else** Toast.*makeText*(MainActivity.**this**,  
 **" Wrong!"**, Toast.***LENGTH\_SHORT***).show();  
 **break**;  
 *//finish switch cases 2-4* }*//end switch* }  
 });  
 imageListener();  
 }*//end buttonListener* **public void** imageListener() {  
  
 **image** = findViewById(R.id.***imageView1***);  
 **image**.setOnClickListener(**new** View.OnClickListener() {  
  
 @Override  
 **public void** onClick(View view) {  
 *// get new question for viewing* **if** (*questionNum* == 4)  
 *//reset count to -1 to start first question again  
 questionNum* = -1;  
 **txtView**.setText(**stringList**.get(++*questionNum*));  
 *//reset radio button (radioTrue) to default* **radioQuestions**.check(R.id.***radioTrue***);  
 }  
 });  
 }*//end imageListener*}*//end activity*

**Next include all the following imports:**

**import** android.app.Activity;  
**import** android.app.ProgressDialog;  
**import** android.os.AsyncTask;  
**import** android.os.Bundle;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.view.View;  
**import** android.view.View.OnClickListener;  
**import** android.widget.Button;  
**import** android.widget.ImageView;  
**import** android.widget.RadioButton;  
**import** android.widget.RadioGroup;  
**import** android.widget.TextView;  
**import** android.widget.Toast;  
**import** java.io.BufferedReader;  
**import** java.io.InputStream;  
**import** java.io.InputStreamReader;  
**import** java.net.HttpURLConnection;  
**import** java.net.URL;  
**import** java.util.ArrayList;

You’ll notice from the code, that you have listener events handling both the display results button and the image you have which is made clickable. Study over the code well as you will tweak some things thru momentarily.

Run your app and test it! You’ll notice a couple of things off the bat. For one, you have 2 stars defaulting that are visible from your rating bar. That’s okay that will be adjusted in a bit. Press your **Display Results** button for the first two questions leaving the True button checked. You should see a Toast reply of **Right!** on the very first round. Hit your Next image and press **Display Results** again and you will a Toast reply of Wrong! for the next question. Of course pressing the False button at this point your should see a message of Right! Upon clicking the Display Results button once again. Meaning the correct radio button has been pressed as per the switch case code above per the buttonListener onClick code.

**STEP 5 Adjust your code.**

Go back to your switch statement in main and finish up your switch cases for cases 2-4 (using similar logic for cases 0-1) which are for cases covering the remaining questions of the quiz .

Study over the quiz questions from the url file for questions 3-5 and determine what is

true and what is false. Code each remaining case to reflect a correct Toast response

message based on your assumptions of what the correct answers maybe for questions 3-5

when cases 2-4 are triggered.

Run your app once again and test all 5 questions and responses. Once you are satisfied with your results, make a **snapshot** of a right answer **and** a wrong answer for a particular

question. You may wish to make your Toast message set to LENGTH\_LONG at this point to

snapshot the message display in ample time.

**STEP 6 Showing the results of questions that were correct to the user.**

You can depict, via shaded Red stars in your Rating Bar how many questions the user had correct. The amount of stars you’ll notice add up to 5 to reflect the 5 questions rated right or wrong. Non shaded Gray stars reflect an incorrect answer by default for this app’s purpose.

To show the number of shaded Red stars which actually will represent *correct* answers, merely place the following code in your activity where you deem fit. Example follows...

RatingBar rb = findViewById(R.id.*ratingBar*);

rb.setRating(3);

where the number 3 would be an arbitrary number *example* depicting the number of correct

responses from the user within **setRating()**’s parameter.

Make sure to include the following import in as well:

**import** android.widget.RatingBar;

In your code then, have the ability to track the number of correct answers your activity has

where you deem necessary and then pass the result into your setRating() method when the last question’s response is displayed so the stars reflect the proper right(s) vs. wrong(s) count in proper detail. Make sure your rating bar is invisible *until* the last question is chosen with the Display button being then pressed.

As you run your app again, **snapshot** question 3 this time with a result of Right! as a displayed message.

**Snapshot** also the rating bar results at the end showing one set of Red stars and another snapshot showing a differing Gray star set of incorrect responses (hope someone gets them all correct even).

Include both portrait and landscape mode UI screenshots.

Grads:

Randomize the question results on startup so a possible different sequence of questions appear at startup to help maintain the integrity of questions presented. Also include a [timer](https://developer.android.com/reference/java/util/Timer) to track long it takes to finish the quiz. Include the lapsed time it took for the quiz to complete when the very last question is answered. **Snapshot** a differing startup question that appears at runtime initially as well as your lapsed time result for the quiz at finishing time.

Include also your MainActivity.java code and snapshots into pdf files as well, for credit.