ITIS/ITCS 4180/5180 Mobile Application Development
Homework 3

Basic Instructions:

1. In every file submitted you MUST place the following comments:
   a. Assignment #.
   b. File Name.
   c. Full name of all students in your group.
2. Each group should submit only one assignment. Only the group leader is supposed to submit the assignment on behalf of all the other group members.
3. Your assignment will be graded for functional requirements and efficiency of your submitted solution. You will lose points if your code is not efficient, does unnecessary processing or blocks the UI thread.
4. Please download the support files provided with this assignment and use them when implementing your project.
5. Export your Android project and create a zip file which includes all the project folder and any required libraries.
6. Submission details: The file name should follow the following format: **Group#_HW03.zip**
7. **Failure to follow the above instructions will result in point deductions.**

Homework 3 (100 Points)

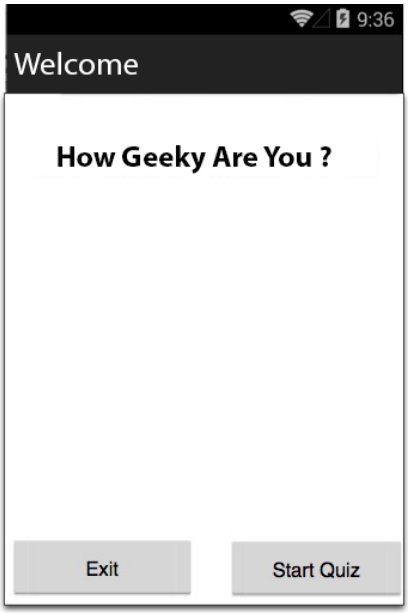
In this assignment you will develop “How Geeky Are You?” personality test. It is Trivia Quiz where you will be given a set of questions and based on the answers that you provide, the game will show your “geeky” level.

Part A: Splash Screen Activity (15 Points)

In the first UI, the user will see a splash screen displaying a photo (**`splash-screen.jpg`**) and a Start Quiz Button. The Splash takes 8 seconds to finish but user can choose to not wait and press the “Start Quiz” button to go directly to the Welcome Activity without waiting for 8 seconds. Figure 1, shows the Splash Screen Activity.

Figure 1, Splash Activity

(a) Downloading the questions



(b) Finished downloading the questions

Figure 2, Welcome Activity

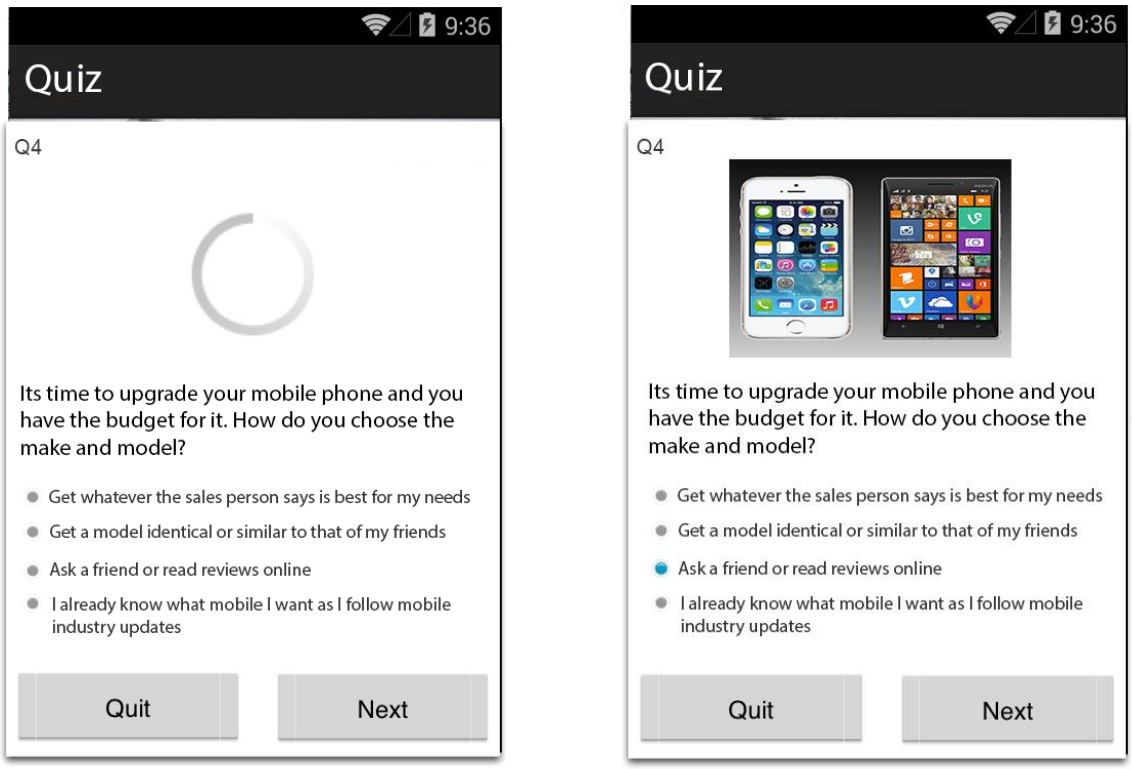
Part B: Loading and Parsing Questions (Welcome Activity) (35 Points)

The questions and answers for this app should be retrieved from the below url: http://dev.theappsdr.com/lectures/trivia/index.php, each question is formatted in a separate line with entries separated by the semicolon ";". An example question is listed below:

```
4;Its time to upgrade your mobile phone and you have the budget for it. How do you choose the make and model?;<IMAGE_URL>;Get whatever the sales person says is best for my needs;0;Get a model identical or similar to that of my friends;2;Ask a friend or read reviews online;5;I already know what mobile I want as I follow mobile industry updates;10
```

The question line is divided as follows: the first item is the question number, then the Question text, then the image url (if there is any), then the option and then the points related to that option. Other options follow similarly. For example, here the third option is "`Ask a friend or read reviews online`" and the next value after the delimiter is "`5`". So if the user selects the 3rd option as done in Figure 3(b), he gets 5 points for that particular question. You also need to shuffle all the options so the options won't necessarily appear in the order as shown in Figure 3(b). You should note that the number of possible answers can vary, in the above example there are 4 possible answers, but in other questions there might be less or more. Below are the requirements:

1. The **Welcome Activity** should use an AsyncTask or Thread to retrieve the questions and answers at the URL provided above. While the questions are being retrieved the Welcome Activity should display a progress bar as indicated in Figure 2(a). The "Start Quiz" button should be disabled while the questions are being retrieved.
2. You should create a Question Class that should hold the parsed question, possible answers and related points, url (if there is any). The AsyncTask or Thread should return a List of Question objects. Make sure the Question class should implement the Parcelable interface. Do not pre-download any of the images and do not pre-store the images in the Question Class.
3. Figure 2(b) shows the activity after the loading and parsing are completed. The "Start Quiz" button should be enabled, the progress bar should be removed indicating that app is ready to view quiz questions.
4. Clicking the enabled "Start Quiz" button should start the Quiz Activity.
5. Clicking the "Exit" button should exit the application.



(a) While loading the image

(b) After Loading the image

Figure 3, Quiz Activity Wireframe

Part C: Quiz Activity (40 Points)

The Quiz activity is started by the Welcome activity and it should receive the list of retrieve questions from the Welcome activity via the intent. Figures 3(a) and 3(b), show the wireframe of the Quiz activity. The activity shows the question number a question text, and the set of answer options. The Quiz activity shows the current question's image(if any). Note that not all question will have a question image, if there is no image, the url entry will be left as "" to indicate no image. You should display, an image to indicate that there was no image found for this question. The number of possible answers is variable, your interface should be able to handle a variable number of answer choices and make sure that the answer choices are shuffled. The requirements are as follows:

1. If the current question has an image, then the image should be downloaded from the specified image url indicated in the question using a separate thread (or AsyncTask) and not using the main thread.
2. Your activity should ensure that the ***downloaded image is displayed only when it's question is the currently displayed question and not when other questions are displayed***.
3. While the question image is still loading you should display a progress bar indicating the image is loading, as indicated in Figure 3(a).

4. Users cannot skip to the next question without selecting atleast one option. If the user clicks on next button without selecting an option, an error Toast message should be displayed saying “Please select atleast one option”.
5. When displaying the next question you should not use a new Question activity instead update the layout of the Quiz activity to display the new question. Note that the number of choices for each question varies, so the views representing the choices should be dynamically generated in your code and should not be statically created.
6. If the user clicks the “Quit” button the activity is finished and the user is sent back to the Welcome activity.
7. When the user answers a question, you should keep track of the number of points earned. You can add the points going further.
8. Upon answering the last question, the Quiz activity should start the Result activity and send it the required information in the intent to display the result.

Part D: Result Activity (10 Points)

Figure 4 shows the wireframe for the Result activity. This activity shows the user the percentage correctly answered quiz questions. Clicking the “Quit” button should send the user to the Welcome activity. Clicking the “Try Again” button should send the user back to the Quiz activity and should redisplay the first quiz question to enable the user to retry the quiz from the first question.

Figure 4, Results Activity

The result will be calculated as follows - The total points should be in the range of 0 – 72. According to the points scored, there will be 3 categories as below:

0 – 10 points: Non-Geek
11- 50 points: Semi-Geek
51 - 72 points: Uber-Geek

The result activity should show the related image according to the points earned. The images are present in the support files.

Description for each of the category is as given below:

Uber Geek:

You are the geek supreme! You are likely to be interested in technology, science, gaming and geeky media such as SciFi and fantasy. All the mean kids that used to laugh at you in high school are now begging you for a job. Be proud of your geeky nature, for geeks shall inherit the Earth!

Semi Geek:

Maybe you're just influenced by the trend, or maybe you just got it all perfectly balanced. You have some geeky traits, but they aren't as "hardcore" and they don't take over your life. You like some geeky things, but aren't nearly as obsessive about them as the uber-geeks. You actually get to enjoy both worlds.

Non-Geek:

There isn't a single geeky bone in your body. You prefer to party rather than study, and have someone else fix your computer, if need be. You're just too cool for this. You probably don't even wear glasses!!