How to Use this Template

1. Create a new document, and copy and paste the text from this template into your new document [Select All → Copy → Paste into new document]
2. Name your document file: "**Capstone_Stage1**"
3. Replace the text in green

GitHub Username: 2ma

MyWay

Description

MyWay is a gps track recording application, that lets you record any outside activities.
You can record biking, hiking, walking and various other activities.
It shows your speed, elevation, distance, time. It lets you revisit previous recordings too.

Intended User

MyWay is for anybody who loves moving. For hikers, bikers, any kind of travel, where you would like to keep track of where you were.

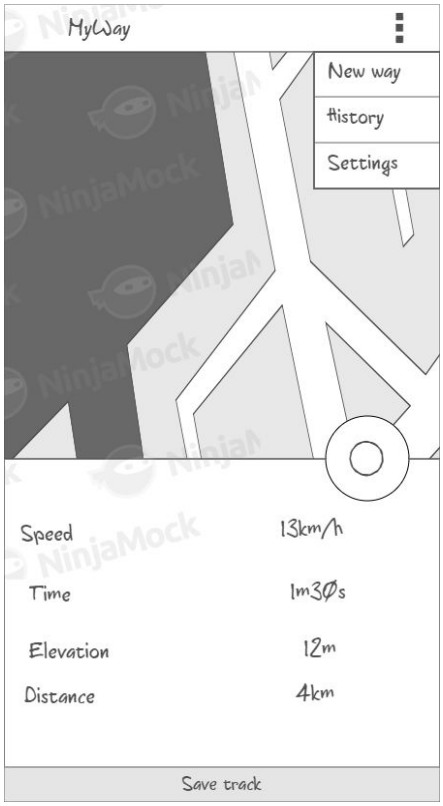
Features

- Save gps track information(distance, time, elevation, speed)
- Displays track on map
- Display previously recorded tracks on map

User Interface Mocks

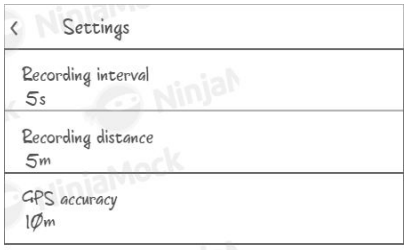
These can be created by hand (take a photo of your drawings and insert them in this flow), or using a program like Google Drawings, www.ninjamock.com, Paper by 53, Photoshop or Balsamiq.

Screen 1



Main screen, shows a map of the current track or shows the previously recorded track.
It has a menu with settings, new way and history options. It has a bottomsheet with recording start/pause, or play/pause fab depending on if it's new way or a recorded one. The bottomsheet displays data regarding the route, it can be minimized if it's too distracting. When the recording is paused, a snackbar is shown with the options to save the track.

Screen 2

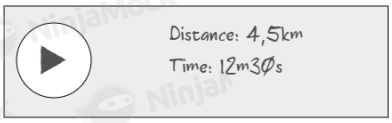


Settings screen displaying options for recording interval, distance, gps accuracy and track name and other options.

Screen 3

History screen, shows previously recorded tracks. Tracks can be dismissed with swipe navigation, when a track is dismissed, a snackbar is shown with undo function. Selecting the recorded track displays it in screen 1.

Screen 4



Widget for the application, displaying a record/pause button and additional information on the current recording.

Key Considerations

How will your app handle data persistence?

Gps data will be recorded locally in a Content Provider.

Describe any edge or corner cases in the UX.

Map displays pin for current position, also shows path for previously recorded track. Information about the track will be displayed in a bottomsheet solution, that can be minimized to show less information, and show the map larger. The same screen will show the previous recording and the active recording as well, the difference will be in the record/play button. This UX choice is not final, further testing needs to be if it's a viable solution.

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

- Location services library for tracking location
- Maps services library for displaying the position and recorded track.
- Admob library for displaying ads.
- ButterKnife for binding views easily

Describe how you will implement Google Play Services or other external services.

- Will use Location services for tracking the devices current position and recording it in a content provider.
- Will use Maps services for displaying recorded path and current position for the user, also for displaying previously recorded paths.
- Admob library will be used for displaying banner ads while on the recording screen, interstitial ads when selecting a previous recording.
- ButterKnife will be used in the Activities and Fragments so I don't have to call findViewByid on each view, also binding click listeners is easier.

Next Steps: Required Tasks

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

Task 1: Project Setup

- Implement project libraries.
- Setup gradle for easy version tracking. (List libraries version that use the same version under root project for easy update)
- Create gradle tasks for release version deployment

Task 2: Implement UI for Each Activity and Fragment

- Build UI for MapActivity
- Build UI for HistoryActivity
- Build UI for SettingsActivity

Task 3: Create content provider

- Create a content provider for storing the recorded tracks.
- Write tests for content provider to make sure it works

Task 4: Create service for location recording

Create a service for using Location services and recording data to the content provider. Service should be capable to run in the background also as a bound service to help connect with the MapActivity

- Implement notifications, to let the user know the app is running in the background, also to control starting/pausing recording.

Task 5: Displaying the recorded data

Displaying the recorded data in the MapActivity.

- Display previously recorded point as a path on the map
- Display current position as a custom marker on the map
- Display speed, elevation, distance values from the data
- Setup record/pause button for recording track
- Setup snackbar for displaying Save track option after track is paused

- Setup dialog for save option, to change track name from default
- Setup menu options for MapActivity(New way, History, Settings)
- ButterKnife will be used for binding the views and click listeners

Task 6: Displaying previously recorded tracks in HistoryActivtiy

- Displaying previously recorded track in HistoryActivity
- Setup swipe navigation in HistoryActivity for deleting unwanted recordings
- Setup snackbar for displaying Undo option for accidental delete
- ButterKnife will be used for binding the views and click listeners

Data for the HistoryActivity will be loaded with LoaderManager and CursorLoader, so it will retain information on screen rotation.
HistoryActivity will include a search function,that will run in the background using AsyncTask to query existing tracks based on name or date.

Task 7: Create PreferenceFragment for SettingsActivity

SettingsActivity will use a PreferenceFragment for changing settings regarding recording accuracy, default name, other options.

Task 8: Implement AppWidget

Create app widget for displaying info about current recording, and showing start/pause recording.

Task 9: Implement ads

- Implement banner ad for MapActivity
- Implement interstitial ad for HistoryActivity

Task 10: Espresso tests

Write espresso test for UI testing between activities.

Submission Instructions

- After you've completed all the sections, download this document as a PDF [File → Download as PDF]

    - Make sure the PDF is named “**Capstone_Stage1.pdf**”
- Submit the PDF as a zip or in a GitHub project repo using the project submission portal

If using GitHub:

- Create a new GitHub repo for the capstone. Name it “**Capstone Project**”
- Add this document to your repo. Make sure it’s named “**Capstone_Stage1.pdf**”