GitHub Username: yoonhok524

Life Logger

Description

We have 24 hours in a day.
How do you spend the time?
You may spend some time in your desk to work or study.
But did you really do your best?
Maybe you can say "yes", but my case it's not easy.
I spend some time to check the sns, web surfing, or playing game.
And before sleep, I regret.
What did I do today!
So I want to check how do I spend the time.
And this app will help to analyze my time spending pattern.

Intended User

Everyone, who want to know how do I spend my time.

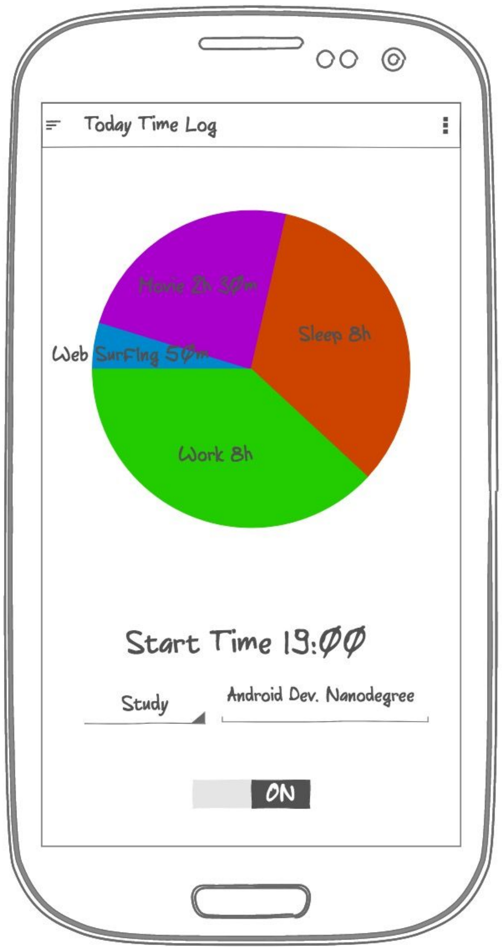
Features

- Saves activities with spend time
- Analyze spending time pattern
- Home Widget to start/stop time log

User Interface Mocks

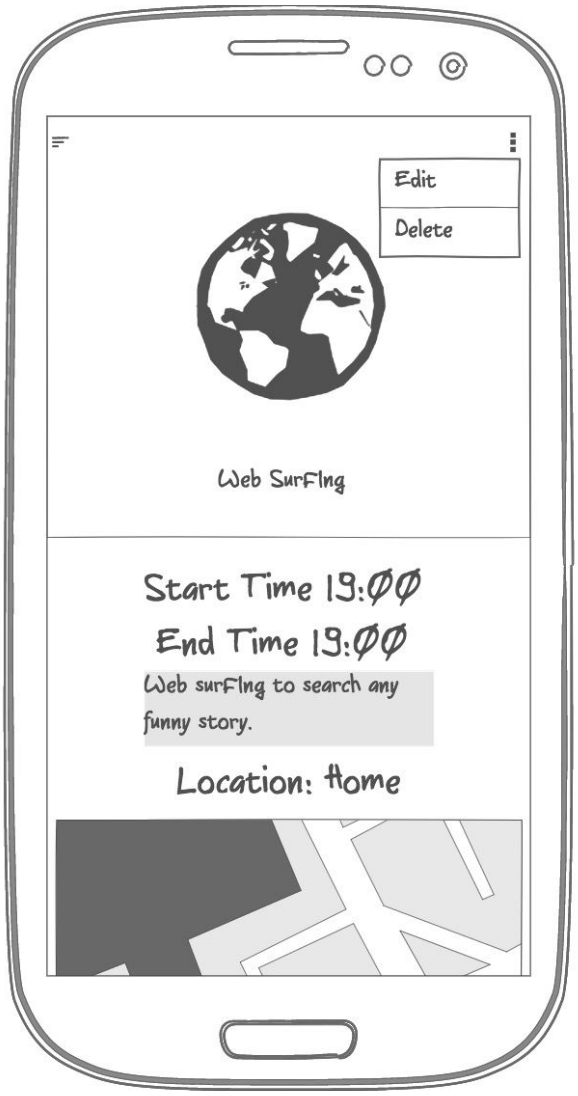
These can be created by hand (take a photo of your drawings and insert them in this flow), or using a program like Google Drawings, www.ninjamock.com, Paper by 53, Photoshop or Balsamiq.

Screen 1



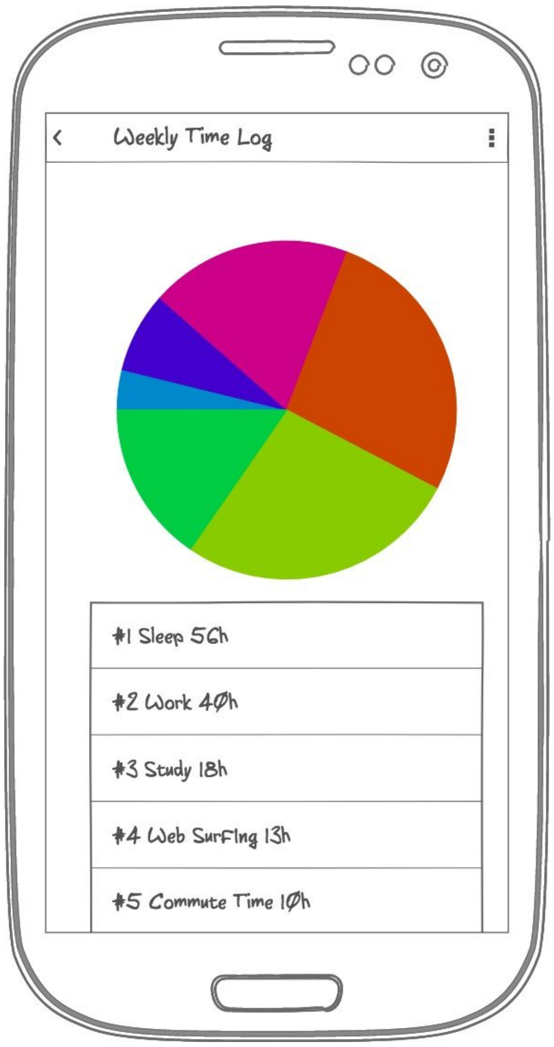
Main View to show today’s time spending states.
And user can start/stop current activity using the on/off switch.
When user start a new activity, user can select a category and input the title.
(Activity doesn’t mean the android component.)

Screen 2



When user click one of the activity, user can see this screen.
It displays Category, title, start/end time and location.
Through the action button, user can edit/delete this activity contents.

Screen 3



User can see the statistic about user's time spending pattern (weekly, monthly)
There are circle chart and list view.
Chart shows all of time spending ratio.
And list view shows the ranking of the time spending activity.
User can notice what activity is needed time.

Add as many screens as you need to portray your app's UI flow.

Key Considerations

How will your app handle data persistence?

Realm and Content Provider are used to save events.

Describe any edge or corner cases in the UX.

This app will provide home widget, to start/stop the time logging.

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

- Realm will be used to store the event data. It doesn't need SQLite code, so I can give my focus more to my business logic.
- ButterKnife will be used to reduce findById code. So I can make my code clean and increase performance.
- MPAndroidChart will be used to show the various chart. It already support various chart api, so I don't need to invent same thing.

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

Geo api will be used to include geo information into the event.

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

- Create a github repository
- Create simple android project
- Configure libraries
- Define data model classes

Task 2: Define UI for Each Activity and Fragment

- Define activity/fragment classes.
- Create/Make xml layout.
- Do same thing for large screen.

Task 3: Define MVP Architecture

- Create source code files for MVP skeleton code.
  - Contract interface
  - Presenter class

Task 4: Create Data Source classes

- Create simple data source classes
  - This class have a responsibility about CRUD operate of event objects stored in Realm.

Task 5: Implements Content Provider

- Create Content Provider class

Task 6: Write Test Code

- Write junit test code
- Write espresso code

Task 7: Implement View/Presenter classes

- Implement View and Presenter classes.
- Implement same thing for large screen.