App Name: FlickrTron
Developers: Eric Xie (hx8rc) and Candace Park (cp6up)
Device Name: Pineco/Android

Pitch

Do you love the feed-based aspect of Instagram, but prefer the richer selection of photos on Flickr? Flickrtron allows you to interact with the Flickr network of photos without the distraction of cluttered menus, just as you would with Instagram. At its core, Flickrtron allows the user to search Flickr for photos related to a keyword or phrase. Queries return a set of 5 or less photos along with their title. Furthermore, the user can search for photos related to their GPS location. And finally, if the user wishes to upload immediately to their account, they can do so from the main screen as well.

Platform Justification

One of the main reasons we chose to develop on Android is the ability to develop on Windows. Since neither of us owned macs we felt that having to rely on lab computers would limit our productivity. Furthermore, from our experience with the bucket list app, we realized that the recent changes to Swift impeded our ability to look up relevant tutorials, whereas we are much more familiar with Java. Finally, we really enjoyed working with the layout .xml files when building our Quiz App, and liked its ease of use for connecting logic to the UI elements in Android.

Key Features

- Login Verification: The user cannot move on to the identity verification screen unless they indicate via a checkbox that they have authorized Flickrtron to access their Flickr information.
- Identity Confirmation: If the user advances to the identity confirmation screen without having authorized Flickrtron, the app detects this and provides a single button to move back to the login page.
- Search by Keywords: Simply enter a query into the search bar to find related photos
- Search by Location: Tap the location button to find photos related to your current location
- Quick Upload: Tap the floating action button to upload a quick snapshot to your account.
- Saved search: Your last search query is saved when you reload the app.

Testing Methodologies

Since our primary features involved heavy consumption of Flickr's REST APIs, we relied heavily on log messages to monitor the API response from the Flickr servers. Each API call has a corresponding Log message for the response which indicated whether or not the call was successful. When the call failed, it was usually easy to determine the correct response by pasting our API request into a browser. However, this was not applicable when forming the

POST request for uploading images. We simply relied on trial and error to test out that particular feature. For the UI logic, we used debugging mode to test out intents and rotations. The intent from each of the screens to the next is accompanied by authorization tokens. We tested the scenarios in which these tokens were present and absent to make sure the user cannot accidentally use the app without proper authorization.

Usage

1. Make sure location is set to “High Accuracy” prior to launching the app
2. Tap “Login” to open up the Flickr Login webpage (flickr.auth.getFrob)
3. Enter in the following credentials:
    a. Username: pinecograder
    b. Password: Pineco4720
4. Tap “Yes, I’ll Authorize” to grant Flickrtron write permission to your account and wait for page refresh
5. Switch back to the Flickrtron App via the recent apps menu.
6. Check the box that says “I’ve authorized Flickrtron”
7. Tap “Continue”
8. If you have successfully authorized Flickrtron, the button should say “Confirm”
9. Confirm that your identity is “Mobile Grader” and tap “Confirm” (flickr.auth.getFrob)
10. Main Screen
    a. To search by keywords (flickr.photos.geo.getLocation)
        i. Tap the search bar and enter your keywords
        ii. Press the enter button
    b. To search by location (flickr.photos.geo.getLocation)
        i. Tap the location icon next to the search bar
    c. To upload (https://up.flickr.com/services/upload/)
        i. Tap the pink floating action button
        ii. Take a photo
        iii. Confirm the photo
        iv. Take another photo, this is the one that will be uploaded (known bug)
        v. Confirm the photo

Lessons Learned

Building this app was a very powerful lesson in consuming RESTful APIs. Although our usage of the API endpoints was trivial, the process of forming each request, setting aside an Asynchronous Task, and retrieving data from the JSON response was very useful. Having to use the URLConnection class to manually make and parse each request taught us how valuable packages like Retrofit are, and we will definitely be using that in the future to increase code efficiency. One annoying aspect of Flickr API authorization is having to hash every parameter in alphabetical order. While we initially build each hash manually, we later switched to using a helper method that assembled the appropriate URL via a stack.

Perhaps the most difficult endpoint was the one for uploading, which was not just a POST request, but also a multipart/form-data submission. This was necessary for sending the photo as a binary file via an HTTP POST request, and it took hours of scouring related articles to realize that a plaintext request was not enough.

Handling rotation was not too difficult, but forced us to refresh our memory of the activity lifecycle. Whereas we had previously saved our state in onStop(), we learned that onPause() is more commonly called during rotation. By saving our shared preferences in onPause() instead of onStop(), we were able to preserve the app state not only during app reloads, but also during rotations.

Taking pictures was another difficult part of our app. Not only did we have to learn how to pass an intent to something other than an activity, we also had to make sure our file was saved in a way that we could retrieve it later for upload. Unlike the other API requests, it was impossible to discern whether the bytecode of the saved photo was a valid file or not.

Finally, we learned to have fun with the process. Finishing the earlier milestones were easily one day tasks, but finishing the final build along with polishing was definitely a weeklong journey. We anticipated this, and the only reason we were able to finish most of our original specifications was because we paced ourselves, set realistic goals, and didn’t try to do everything in one session.