GitHub Username: neiljaywarner

DisciplesToday

Description

- Portal for disciplestoday.org
- News for International Church of Christ Ministries
- Church locator to find contact info for churches around the world.

Intended User

Members of the ICOC (International Church of Christ) and people interested in information about these churches and/or their ministries, as well as those interested in visiting.

Particularly nice would be attendees of the 2016 conference.

Features

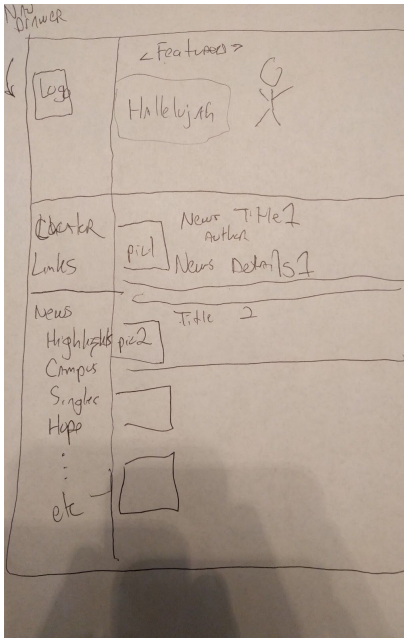
List the main features of your app. For example:

- Saves information
- Takes pictures
- Other features like that

User Interface Mocks

These can be created by hand (take a photo of your drawings and insert them in this flow), or using a program like Photoshop or Balsamiq.

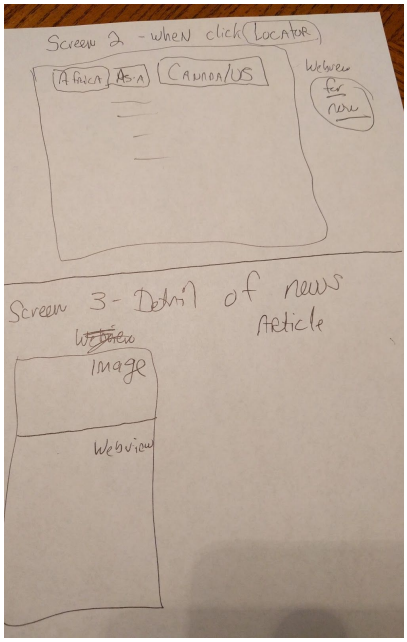
Screen 1



Opening screen

- Navdrawer to switch sections of news
- Navdrawer option to use church locator
- Navdrawer option to go to collection of links. (Or perhaps 3 links at the bottom)
- Sections in the navdrawer to make clear one is news, one is the locator, different types of function.

Screen 2



Screen 2 - Locator

- For now a webview to http://www.dtodayinfo.net/Dtoday (mobile site doesnt' include the map, so it is almost usable)

Screen 3 - Detail Screen

- Full bleed image on top, webview on bottom for html loaded from the feed
- Clicking on it goes to the site in Chrome in case the article is richer or fuller, etc.

Key Considerations

How will your app handle data persistence?

- Maybe cupboard because it handles content providers nicely and is pretty simple
- Content provider as required - for the news feeds which will be savedd to the database.

Describe any corner cases in the UX.

- It is unlikely the user will click on the article often, so it's an edge case, they'll be in Chrome for now and the back button will be used to return to the app.

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

- Picasso - image caching.
- Retrofit - for loading the data from the rest endpoint and parsing the JSON
- Cupboard - simple data persistence.

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

- Android Studio 2.2 to learn and play with ConstraintLayout
- Build.gradle file to include cupboard, retrofit, picasso
- Material Design Support Library NavigationView to provide template for navigation drawer with best practices and

Task 2: Prototype

- Octocat example from github for retrofit v2.
- Listview for articles - image and title only
- Retrofit to get the data - title,summary, text only
- Populate listview
- Pass arguments to Detail screen for title,image, text
- Implement detail screen with image, title, webview

Task 3: Flesh out prototype listview/detailview to match design

- Change model to match news articles
- Change model to match feed
- Modify list_item xml file as needed
- Optional :Shared transitions based

Task 4: Featured article

- Get featured article for now from first in “Highlighted” feed.
- Implement in top half of main screen (wire up)

Task 5: Locator

- Implement webview for now.
- This functionality is #1 for users.
- Time permitting this will be local via REST request
- API is not built yet, it is being built by back-end team now

Task 6: Other feeds (singles, campus, etc)

- Add titles to navdrawer
- Use same UI and architecture with different parameters for url to REST request.

Task 7: Links section of navdrawer

- Add links (icochotews.com, mediastore) to navdrawer
- Probably display in webview so still seems you’re in the app
- This could represent a working prototype to show the client

Task 8: Analytics

- Add firebase analytics (register app with firebase/google console/admin site)
- Add suggested items for all apps for applicable items
    - https://support.google.com/firebase/answer/6317498?hl=en&ref_topic=6317484
- Test briefly in console

Task 9: Firebase invites

- Add firebase invites
- Let people share an article with deep link invite

Task 10: Contentprovider, Syncadapter, cupboard

- Remove asynctask
- Content only needs updated once per day - do this with syncadadpter
- ContentProvider + cupboard + loaders to get data from sqllite db to screen

Task 11: Tablet Layout

- Do tablet layout with Listview and detail screen combined

Task 12: Misc

- Dpad
- RTL
- Make sure I meet rubric

Submission Instructions

1. After you've completed all the sections, download this document as a PDF [File → Download as PDF]
2. Create a new GitHub repo for the capstone. Name it "**Capstone Project**"
3. Add this document to your repo. Make sure it's named "**Capstone_Stage1.pdf**"