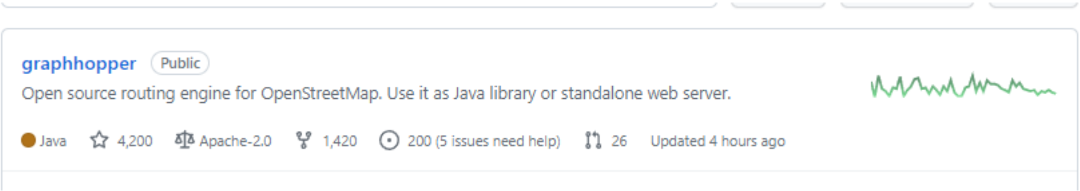
Graphhopper 오픈소스

graphhopper 오픈소스

https://github.com/graphhopper/graphhopper



graphhopper Public

Open source routing engine for OpenStreetMap. Use it as Java library or standalone web server.

● Java ☆ 4,200 Apache-2.0 1,420 200 (5 issues need help) 26 Updated 4 hours ago

깃허브 4.2k 라이브러리인 graphhopper에 대해서 간단하게 적어봄

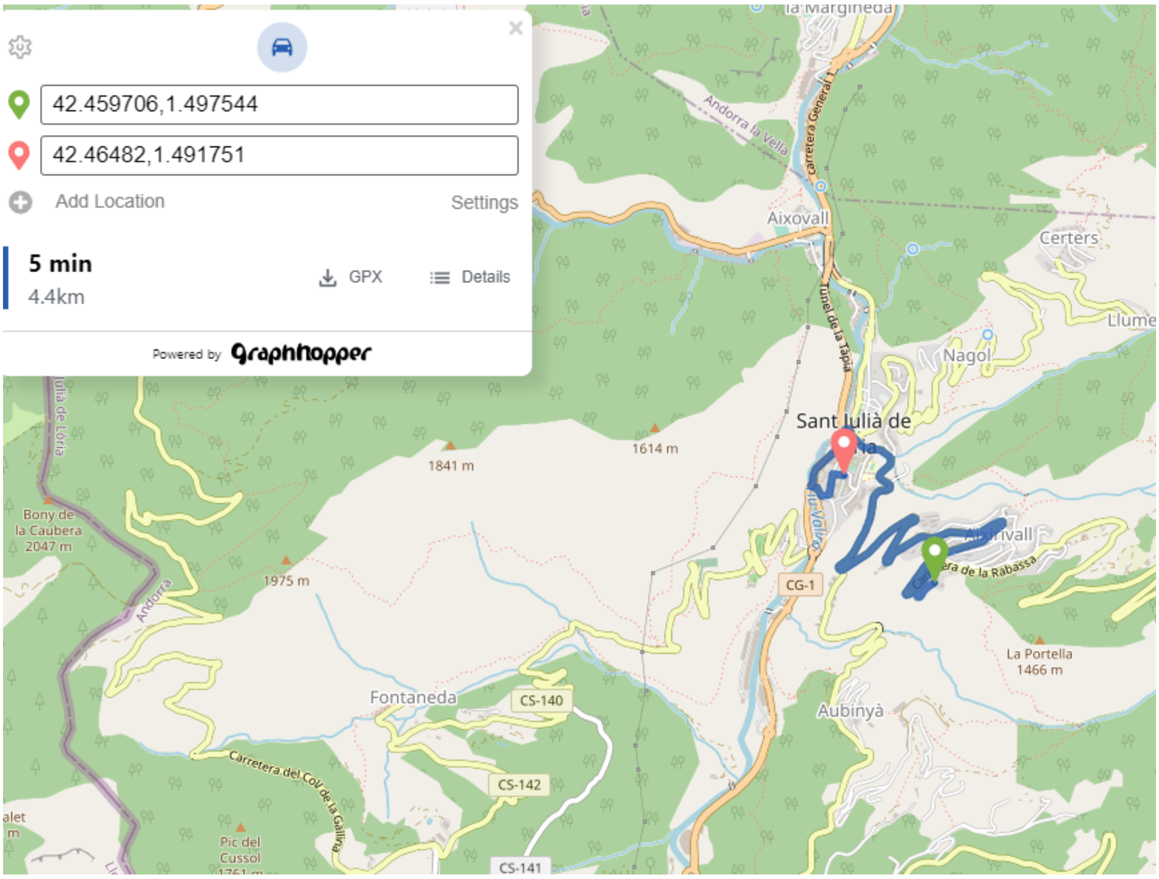
일단 OSM PBF 기반으로 경로 탐색 알고리즘을 지원하는 라이브러리이며 주 언어는 자바로 구성 되 어있다.

실행방법은 간단하다

도커로 구성 되어있어 아래와 같이 입력하면된다.

```
docker run -p 4141:8989 israelhikingmap/graphhopper \
  --url https://download.geofabrik.de/asia-latest.osm.pbf
```

url 뒤에 나와 있는 pbf 주소로 타겟을 변경 해주면 된다,



42.459706,1.497544

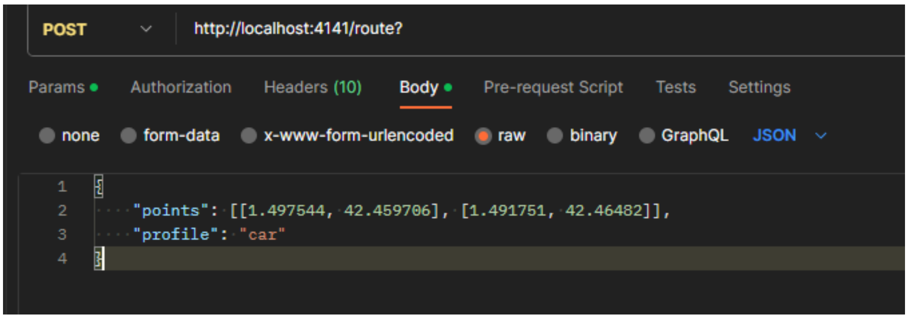
42.46482,1.491751

5 min
4.4km

사용법은 간단하다. 아래와 같이 경로를 입력해주면
위치를 찾아준다.

기능은 일반적인 길찾기와 비슷하며 내가 볼것은 여기서 제공해주는 API다.
일반적으로 POST 요청을 통해 보낼 수 있다.

사진과 같이 requests를 보내면 된다. 그럼 아래와 같이 이쁜 결과 값으로 나온다 이것을 잘 파싱해서 지도에 뿌려주면 된다. 끝!!!!



POST http://localhost:4141/route?

Params
Authorization
Headers (10)
Body
Pre-request Script
Tests
Settings
○ none ○ form-data ○ x-www-form-urlencoded ● raw ○ binary ○ GraphQL JSON

```
{
    "points": [[1.497544, 42.459706], [1.491751, 42.46482]],
    "profile": "car"
}
```

Graphhoper 길찾기 알고리즘

유명한 알고리즘인 dijkstyra 알고리즘과 Contraction Hierachies 알고리즘을 사용하여 최단거리를 계산한다. (도로로 연결된)

```
{
    "hints": {
        "visited_nodes.sum": 60,
        "visited_nodes.average": 60.0
    },
    "info": {
        "copyrights": [
            "GraphHopper",
            "OpenStreetMap contributors"
        ],
        "took": 3
    },
    "paths": [
        {
            "distance": 4364.453,
            "weight": 623.545577,
            "time": 318037,
            "transfers": 0,
            "points_encoded": true,
            "bbox": [
                1.489602,
                42.459028,
                1.501934,
                42.467055
            ],
            "points": "k|cbGmncHz@tBJL`At@DHAJEHMBu@]m@_@cAq@a@y@EAIBN^nAvBX\\pAhAFP@\\CFGDOEg@w@_Ay@e@i@i@_@m@i@eAoAe@w@Ki@Cg@c@cCo@o
            "instructions": [
                {
                    "distance": 277.146,
                    "heading": 235.45,
                    "sign": 0,
                    "interval": [
                        0,
                        13
                    ],
                    "text": "Continue",
                    "time": 33258,
                    "street_name": ""
                },
                {
                    "street_ref": "CS-131",
                    "distance": 3382.803,
                    "sign": -3,
                    "interval": [
                        13,
                        130
                    ],
                    "text": "Turn sharp left onto Carretera de la Rabassa",
                    "time": 216002,
                    "street_name": "Carretera de la Rabassa"
                },
                {
                    "exit_number": 3,
                    "distance": 399.719,
                    "sign": 6,
                    "exited": true,
                    "turn_angle": -6.03,
                    "interval": [
                        130,
                        149
                    ],
                    "text": "At roundabout, take exit 3 onto Carrer Francesc Carat",
                    "time": 29237,
                    "street_name": "Carrer Francesc Carat"
                },
                {
                    "distance": 21.661,
                    "sign": -2,
                    "interval": [
                        149,
                        151
                    ],
                    "text": "Turn left",
                    "time": 1200,
                    "street_name": ""
                },
                {
                    "distance": 63.753,
                    "sign": 0,
                    "interval": [
                        151,
                        156
                    ],
                    "text": "Continue onto Carrer Isidre Valls",
                    "time": 7650,
                    "street_name": "Carrer Isidre Valls"
                },
                {
                    "distance": 43.294,
                    "sign": -2,
                    "interval": [
                        156,
                        159
                    ],
                    "text": "Turn left onto Placeta del Mallador",
                    "time": 5195,
                    "street_name": "Placeta del Mallador"
                },
                {
                    "distance": 112.86,
                    "sign": -2,
                    "interval": [
                        159,
                        161
                    ],
                    "text": "Turn left onto Avinguda Verge de Canòlich",
                    "time": 16252,
                    "street_name": "Avinguda Verge de Canòlich"
                },
                {
                    "distance": 60.456,
                    "sign": 2,
                    "interval": [
                        161,
                        162
                    ],
                    "text": "Turn right onto carrer Casa Comuna",
                    "time": 7255,
                    "street_name": "carrer Casa Comuna"
                },
                {
                    "distance": 2.761,
                    "sign": -2,
                    "interval": [
                        162,
                        163
                    ],
                    "text": "Turn left",
                    "time": 1988,
                    "street_name": ""
                },
                {
                    "distance": 0.0,
                    "sign": 4,
                    "last_heading": 4.131025702893235,
                    "interval": [
                        163,
                        163
                    ],
                    "text": "Arrive at destination",
                    "time": 0,
                    "street_name": ""
                }
            ],
            "legs": [],
            "details": {},
            "ascend": 0.0,
            "descend": 0.0,
            "snapped_waypoints": "k|cbGmncHw^fc@"
        }
    ]
}
```

관련 설명이다.

Routing Web API Docs

In order to communicate with your or our hosted GraphHopper server you need to understand how to use it. There is a separate JavaScript and Java client for this API or use the plain JSON response for your language. To find out how to use the hosted GraphHopper Directions API you should refer to the online documentation here. This file describes the web API of the open source routing server.

A simple example

http://localhost:8989/route?point=52.5300591%2C13.3565022&point=52.5060440%2C13.4378107

The URL path of the local instance is http://localhost:8989

The endpoint to obtain the route is `/route` via GET.

HTTP POST

The GET request has an URL length limitation, so it won't work for many locations per request. In those cases use a HTTP POST request with JSON data as input. The POST request is identical except that all singular parameter names are named as their plural for a POST request. All effected parameters are: `points`, `snap_preventions`, `curbsides` and `point_hints`. (`details` stays `details`)

Please note that unlike to the GET endpoint, points are specified in `[longitude, latitude]` order. For example `point=10,11&point=20,22` will be the following JSON:

```
{ "points": [[11,10], [22,20]] }
```

Parameters

All official parameters are shown in the following table

Parameter	Default	Description
point	-	Specify multiple points for which the route should be calculated. The order is important. Specify at least two points.
locale	en	The locale of the resulting turn instructions. E.g. `pt_PT` for Portuguese or `de` for German
instructions	true	If instruction should be calculated and returned
profile	-	The profile to be used for the route calculation.
elevation	false	If `true` a third dimension - the elevation - is included in the polyline or in the GeoJson. IMPORTANT: If enabled you have to use a modified version of the decoding method or set points_encoded to `false`. See the points_encoded attribute for more details. Additionally a request can fail if the vehicle does not support elevation. See the features object for every vehicle.
points_encoded	true	If `false` the coordinates in `point` and `snapped_waypoints` are returned as array using the order [lon,lat,elevation] for every point. If `true` the coordinates will be encoded as string leading to less bandwidth usage. You'll need a special handling for the decoding of this string on the client-side. We provide open source code in Java and JavaScript. It is especially important to use no 3rd party client if you set `elevation=true`!
debug	false	If true, the output will be formatted.
calc_points	true	If the points for the route should be calculated at all printing out only distance and time.
point_hint	-	Optional parameter. Specifies a hint for each `point` parameter to prefer a certain street for the closest location lookup. E.g. if there is an address or house with two or more neighboring streets you can control for which street the closest location is looked up.
snap_prevention	-	Optional parameter to avoid snapping to a certain road class or road environment. Current supported values: `motorway`, `trunk`, `ferry`, `tunnel`, `bridge` and `ford`. Multiple values are specified like `snap_prevention=ferry&snap_prevention=motorway`
details	-	Optional parameter. You can request additional details for the route: `average_speed`, `street_name`, `edge_id`, `road_class`, `road_environment`, `max_speed` and `time` (and see which other values are configured in `graph.encoded_values`). Multiple values are specified like `details=average_speed&details=time`. The returned format for one detail segment is `[fromRef, toRef, value]`. The `ref` references the points of the response. Value can also be `null` if the property does not exist for one detail segment.
curbside	any	Optional parameter applicable to edge-based routing only. It specifies on which side a query point should be relative to the driver when she leaves/arrives at a start/target/via point. Possible values: right, left, any. Specify for every point parameter. See similar heading parameter.
force_curbside	true	Optional parameter. If it is set to true there will be an exception in case the curbside parameters cannot be fulfilled (e.g. specifying the wrong side for one-ways).
timeout_ms	inf	Optional parameter. Limits the request runtime to the minimum between the given value in milli-seconds and the server-side timeout configuration

Hybrid

If you enabled hybrid mode in the config you can use most of the features from flexible mode and still benefit from a speed up.

Parameter	Default	Description
ch.disable	`false`	Set to `true` in order to use the hybrid mode for the given profile, works only if the hybrid mode was enabled for this profile
lm.active_landmarks	4	Not recommended to change this

Flexible

Unlock certain flexible features via `ch.disable=true` per request or disable CH on the server-side by using an empty list for `profiles_ch`.

Parameter	Default	Description
ch.disable	`false`	Use this parameter in combination with one or more parameters of this table
custom_model	-	Customize the route calculations. See the documentation for more information.
algorithm	`astarbi`	The algorithm to calculate the route. Other options are `dijkstra`, `astar`, `astarbi`, `alternative_route` and `round_trip`
heading	NaN	Favour a heading direction for a certain point. Specify either one heading for the start point or as many as there are points. In this case headings are associated by their order to the specific points. Headings are given as north based clockwise angle between 0 and 360 degree. This parameter also influences the tour generated with `algorithm=round_trip` and forces the initial direction.
heading_penalty	120	Penalty for omitting a specified heading. The penalty corresponds to the accepted time delay in seconds in comparison to the route without a heading.
pass_through	`false`	If `true` u-turns are avoided at via-points with regard to the `heading_penalty`.
round_trip.distance	10000	If `algorithm=round_trip` this parameter configures approximative length of the resulting round trip
round_trip.seed	0	If `algorithm=round_trip` this parameter introduces randomness if e.g. the first try wasn't good.
alternative_route.max_paths	2	If `algorithm=alternative_route` this parameter sets the number of maximum paths which should be calculated. Increasing can lead to worse alternatives.
alternative_route.max_weight_factor	1.4	If `algorithm=alternative_route` this parameter sets the factor by which the alternatives routes can be longer than the optimal route. Increasing can lead to worse alternatives.
alternative_route.max_share_factor	0.6	If `algorithm=alternative_route` this parameter specifies how much alternatives routes can have maximum in common with the optimal route. Increasing can lead to worse alternatives.

Public Transit

Only applicable when profile `pt` is used.

Parameter	Default	Description
point	-	Specify multiple points for which the route should be calculated. The order is important. Specify at least two points.
locale	en	The locale of the resulting turn instructions. E.g. `pt_PT` for Portuguese or `de` for German.
pt.earliest_departure_time	-	Specify the earliest departure time of the itineraries. In ISO-8601 format `yyyy-MM-ddTHH:mm:ssZ` e.g. `2020-12-30T12:56:00Z`.
pt.arrive_by	false	If true the `pt.earliest_departure_time` parameter is used to define the latest time of arrival of the itineraries.
pt.profile	false	If true you request a list of all itineraries where each one is the best way to get from A to B, for some departure time within a specified time window. This profile query is also called "range query". The time window is specified via `pt.profile_duration`. Limited to 50 by default, change this via `pt.limit_solutions`.
pt.profile_duration	PT60M (1 hour)	The time window for a profile query and so only applicable if `pt.profile` is `true`. Duration string e.g. `PT200S`.
pt.limit_street_time	unlimited	Maximum duration on street for access or egress of public transit i.e. time outside of public transit. Duration string e.g. `PT30M`.
pt.ignore_transfers	false	Specifies if transfers as criterion should be ignored.
pt.limit_solutions	unlimited	The number of maximum solutions that should be searched.

Example output for the case `type=json`

Keep in mind that attributes which are not documented here can be removed in the future - you should not rely on them! The JSON result contains the following structure:

JSON path/attribute	Description
paths	An array of possible paths
paths[0].distance	The total distance of the route, in meter
paths[0].time	The total time of the route, in ms
paths[0].ascend	The total ascend (uphill) of the route, in meter
paths[0].descend	The total descend (downhill) of the route, in meter
paths[0].points	This value contains the coordinates of the path. If `points_encoded=true` or no `points_encoded` specified an encoded string will be returned, otherwise an array with order [lon,lat,elevation] is returned. See the parameter `points_encoded` for more information.
paths[0].points_encoded	Is true if the points are encoded, if not paths[0].points contains the geo json of the path (then order is lon,lat,elevation), which is easier to handle but consumes more bandwidth compared to encoded version
paths[0].bbox	The bounding box of the route, format:minLon, minLat, maxLon, maxLat
paths[0].snapped_waypoints	This value contains the snapped input points. If `points_encoded=true` or no `points_encoded` parameter was specified then an encoded string will be returned, otherwise an array is returned. See the parameter `points_encoded` for more information.
paths[0].instructions	Contains information about the instructions for this route. The last instruction is always the Finish instruction and takes 0ms and 0meter. Keep in mind that instructions are currently under active development and can sometimes contain misleading information, so, make sure you always show an image of the map at the same time when navigating your users!
paths[0].instructions[0].text	A description what the user has to do in order to follow the route. The language depends on the locale parameter.
paths[0].instructions[0].street_name	The name of the street to turn onto in order to follow the route.
paths[0].instructions[0].distance	The distance for this instruction, in meter
paths[0].instructions[0].time	The duration for this instruction, in ms
paths[0].instructions[0].interval	An array containing the first and the last index (relative to paths[0].points) of the points for this instruction. This is useful to know for which part of the route the instructions are valid.
paths[0].instructions[0].sign	A number which specifies the sign to show e.g. 2 for a right turn.KEEP_LEFT=-7TURN_SHARP_LEFT = -3TURN_LEFT = -2TURN_SLIGHT_LEFT = -1CONTINUE_ON_STREET = 0TURN_SLIGHT_RIGHT = 1TURN_RIGHT = 2TURN_SHARP_RIGHT = 3FINISH = 4REACHED_VIA = 5USE_ROUNDABOUT = 6KEEP_RIGHT=7implement some default for all other
paths[0].instructions[0].exit_number	[optional] Only available for USE_ROUNDABOUT instructions. The count of exits at which the route leaves the roundabout.
paths[0].instructions[0].exited	[optional] Only available for USE_ROUNDABOUT instructions. True if the roundabout should be exited. False if a via point or end is placed in the roundabout, thus, the roundabout should not be exited due to this instruction.
paths[0].instructions[0].turn_angle	[optional] Only available for USE_ROUNDABOUT instructions. The radian of the route within the roundabout: `0 < r < 2*PI` for clockwise and `-2PI < r < 0` for counterclockwise transit. `NaN` if the direction of rotation is undefined.

```
{
  "paths": [{
    "bbox": [
      13.362853824187303,
      52.469481955531585,
      13.385836736460217,
      52.473849308838446
    ],
    "distance": 2138.3027624572337,
    "instructions": [
      {
        "distance": 1268.519329705091,
        "interval": [
          0,
          10
        ],
        "sign": 0,
        "text": "Geradeaus auf A 100",
        "time": 65237
      },
      {
        "distance": 379.74399999999997,
        "interval": [
          10,
          11
        ],
        "sign": 0,
        "text": "Geradeaus auf Strasse",
        "time": 24855
      },
      {
        "distance": 16.451,
        "interval": [
          11,
          11
        ],
        "sign": 0,
        "text": "Geradeaus auf Tempelhofer Damm",
        "time": 1316
      },
      {
        "distance": 473.58843275214315,
        "interval": [
          11,
          12
        ],
        "sign": -2,
        "text": "Links abbiegen auf Tempelhofer Damm, B 96",
        "time": 37882
      },
      {
        "distance": 0,
        "interval": [
          12,
          12
        ],
        "sign": 4,
        "text": "Ziel erreicht!",
        "time": 0
      }
    ],
    "points": "oxg_Iy|ppAl@wCdE}LfFsN|@_Ej@eEtAaMh@sGVuDNcDb@{PFyGdAi]FoC?q@sXQ_@?",
    "points_encoded": true,
    "details":{
        "street_name":[[0,1,"Rue Principale"],[1,13,"D19E"],[13,18,"D19"],..]
    },
    "time": 129290
  }]
}
```

Area information

If you need to find out details about the area or need to ping the service use '/info'

http://localhost:8989/info

Example output:

```
{ "build_date":"2014-02-21T16:52",
  "bbox":[13.072624,52.333508,13.763972,52.679616],
  "version":"0.3",
  "features": { "foot" : { "elevation" : true  },
                "car"  : { "elevation" : false } }
}
```

JSON path/attribute	Description
version	The GraphHopper version
bbox	The maximum bounding box of the area, format:minLon, minLat, maxLon, maxLat
features	A json object per supported vehicles with name and supported features like elevation
build_date	[optional] The GraphHopper build date
import_date	[optional] The date time at which the OSM import was done
prepare_date	[optional] The date time at which the preparation (contraction hierarchies) was done. If nothing was done this is empty
supported_vehicles	[deprecated] An array of strings for all supported vehicles

Error Output

```
{
  "message": "Cannot find point 2: 2248.224673, 3.867187",
  "hints": [{"message": "something", ...}]
}
```

Sometimes a point can be "off the road" and you'll get 'cannot find point', this normally does not indicate a bug in the routing engine and is expected to a certain degree if too far away.

JSON path/attribute	Description
message	Not intended to be displayed to the user as it is not translated
hints	An optional list of details regarding the error message e.g. `[{"message": "first error message in hints"}]`

HTTP Error codes

HTTP error code	Reason
500	Internal server error. It is strongly recommended to send us the message and the link to it, as it is very likely a bug in our system.
501	Only a special list of vehicles is supported
400	Something was wrong in your request

Isochrone

In addition to routing, the end point to obtain an isochrone is `/isochrone`. To get a point list instead of a polygon you can have a look into the /spt endpoint.

http://localhost:8989/isochrone

All parameters are shown in the following table.

Parameter	Default	Description
profile		The profile to be used for the isochrone calculation.
buckets	1	Number by which to divide the given `time_limit` to create `buckets` nested isochrones of time intervals `time_limit-n*time_limit/buckets` for `n=[0,buckets)`. Applies analogously to `distance_limit`.
reverse_flow	false	If false the flow goes from point to the polygon, if true the flow goes from the polygon inside to the point. Example usage for false: *How many potential customer can be reached within 30min travel time from your store* vs. true: *How many customers can reach your store within 30min travel time.* (optional, default to false)
point		Specify the start coordinate (required). A string organized as `latitude,longitude`.
time_limit	600	Specify which time the vehicle should travel. In seconds. (optional, default to 600)
distance_limit	-1	Specify which distance the vehicle should travel. In meter. (optional, default to -1)
pt.earliest_departure_time		Specify the earliest departure time of the trip. Only applicable and required when profile `pt` is used. See the public transit section above for more details and other parameters.