Contents

1 Abstract

2 Introduction

2.1 The Idea

The Digital Life Tracking App is designed to enable users to track their daily tasks and the time spent on them. To make it more convenient to trigger an activity, the start and end of an activity may be triggered in different ways. Possible activation mechanisms could be:

- Two claps
- moving the smartphone in a certain way (gestures)
- activation based on GPS position
- pressing a button

Users should be able to define their own activities and select an activation function from a list and assign it to one of their activities. The user should then be able to display his history in form of charts. A separate device (particle photon board) could also be used to trigger a specific activity.

2.1.1 The Deep Dive

For the prototype in the context of the assigment we decided to focus mainly on the detection of double clapping as an activity trigger.

3 First Prototype (Board)

TONY

4 Second Prototype (Android)

4.1 Research for clap detection in android

Java is known to have many open source libraries. For the processing of audio signals, the Tarsos library was a good choice. Among other things, this offers a ready-made percussion detection which enables to detect sudden peaks in a frequency. However, Tarsos also offers more basic functionalities, such as transforming an audio signal into the frequency domain using Fast Fourier Transformation.

In order to implement and test the necessary functionality for the state machine and the UI, the percussions detection from Tarsos was initially used to detect a loud noise.

4.2 Implementation for data analysis

Some code was soley written for getting data out of the app and was removed later for the final app.

4.2.1 Google Drive

When working on the first prototype with the Particle Photon Board, the roundtrip time required to get the data from the board to a PC, where we can analyze it, was particularly noticeable. For this reason, an automatic upload of the CSV data to Google Drive has been implemented. This made it possible to quickly transfer all recorded data to a central location and analyze it from there on a PC.

4.2.2 CSV Button

Another functionality that was only necessary for the initial phase is the recording of the audio signal for a certain period of time and afterwards writing the transformed (FFT) data into a CSV file.

An extra button was added to the page, which adds a new AudioProcessor to the AudioDispatcher, which then writes the data to the mobile phone. The existing CSVWriter class was used and adapted for this.

Unfortunately it turned out that you can't call the method AudioDispatcherFactory.fromDefaultMicrop twice to run two dispatchers with different buffer sizes at the same time. For the recording of test data a longer period of time is useful over which the FFT is then applied. A buffer size of 3*sampling size was needed, to allow 3 seconds of audio recording per test. For real application, however, shorter periods of time are needed to detect a double clap and to keep the delay of the detection small.

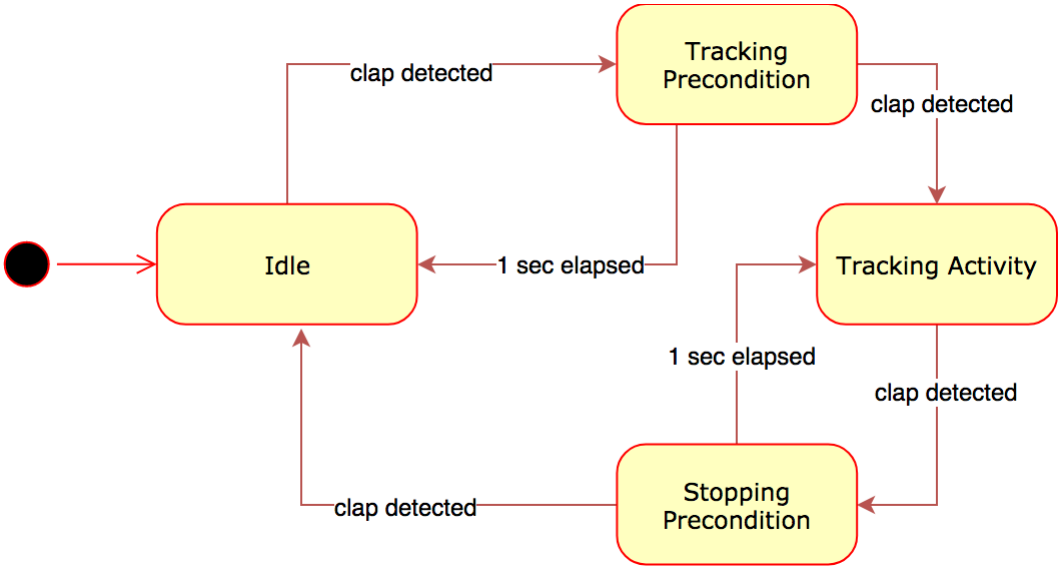
4.3 Implementation

4.3.1 Clap Detector

- Tarsos library (for FFT)

4.3.2 State-machine

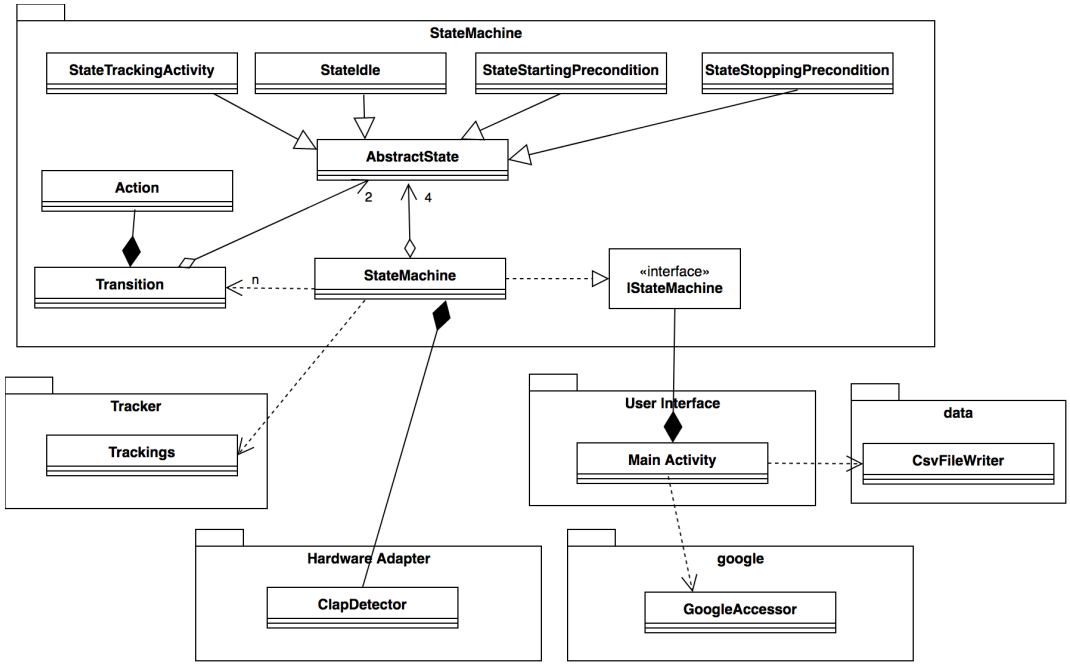
A state machine was implemented to represent the different states of the app. The following diagram shows the implemented state machine.



There are a total of four states in which the app can be in.

- Idle: The initial state when starting the app and the state after an activity tracking has been completed.
- StartPrecondition: If the app is in the idle state and a clap is detected, the state machine switches to this state. When switching to this state, a timer is started which defines the time window in which the second clap must occur in order to switch the state to TrackingActivity. If the timer expires before another clap is detected, the state machine switches back to the idle state.
- TrackingActivity: After a second clap is detected while the timer of the start precondition has not yet elapsed, the state machine changes to this state and starts capturing the time by saving a time stamp.
- StoppingPrecondition: If the state machine is in the TrackingActivity state and a clap occurs, then the state machine switches to this state, which behaves in the same way as the StartingPrecondition, except that on a successful second clap, it changes to the idle state and the tracking of the current activity is ended.

4.3.3 Architecture



1. StateMachine

   The state machine presented in the previous chapter takes care of the logic of the app and switches back and forth between states when registering the corresponding event. The Statemachine class is the center of the package of the same name and performs the tasks described in the previous chapter. It holds instances of all four states and a configuration of all possible transitions between the individual states. Each transition in the configuration is also linked to an action that determines the event that triggers the transition. Each transaction also defines the previous state and the next state. When a transition is triggered, other operations are also executed that either adapt UI elements to the new state or execute other functions that are important for parts of the business logic.

2. Hardware Adapter The hardware adapter package includes the ClapDetector class, which uses the SmartPhone's microphone to listen for ambient sounds and tries to detect a clap from the recorded frequencies. When a clap is detected, a handler is triggered in the state machine. This triggers a transition between two states in the state machine. This is illustrated in figure TODO: REF EINBAUEN !!!!.

3. Tracker

   One of the operations triggered by the investments in the state machine is to start or stop the tracker, which stores the activities of the user together with the measured times in a list.

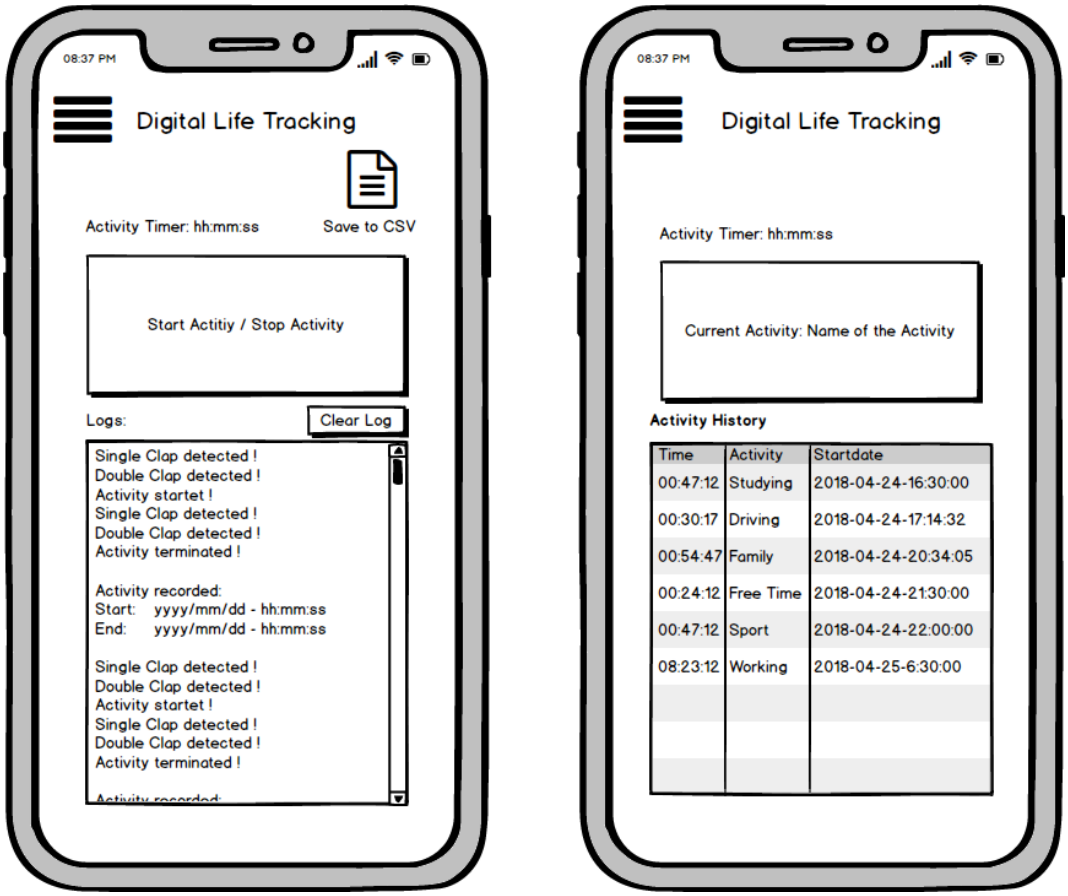
4. User Interface

   The User Interface package contains the main activity of the app, which initializes the other functions of the program including the state machine. The goal was that the Main Activity class should contain as little business logic as possible. The business logic has been implemented almost completely in the state

machine. If possible, the class should only provide and control the UI elements of the app. It also controls the permissions request to the user, which the user must give in order for the apps to work properly.

5. Google This package contains the GoogleAccessor class, which was initially designed to upload the test data stored on the smartphone via the CSVWriter class to the Google Cloud. Once enough data has been collected, this functionality has been disabled for the release of the app. The reason for this was that access to the cloud required signed apk, which had nothing to do with the pure functionality of the app. Commenting out the code of this feature should avoid unnecessary problems when starting the app.
6. Data The data package contains the CsvFileWriter class, which stores the frequencies recorded by the ClapDetector in CSV files on the memory of the smartphone. This functionality was used during development to collect test data for later analysis.

4.3.4 UI Design



5 Conclusion

5.1 Current State

5.1.1 Evalutation

- How reliable can our implementation detect clap.
- Benchmark
  - How many time false positives where detected (20x husten, 20x schnipsen, 20 klatschen)
  - Show statistics by trying it out (maybe in different environments (loud, silent rooms, outdoors)

Refer to to evalution part above. State how difficult this was and the time needed to try out more advanced solutions (AI) was not enough.

5.2 Project Outlook

Maybe add more debug functionallity inside the App, be able to not only tweak parameters inside the code, but also with UI Controls inside the app.

Whistling detection instead of clapping.

6 Referernces

http://www.klangfuzzis.de/showthread.php?679817-Was-hat-in-etwa-wie-viel-hz

7 Aufgaben für uns noch:

7.0.1 TODO Tarsos Code rausnkopieren

7.0.2 TODO State-machine implement

7.0.3 TODO Fork vom android und unsere repo reinkopieren