IEMS 5722

Mobile Network Programming and Distributed Server Architecture

2014-2015 Semester 3

Assignment 3: Google Cloud Messaging and Notifications

Due Date: 5th March, 2015 (Thursday)

Notes:

i.) See the instructions at the end of this assignment, follow them to submit your files for marking

ii.) Late submissions will receive 30% mark penalty

1. Aim

- To learn how to develop an Android app that is able to receive push messages from our server and fetch data from a URL embedded in the message

2. Objectives

- Set up the app to use Google Cloud Messaging (GCM) to receive push messages
- Handle the push message received
- Create a notification to alert the user that a push

3. Instructions

In this exercise, your app will register with the **GCM service** to receive push messages sent from our server. The server will send push message to registered apps periodically (**every 2 minutes**). The GCM message contains **a title**, **a short description** and **a URL**, among other information.

On receiving the push message, your app should display a **notification** to the user. If the user clicks on the notification, your app should then use the URL embedded in the message to retrieve additional information and display it to your user.

The fetched data will be in **JSON** format, which contains a list of objects - each has a title, description and the URL of an image. This list needs to be displayed as a list view with the picture, title and description as a list view item.

For your app to work correctly, you should:

1) Set the package name to **com.iems5722.assignment3**,
2) Use **145180457203** as the GCM Sender ID.

4. Tasks

4.1. Set up to use GCM

The first task is to ensure the Android device has **Google Play Services** installed. This is required if you want to use GCM. Additionally, you have to ensure the right permissions and listeners are stated in the manifest.

Assuming the check is successful, **register** your app with Google to receive GCM messages. Google will reply with a **token**. This token should be stored in your app for future use. Store the token in the Shared Preference. Refer to the Data Storage Options in the appendix for further information.

The token also needs to be submitted to our server so that it can push messages to your app. You should submit it to this URL: http://iems5722v.ie.cuhk.edu.hk:8080/gcm_register.php with the following parameters using the **POST** method:

- **sid** (your student ID),
- **gcm_id** (The GCM token you got by registering your app against GCM).

Our server will respond with a 200 STATUS OK message if you have registered successfully.

4.2. Handle GCM messages and notify the user

To listen for the GCM message, implement a **WakefulBroadcastReceiver**. This receiver ensures your device does not sleep while a message is being processed. On receiving a push message, the receiver will then start an **IntentService** to process it. You will need to extract the **title**, **description** and **URL** from the message. The relevant keys are "**title**", "**desc**" and "**url**".

If your app does not receive any message after 2 minutes, check whether your permissions are set up correctly. In particular, check whether the class that implements WakefulBroadcastReceiver is registered properly to listen for incoming GCM messages.

After the data has been processed by the IntentService, a **notification** needs to be displayed to the user. The notification should include the GCM title as the content title, and the GCM description as the content text. The notification should also play a **sound** and **vibrate** to alert the user.

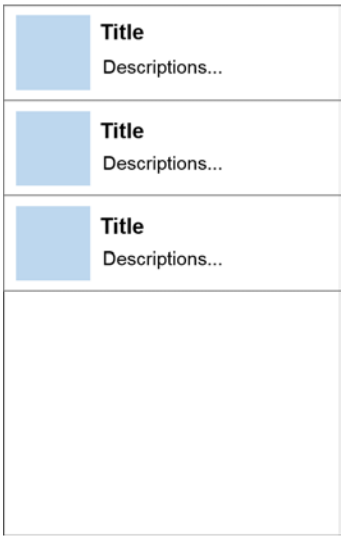
When the user taps on the notification, it should start your app's main activity and pass in the URL from GCM for further actions.

4.3. Fetch data from embedded URL

When the app is launched when the user clicks on the notification, the app should fetch data from the URL and display the list of items in a **ListView** in the main activity. The JSON data will be in the following format.

```
{
    "status": "OK",
    "data": [
        {
            "title": "...",
            "desc": "...",
            "image": "http://...."
        },
        {
            "title": "...",
            "desc": "...",
            "image": "http://...."
        },
        {
            "title": "...",
            "desc": "...",
            "image": "http://...."
        }
    ]
}
```

We are interested in displaying the **data** array. For each row in the ListView, the item should have a square **ImageView**, a **TextView** for displaying the title, and another **TextView** for displaying the description. Note that the image is a URL and your app will need to fetch it in order to render the ImageView. The resultant user interface will be something like what is presented on the right.



When using a ListView, you will need to use a **ListView adapter** to help display the items as the row design is relatively complex. One possible solution is to use an ArrayAdapter, but you are free to choose other options.

5. Bonus Goal

Try to implement a method that allows the user to choose how they would like to be alerted about the notification. The choices are “**sound and vibrate**” (the default option), “**vibrate only**” and “**silent**”. You can add a button somewhere on the main activity, and when the user clicks on it the app will prompt the user to choose one of the three options mentioned above. You will need to store this chosen option in the app, and retrieve this user preference before creating the notification. Refer to Appendix for related articles.

6. References and Resources

Google Cloud Messaging

- Tutorial on implementing a GCM Client
  https://developer.android.com/google/gcm/client.html
- Tutorial on implementing a GCM Server (not required for this assignment)
  https://developer.android.com/google/gcm/server.html

Data Storage in Android Apps

- Data Storage Options
  http://developer.android.com/guide/topics/data/data-storage.html
- User Preferences
  http://developer.android.com/guide/topics/ui/settings.html
- Find existing preferences
  http://developer.android.com/guide/topics/ui/settings.html#ReadingPrefs

Notifications

- http://developer.android.com/guide/topics/ui/notifiers/notifications.html

ListView and Adapters

- http://developer.android.com/guide/topics/ui/layout/listview.html
- http://www.javacodegeeks.com/2013/09/android-listview-with-adapter-example.html

7. Submission

To submission your assignment, create a folder with a name in the following format:
<your_student_id>_assgn3

Copy the follow materials into the folder you created:

- The **src** folder (including all Java source code files)
- The **res** folder (including all the sub-folders and files)
- The **AndroidManifest.xml** file

Compress the folder into a .zip file, and submit it in the CUHK eLearning System online:
https://elearn.cuhk.edu.hk/