ITIS/ITCS 5180 Mobile Application Development
Homework 4

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 loose 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:
   a. Only a single group member is required to submit on Canvas for each group.
   b. The file name is very important and should follow the following format: **Group#_HW04.zip**
   c. You should submit the assignment through Canvas: Submit the zip file.
7. **Failure to follow the above instructions will result in point deductions.**
8. **The required Android Virtual Device (AVD) should have minimum SDK version set to 20 and target SDK at 25.**

Homework 04 (100 Points)

In this assignment you will develop a Trivia quiz. Trivia questions will be downloaded as an JSON file from a remote server. You will get familiar with parsing JSON and generating the proper user interface for showing each parsed question. This project is composed of three Activities, which include: **Main**, **Trivia**, and **Stats** Activities.

Part 1: Main Activity (40 points)

The **Main** activity is responsible for the loading of the trivia contents (questions and answers) included in the JSON web service located at the below address: http://dev.theappsdr.com/apis/trivia_json/index.php. The content of the JSON file should be retrieved by establishing a HTTP connection to the service. The web service return a JSON response that contains a **questions** array which contains question elements.

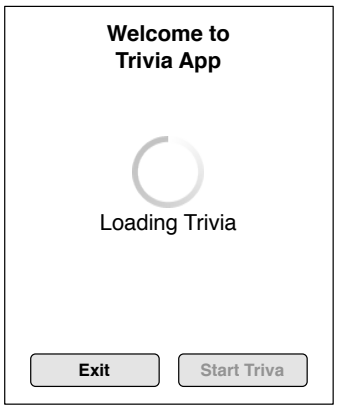
Each object in the question array will contain a **text** element, and a **choices** element. The **choices** element has an array of **choice** elements (number of choices is variable). There is an **answer** element which holds the sequence number of the correct answer. For example, in the above question the answer is the choice containing "George Washington" which has the sequence number of 1. A **question** element can also have an **image** element that should be displayed to a user when viewing the question. A **image** element will hold a URL to an image on a remote server.

To parse the JSON, you should follow the instructions in the video lectures. All the parsing and HTTP connections should be performed by a worker thread or an AsyncTask and should not be performed by the main thread. When a question is parsed from the JSON, it should be mapped to a **Question** object. Your **Question** class must implement the Serializable interface (or you can use Parcelable interface). You should keep track of all questions in a List, as you will use them to show the questions to the user later on. **All the AsyncTasks should be implemented as a separate class and should not be an inner class to any activity.**

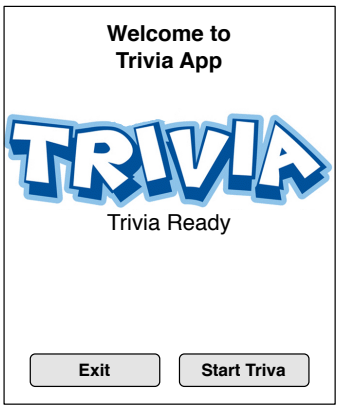
The below figure shows the wireframe of the **Main** activity. Note that when the activity is created it should retrieve and parse the trivia questions. The activity progress should be presented as shown in Figure 1(a). Note that the "Start Trivia" button is disabled while the loading and parsing are being performed. Figure 1(b) shows the activity after the loading and parsing are completed. Note that the "Start Trivia" button is enabled. The parsing should generate a list of questions. Clicking the "Start Trivia" button should start the **Trivia** activity. Clicking the "Exit" button should exit the application.

Part 2: Trivia Activity (50 points)

When the **Trivia** activity is instantiated by the **Main** activity it will receive a list of the trivia questions. Figures 2(a) and 2(b), show the wireframe of the Trivia activity. The activity shows the question number, a countdown timer, a question text, and the set of answer options. The Trivia activity shows an image if one exists for the current question.



1(a) While Loading and Parsing

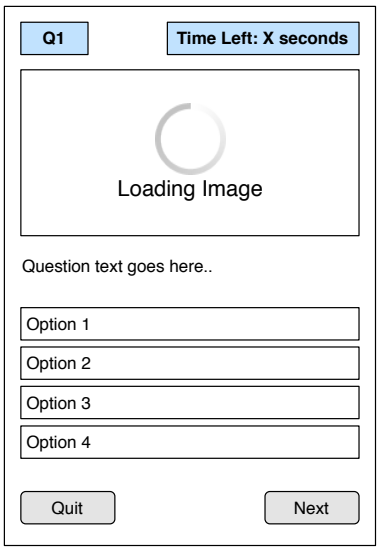


1(b) After Loading and Parsing

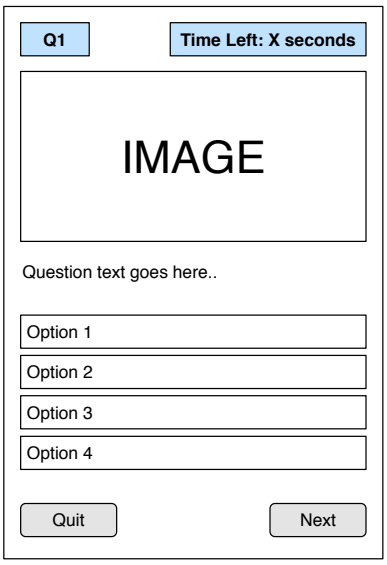
If the current question has an image, then the image should be downloaded from the specified image url indicated in the question using a separate thread (or AsyncTask) and not using the main thread. Your activity should ensure that the downloaded image is displayed only when it's question is the currently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

When the user answers a question, you should detect whether the selected answer was correct or not, and keep track the number of correctly answered questions. Then you should update the **Trivia** activity to display the next question. Do not use a separate activity for each question, but instead update the layout of the **Trivia** activity to show the new question. The choices could be displayed using a RadioGroup. 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 in the layout xml.

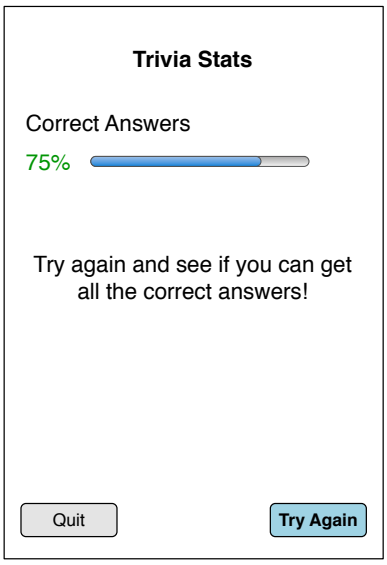
The countdown timer should start once the user starts the first question. If the countdown timer reaches 0 before user answers all the questions, it should be assumed that the remaining questions were answered incorrectly, then the user should be sent directly to the "Stats" activity. If the user manages to answer all the questions within the allotted time (2 minutes), then upon clicking next the user should be sent to **Stats** activity.



2(a) Trivia Activity (Image Loading)



2(b) Trivia Activity (Loaded Image)



2(c) Stats Activity

Part 3: Stats Activity (10 points)

Figure 2(c) shows the wireframe for the Stats activity. This activity shows the user the percentage correctly answered trivia questions. Clicking the "Quit" button should send the user to the **Main** activity. Clicking the "Try Again" button should send the user back to the **Trivia** activity and should redisplay the first trivia question to enable the user to retry the trivia from the first question.