GitHub Username: trogdan

Marker

Google Play Description

Marker will help you discover new and exciting life experiences by showing you exactly where other people have blogged about those experiences. While reading your favorite martial arts blog, easily find the location of that kung-fu weapon store. Or when visiting a new city, search a map for food blog entries describing delectable meals near you. Find out what other bloggers have written about that place, and then go off and explore.

App Description

Marker solves the problem of bloggers (food bloggers initially) not having a easy way to reference their blog entries with a geographic location. The hope is that Marker will allow someone (for example) to easily find life (food) experiences near a geographical location, and the blog entries that describe those experiences.

Marker is the second half of a two app blogging system. The first app (not implemented here) will be used to connect blog writers (and their blog entries) with locations stored in a server and

eventually loaded into a custom Google Maps Place list. The second app is for blog viewers (such as foodies) to find blog entries near a location or a location associated with a blog entry.. Eventually this location / blog entry list could be sorted by blog type (tags), blog authors, date, etc. But initially, this Proof Of Concept will be demonstrated using a single blog and blog author (specifically a food blog).

Intended User

This app is for blog readers, who want to learn more about *where* the interesting stores bloggers are writing about.

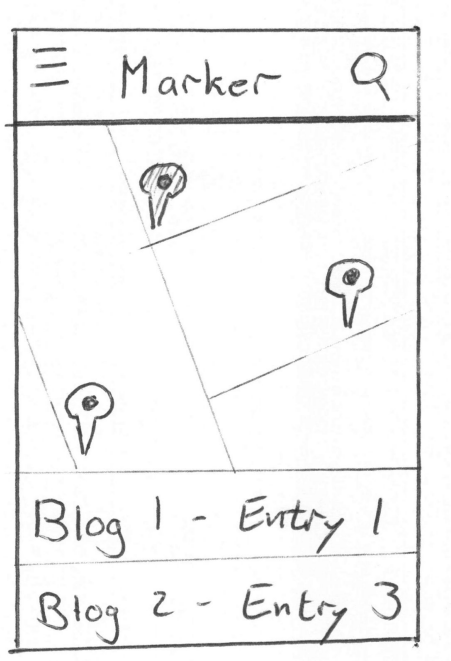
Features

List the main features of your app. For example:

- Add a blog
- View List of favorite Blogs
- View List of Blog Entries for a single Blog
- View Single Blog Entry in a WebView
- View some subset of blog entry locations in a custom map view.

User Interface Mocks

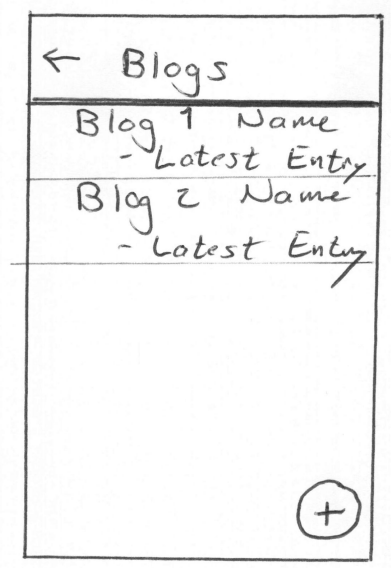
Screen 1



The primary goal of this app is for discovery, so we want to allow the user to find new and exciting blog locations. First screen is a map display which will take the user to her current location, and display nearby blog entries. There may be no blog entries nearby. The user will be given the option to search for a location. The map will change to that location, and display blog entries for the surrounding area. Clicking on a blog entry will take you to screen 4.

Lastly, there will be a hamburger-menu that will allow the user to navigate between map view, blog view, and settings.

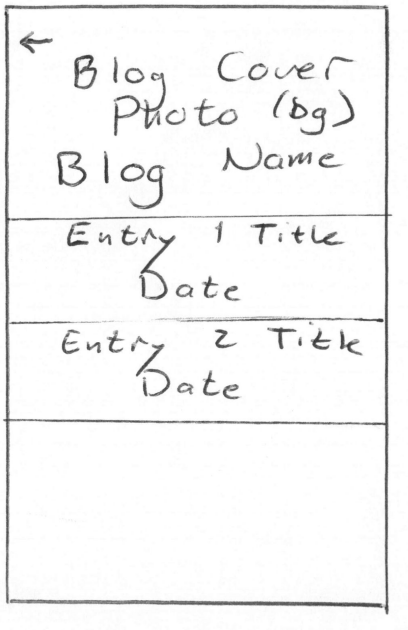
Screen 2



The second screen allows the user to add blogs. Currently this will be limited to blogs from blogger.com, and the user must enter the url of the blog to add. (future options to allow add by user id, blog id, and generic search). Blogs can be removed by swiping.

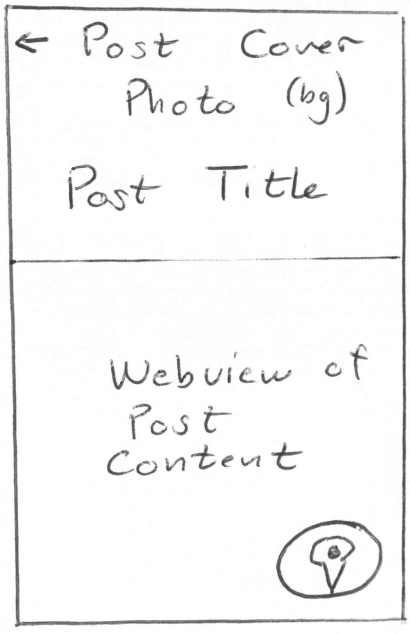
Clicking on a blog opens up screen 3. Although this shows a back-arrow, this will be a hamburger icon.

Screen 3



Third screen shows a list of all the blog entries for a selected blog. Clicking on an entry will take the user to that post in a new view. Entries with a location associated with them will show that a location is available for that post, and can be long-pressed to take the user directly to the map view of that post.

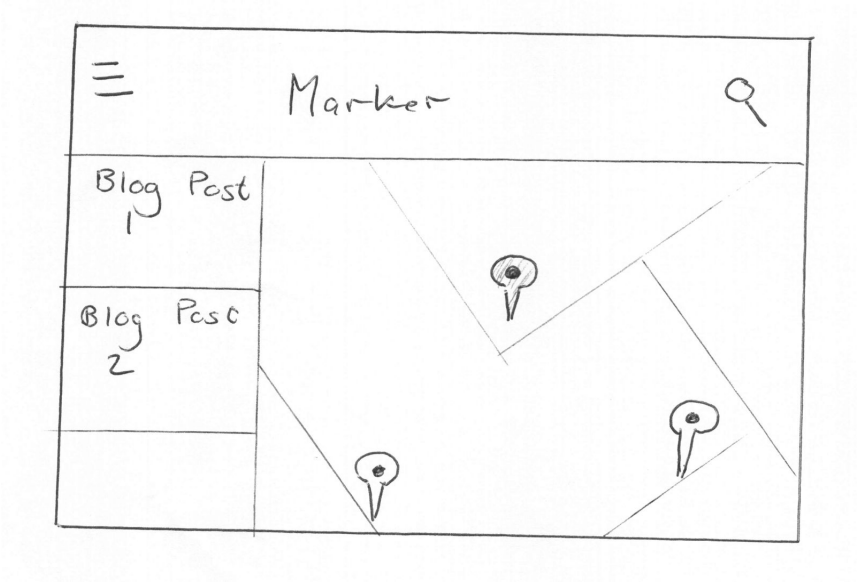
Screen 4



Fourth screen shows the content of the blog post at the bottom, and the cover photo of the post at the top, which scrolls away with parallax, to be just a toolbar with the post title. A back button

takes the user back to either the blog view, or to the map view. A FAB takes the user to the marker for the location, on Google Maps.

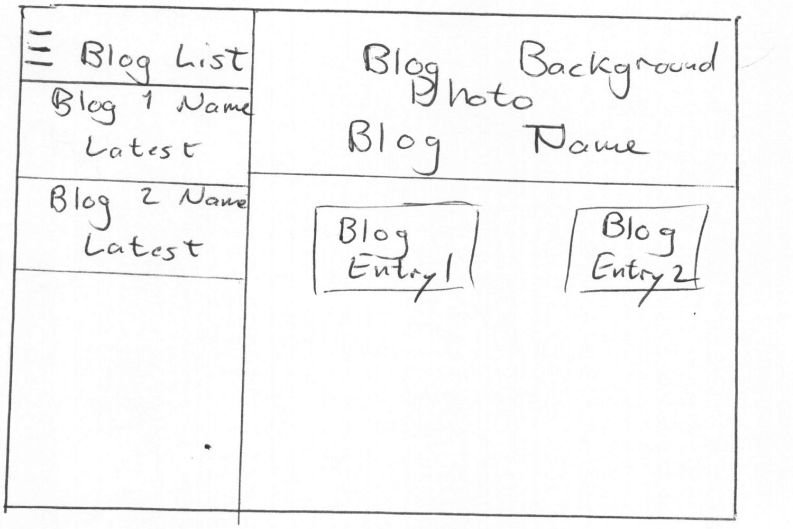
Tablet Screen 1



The front page is again the map view. Initially just a toolbar and the map view. Searching and clicking on a place will reveal the blog posts available for that location. Clicking the blog post will open the view of that post.

Clicking on the hamburger will bring up the option to switch from map view, blog view, and settings. This will also allow the addition of a blog via a button at the top.

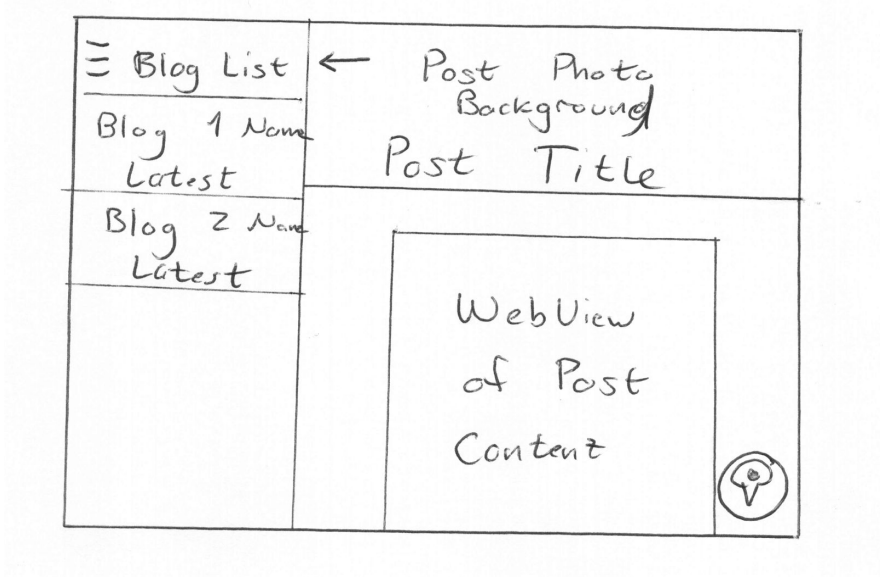
Tablet Screen 2



The Blog view will have the background photo of the blog at the top, with the blog name. The Blog entries will be re-flown as cards depending on dpi of the display.

Entries with a location associated with them will show that a location is available for that post, and can be long-pressed to take the user directly to the map view of that post.

Tablet Screen 3



The post view will have the background photo of the post at the top, with the post title. The post content will be in a web-view, and the fab for going to the location of the post will take you to Google Maps view of that place.

Key Considerations

How will your app handle data persistence?

The app will use a content provider, so one will be built to normalize the data coming from the custom Marker web service, Google Maps, and Blogger APIs. This will also reduce the need to constantly poll the servers.

The content provider will be backed by a sqlite database.

Describe any corner cases in the UX.

The post view can (in the phone case) go back to the map view, or to the blog view. Also, since the tablet keeps the blog list in the navigation bar, the back button on the blog view takes the user to the map view, NOT the blog list view.

Describe any libraries you'll be using and share your reasoning for including them.

Google Maps Api will be used for rendering maps views of Marker data.
Glide will be used for loading and caching of images.
https://github.com/bumptech/glide

GeoTools will be used for accessing the WFS layer (as JSON) of the Marker data service. This stores blogger saved locations associated with a blog entry and a google maps place.
http://docs.geotools.org/stable/userguide/library/data/wfs.html

OKHttp will be use for optimized access to the Marker data service if necessary for the standard rest interface.

Try the Marker data service yourself, for geospatial queries
http://marker.cloudapp.net:8080/geoserver/wfs?srsname=EPSG:4326&BBOX=-81,28,-80,29&typename=go:place_post&version=1.0.0&request=GetFeature&service=WFS&outputFormat=json

Or queries on the underlying databases, via standard rest

Getting ALL the places
http://marker.cloudapp.net:3000/place

Getting the post(s) matching a google place id.
http://marker.cloudapp.net:3000/place_post?goo_id=eq.ChIJXfpvGV0S3ogR6JlNWP7N5Pc

Next Steps: Required Tasks

This is the section where you can take the main features of your app (declared above) and decompose them into tangible technical tasks that you can complete incrementally until you have a finished app.

Task 1: Project Setup

- Create a new project in Android studio
- Setup git repo
- Add project libraries for OkHttp, Glide, and GeoTools.
- Create API keys in Google Developer console for Google Maps, Google Places and Blogger and add to the Android studio project.
- Create color pallette for app.
- Pull together resources such as icons.
- Create flavors for debug and release builds, support for APK signing.

Task 2: Implement UI for Each Activity and Fragment

- Build UI for MainActivity/Fragment (MapView using Google Maps fragment)
- Build UI for the BlogList activity/fragment.
- Build UI for the Blog fragment.
- Build UI for the Post view fragment
- Create layouts for phone and tablet

Task 3: Implement Marker Content Provider

- Create Marker data contract
- Create Marker Sqlite DB helper
- Create Marker Content provider
- Store list of blogs in SharedPreferences
- Create Marker SyncAdapter(s) to pull from Marker API.
- Create unit tests for DB helper, provider, and sync adapter.

Task 4: UI and Content Provider Integration

- Connect content provider with UI MainActivity/Fragment
- Connect content provider with BlogList fragment
- Connect content provider with Blog fragment
- Connect content provider with Post view.

Task 5: Accessibility and Localization

- Check and debug accessibility options
- Check support for localization including RTL layouts.
- Check for error conditions, and add error handling.