

**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. Please download the support files provided with this assignment and use them when implementing your project.
4. Export your Android project as follows:
  - a. From eclipse, choose “*Export...*” from the *File* menu.
  - b. From the Export window, choose *General* then *File System*. Click *Next*.
  - c. Make sure that your Android project for this assignment is selected. Make sure that all of its subfolders are also selected.
  - d. Choose the location you want to save the exported project directory to. For example, your *Desktop* or *Documents* folder.
  - e. When exporting make sure you select *Create directory structure for files*.
  - f. Click Finish, and then go to the directory you exported the project to. Make sure the exported directory contains all necessary files, such as the .java and resource files.
5. Submission details:
  - a. When you submit the assignment, compress your exported Android project into a single zip file. The format of compressed file name is HW#.zip
  - b. You should submit the assignment through Moodle: Submit the zip file.
6. **Failure to follow the above instructions will result in point deductions.**

## Homework 4 (200 Points)

In this assignment you will develop a Trivia quiz. Trivia questions will be downloaded as an XML file from a remote server. You will get familiar with parsing XML 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.

### Notes:

1. The recommended Android Virtual Device (AVD) should have a Target = Android 4.0.3, and a Skin = HVGA.
2. All API calls and parsing should happen through threads and should not block the main thread.

### Part A: Main Activity (80 Points)

The **Main** activity is responsible for the loading of the trivia contents (questions and answers) included in the XML web service located at the below address:

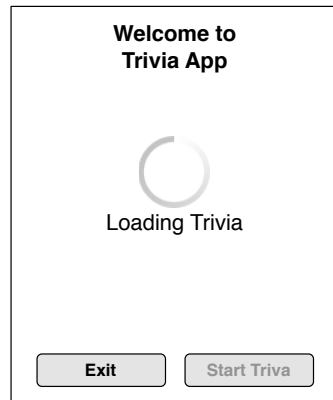
<http://cci-webdev.uncc.edu/~mshehab/api/trivia.xml>. The content of the XML file should be retrieved by establishing a HTTP connection to the service. The trivia.xml file contains a **questions** element as the root element, which contains a number of **question** elements. An example question element is as follows:

```
<questions>
  <question id="1">
    <text>Who is the first President of the United States of America?</text>
    <image>http://cci-webdev.uncc.edu/~mshehab/api/figures/georgewashington.png</image>
    <choices answer="0">
      <choice>George Washington</choice>
      <choice>Thomas Jefferson</choice>
      <choice>James Monroe</choice>
      <choice>John Adams</choice>
    </choices>
  </question>
  .....
</questions>
```

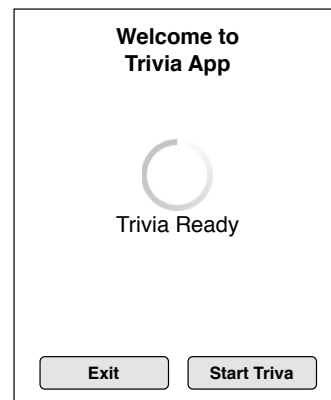
A **question** element will contain a **text** element, and a **choices** element. The **choices** element has a number of **choice** elements (number of choices is variable). The **answer** attribute of the **choices** element indicates the index of the correct answer and starts from 0. A **question** element can also have a **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 XML, you should use one of the methods described in class (SAX, or Pull). 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 XML, it should be mapped to a **Question** object. Your **Question** class must extend the provided abstract class **IQuestion** (In support files). You should keep track of all questions in a List, as you will use them to show the questions to the user later on.

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.



1(a) While Loading and Parsing



1(b) After Loading and Parsing

## **Part B: Trivia Activity (100 Points)**

When this activity is instantiated by the **Main** activity it will receive a list of the trivia questions. Figure 2(a), shows the wireframe of the Trivia activity. The activity shows the question number, a countdown timer, a question text, and the set of answer options. It also shows an image if one exists for the current question. The user should have at most 4 minutes to finish all the trivia questions. The countdown timer monitors the start and end of the 4 minutes. Users can answer the question or skip to the next one. A skipped question should be counted as an incorrect answer. If the user clicks the “Quit” button the activity is finished and the user is sent back to the **Main** 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 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. (Check the Android’s **CountDownTimer** class). If the user manages to answer all the questions within the allotted time (4 minutes), then upon clicking next the user should be sent to **Stats** activity.

Q1 Time Left: X seconds

IMAGE

Question text goes here..

Option 1

Option 2

Option 3

Option 4

Quit Next

2(a) Trivia Activity

Trivia Stats

Correct Answers

75%

Try again and see if you can get all the correct answers!

Quit Try Again

2(b) Stats Activity

### **Part C: Stats Activity (20 Points)**

Figure 2(b) shows the wireframe for this activity. This activity shows the user the percentage of correct answers they have got. 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 redisplay the first trivia question to enable the user to retry the trivia from the first question.