How to Use this Template

1. Make a copy [File → Make a copy...]
2. Rename this file: "**Capstone_Stage1**"
3. Replace the text in green

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**"

GitHub Username: cameocoder

Ottawa Waste Schedule

Description

View the Garbage and recycling schedule Ottawa, Ontario, Canada. Receive reminder notification before pickup day.

Intended User

Residents of Ottawa, Ontario, Canada. Actual API applies to many cities and towns in Canada, but it is difficult to enumerate who can and cannot use the app. It may be easier to create different flavors for different jurisdictions.

Features

List the main features of your app. For example:

- Onboarding to help user select location
- Show garbage and recycling schedule based on location
- Display notification reminder the day before collection.

User Interface Mocks

Screen 1

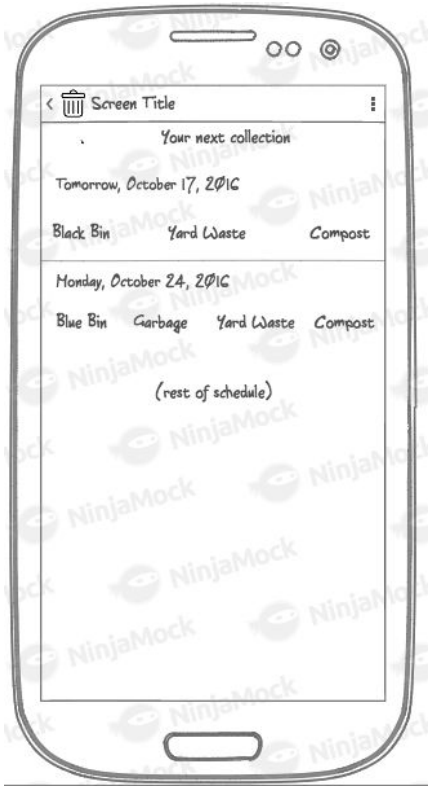
OnboardingActivity used to get the user's location. They could open the app from a location other than home so we need to handle current location or search for location.

Screen 2

Choose location from map

Screen 3

After we have a location we can retrieve the garbage collection schedule.



Screen 4

Widget at 1x1 size may just be color coded. Supports at most 3 entries.

Key Considerations

How will your app handle data persistence?

Some data like location (lat/long) and determined “zone” can be stored in preferences.

Garbage collection schedule will be stored in a database.

Describe any corner cases in the UX.

Handling errors around determining the user location and “zone”. If location cannot be determined, or is inaccurate, then will need to display a map and have the user select location.

App has to be bilingual in English and French. Api returns some strings in English and French and will need to display the proper string to the user.

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

Retrofit or similar library for handling http api requests. Preferably to ease the pain of parsing json.

Butterknife for view injection to lessen some view handling boilerplate code.

Describe how you will implement Google Play Services.

Google Play Services will be used to determine user’s location

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

Start with Android studio new project from template Basic Activity with Fragment.

Add common libraries to gradle build file (prune out later if needed)

```
compile "com.android.support:support-annotations:$supportVersion"
compile "com.android.support:appcompat-v7:$supportVersion"
compile "com.android.support:support-v4:$supportVersion"
compile "com.android.support:design:$supportVersion"
compile "com.android.support:recyclerview-v7:$supportVersion"
compile "com.google.android.gms:play-services-gms:$playVersion"
compile "com.google.android.gms:play-services-location:$playVersion"
```

Add dexcount gradle plugin

https://github.com/KeepSafe/dexcount-gradle-plugin

Add LeakCanary

https://github.com/square/leakcanary

Task 2: Implement UI for each planned Activity and Fragment

Stub in UI for main elements

- Build UI for OnboardingActivity
- Build UI for MainActivity
- Build UI for Schedule list screen
- Add Settings activity

Task 3: Handle getting current location

Handle getting location from current location. This gives me location data to start with. Getting location from Map can come later.

- Handle click on “Use current location” button
- Handle permission checks
- Handle permission deny
- Handle location error

Task 4: Add back end tasks

Add sync service and sync adapter

- Add basic authentication
- Two initial types of sync
    - Get Zone from location
    - Get Schedule for Zone

Retrieve Zone from the given location

- Send location to api to retrieve Zone
- Handle result and parse data
- Handle Zone not found or other errors with data

Task 5: Add database backend

Retrieve Schedule from the given found Zone

- Send Zone to api to retrieve Schedule

Add tables to store Zone and Schedule data

Task 6: Handle location selection from map

Now that we can retrieve schedule data, provide alternate way to get location

- Add Ui for retrieving location from map

Task 7: Add setting for changing location

Add a settings link that allows a user to change their location.

- Bring user back to the onboarding activity

Task 8: Service for handling notifications

Add service for handling notifications

- Notification time should be configurable
    - Day before pickup
    - Morning of pickup
- Add settings for choosing notification time
- Raise notification with info for current pickup

Task 9: Widget

Add widget displaying next garbage pickup schedule item.

- Support 1x1 size so it is unobtrusive and can be interpreted at a glance.
- Maximum 3 items to display.

Task 10: Polish

Check UI against Material guidelines

- Adjust layouts as necessary
- Add transitions between screens where necessary
- Add coordinator behavior such as sliding action bar on schedule list if necessary
- Choose color scheme

Add as many tasks as you need to complete your app.

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**"