ITIS/ITCS 4180/5180 Mobile Application Development
Homework 3
Date Posted: 06/07/2013 at 03:00am
Due Date: 06/13/2013 at 11:55pm

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 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.
6. 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.
7. **Failure to follow the above instructions will result in point deductions.**

Homework 3 (200 Points)

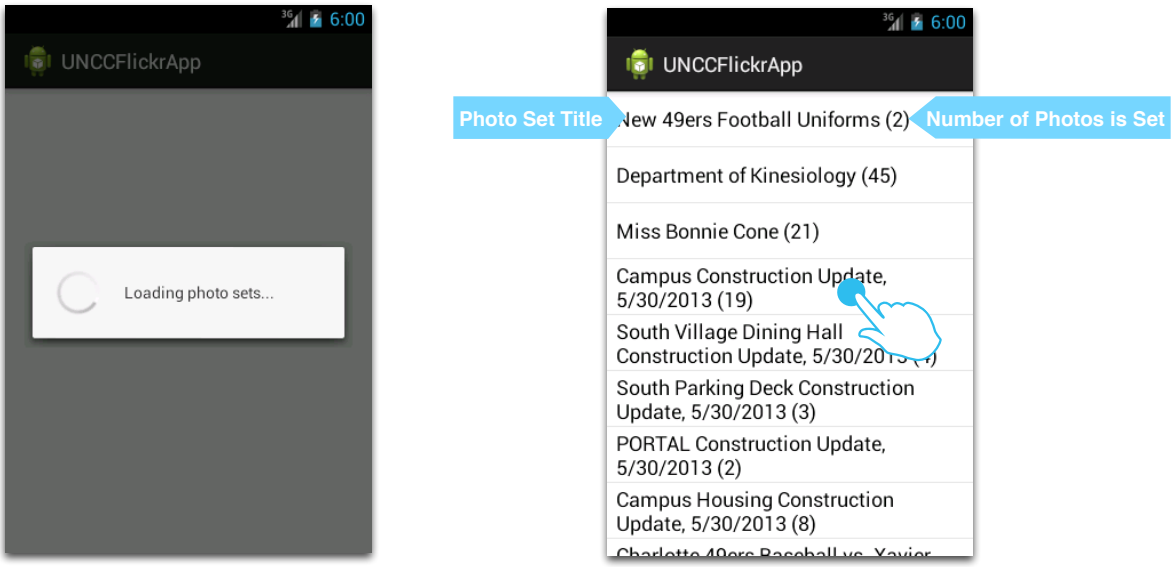
In this assignment you will develop a simple UNC Charlotte photo gallery application for android. The Flickr API will be used to retrieve the different photo streams and photos posted on the UNC Charlotte official Flickr account. The app will enable the user to pick from different photo albums, display contents of each album, and show full sized specific photos. The app is composed of 3 activities, namely **MainActivity**, **GalleryActivity**, and **ImageViewerActivity**. In this assignment you will get familiar with multi-activity communication using Intents, multi-threading, ListView, GridViews, and XML or JSON parsing.

Notes:

1. The recommended Android Virtual Device (AVD) should have minimum SDK version set to 11 and target SDK at least 17. The app should display correctly on 3.2" QVGA (ADP2) (320x480: mdpi).
2. All image downloading should be performed using Threads (or AsyncTask) and your code should not block the main thread.

API Description

- The API used in this assignment is the Flickr API. You should create a Flickr developer account and acquire an API key to be used for this application.
- The Flickr apis used in this assignment are the **flickr.photosets.getList** and **flickr.photosets.getPhotos** apis.
- The **flickr.photosets.getList**: Returns the photosets belonging to the specified user.
  - This api provides both XML and JSON output, you are free to use either.
  - This api requires a "user_id" parameter, set the user_id to "40729938@N03", which is UNC Charlotte's Official flickr user id.
  - Make sure to select the response format and to include your api key as part of the parameters.
  - For information http://www.flickr.com/services/api/flickr.photosets.getList.html, at the bottom of this page there is an API Explorer link that is very useful for testing.
- The **flickr.photosets.getPhotos**: Returns the list of photos in a specific photo set.
  - This api provides both XML and JSON output, you are free to use either.
  - This api requires a "photoset_id", which is the id of the photo set for which the list of photos is to be retrieved. Make sure to include the "extras" parameter and set it to "url_sq,url_m", so that for each photo the photo square url and photo medium url are included in the response.
  - Make sure to select the response format and to include your api key as part of the parameters.
  - For information http://www.flickr.com/services/api/flickr.photosets.getPhotos.html, at the bottom of this page there is an API Explorer link that is very useful for testing.



(a) Main Activity While Loading

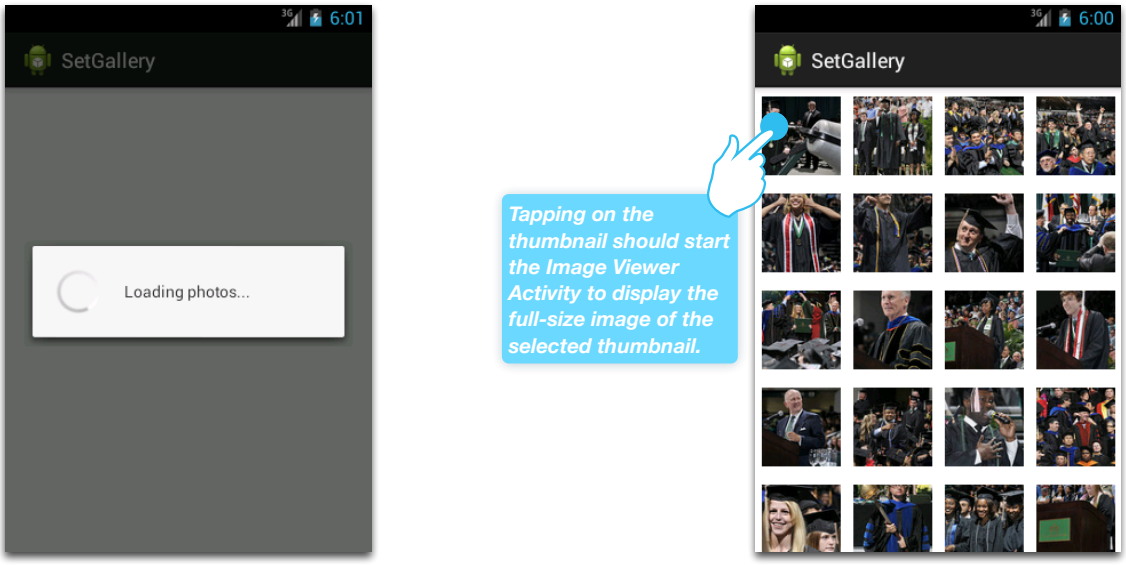
(b) Main Activity After Loading

Figure 1, Main Activity Screen Shot

Part 1: MainActivity (80 Points)

The **MainActivity** is responsible for the loading the titles for the different photo sets provided by the UNC Charlotte Official Flickr account. The interface should be created to match the user interface (UI) presented in Figure 1. The implementation requirements include:

1. Your application should have an application launcher icon, please select your launcher icon to represent your app. Set the application title to “Photo Gallery”.
2. Use a thread pool (or AsyncTask) to communicate with **flickr.photosets.getList** api and to parse the generate result. While the communicating with the api and parsing the data display a ProgressDialog indicating that the photo sets are being loaded, see Figure1(a). Do not use the main threads to download the api results.
3. The ProgressDialog should not be cancelable. The ProgressDialog should be dismissed when all the results are loaded.
4. The retrieved photo set titles and the number of photos in each photo set should be displayed as list item in a ListView, see Figure 1(b). You should setup the ListView and the required adapter.
5. Tapping on a list item for a specific photo set should setup the required intent to start the **GalleryActivity** and to pass it the required photo set information. Hint: Using the Intent Extras the GalleryActivity should be sent the id of the photo set.
6. Tapping on the device’s back button should finish this activity.



(a) Loading Gallery Thumbnails (b) Loaded Gallery Thumbnails

Figure 2, GalleryActivity Screen Shot

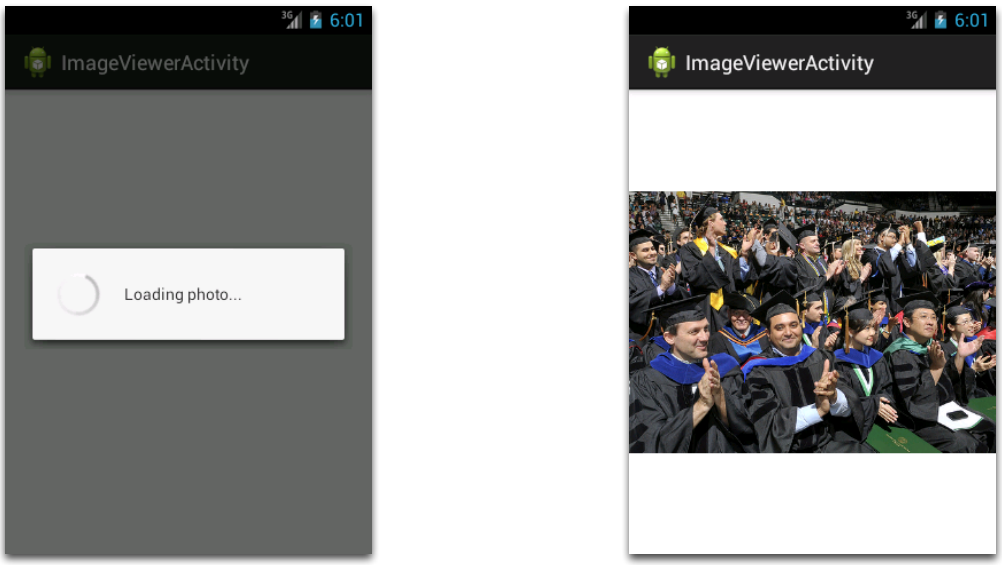
Part 2: GalleryActivity (80 Points)

The **GalleryActivity** is started by the MainActivity when the user selects one of the photo set list items. The GalleryActivity should display all the thumbnail photos of the selected photo set. Note that, the photos' information for a photo set should be retrieved by using the **flickr.photosets.getPhotos** api. Below are the main implementation requirements:

1. The **flickr.photosets.getPhotos** flickr api should be used to retrieve the photos in the photo set that was selected in the MainActivity. Please refer to the API hints listed above setup this api parameters.
2. You should use a GridView to display the thumbnails of all the photos included in the photo set that was selected in the MainActivity. The GridView is a UI component that is very similar to the ListView. To be able to display download the thumbnail images you will need to extend the BaseAdapter. For more information and for a very related example please visit: http://developer.android.com/guide/topics/ui/layout/gridview.html
3. Use a thread pool (or AsyncTask) to communicate with the api, parse the results, and download the thumbnail images for photo urls retrieved. While the communicating with the api and parsing the data display a ProgressDialog indicating that the photos are being loaded, see Figure2(a). Do not use the main threads to download the api results or to download the thumbnails.
4. The ProgressDialog should not be cancelable. The ProgressDialog should be dismissed when all the api data is retrieved and parsed.
5. You should use a GridView to hold the thumbnail image ImageViews. The user should be able to scroll up/down to view all displayed thumbnail images. See Figure2(b).
6. Clicking on a thumbnail image should start the ImageViewerActivity and should pass it the required image information, which is the large image url. Setup the required

onItemClick event handlers. The Gallery Activity should use Intent Extras to pass the the ImageViewerActivity the url of the large photo of selected photo that should be displayed.

7. Tapping on the device's back button should finish this activity.



(a) Loading Full Size Image (b) Full Size Image Loaded

Figure 3, ImageViewerActivity Screen Shot

Part 3: ImageViewerActivity (40 Points)

The **ImageViewerActivity** is started by the GalleryActivity when the user taps on one of the thumbnail images. The ImageViewerActivity should display the full size photo of the selected thumbnail. The implementation requirements include:

1. Use a thread pool (or AsyncTask) to download the full sized photo. While the photo is being downloaded display a ProgressDialog indicating that the photo is being loaded, see Figure 3(a). Do not use the main thread to download any photos.
2. The ProgressDialog should not be cancelable. The ProgressDialog should be dismissed after the photo is downloaded and displayed.
3. You should use a FrameLayout to hold the full sized photo ImageView, see Figure 3(b). Hint: For the full sized photo ImageView, make sure to set the `android:scaleType` attribute to "centerInside".
4. Tapping on the device's back button should finish this activity.

Part 4: Bonus (20 Points)

This bonus involves using swipe events to display the next image in the ImageViewerActivity. The implementation requires include.

1. Use the finger swipe direction on the screen to decide which full size image to be displayed next. If the user swipes from left to right, this should display the next full sized photo in this category. For example, if the currently displayed image has an

index value of 5 in photo url ArrayList, then swiping from left to right should display the image with the index value 6. If the index value of the currently displayed image is N-1, then swiping from left to right should rollover and display image 0, where N is the size of the ArrayList holding the photo objects for the selected photo set.

2. Similarly, If the user swipes from right to left should display the previous full sized photo in this category. For example, if the currently displayed image has an index value of 5, then swiping from right to left should display the image with the index value 4. If the index value of the currently displayed image is 0, then swiping from right to left should rollover and display image N-1, where N is the size of the ArrayList holding the photo objects for the selected photo set.
3. For every full sized image being displayed a Thread pool (or AsyncTask) should be used to download the full sized image. Also the ProgressDialog should be displayed to indicate that the photo is being loaded.

Figure 4, Bonus