Project 3 – Photo Notes

Overview

In this homework assignment, you will create a simple note-taking app that allows you to take a photo and associate a caption for each picture.

This homework will give familiarity with SQLite storage, file storage, and the camera.

Requirements

The app has the following requirements:

Activities

The app has 3 activities:

1. List activity

The list activity should display the list of saved notes. Each row can be represented with a thumbnail picture plus a simple of piece of text. One must use CursorAdaptor to link SQLite DB and the ListView (unless you are implementing the RecyclerView, in which you will need to use RecyclerView.Adaptor)

The ActionBar should have a button for adding a new photo, which launches the "Add Photo Activity" (described below). Additionally, you may have a floating action button that does the same.

Make sure long caption text is properly ellipsized.

2. View Photo Activity

This activity is simple: it displays the photo and the caption that was clicked on in the List Activity. The RETURN or a “Return” button will bring screen back to List activity.

3. Add Photo Activity

This activity should have at least 3 fields:

- EditText field for the caption

- Button for taking the photo. This should launch the camera intent.
- Save button. This should return to the list activity (use the finish()method).

Alternatively, when you click “Add” button in List activity, you may go directly to camera app to take a picture, followed by coming back to Add Photo activity. This likely gives better user experience.

.

SQLite

This app should use SQLite. You should have a table with at least two fields:

- Caption field - This is a user-specified text field.
- ImagePath field - This holds the absolute path to the photo.

It is likely you may have more fields than above listed. The photos and thumbnail files (if any) should be stored on external storage.

Menu Items

For List activity Option Menu should have an “Add” option (in ActionBar) and an “Uninstall” option (only shown in overflow menu) (similar to Homework 2). For other activities, Option Menu is optional.

Bonus Features

Implement List activity with RecyclerView. Note that you don’t have to use CardView for each row. You can use regular views such as LinearLayout.

Support swiping gesture which will delete the photo note. Support long-press-and-drag gesture to re-order the listing of photo notes.

Note that when you delete a photo note, you should delete associated picture and thumbnail files.

Notes

- It's not possible to view internal storage through adb/sqlite3 on a physical Android device (unless it's rooted, which I don't recommend doing). This technique only works on the emulator.
- To avoid Android 6.0 run-time permission, it is OK to set targetSDK to 22 in your app build.gradle file. Alternatively you need to implement run-time permission request.
- You may use the class provided in appendix to create thumbnail file from an image file. Modify to suit your need.
- It is OK to user Cursor.requery() method for this assignment even though it is deprecated. Of course you are also encouraged to use LoaderManager to do it more properly.

- If you use bitmap to show the picture, displaying large bitmaps may cause your app to run out of memory. You should load bitmaps from the filesystem with the following to avoid OutOfMemoryErrors:

```
BitmapFactory.Options options = new BitmapFactory.Options();
options.inSampleSize = 8;  // Experiment with different sizes
Bitmap b = BitmapFactory.decodeFile(filePath, options);
```

Scoring

Total points are 10. Bonus points are 2.

- Functionality (8 pt)
  - Meet all functional requirements (3 activities, option menu for List activity, ellipsizing large caption text, etc)
  - No crashes
- Layout & UI (1 pt)
  - Aesthetically pleasing with customized launch icon
  - Render well in both landscape mode and portrait mode; Or fix activities in certain orientation
  - Scale well from 4 inch to 7 inch screen
- Programming & coding (1 pt)
  - Neat & clean programming with clear naming
  - Correct usage of framework (e.g., use CursorAdapter instead of ArrayAdapter)

Bonus features, if implemented, are worth of 2pt.

Submission

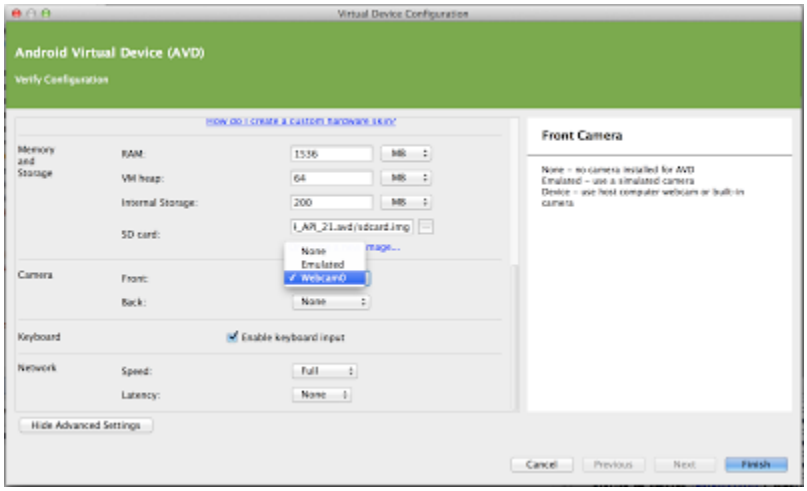
- Make sure your project builds on the latest Android Studio and runs on emulator before submission
- Include a README.txt file at top level of project directory. Teacher/grader will read this file first before building and testing your program.
  - List any special studio settings, project settings or demo/testing instructions if necessary.
  - Where are image files stored? Do you use thumbnails?
  - Does “Add” button to Camera picture capture directly or go to “Add Photo” activity first?
  - Do you implement 6.0 run-time permission request or set targetSDK to 22 in build.gradle file?
  - Do you support both viewing modes or have fixed viewing modes?
  - Do you implement the bonus features?
- Rename your project directory as *lastname.firstname.pa3* (e.g., sun.jun.pa3) Recall that project directory can be located by switching to “Project” view in Android Studio.
- Zip the whole project directory as a ZIP file with naming as *lastname.firstname.pa2.zip*
  - *Note: Wrong file/directory name will deduct up to 1 point*

- Submit the zip file via Camino/Canvas

Appendix – Camera on Emulator

You may use an emulator or a physical device to test your homework. This section describes specific details for getting the emulator to work with the camera. Skip this section if you're using a physical device.

In the Android Virtual Device settings, there are 3 options for the camera (shown in the screenshot to the right):



- None - If you choose this, your app will crash when you try to take a picture.
- Emulated - If you choose this, then a checkered pattern will show as the image for the camera. This is a completely valid option for the homework.
- Webcam - If you choose this, then your computer's webcam will be used.

On the Mac, the Webcam option should just work. On a Windows PC, the Webcam option may have issues if you don't have an appropriate driver. You can use an app like ManyCam to feed video to the emulator.

If you receive the error "Insert an SD card before using the camera." when running the camera, make sure you have a storage size specified for the SD Card. In the screenshot to the right, the SD card storage is 200 MiB.

Appendix – Generate thumbnail file

```
public class Thumbify {
    static public void generateThumbnail(String imgFile, String thumbFile) {
        try {
            Bitmap picture = BitmapFactory.decodeFile(imgFile);
            Bitmap resized = ThumbnailUtils.extractThumbnail(picture, 120, 120);
            FileOutputStream fos = new FileOutputStream(thumbFile);
            resized.compress(Bitmap.CompressFormat.JPEG, 90, fos);
            fos.flush();
            fos.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```