Developer Guide

Developer Guide

[GPX 1](#_Toc112102124)

[sample-MTB-gpx-6938 1](#_Toc112102125)

[Original\_GPX 1](#_Toc112102126)

[Matched\_GPX 1](#_Toc112102127)

[segments\_hors\_osm.csv 2](#_Toc112102128)

[graphhopper\_config 2](#_Toc112102129)

[config.yml 2](#_Toc112102130)

[mtb\_test.json 2](#_Toc112102131)

[mtb-router 2](#_Toc112102132)

[oprouting 2](#_Toc112102133)

[scripts 2](#_Toc112102134)

[\_\_\_main\_\_\_.py 2](#_Toc112102135)

# GPX

## sample-MTB-gpx-6938

This file contains the starting GPX files that will be used to add a weight to the OSM roads on graphhopper. More GPX files can be added in this folder, that actually contains 1233 GPX routes, located in Auvergne-Rhône-Alpes, in France.

## Original\_GPX

Contains the original GPX files, converted in UTF8 format, in order to be manipulated with python script and postgresql.

## Matched\_GPX

Contains the GPX files that have been snapped on the OSM roads using graphhopper’s map matching module. Those GPX files are used to calculate the weight of every OSM road, by counting the number of time every GPX route have been taken.

## segments\_hors\_osm.csv

.csv file that contains the map matching accuracy of every GPX route. It calculates the share of each GPX snapped route that is too far from the original GPX route, within a given distance. It gives, for each GPX file, the accuracy, and the remaining geometry of the snapped GPX file that too far from the original file.

# graphhopper\_config

## config.yml

Configuration file of the graphhoper server: sets the profiles (bike, car, …), the server ports and address and the tags that will be used by graphhopper’s routing module.

## mtb\_test.json

This file sets the custom routing model used for the Bike profile by graphhopper. It’s sets the weight attributed to every OSM road, based on the roads tags values (speed limit, road type, surface, and the custom weight gpx\_weight).

# mtb-router

This folder contains the grapphopper files. The mtb-router API uses the graphhopper engine, and adds its own files to use this server. The custom files are located in the *mtb-router/MtbRouter* folder. It contains the java classes that are used to set the graphhopper server, and to add the custom tag gpx\_weight to every OSM way.

* *src/main/java/com/mtb/router/MtbRouterApplication.java* configures the graphhopper application
* *src/main/java/com/mtb/router/entities/* contains the encoders of the new graphhopper encoded value, gpx\_weight, that will be used to configurate the routing engine, and the custom model. The encoded value *gpx\_weight.java* makes the link between the Java application and the postgresql database that hosts the weight of every GPX file.

# oprouting

The oprouting module is used to initialize and build the postgresql database that will store the gpx\_weight value for every OSM way. The oprouting scripts are called in the *script.py* python fil, that initializes the map-matching module, and builds the postgresql database.

# scripts

The *scripts* folder containts all the scripts that are used to initialize mtb-router.

## \_\_\_main\_\_\_.py

This python script groups and launches all the processes. The \_\_\_init\_\_\_() function initializes the postgresql database by calling the otrouting scripts. The \_\_\_main\_\_\_() function requires first the user to initialize and launch the graphhopper application. The user can edit its parameters (server port and url, GPX files paths and the accuracy precision required to calculate the snapping accuracy). This function creates a database with everi GPX route, uses the map-matching module from graphhopper to snap the routes with the OSM roads, and calculates the gpx\_weight for each OSM way, by counting the number of time each OSM roads has been taken by a cycling journey. All the functions called in the \_\_\_main\_\_\_() are described in the *scripts.py* file.

# output

The output folder contains several web files that have been used during the development to validate the map-matching method and the gpx-weight attribution, by displaying the map-matched GPX routes and the OSM ways that are taken by those routes. It is not used by mtb-router itself, but can be used to validate some improvements on the API.