

ITIS/ITCS 4180/5180 Mobile Application Development

Homework 5

Date Posted: 03/13/2015 at 20:00

Due Date: 03/20/2015 at 23:55

---

**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 lose 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. All the group members should submit the same zip file.
  - b. The file name is very important and should follow the following format:  
**Group#\_HW05.zip**
  - c. 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 build on your midterm app, which makes simple HTTP requests and will parse the iTunes RSS feed data, **and use [parse.com](https://parse.com) to store and retrieve user and app information**. The new app will include Login Activity and SignUp Activity, Apps Activity, Preview Activity and WebView Activity.

### Important App Requirements:

Points will be deducted if your application does not follow the below requirements:

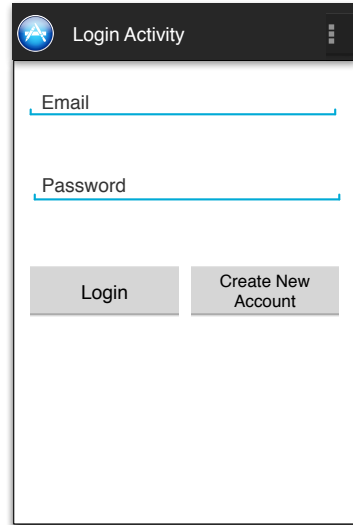
1. The required Android Virtual Device (AVD) should have **minimum SDK version set to 14 and target SDK at least 19**. The app should display correctly on Nexus 5. Your assignment will not be graded if it does not meet these requirements, and you will not be granted any points on your submission.
2. All API calls, XML parsing and image downloading should be performed using Threads (or AsyncTask) and your code should not block the main thread.
3. All [parse.com](https://parse.com) communication should be performed using the background mechanisms provided by [parse.com](https://parse.com) and should not block the main thread.

## API Description

- The API used in this assignment is the iTunes RSS API. The api retrieves 100 top grossing applications provided by the iTunes store. Figure 1 shows the XML response and the required fields. To access the XML stream issue a GET request to the below url:
  - <https://itunes.apple.com/us/rss/topgrossingapplications/limit=100/xml>
  - You are required to access the api and retrieve for each app the following fields: id, app title, developer name, url, small photo url, large photo url, and app price. The required fields are highlighted in Figure 1.

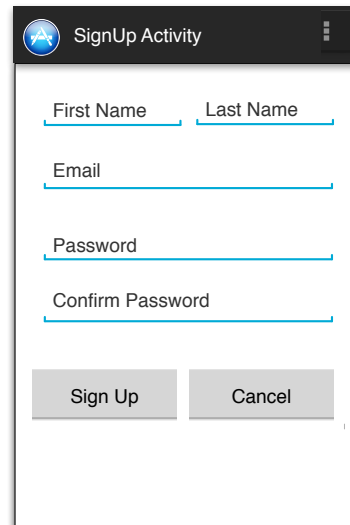


### Figure 1, Required XML Components



The Login Activity wireframe shows a mobile app interface with a dark header bar containing a logo and the title "Login Activity". Below the header, there are two text input fields: "Email" and "Password". At the bottom, there are two buttons: "Login" and "Create New Account".

(a) Login Activity



The SignUp Activity wireframe shows a mobile app interface with a dark header bar containing a logo and the title "SignUp Activity". Below the header, there are four text input fields: "First Name", "Last Name", "Email", and "Password", followed by a "Confirm Password" field. At the bottom, there are two buttons: "Sign Up" and "Cancel".

(a) SignUp Activity

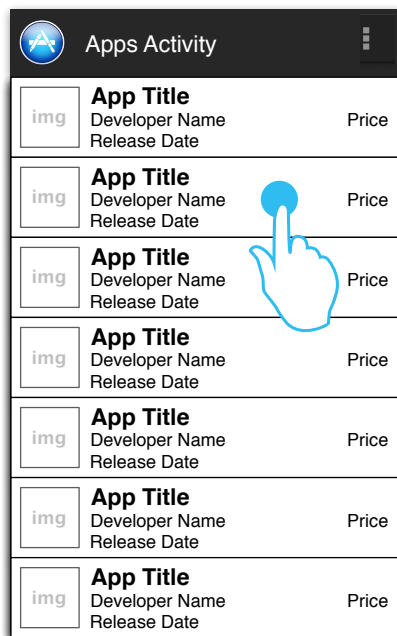
**Figure 2, Wireframe for Login and SignUp Activities**

### **Part 1: User Signup and Login (10 Points)**

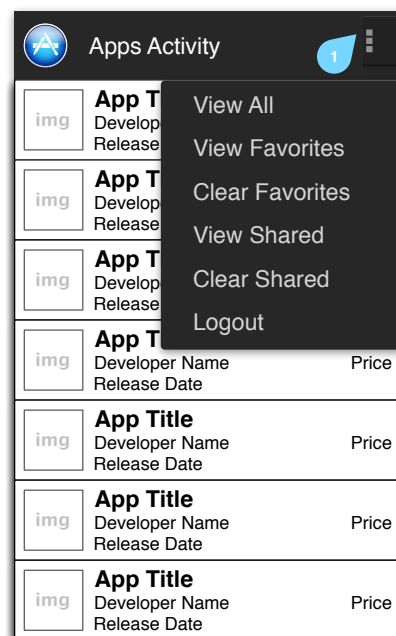
Your app should implement both login and signup functionalities. You should use [parse.com](https://parse.com) to Store the user's full name, email address and password in the User class. The requirements are as follows:

1. The launcher activity should be set to the Login activity. When the app first starts, the Login activity should check if there is a current user session, by using the parse provided methods to check if there is a valid current user:
  - a) If there is a current valid user, then start the Apps activity, and finish the Login activity.
  - b) If there is no current valid user, then the Login activity should be used to provide user login.
2. Create a Login activity (Figure 2(a)):
  - a) The user should provide their email and password.
    - Check the user input, if the email or password field is left empty, print a toast message indicating that these fields are required and don't submit the login information to [parse.com](https://parse.com).
  - b) The provided credentials should be used to authenticate the user using [parse.com](https://parse.com). Clicking the "Login" button should submit the login information to [parse.com](https://parse.com) to verify the user's credentials.
    - If the user is successfully logged in then start the Apps activity, and finish the Login activity.
    - If the user is not successfully logged in, then show a toast message indicating that the login was not successful.
  - c) Clicking the "Create New Account" button should start the Signup activity and finish the Login activity.

3. Create a Signup activity (Figure 2(b)):
  - a) Clicking the “Cancel” button should finish the Signup activity and start the Login activity.
  - b) The user should provide their first name, last name, email and password. The provided credentials should be stored in the User class in [parse.com](https://parse.com). Clicking the “Sign Up” button should submit the user’s information to [parse.com](https://parse.com) to verify the user’s credentials.
    - Check the user input, if any of the fields is left empty, or if the password and confirm password do not match, print a toast message indicating the corresponding error message and don’t submit the provided information to parse.com.
    - If an account with the same email already exists, display an error message indicating that the account account was not created and the user should select a different email.
    - If an account with the provided credentials does not already exist, then store the new account information and display a Toast indicating that the user has successfully login. Then start the Apps activity and finish the Signup activity.
    - Note that, the username and email should set to the same value in the user’s table.



(a) Apps Activity



(b) Menu Options

Figure 3, Wireframes of the Apps Activity

## Part 2 (40 Points): Apps Activity

The interface should be created to match the user interface (UI) presented in Figures 3(a) and 3(b). The Apps activity is responsible for the loading the list of apps retrieved from the api or the list of favorite apps. The implementation requirements include:

1. By default the apps activity should load the app list by retrieving it from the Apple RSS feed api.
2. **For the apps' list retrieved using the api:**
  - a. Use a thread pool (or AsyncTask) to communicate with the iTunes api and to parse the result. Do not use the main threads to download the api results. Care should be taken while parsing and should take into consideration the namespace.
  - b. The retrieved apps' information should be displayed in a ListView as shown in Figure 3.
3. **For the apps' list retrieved from the stored Favorites:**
  - a. Retrieve the list of favorite apps stored at parse.com. You **must use [parse.com](http://parse.com)** to implement the required favorite storage and retrieval functionality. Note that the favorites should store all the apps information required to display the app list as shown in Figure 3.
  - b. The retrieved apps' information should be displayed in a ListView as shown in Figure 3.
4. **For the apps' list retrieved from the apps shared with the current user:**
  - a. Retrieve the list of apps shared with the current user from parse.com. You **must use [parse.com](http://parse.com)** to implement the required shared apps storage and retrieval functionality. Note that the shared apps should store all the apps information required to display the app list as shown in Figure 3.
  - b. The retrieved apps' information should be displayed in a ListView as shown in Figure 3.
5. Create a custom layout and adapter in order to display the required list view items as shown in Figure 3.
  - a. For image loading/caching you must use the Picasso library.
  - b. For the thumbnail image use the Small Photo URL retrieved from the parsed XML for each app item.
  - c. For free apps, indicate the price as "0.0".
6. Clicking a list item should display the app details by starting the Preview Activity.
7. **Menu Items (View All)**
  - a. Clicking the menu item "View All" should retrieve the app information from the Apple API and display on the retrieved items in the apps list.
  - b. Clicking the menu item "View All" should only work when the apps list is currently displaying the Favorites or Shared list.
8. **Menu Items (View Favorites)**
  - a. Clicking the menu item "View Favorites" should query the Favorites stored in [parse.com](http://parse.com) and display on those selected items in the apps list.
  - b. Clicking the menu item "View All" should redisplay all apps in the list when the Favorites are currently showing.
9. **Menu Items (Clear Favorites)**

- a. Clicking the menu item “Clear Favorites” should clear all the stored app favorites. If the current list being displayed is the favorites list then it should be redisplayed to reflect clearing of the favorites.

#### 10.Menu Items (View Shared)

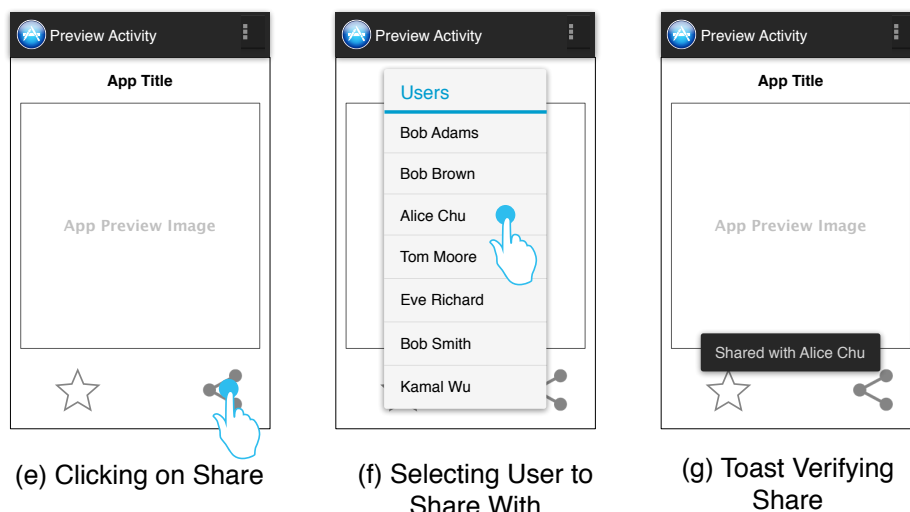
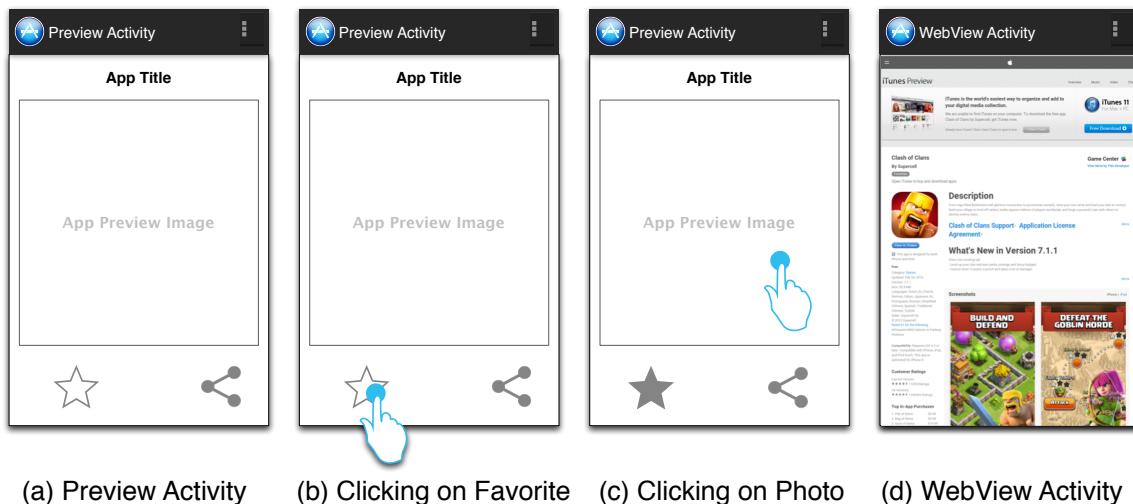
- a. Clicking the menu item “View Shared” should query the Shared Apps stored in [parse.com](http://parse.com) and display on the apps shared with the current logged in user in the apps list.
- b. Clicking the menu item “View All” should redisplay all apps in the list when the Favorites are currently showing.

#### 11.Menu Items (Clear Shared)

- a. Clicking the menu item “Clear Shared” should clear all the apps shared with the current user. If the current list being displayed is the favorites list then it should be redisplayed to reflect clearing of the shared apps.

#### 12.Menu Items (Logout)

- a. Clicking the menu item “Logout” should sign out the current user, finish the Apps activity, and start the Login activity.



**Figure 4, Preview and WebView Activities Wireframes**

### **Part 3 (50 Points): Preview Activity and WebView Activity**

This activity should display the app title, and the application preview image, and the Favorite status of the app. The Preview Activity should receive the app information from the Apps Activity. Figure 4(a) shows the PreviewActivity. The implementation requirements include:

1. Use the “App Large Photo” retrieved for each app to display the app preview image.
2. The activity should also indicate if the loaded app is either in the favorite list or not. If the app is not in the favorites this should be indicated by using the icon in Figure 4(b) indicated by the white star, and if it is in the favorites it should be indicated using the icon in Figure 4(c) indicated by the grey star.
3. Clicking on the star not in favorites icon should add the current app to the favorites and should change the icon to the in favorites icon. Similarly clicking the in favorites icon should remove the current app from the favorites and should change the icon to the not in favorites icon. You can get these icons from the android icon pack available at [http://developer.android.com/downloads/design/Android\\_Design\\_Icons\\_20120814.zip](http://developer.android.com/downloads/design/Android_Design_Icons_20120814.zip)
4. Clicking on the “App Preview Image” ImageView should start the WebView Activity and display the app url in the embedded WebView in the WebView activity as shown in Figure 4(d). The WebView should be setup to handle url redirects.
5. Clicking on the “Share” icon should query the list of users stored in the system from [parse.com](http://parse.com) and display the list of users (First and last names) sorted by last name as shown in Figure 4(f). Upon selecting a user from the list should store the current app in the shared apps for the selected user. When [parse.com](http://parse.com) successfully completes the storing of the shared app you should display a toast message indicating that the app has been successfully shared with the selected user as indicated in Figure 4(e).