GotHiite

GitHub Username: RBische

GotHiite

Description

Ever dreamt about an app that can follow you during your interval training ? GotHiite app is here for you.

GotHiite uses GPS to track your training, give you a real feedback during session.
Create your own exercise, choose your vVO2max, start to run and boost your performances.
Sync your activities to retrieve them later and see the evolution

Features of the app:
Track your activities in real time
Calculates distances that needs to be done during your running time using your vVO2max
Give you a vocal feedback when you are late or ahead
Automatic synchronization of sessions
Sync your work with Google Fit
Share your workout on social networks

Intended User

This app is created for runners. It will be used when they do HIIT training (https://en.wikipedia.org/wiki/High-intensity_interval_training)

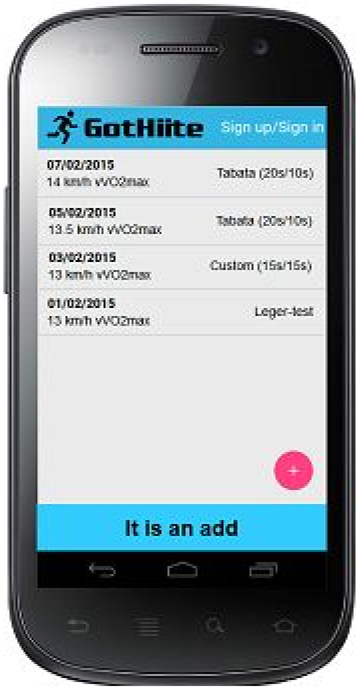
Features

- Track your activities in real time
- Calculates distances that needs to be done during your running time using your vVO2max
- Give you a vocal feedback when you are late or ahead
- Automatic synchronization of sessions
- Sync your workout with Google Fit
- Share your workout on social networks

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 (Main)



07/02/2015
14 km/h vVO2max
Tabata (20s/10s)

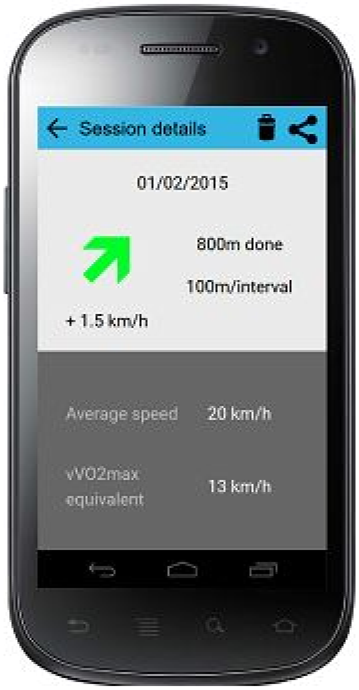
05/02/2015
13.5 km/h vVO2max
Tabata (20s/10s)

03/02/2015
13 km/h vVO2max
Custom (15s/15s)

01/02/2015
13 km/h vVO2max
Leger-test

First screen of the app, the user has a TextView asking him to do his first session or a list of his previous sessions.
The user can also connect or sign in (screen 7) to save his data so that he can retrieve running sessions and session types on other devices (another android or a website or…)

Screen 2 (Session Details)



01/02/2015

800m done

100m/interval

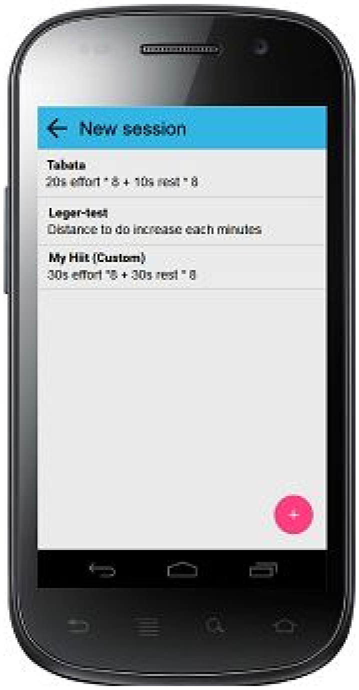
+ 1.5 km/h

Average speed 20 km/h

vVO2max equivalent 13 km/h

When the user clicks on a previous session in the main screen, details are displayed, like the date of the previous session, the number of meters done, the delta between his average vVO2max and this session vVO2max, etc..

Screen 3 (New Session)



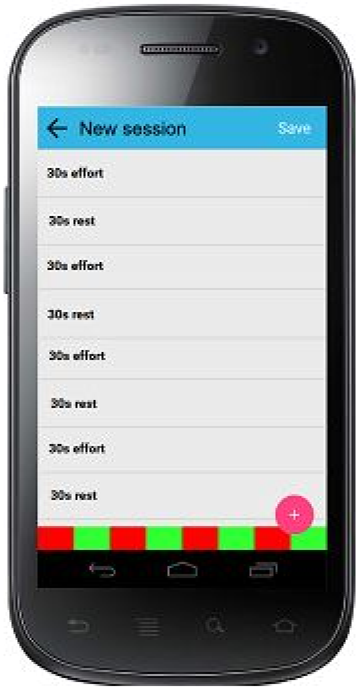
Tabata
20s effort * 8 + 10s rest * 8

Leger-test
Distance to do increase each minutes

My Hiit (Custom)
30s effort *8 + 30s rest * 8

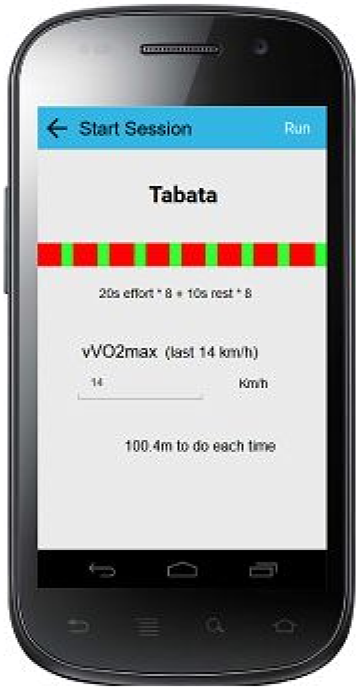
If the user start the app for the first time, 2 preprogrammed session type will be shown : Tabata and Leger-test. The user can click the floating action button to add a new session type or use a session type ever created.

Screen 4 (New Session Type)



When the user creates a new session type, he must add each interval one by one.The bar at the bottom resumes the splitting of the session. When the editing is finished, he clicks on save button.

Screen 5 (Start Session)



Tabata

20s effort * 8 + 10s rest * 8

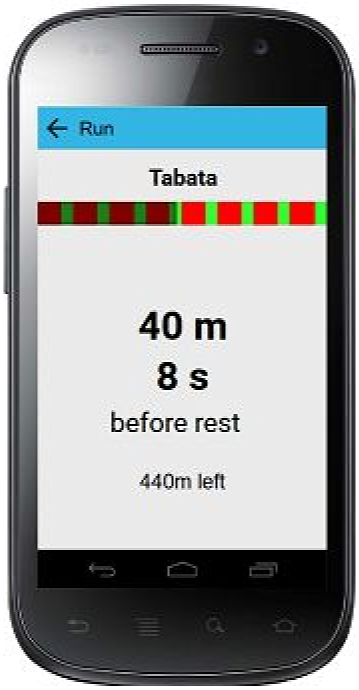
vVO2max (last 14 km/h)

14 Km/h

100.4m to do each time

Once the user choosed a session type, some more informations are shown, like the number of meters that the user will do for each effort. The user can change the vVO2max. That change modifies the number of meters that the user have to do.
The user then clicks on “Run” if he is ready.

Screen 6 (Running)



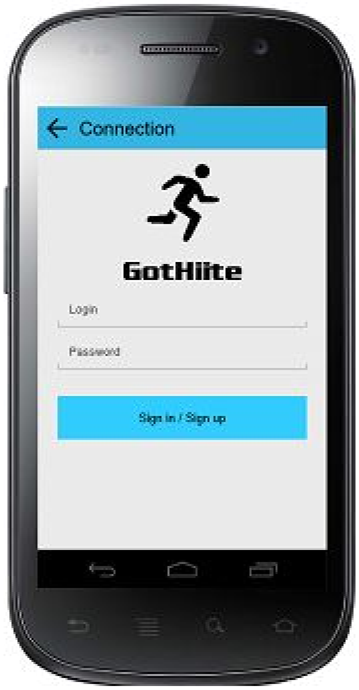
Tabata

40 m
8 s
before rest

440m left

This screen is shown when the user is running. The bar resume the session and the user's position in the session. During that time, the user receive a vocal feedback if he is late or ahead.

Screen 7 (Connection)



The user type an existing or not existing login and password. If the login and password ever exists, data from the parse account and from the android merge.

Key Considerations

How will your app handle data persistence?

The app will sync its data with a ContentProvider and Loaders connected to a Parse account (http://parse.com/)

Describe any corner cases in the UX.

The app will contain a bound service that control the current activity. The bound service will show a notification if the app is hidden. If the user go back in the app, a "Now Running" bar will appear at the bottom of the app.

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

Parse
(To do data persistence and account creation)
Google Play Services
(To handle location)
Google Play Services Ads
(To make profits from the app ;-))
ButterKnife
(To make a clean code)

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

- Create project
- Configure Github
- Create Parse.com account
- Configure libraries

Task 2: Implement UI for MainActivity and ConnectionActivity

- Build UI for MainActivity and MainFragment
- Build UI for ConnectionActivity and ConnectionFragment

Task 3: Preparing data

- Create dummy data on Parse

Task 4: Link parse data

- Create ContentProvider
- Create SyncAdapter to fetch Parse dummy datas

Task 5: Details view

- Create SessionDetailsActivity and SessionDetailsFragment UI
- Links it to the provider
- Add the share button and the delete button

Task 6: Session type view

- Create SessionTypeActivity and SessionTypeFragment UI
- Add by default the two session types
- Links it to the provider to retrieve the dummy data

Task 7: New Session Type View

- Create CreateSessionTypeActivity and CreateSessionTypeFragment UI
- Handle the fab click and feed the listview/recyclerview
- Create the custom view (bar that describes the training)

Task 8: Create the Start Session View

- Create StartSessionActivity and StartSessionFragment UI
- Find the correct formula to calculate meters that have to be done depending on the vVO2max
- Refreshing meters that have to be done according to the formula

Task 9: Prepare assets

- Record the vocal assets

Task 10: Create the Running View

- Create RunningActivity and RunningFragment UI
- Create a BoundService that will be bound to this fragment/activity
- Handle location to follow the interval training
- Handle session finish and session interrupted
- Make that at the end of the training, performance will be stored and synchronized if connected