The Photo Diary

Group members: Boyang An, Zehua Shao, ZhenYu Han, Jerry Wang

Inspiration behind Photo Diary

Our app was designed after our group members brainstormed a way for users to have access to a visualization of where they took photos around the world.

The app itself is a diary that keeps track of where users have been around the world. The user will be able to see ‘clusters’ of markers on the world map where he or she has taken the most photos, and be able to view individual photos by clicking on the marker.

We hope to add more to this application so that people who are interested in traveling and photography, whether amateurs or professionals, will be able to organize and access their photos according to their place in the larger world. The possibilities of sharing photos through a world map with clickable markers is very enticing.

App Features

The Photo Diary App is an application for the organization of user photos based on time taken and location. This app will aggregate exif data from all photos stored in a user’s android phone, gather the geotags associated with each photo, and plot a marker on a google map for each photo taken.

The App organizes user information with a database that stores user profile information and user photo information. Each user must enter a user name, password and email address to be registered. Each photo from the user will have a name, a date taken, a latitude, longitude and user id for each entry.

The google map displayed to the user will have clickable markers that display a thumbnail for individual pictures or an image switcher view of successive photos if a cluster is clicked.
The cluster is merely an aggregate of separate markers that are relatively close together on the google map. If the map view is enlarged, the cluster of markers will separate to individual markers.

Application Implementation

Photo Diary Layout files:

- **AboutUs.xml**: two text views that display our group names, a button to return to previous activity.
- **ClusterImageDialogue.xml**: text switcher and image switcher to display thumbnail, three buttons to iterate through photo collection.
- **Infowindow.xml:** image view inside of linear layout. This will display the thumbnail image for a single marker
- **Login.xml:** two linear layouts consisting of text views and edit views that guide user to enter name and password. Two Buttons to cancel activity or to proceed to main activity.
- **Register.xml:** four linear layouts that ask users for name, password, reenter password and email. Two buttons to cancel activity or proceed to main activity.
- **Startup.xml:** two buttons in a linear layout. Buttons launch either login or register activity.
- **Main.xml**: fragment that displays google map. Linear layout, image button and list view on top of the fragment which updates when there are markers plotted on the google map.
- **Setpassword.xml**: three linear layouts each containing a text view and edit text view to guide user to enter old password, new password, and re-enter new password, all on top of relative layout.

Additional Resources:

Various images stored in drawable-hdpi used as background images

- **PhotoDiary:** constructor to create an photo object containing the name, geo location (latitude,longitude), create time and absolute path on the phone.
- **User:** constructor that create an user object that containing the some basic information about user, like username, userid, password and also email address.
- **AboutUsActivity:** his shows information about our team-members.
- Startup Activity: the first activity when we start the APP and two buttons was setup there and one is for login and the other is for register an new account.
- **LoginActivity:** user have to obtain an user account to login into the main activity through this activity.
- **RegisterActivity:** when people first use this APP it will requires people to create a new account. With in this activity it actually need people to type in a user name which should be no more than 6 characters for the secure reason and also set up a password.
- **MainActivity**: this activity will draw a google map and plot markers with the geo data accessed from the local phone’s photo album. The activity itself is a fragment activity because the separate markers must be displayed on top of the google map all at once.
    - satellite view of google map
    - google map key
- **SetPasswordActivity:** this activity allows user to reset the password when they already login.

External APIs Used:

- Google Play Services
- Google android map utilities (for clusters)

Permissions required:

- Google maps service: requires key + computer virtual fingerprint registered at google site
- Access network state
- Internet
- Write external storage : required to write to database
- Read gservice: required to location user on google map
- Access coarse location
- Access fine location

Problems encountered

The file system in java doesn’t work at all with emulators. The java method directory.listfiles() functions correctly on an android phone but for some reason will never return the correct files on an emulator, always returning null and crashing the application.

Exif interface returned a pair of gps coordinates that were not in the correct format for plotting on a google map. we had to convert the coordinates into two floats.

Database persistence was a bit tricky to implement. The database would sometimes erase user info, but we found that using the method onUpgrade() during the initial login phase was the culprit and erased the method.

Occasionally, clicking on the marker, whether an individual marker or a cluster, will crash the app. We think that if the absolute path of the photo on the phone’s local storage is null, then that could throw a fatal exception and halt the application.

Inflating a list of more than 2 photos in a listview on the layout slowed down our app significantly. At first we tried to multithread the application, but we sped up the application by switching to an image switcher to render 1 image thumbnail of each photo at a time. We no longer had a need for multithreading after that.

Draw line function was unable to be implemented in time for presentation. But we tried to draw a line between two markers based on order of photo taken. The line function was too time-consuming to debug.

We didn’t have enough time to implement a list of all the cities the user has visited. This feature would have been a great addition , and would have used Google Maps API’s ‘reverse geocoding' methods to look up a geotag's city location, then adding it to a list stored in the database. Alas, it involved too much debugging to implement.

Instruction

1. Replace the Google map key in AndroidManifest.xml with your own key.
2. Import the Google Maps Android API Utility Library which is already included in the ZIP file.
3. Add the library to this project.
4. The minimal version this project support is Android 4.0.3, API 15.
5. Default user account is set as:
    - username: aaaaaa
    - password: a