ITIS/ITCS 4180/5180 Mobile Application Development
Homework 5
Date Posted: 10/14/2013 at 01:30pm
Due Date: 10/21/2013 at 11:55pm

Basic Instructions:

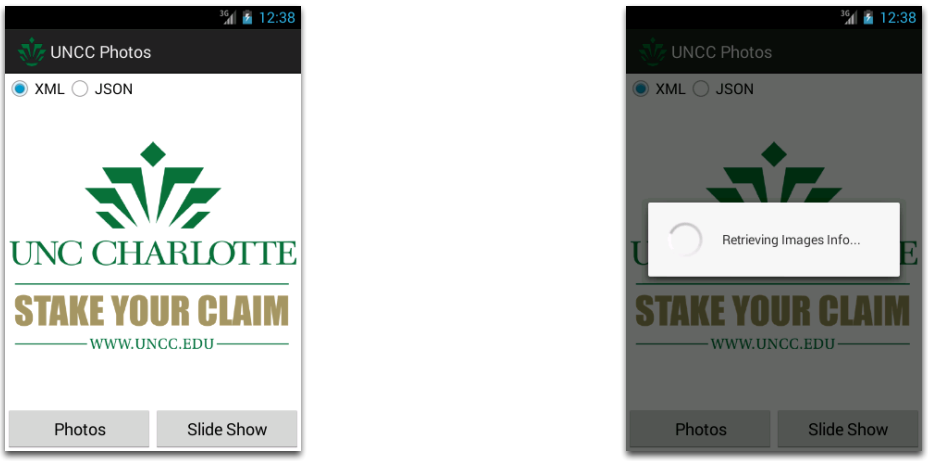
1. In every file submitted you MUST place the following comments:
   a. Assignment #.
   b. File Name.
   c. Full name of all students in your group.
2. Each group should submit only one assignment. Only the group leader is supposed to submit the assignment on behalf of all the other group members.
3. 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 5 (100 Points)

In this assignment you will develop a simple UNC Charlotte photo gallery application for android. The app is composed of 2 activities, namely **MainActivity**, and **PhotosActivity**. In this assignment you will get familiar with multi-activity communication using Intents, multi-threading, XML, and JSON.

Notes:

1. The recommended Android Virtual Device (AVD) should have minimum SDK version set to 12 and target SDK at least 18. The app should display correctly on 3.2" QVGA (ADP2) (320x480: mdpi).
2. All strings should be read from your strings.xml, all dimensions from dimens.xml, and all images from drawable-ldpi. The string values used for the text labels, and button labels should be read from the `strings.xml` file and should not be hardwired in the layout file.
3. All image downloading should be performed using Threads (or AsyncTask) and your code should not block the main thread.
4. For further information about the Flickr api check the documentation available at the ***flickr.photos.search*** api go to the api documentation available at: http://www.flickr.com/services/api/flickr.photos.search.html. You will need to apply for an api key.



(a) Main Activity (b) Loading Images' Information

Figure 1, Main Activity

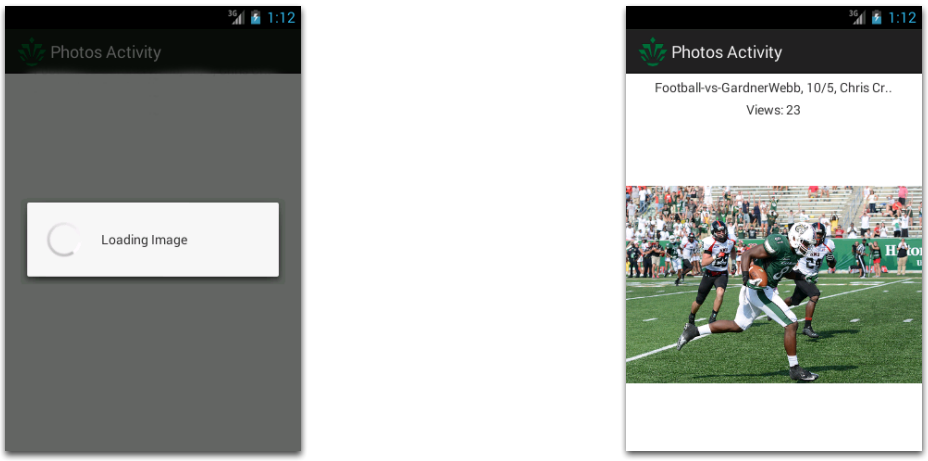
Part 1: MainActivity (80 Points)

The **MainActivity** is a simple activity that contains two buttons and a radio group. The interface should be created to match the user interface (UI) presented in Figure 1(a). 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. The parsing and retrieval of the photos with the UNCC tag should be performed in this activity using an AsyncTask or Thread. The content of the Flickr API should be

retrieved by establishing a HTTP connection to the service. 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. While, the Flickr api result is being loaded you should indicate that using the progress alert as indicated in Figure 1(b).

3. Tapping either the “Photos” or “Slide Show” buttons should start the communication and parsing of the Flickr API. The retrieved list of photos’ information (title, url, and number of views) should then be sent to the Photo Activity using extras. Hint: Use the Parcelable approach.
   a. The “flickr.photo.search” parameters should be configured such that tags=uncc, the extras=views,url_m, and the per_page=100.
4. The format of the API result should be selected based on the radio buttons at the top of the app. If XML is selected then the API should be configured to return a XML result. Similarly if the JSON is selected then the API should be configured to return a JSON result. Your code should handle the parsing of both formats.
5. Tapping the “Photos” button should start the Photo Activity in photo display mode. Tapping the “Slide Show” button should start the Photo Activity in the slide show mode.



(a) Loading Image (b) Loaded Image and Information

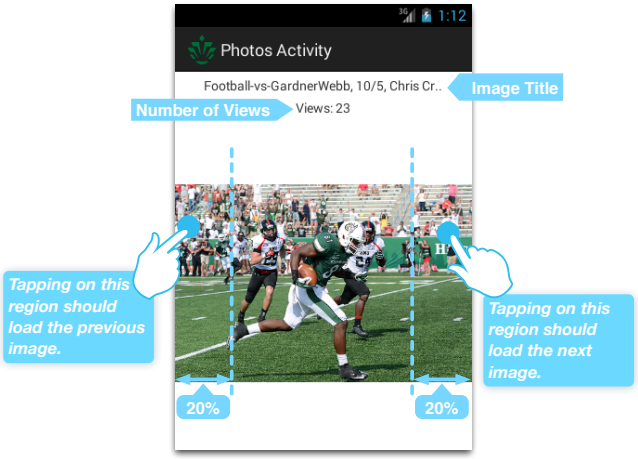
Figure 2, Photo Activity

Part 2: Photo Activity in Photo Display Mode (10 Points)

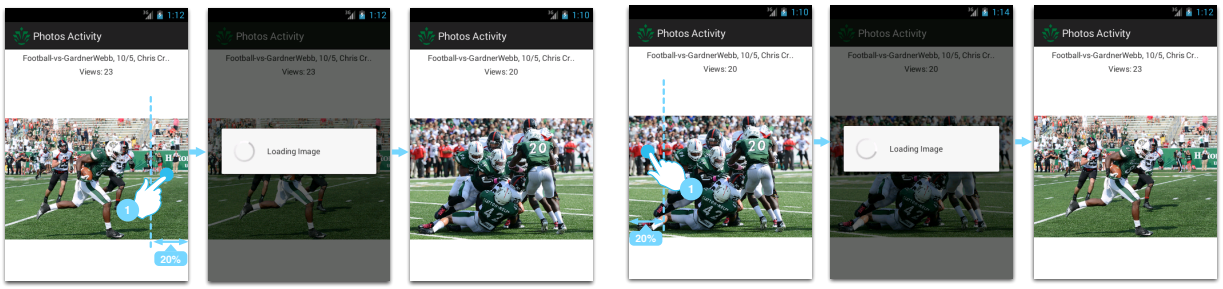
The **PhotoActivity** is started in the Photo Display Mode by the MainActivity when the user taps the “Photos” button. The photo information list should be sent to the Photo Activity from the Main Activity. The photos information list should be sorted based on number of views in descending order. In this mode the Photo Activity should start by displaying the first full size image based on the image url provided in sorted photos information list. See Figures 2(a) and 2(b). The implementation requirements include:

1. Use a thread pool (or AsyncTask) to download the photo. While the photo is being downloaded display a ProgressDialog indicating that the photo is being loaded, see Figure 2(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. The images should be downloaded only when needed, your implementation should not pre-download all the images. At any given time only maintain a reference to the displayed photo. Do not store the downloaded images in memory as this is not efficient and will crash your application.
4. The photo title and number of views should be displayed above the image as indicated in Figure 2(b).
5. Tapping on the right region of the ImageView (20% of the ImageView Width) should display the next photo. For example, if the currently displayed image has an index value of 5 in photos information list, then tapping on right region should display the image with the index value 6. If the index value of the currently displayed image is N-1, then tapping the right region button should rollover and display image 0, where N is the size of the list holding the photos information. Refer to Figure 3(a), and Figure 3(b) for an example.
6. Tapping on the left region of the ImageView (20% of the ImageView Width) should display the previous photo. For example, if the currently displayed image has an index value of 5, then tapping on left region should display the image with the index value 4. If the index value of the currently displayed image is 0, then tapping the left region should rollover and display image N-1, where N is the size of the list holding the photos information. Refer to Figure 3(a), and Figure 3(c) for an example.



(a) Tapping Regions in the Photo Activity



(b) Tapping the Right Side of the Image

(c) Tapping the Left Side of the Image

Figure 2, The Photo Activity

Part 3: Photo Activity in Slide Show Mode (10 Points)

The **PhotoActivity** is started in the Slide Show Mode by the MainActivity when the user taps the “Slide Show” button. The photo information list should be sent to the Photo Activity from the Main Activity. The photos information list should be sorted based on number of views in descending order. In this mode the Photo Activity should start by displaying the first full size image based on the sorted photos information list. In this mode the photos should be displayed automatically in sequence with a 2 seconds delay between each photo being displayed. See Figure 2(b). The implementation requirements include:

1. Use a thread pool (or AsyncTask) to download the full sized photo. Do not use the main thread to download any photos. While the photo is being downloaded **do not** display a ProgressDialog.
2. In this mode the right and left regions indicated in Part 2 should be disabled.
3. The images should be downloaded only when needed, your implementation should not pre-download all the images. At any given time only maintain a reference to the displayed photo. Do not store the downloaded images in memory as this is not efficient and will crash your application.
4. If the index value of the currently displayed image is N-1, then after waiting 2 seconds rollover and display image 0, where N is the size of the list holding the photos information.