python-gmaps Documentation

Release 0.0.2

Michał Jaworski

February 22, 2017

Contents

API:

CHAPTER 1

gmaps package

Submodules

gmaps.client module

class gmaps.client.**Client**(*sensor=False*, *api_key=None*, *use_https=True*)
Bases: object

Base class for Google Maps API endpoints

Parameters

- **sensor** – boolean value indicating if application is using sensor (such as a GPS locator) to determine the user's location.
- **api_key** – google business API key
- **use_https** – boolean indicating if https should be use to make requests

Note: Google API won't allow you to make plain http requests with API key. If you would like to use api_key you should use https too.

BASE_API_HTTPS_URL = 'https://maps.googleapis.com/maps/api/'

BASE_API_HTTP_URL = 'http://maps.googleapis.com/maps/api/'

static assume_latlon(*location*)

static assume_latlon_or_address(*location*)

gmaps.directions module

class gmaps.directions.**Directions**(*sensor=False*, *api_key=None*, *use_https=True*)
Bases: *gmaps.client.Client*

DIRECTIONS_URL = 'directions/'

directions(*origin*, *destination*, *mode=None*, *alternatives=None*, *waypoints=None*, *optimize_waypoints=False*, *avoid=None*, *language=None*, *units=None*, *region=None*, *departure_time=None*, *arrival_time=None*, *sensor=None*)
Get directions between locations

Parameters

- **origin** – Origin location - string address; (latitude, longitude) two-tuple, dict with ("lat", "lon") keys or object with (lat, lon) attributes
- **destination** – Destination location - type same as origin
- **mode** – Travel mode as string, defaults to "driving". See google docs details
- **alternatives** – True if provide it has to return more then one route alternative
- **waypoints** – Iterable with set of intermediate stops, like ("Munich", "Dallas") See google docs details
- **optimize_waypoints** – if true will attempt to re-order supplied waypoints to minimize overall cost of the route. If waypoints are optimized, the route returned will show the optimized order under "waypoint_order". See google docs details
- **avoid** – Iterable with set of restrictions, like ("tolls", "highways"). For full list refer to google docs details
- **language** – The language in which to return results. See list of supported languages
- **units** – Unit system for result. Defaults to unit system of origin's country. See google docs details
- **region** – The region code. Affects geocoding of origin and destination (see *gmaps.Geocoding.geocode* region parameter)
- **departure_time** – Desired time of departure as seconds since midnight, January 1, 1970 UTC
- **arrival_time** – Desired time of arrival for transit directions as seconds since midnight, January 1, 1970 UTC.

gmaps.distance_matrix module

gmaps.elevation module

class gmaps.elevation.**Elevation**(*sensor=False*, *api_key=None*, *use_https=True*)

Bases: *gmaps.client.Client*

ELEVATION_URL = 'elevation/'

elevation(*locations=None*, *samples=None*, *sensor=False*)

Parameters

- **locations** – list of lat/lon positions
- **samples** – specifies the number of sample points along a path for which to return elevation data. The samples parameter divides the given path into an ordered set of equidistant points along the path. If not set then the result will be elevation for every point in list of locations.

Returns

gmaps.errors module

exception gmaps.errors.**GmapException**
Bases: exceptions.Exception

Base exception for all python-gmap exceptions

exception gmaps.errors.**InvalidRequest**
Bases: *gmaps.errors.GmapException*

Raised when request to Google API was invalid

exception gmaps.errors.**NoResults**
Bases: *gmaps.errors.GmapException*

Raised when api returned no results

exception gmaps.errors.**RateLimitExceeded**
Bases: *gmaps.errors.GmapException*

Raised when rate limit to API endpoint was exceeded

exception gmaps.errors.**RequestDenied**
Bases: *gmaps.errors.GmapException*

Raised when request to API was denied

gmaps.geocoding module

class gmaps.geocoding.**Geocoding**(*sensor=False*, *api_key=None*, *use_https=True*)
Bases: *gmaps.client.Client*

GEOCODE_URL = 'geocode/'

geocode(*address=None*, *components=None*, *region=None*, *language=None*, *bounds=None*, *sensor=None*)
Geocode given address. Geocoder can queried using address and/or components. Components when used with address will restrict your query to specific area. When used without address they act like more precise query. For full details see Google docs.

Parameters

- **address** – address string
- **components** – ditc of components
- **region** – region code specified as a ccTLD ("top-level domain") two-character value, influences but not restricts query result
- **language** – the language in which to return results. For full list of laguages go to Google Maps API docs
- **bounds** – two-tuple of (latitude, longitude) pairs of bounding box. Influences but not restricts result (same as region parameter)
- **sensor** – override default client sensor parameter

reverse(*lat*, *lon*, *result_type=None*, *location_type=None*, *language=None*, *sensor=None*)
Reverse geocode with given latitude and longitude.

Parameters

- **lat** – latitude of queried point
- **lon** – longitude of queried point
- **result_type** – list of result_type for filtered search. Accepted values: https://developers.google.com/maps/documentation/geocoding/intro#Types **Important**: this feature may require using API key to work.
- **location_type** – list of location_type for filtered search.
- **language** – the language in which to return results. For full list of laguages go to Google Maps API docs
- **sensor** – override default client sensor parameter

Note: Google API allows to specify both latlng and address params but it makes no sense and would not reverse geocode your query, so here geocoding and reverse geocoding are separated

---

gmaps.status module

gmaps.timezone module

class gmaps.timezone.**Timezone**(*sensor=False*, *api_key=None*, *use_https=True*)

Bases: *gmaps.client.Client*

TIMEZONE_URL = 'timezone/'

timezone(*lat*, *lon*, *datetime*, *language=None*, *sensor=None*)

Get time offset data for given location.

Parameters

- **lat** – Latitude of queried point
- **lon** – Longitude of queried point
- **language** – The language in which to return results. For full list of laguages go to Google Maps API docs
- **datetime** (*datetime.datetime*) – Desired time. The Time Zone API uses the timestamp to determine whether or not Daylight Savings should be applied. datetime should be timezone aware. If it isn't the UTC timezone is assumed.
- **sensor** – Override default client sensor parameter

gmaps.timezone.**total_seconds**(*td*)

Take a timedelta and return the number of seconds it represents

gmaps.timezone.**unixtimestamp**(*datetime*)

Get unix time stamp from that given datetime. If datetime is not tzaware then it's assumed that it is UTC

Module contents

repo on github

CHAPTER 2

Indices and tables

- genindex
- modindex
- search

g