IEMS 5722

Mobile Network Programming and Distributed Server Architecture

2014-2015 Semester 2

Assignment 2: Basic Network Programming

Due Date: 12th February, 2014 (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 understand how network programming, including sockets and HTTP requests, and asynchronous operations are handled in Android

2. Objectives

- Modify the translate app you developed in Assignment 1 to request the translation from a server instead of a local hash map
- Send the user input to the server and receive the response via network programming
- Implement the network programming using AsyncTask

3. Basic Settings

3.1. Starting Point

You are encouraged to use the app you submitted in Assignment 1 as the starting point for this assignment. If you do not wish to, you may use the solution to Assignment 1 from GitHub at https://github.com/IEMS5722-Spring2015/A1Solution.

3.2. Server Implementation

In this assignment, you are not required to implement the server by yourself. We have set up a server program with a translation hashmap that is identical to that provided in Assignment 1. The implementation is similar to the solution of Assignment 1: the user input is used as a key to look up the hashmap. If no key is found, it will return an error message. You can check out the source code of the server program at https://github.com/IEMS5722-Spring2015/TranslateServer and at https://github.com/IEMS5722-Spring2015/TranslateServerHTTP.

4. Tasks

In Assignment 1, the translation implementation looks up the supplied hashmap and returns the result to the user. However, in real scenarios, this is not ideal as a useful translation app would have many more languages with a much larger vocabulary. Additionally translation is a complex subject and language processing engines may be required. This processing usually happens on servers which are much more powerful than mobile devices. Therefore, the translation algorithm task has now been shifted to the server.

4.1. Network Programming

In this assignment, you are required to extend your translation app and make it capable of sending requests to a server to ask for the translation of the word input by the user. You need to implement **two methods** to do so.

4.1.1. Using a TCP Connection

Firstly, you need to implement your app such that it will establish a TCP connection to a server. This can be achieved by using socket programming mentioned in the lectures. More specifically, you need to perform the following task:

1. Receive user input
2. On receiving the "submit" action of the user (e.g. when the user clicks on the submit button), perform translation by carry out the following actions
3. Create a TCP socket connection to the server (**host**: iems5722v.ie.cuhk.edu.hk, **port**: 3001)
4. Send the text input by the user as a request to the server
5. Wait for the response of the server
6. Once the response is received, present the result to the user

4.1.2. Using an HTTP Connection

Secondly, you need to implement also the method of using HTTP to send request and receive response from the server. The following tasks are to be performed:

1. Receive user input
2. On receiving user action, perform translation by carry out the following actions
3. Create an HTTP connection to the server
   (**URL**: http://iems5722v.ie.cuhk.edu.hk:8080/translate.php)
4. Send the text input using the **GET** method, by appending a query string to the URL, the word input by the user should be passed using a parameter called "**word**"
5. Wait for the response of the server
6. Once the response is received, present the result to the user

Note that in order to allow the user to choose which method to use, you need to extend your user interface by adding one more "submit" button (e.g. a "TCP Submit" and a "HTTP Submit" button).

In addition, since these network operations may take up to a few seconds, especially when the network is slow, you must implement them using **AsyncTask**, such that the operations can be run on a new thread other than the UI thread.

To create an AsyncTask, create a new class that extends the AsyncTask. The AsyncTask expects three inputs - the first is the input given to the task, the second is an update value, and the third is an output value expected when the task completes. The update value can be used to notify the user of progress, such as a percentage complete dialog. This is not needed for this assignment.

The only mandatory method in AsyncTask is the **doInBackground** method. Other methods include **onPreExecute**, which executes before doInBackground, **publishProgress** and **onProgressUpdate**, which lets you update interface before the task completes, and **doPostExecute**, which runs after doInBackground is complete. A more complete description of AsyncTask can be seen at http://developer.android.com/reference/android/os/AsyncTask.html.

For more information about processes and threads in Android, refer to https://developer.android.com/guide/components/processes-and-threads.html, and for information about sockets, refer to http://developer.android.com/reference/java/net/Socket.html.

4.2. Keeping Records in the App

To allow the user to view the history of his action, you are required to implement a new page (let's call it the "History Page") in your app, which will display a list of words that has been input by the user so far and their corresponding translations. To fulfill this requirement, here is a list of steps you can follow.

1. Create a new button on the main activity, say "Translation Record"
2. Program the button to start a new activity when clicked
3. When the activity is started, load the records from somewhere you have stored (to be discussed below), and display it as a list.

4.2.1. Storing Data in the App

In order to implement the "History Page", you need to keep a record of what the user has input so far. Android offers several different ways for you to store data in the device, such that you can retrieve at a later time. For more details, you can refer to the following page: http://developer.android.com/guide/topics/data/data-storage.html.

How you store your data is up to you in this assignment. Here are some hints.

The easiest way is to use **Shared Preferences**, which is a simple key-value store. You can create a key called "records", and then convert all the records into a String and put it as the value of this key. Of course, you will need to think about how to format the records into a String, and how you can add new records to this.

Another easy way is to use **files**. Create a new file and write a record to a new line. When you want to retrieve the records, simply open the file and read the file line by line. This method is recommended in this assignment.

4.2.2. Presenting a List of Items in Android

In order to present the records, you will probably use the **ListView** UI component in Android. ListView is for presenting a list of cells to the user, and it will handle scrolling automatically if your list is long. ListView is a very common UI component, and can be seen in many popular apps. For more information, read the following links:
http://developer.android.com/guide/topics/ui/layout/listview.html
https://guides.codepath.com/android/Using-an-ArrayAdapter-with-ListView

When using ListView, you need to use an **Adapter** to control what data is presented in the list. In this assignment, you can simply use an **ArrayAdapter** for strings, as the data you are going to present only involves strings.

Note: In this assignment, you have to implement the "History Page" using ListView, such that each row of the list presents one record of the translation history. The format can be something like this: "four : 四".

4.3. Bonus Tasks: Removing a Record

Once you have finished implementing the "History Page", try to allow the user to **delete** a record by long-pressing on a record. You should prompt the user first (e.g. Ask the user whether he really wants to delete the record using a dialog box), before really removing the record for him. After removal, that row should disappear from the UI.

Hint: you will probably need to use the **setOnLongClickListener**() function of ListView. For displaying a dialog box and get the response of the user, check out the following link:
http://developer.android.com/guide/topics/ui/dialogs.html

5. Submission

To submission your assignment, create a folder with a name in the following format:
<your_student_id>_assgn2

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/