GeoPy Documentation

Release 1.14.0

GeoPy Contributors

May 13, 2018

Contents

geopy is a Python 2 and 3 client for several popular geocoding web services.

geopy makes it easy for Python developers to locate the coordinates of addresses, cities, countries, and landmarks across the globe using third-party geocoders and other data sources.

geopy is tested against CPython (versions 2.7, 3.4, 3.5, 3.6), PyPy, and PyPy3. geopy does not and will not support CPython 2.6.

CHAPTER 1

Geocoders

Each geolocation service you might use, such as Google Maps, Bing Maps, or Nominatim, has its own class in `geopy.geocoders` abstracting the service's API. Geocoders each define at least a `geocode` method, for resolving a location from a string, and may define a `reverse` method, which resolves a pair of coordinates to an address. Each Geocoder accepts any credentials or settings needed to interact with its service, e.g., an API key or locale, during its initialization.

To geolocate a query to an address and coordinates:

```
>>> from geopy.geocoders import Nominatim
>>> geolocator = Nominatim()
>>> location = geolocator.geocode("175 5th Avenue NYC")
>>> print(location.address)
Flatiron Building, 175, 5th Avenue, Flatiron, New York, NYC, New York, ...
>>> print((location.latitude, location.longitude))
(40.7410861, -73.9896297241625)
>>> print(location.raw)
{'place_id': '9167009604', 'type': 'attraction', ...}
```

To find the address corresponding to a set of coordinates:

```
>>> from geopy.geocoders import Nominatim
>>> geolocator = Nominatim()
>>> location = geolocator.reverse("52.509669, 13.376294")
>>> print(location.address)
Potsdamer Platz, Mitte, Berlin, 10117, Deutschland, European Union
>>> print((location.latitude, location.longitude))
(52.5094982, 13.3765983)
>>> print(location.raw)
{'place_id': '654513', 'osm_type': 'node', ...}
```

Locators' `geocode` and `reverse` methods require the argument `query`, and also accept at least the argument `exactly_one`, which is `True` by default. Geocoders may have additional attributes, e.g., Bing accepts `user_location`, the effect of which is to bias results near that location. `geocode` and `reverse` methods may return three types of values:

- When there are no results found, returns `None`.
- When the method's `exactly_one` argument is `True` and at least one result is found, returns a *geopy.location.Location* object, which can be iterated over as:

  ```
  (address<String>, (latitude<Float>, longitude<Float>))
  ```

  Or can be accessed as `location.address`, `location.latitude`, `location.longitude`, `location.altitude`, and `location.raw`. The last contains the full geocoder's response for this result.
- When `exactly_one` is `False`, and there is at least one result, returns a list of *geopy.location.Location* objects, as above:

  ```
  [location, [...]]
  ```

If a service is unavailable or otherwise returns a non-OK response, or doesn't receive a response in the allotted timeout, you will receive one of the *Exceptions* detailed below.

Every geocoder accepts an argument `format_string` that defaults to `'%s'` where the input string to geocode is interpolated. For example, if you only need to geocode locations in *Cleveland, Ohio*, you could do:

```
>>> from geopy.geocoders import GoogleV3
>>> geolocator = GoogleV3(format_string="%s, Cleveland OH")
>>> address, (latitude, longitude) = geolocator.geocode("11111 Euclid Ave")
>>> print(address, latitude, longitude)
Thwing Center, 11111 Euclid Ave, Cleveland, OH 44106, USA 41.5074066 -81.
↪60832649999999
```

class geopy.geocoders.**options**

The *options* object contains default configuration values for geocoders, e.g. *timeout* and *User-Agent*. Instead of passing a custom value to each geocoder individually, you can override a default value in this object.

Please note that not all geocoders use all attributes of this object. For example, some geocoders don't respect the `default_scheme` attribute. Refer to the specific geocoder's initializer doc for a list of parameters which that geocoder accepts.

Example for overriding default `timeout` and `user_agent`:

```
>>> import geopy.geocoders
>>> from geopy.geocoders import Nominatim
>>> geopy.geocoders.options.default_user_agent = 'my_app/1'
>>> geopy.geocoders.options.default_timeout = 7
>>> geolocator = Nominatim()
>>> print(geolocator.headers)
{'User-Agent': 'my_app/1'}
>>> print(geolocator.timeout)
7
```

Attributes:

default_format_string String containing `'%s'` where the string to geocode should be interpolated before querying the geocoder. Used by *geocode* calls only. For example: `'%s, Mountain View, CA'`.

default_proxies If specified, tunnel requests through the specified proxy. E.g., `{"https": "192.0.2.0"}`. For more information, see documentation on `urllib.request.ProxyHandler`.

default_scheme Use `'https'` or `'http'` as the API URL's scheme.

default_ssl_context An `ssl.SSLContext` instance with custom TLS verification settings. Pass `None` to use the interpreter's defaults (starting from Python 2.7.9 and 3.4.3 that is to use the system's trusted CA certificates; the older versions don't support TLS verification completely).

For older versions of Python (before 2.7.9 and 3.4.3) this argument is ignored, as *urlopen* doesn't accept an ssl context there, and a warning is issued.

To use the CA bundle used by *requests* library:

```
import ssl
import certifi
import geopy.geocoders
ctx = ssl.create_default_context(cafile=certifi.where())
geopy.geocoders.options.default_ssl_context = ctx
```

To disable TLS certificate verification completely:

```
import ssl
import geopy.geocoders
ctx = ssl.create_default_context()
ctx.check_hostname = False
ctx.verify_mode = ssl.CERT_NONE
geopy.geocoders.options.default_ssl_context = ctx
```

See docs for the `ssl.SSLContext` class for more examples.

default_timeout Time, in seconds, to wait for the geocoding service to respond before raising a *geopy.exc.GeocoderTimedOut* exception. Pass *None* to disable timeout.

default_user_agent User-Agent header to send with the requests to geocoder API.

default_format_string = '%s'

default_proxies = None

default_scheme = 'https'

default_ssl_context = None

default_timeout = 1

default_user_agent = 'geopy/1.14.0'

geopy.geocoders.**get_geocoder_for_service**(*service*)

For the service provided, try to return a geocoder class.

```
>>> from geopy.geocoders import get_geocoder_for_service
>>> get_geocoder_for_service("nominatim")
geopy.geocoders.osm.Nominatim
```

If the string given is not recognized, a *geopy.exc.GeocoderNotFound* exception is raised.

class geopy.geocoders.**ArcGIS**(*username=None*, *password=None*, *referer=None*, *token_lifetime=60*, *scheme=None*, *timeout=DEFAULT_SENTINEL*, *proxies=DEFAULT_SENTINEL*, *user_agent=None*, *format_string=None*, *ssl_context=DEFAULT_SENTINEL*)

Geocoder using the ERSI ArcGIS API.

Documentation at: https://developers.arcgis.com/rest/geocode/api-reference/overview-world-geocoding-service.htm

__init__(*username=None*, *password=None*, *referer=None*, *token_lifetime=60*, *scheme=None*, *timeout=DEFAULT_SENTINEL*, *proxies=DEFAULT_SENTINEL*, *user_agent=None*, *format_string=None*, *ssl_context=DEFAULT_SENTINEL*)

Parameters

- **username** (*str*) – ArcGIS username. Required if authenticated mode is desired.
- **password** (*str*) – ArcGIS password. Required if authenticated mode is desired.
- **referer** (*str*) – Required if authenticated mode is desired. *Referer* HTTP header to send with each request, e.g., `'http://www.example.com'`. This is tied to an issued token, so fielding queries for multiple referrers should be handled by having multiple ArcGIS geocoder instances.
- **token_lifetime** (*int*) – Desired lifetime, in minutes, of an ArcGIS-issued token.
- **scheme** (*str*) – See *geopy.geocoders.options.default_scheme*. If authenticated mode is in use, it must be `'https'`.
- **timeout** (*int*) – See *geopy.geocoders.options.default_timeout*.
- **proxies** (*dict*) – See *geopy.geocoders.options.default_proxies*.
- **user_agent** (*str*) – See *geopy.geocoders.options.default_user_agent*.

  New in version 1.12.0.
- **format_string** (*str*) – See *geopy.geocoders.options.default_format_string*.

  New in version 1.14.0.
- **ssl_context** (`ssl.SSLContext`) – See *geopy.geocoders.options.default_ssl_context*.

  New in version 1.14.0.

geocode(*query*, *exactly_one=True*, *timeout=DEFAULT_SENTINEL*, *out_fields=None*)

Return a location point by address.

Parameters

- **query** (*str*) – The address or query you wish to geocode.
- **exactly_one** (*bool*) – Return one result or a list of results, if available.
- **timeout** (*int*) – Time, in seconds, to wait for the geocoding service to respond before raising a *geopy.exc.GeocoderTimedOut* exception. Set this only if you wish to override, on this call only, the value set during the geocoder's initialization.
- **out_fields** (*str or iterable*) – A list of output fields to be returned in the attributes field of the raw data. This can be either a python list/tuple of fields or a comma-separated string. See https://developers.arcgis.com/rest/geocode/api-reference/geocoding-service-output.htm for a list of supported output fields. If you want to return all supported output fields, set `out_fields="*"`.

  New in version 1.14.0.

Return type `None`, *geopy.location.Location* or a list of them, if `exactly_one=False`.

reverse(*query*, *exactly_one=True*, *timeout=DEFAULT_SENTINEL*, *distance=None*, *wkid=4326*)

Return an address by location point.

Parameters

- **query** (*geopy.point.Point*, list or tuple of `(latitude, longitude)`, or string as `"%(latitude)s, %(longitude)s"`.) – The coordinates for which you wish to obtain the closest human-readable addresses.

- **exactly_one** (*bool*) – Return one result or a list of results, if available.
- **timeout** (*int*) – Time, in seconds, to wait for the geocoding service to respond before raising a *geopy.exc.GeocoderTimedOut* exception. Set this only if you wish to override, on this call only, the value set during the geocoder's initialization.
- **distance** (*int*) – Distance from the query location, in meters, within which to search. ArcGIS has a default of 100 meters, if not specified.
- **wkid** (*str*) – WKID to use for both input and output coordinates.

  Deprecated since version 1.14.0: It wasn't working before because it was specified incorrectly in the request parameters, and won't work even if we fix the request, because *geopy.point.Point* normalizes the coordinates according to WKID 4326. Please open an issue in the geopy issue tracker if you believe that custom wkid values should be supported.

Return type None, *geopy.location.Location* or a list of them, if `exactly_one=False`.

class geopy.geocoders.**Baidu**(*api_key*, *scheme=None*, *timeout=DEFAULT_SENTINEL*, *proxies=DEFAULT_SENTINEL*, *user_agent=None*, *format_string=None*, *ssl_context=DEFAULT_SENTINEL*)

Geocoder using the Baidu Maps v2 API.

Documentation at: http://lbsyun.baidu.com/index.php?title=webapi/guide/webservice-geocoding

New in version 1.0.0.

__init__(*api_key*, *scheme=None*, *timeout=DEFAULT_SENTINEL*, *proxies=DEFAULT_SENTINEL*, *user_agent=None*, *format_string=None*, *ssl_context=DEFAULT_SENTINEL*)

Parameters

- **api_key** (*str*) – The API key required by Baidu Map to perform geocoding requests. API keys are managed through the Baidu APIs console (http://lbsyun.baidu.com/apiconsole/key).
- **scheme** (*str*) – See *geopy.geocoders.options.default_scheme*.

  Changed in version 1.14.0: Default scheme has been changed from `http` to `https`.
- **timeout** (*int*) – See *geopy.geocoders.options.default_timeout*.
- **proxies** (*dict*) – See *geopy.geocoders.options.default_proxies*.
- **user_agent** (*str*) – See *geopy.geocoders.options.default_user_agent*.

  New in version 1.12.0.
- **format_string** (*str*) – See *geopy.geocoders.options.default_format_string*.

  New in version 1.14.0.
- **ssl_context** (`ssl.SSLContext`) – See *geopy.geocoders.options.default_ssl_context*.

  New in version 1.14.0.

geocode(*query*, *exactly_one=True*, *timeout=DEFAULT_SENTINEL*)

Return a location point by address.

Parameters

- **query** (*str*) – The address or query you wish to geocode.
- **exactly_one** (*bool*) – Return one result or a list of results, if available.
- **timeout** (*int*) – Time, in seconds, to wait for the geocoding service to respond before raising a *geopy.exc.GeocoderTimedOut* exception. Set this only if you wish to override, on this call only, the value set during the geocoder's initialization.

Return type `None`, *geopy.location.Location* or a list of them, if `exactly_one=False`.

reverse(*query*, *exactly_one=True*, *timeout=DEFAULT_SENTINEL*)

Return an address by location point.

Parameters

- **query** (*geopy.point.Point*, list or tuple of `(latitude, longitude)`, or string as `"%(latitude)s, %(longitude)s"`.) – The coordinates for which you wish to obtain the closest human-readable addresses.
- **exactly_one** (*bool*) – Return one result or a list of results, if available. Baidu's API will always return at most one result.

  New in version 1.14.0.
- **timeout** (*int*) – Time, in seconds, to wait for the geocoding service to respond before raising a *geopy.exc.GeocoderTimedOut* exception. Set this only if you wish to override, on this call only, the value set during the geocoder's initialization.

Return type `None`, *geopy.location.Location* or a list of them, if `exactly_one=False`.

class geopy.geocoders.**Bing**(*api_key*, *format_string=None*, *scheme=None*, *timeout=DEFAULT_SENTINEL*, *proxies=DEFAULT_SENTINEL*, *user_agent=None*, *ssl_context=DEFAULT_SENTINEL*)

Geocoder using the Bing Maps Locations API.

Documentation at: https://msdn.microsoft.com/en-us/library/ff701715.aspx

__init__(*api_key*, *format_string=None*, *scheme=None*, *timeout=DEFAULT_SENTINEL*, *proxies=DEFAULT_SENTINEL*, *user_agent=None*, *ssl_context=DEFAULT_SENTINEL*)

Parameters

- **api_key** (*str*) – Should be a valid Bing Maps API key (https://www.microsoft.com/en-us/maps/create-a-bing-maps-key).
- **format_string** (*str*) – See *geopy.geocoders.options.default_format_string*.
- **scheme** (*str*) – See *geopy.geocoders.options.default_scheme*.
- **timeout** (*int*) – See *geopy.geocoders.options.default_timeout*.
- **proxies** (*dict*) – See *geopy.geocoders.options.default_proxies*.
- **user_agent** (*str*) – See *geopy.geocoders.options.default_user_agent*.

  New in version 1.12.0.
- **ssl_context** (`ssl.SSLContext`) – See *geopy.geocoders.options.default_ssl_context*.

  New in version 1.14.0.

geocode(*query*, *exactly_one=True*, *user_location=None*, *timeout=DEFAULT_SENTINEL*, *culture=None*, *include_neighborhood=None*, *include_country_code=False*)

Return a location point by address.

Parameters

- **query** (*str*) – The address or query you wish to geocode.

  For a structured query, provide a dictionary whose keys are one of: *addressLine*, *locality* (city), *adminDistrict* (state), *countryRegion*, or *postalcode*.

- **exactly_one** (*bool*) – Return one result or a list of results, if available.

- **user_location** (*geopy.point.Point*) – Prioritize results closer to this location.

- **timeout** (*int*) – Time, in seconds, to wait for the geocoding service to respond before raising a *geopy.exc.GeocoderTimedOut* exception. Set this only if you wish to override, on this call only, the value set during the geocoder's initialization.

- **culture** (*str*) – Affects the language of the response, must be a two-letter country code.

  New in version 1.4.0.

- **include_neighborhood** (*bool*) – Sets whether to include the neighborhood field in the response.

  New in version 1.4.0.

- **include_country_code** (*bool*) – Sets whether to include the two-letter ISO code of the country in the response (field name 'countryRegionIso2').

  New in version 1.4.0.

Return type `None`, *geopy.location.Location* or a list of them, if `exactly_one=False`.

reverse(*query*, *exactly_one=True*, *timeout=DEFAULT_SENTINEL*, *culture=None*, *include_country_code=False*)

Return an address by location point.

Parameters

- **query** (*geopy.point.Point*, list or tuple of `(latitude, longitude)`, or string as `"%(latitude)s, %(longitude)s"`.) – The coordinates for which you wish to obtain the closest human-readable addresses.

- **exactly_one** (*bool*) – Return one result or a list of results, if available.

- **timeout** (*int*) – Time, in seconds, to wait for the geocoding service to respond before raising a *geopy.exc.GeocoderTimedOut* exception. Set this only if you wish to override, on this call only, the value set during the geocoder's initialization.

- **culture** (*str*) – Affects the language of the response, must be a two-letter country code.

- **include_country_code** (*bool*) – Sets whether to include the two-letter ISO code of the country in the response (field name 'countryRegionIso2').

Return type `None`, *geopy.location.Location* or a list of them, if `exactly_one=False`.

class geopy.geocoders.**DataBC**(*scheme=None*, *timeout=DEFAULT_SENTINEL*, *proxies=DEFAULT_SENTINEL*, *user_agent=None*, *format_string=None*, *ssl_context=DEFAULT_SENTINEL*)

Geocoder using the Physical Address Geocoder from DataBC.

Documentation at: http://www.data.gov.bc.ca/dbc/geographic/locate/geocoding.page

__init__(*scheme=None*, *timeout=DEFAULT_SENTINEL*, *proxies=DEFAULT_SENTINEL*, *user_agent=None*, *format_string=None*, *ssl_context=DEFAULT_SENTINEL*)

Parameters

- **scheme** (*str*) – See *geopy.geocoders.options.default_scheme*.
- **timeout** (*int*) – See *geopy.geocoders.options.default_timeout*.
- **proxies** (*dict*) – See *geopy.geocoders.options.default_proxies*.
- **user_agent** (*str*) – See *geopy.geocoders.options.default_user_agent*.

  New in version 1.12.0.
- **format_string** (*str*) – See *geopy.geocoders.options.default_format_string*.

  New in version 1.14.0.
- **ssl_context** (ssl.SSLContext) – See *geopy.geocoders.options.default_ssl_context*.

  New in version 1.14.0.

geocode(*query*, *max_results=25*, *set_back=0*, *location_descriptor='any'*, *exactly_one=True*, *timeout=DEFAULT_SENTINEL*)

Return a location point by address.

Parameters

- **query** (*str*) – The address or query you wish to geocode.
- **max_results** (*int*) – The maximum number of resutls to request.
- **set_back** (*float*) – The distance to move the accessPoint away from the curb (in meters) and towards the interior of the parcel. location_descriptor must be set to accessPoint for set_back to take effect.
- **location_descriptor** (*str*) – The type of point requested. It can be any, accessPoint, frontDoorPoint, parcelPoint, rooftopPoint and routingPoint.
- **exactly_one** (*bool*) – Return one result or a list of results, if available.
- **timeout** (*int*) – Time, in seconds, to wait for the geocoding service to respond before raising a *geopy.exc.GeocoderTimedOut* exception. Set this only if you wish to override, on this call only, the value set during the geocoder's initialization.

Return type None, *geopy.location.Location* or a list of them, if exactly_one=False.

class geopy.geocoders.**GeocodeFarm**(*api_key=None*, *format_string=None*, *timeout=DEFAULT_SENTINEL*, *proxies=DEFAULT_SENTINEL*, *user_agent=None*, *ssl_context=DEFAULT_SENTINEL*, *scheme=None*)

Geocoder using the GeocodeFarm API.

Documentation at: https://www.geocode.farm/geocoding/free-api-documentation/

__init__(*api_key=None*, *format_string=None*, *timeout=DEFAULT_SENTINEL*, *proxies=DEFAULT_SENTINEL*, *user_agent=None*, *ssl_context=DEFAULT_SENTINEL*, *scheme=None*)

Parameters

- **api_key** (*str*) – (optional) The API key required by GeocodeFarm to perform geocoding requests.
- **format_string** (*str*) – See *geopy.geocoders.options.default_format_string*.
- **timeout** (*int*) – See *geopy.geocoders.options.default_timeout*.
- **proxies** (*dict*) – See *geopy.geocoders.options.default_proxies*.
- **user_agent** (*str*) – See *geopy.geocoders.options.default_user_agent*.

  New in version 1.12.0.
- **ssl_context** (ssl.SSLContext) – See *geopy.geocoders.options.default_ssl_context*.

  New in version 1.14.0.
- **scheme** (*str*) – See *geopy.geocoders.options.default_scheme*.

  New in version 1.14.0.

geocode(*query*, *exactly_one=True*, *timeout=DEFAULT_SENTINEL*)

Return a location point by address.

Parameters

- **query** (*str*) – The address or query you wish to geocode.
- **exactly_one** (*bool*) – Return one result or a list of results, if available.
- **timeout** (*int*) – Time, in seconds, to wait for the geocoding service to respond before raising a *geopy.exc.GeocoderTimedOut* exception. Set this only if you wish to override, on this call only, the value set during the geocoder's initialization.

Return type None, *geopy.location.Location* or a list of them, if exactly_one=False.

reverse(*query*, *exactly_one=True*, *timeout=DEFAULT_SENTINEL*)

Return an address by location point.

Parameters

- **query** (*geopy.point.Point*, list or tuple of (latitude, longitude), or string as "%(latitude)s, %(longitude)s".) – The coordinates for which you wish to obtain the closest human-readable addresses.
- **exactly_one** (*bool*) – Return one result or a list of results, if available. GeocodeFarm's API will always return at most one result.
- **timeout** (*int*) – Time, in seconds, to wait for the geocoding service to respond before raising a *geopy.exc.GeocoderTimedOut* exception. Set this only if you wish to override, on this call only, the value set during the geocoder's initialization.

Return type None, *geopy.location.Location* or a list of them, if exactly_one=False.

class geopy.geocoders.**GeoNames**(*country_bias=None*, *username=None*, *timeout=DEFAULT_SENTINEL*, *proxies=DEFAULT_SENTINEL*, *user_agent=None*, *format_string=None*, *ssl_context=DEFAULT_SENTINEL*)

GeoNames geocoder.

Documentation at: http://www.geonames.org/export/geonames-search.html

Reverse geocoding documentation at: http://www.geonames.org/export/web-services.html#findNearbyPlaceName

__init__(*country_bias=None*, *username=None*, *timeout=DEFAULT_SENTINEL*, *proxies=DEFAULT_SENTINEL*, *user_agent=None*, *format_string=None*, *ssl_context=DEFAULT_SENTINEL*)

Parameters

- **country_bias** (*str*) –
- **username** (*str*) – GeoNames username, required. Sign up here: http://www.geonames.org/login
- **timeout** (*int*) – See *geopy.geocoders.options.default_timeout*.
- **proxies** (*dict*) – See *geopy.geocoders.options.default_proxies*.
- **user_agent** (*str*) – See *geopy.geocoders.options.default_user_agent*.

  New in version 1.12.0.
- **format_string** (*str*) – See *geopy.geocoders.options.default_format_string*.

  New in version 1.14.0.
- **ssl_context** (ssl.SSLContext) – See *geopy.geocoders.options.default_ssl_context*.

  New in version 1.14.0.

geocode(*query*, *exactly_one=True*, *timeout=DEFAULT_SENTINEL*)

Return a location point by address.

Parameters

- **query** (*str*) – The address or query you wish to geocode.
- **exactly_one** (*bool*) – Return one result or a list of results, if available.
- **timeout** (*int*) – Time, in seconds, to wait for the geocoding service to respond before raising a *geopy.exc.GeocoderTimedOut* exception. Set this only if you wish to override, on this call only, the value set during the geocoder's initialization.

Return type None, *geopy.location.Location* or a list of them, if `exactly_one=False`.

reverse(*query*, *exactly_one=DEFAULT_SENTINEL*, *timeout=DEFAULT_SENTINEL*)

Return an address by location point.

New in version 1.2.0.

Parameters

- **query** (*geopy.point.Point*, list or tuple of `(latitude, longitude)`, or string as `"%(latitude)s, %(longitude)s"`.) – The coordinates for which you wish to obtain the closest human-readable addresses.
- **exactly_one** (*bool*) – Return one result or a list of results, if available.

  Changed in version 1.14.0: Default value for `exactly_one` was `False`, which differs from the conventional default across geopy. Please always pass this argument explicitly, otherwise you would get a warning. In geopy 2.0 the default value will become `True`.
- **timeout** (*int*) – Time, in seconds, to wait for the geocoding service to respond before raising a *geopy.exc.GeocoderTimedOut* exception. Set this only if you wish to override, on this call only, the value set during the geocoder's initialization.

Return type `None`, *geopy.location.Location* or a list of them, if `exactly_one=False`.

class geopy.geocoders.**GoogleV3**(*api_key=None*, *domain='maps.googleapis.com'*, *scheme=None*, *client_id=None*, *secret_key=None*, *timeout=DEFAULT_SENTINEL*, *proxies=DEFAULT_SENTINEL*, *user_agent=None*, *format_string=None*, *ssl_context=DEFAULT_SENTINEL*, *channel=''*)

Geocoder using the Google Maps v3 API.

Documentation at: https://developers.google.com/maps/documentation/geocoding/

API authentication is only required for Google Maps Premier customers.

__init__(*api_key=None*, *domain='maps.googleapis.com'*, *scheme=None*, *client_id=None*, *secret_key=None*, *timeout=DEFAULT_SENTINEL*, *proxies=DEFAULT_SENTINEL*, *user_agent=None*, *format_string=None*, *ssl_context=DEFAULT_SENTINEL*, *channel=''*)

Parameters

- **api_key** (*str*) – The API key required by Google to perform geocoding requests. API keys are managed through the Google APIs console (https://code.google.com/apis/console).
- **domain** (*str*) – Should be the localized Google Maps domain to connect to. The default is 'maps.googleapis.com', but if you're geocoding address in the UK (for example), you may want to set it to 'maps.google.co.uk' to properly bias results.
- **scheme** (*str*) – See *geopy.geocoders.options.default_scheme*.
- **client_id** (*str*) – If using premier, the account client id.
- **secret_key** (*str*) – If using premier, the account secret key.
- **timeout** (*int*) – See *geopy.geocoders.options.default_timeout*.
- **proxies** (*dict*) – See *geopy.geocoders.options.default_proxies*.
- **user_agent** (*str*) – See *geopy.geocoders.options.default_user_agent*.

  New in version 1.12.0.
- **format_string** (*str*) – See *geopy.geocoders.options.default_format_string*.

  New in version 1.14.0.
- **ssl_context** (`ssl.SSLContext`) – See *geopy.geocoders.options.default_ssl_context*.

New in version 1.14.0.

- **channel** (*str*) – If using premier, the channel identifier.

  New in version 1.12.0.

geocode(*query=None*, *exactly_one=True*, *timeout=DEFAULT_SENTINEL*, *bounds=None*, *region=None*, *components=None*, *language=None*, *sensor=False*)

Return a location point by address.

Parameters

- **query** (*str*) – The address or query you wish to geocode. Optional, if `components` param is set:

```
>>> g = GoogleV3()
>>> g.geocode(components={"city": "Paris", "country": "FR"})
Location(France, (46.227638, 2.213749, 0.0))
```

  Changed in version 1.14.0: Now `query` is optional if `components` param is set.

- **exactly_one** (*bool*) – Return one result or a list of results, if available.
- **timeout** (*int*) – Time, in seconds, to wait for the geocoding service to respond before raising a *geopy.exc.GeocoderTimedOut* exception. Set this only if you wish to override, on this call only, the value set during the geocoder's initialization.
- **bounds** (*list or tuple*) – The bounding box of the viewport within which to bias geocode results more prominently.
- **region** (*str*) – The region code, specified as a ccTLD ("top-level domain") two-character value.
- **components** (*dict*) – Restricts to an area. Can use any combination of: route, locality, administrative_area, postal_code, country.
- **language** (*str*) – The language in which to return results.
- **sensor** (*bool*) – Whether the geocoding request comes from a device with a location sensor.

Return type `None`, *geopy.location.Location* or a list of them, if `exactly_one=False`.

reverse(*query*, *exactly_one=DEFAULT_SENTINEL*, *timeout=DEFAULT_SENTINEL*, *language=None*, *sensor=False*)

Return an address by location point.

Parameters

- **query** (*geopy.point.Point*, list or tuple of `(latitude, longitude)`, or string as `"%(latitude)s, %(longitude)s"`.) – The coordinates for which you wish to obtain the closest human-readable addresses.
- **exactly_one** (*bool*) – Return one result or a list of results, if available.

  Changed in version 1.14.0: Default value for `exactly_one` was `False`, which differs from the conventional default across geopy. Please always pass this argument explicitly, otherwise you would get a warning. In geopy 2.0 the default value will become `True`.

- **timeout** (*int*) – Time, in seconds, to wait for the geocoding service to respond before raising a *geopy.exc.GeocoderTimedOut* exception. Set this only if you wish to override, on this call only, the value set during the geocoder's initialization.
- **language** (*str*) – The language in which to return results.

- **sensor** (*bool*) – Whether the geocoding request comes from a device with a location sensor.

Return type None, *geopy.location.Location* or a list of them, if `exactly_one=False`.

timezone(*location*, *at_time=None*, *timeout=DEFAULT_SENTINEL*)

This is an unstable API.

Finds the timezone a *location* was in for a specified *at_time*, and returns a pytz timezone object.

New in version 1.2.0.

Parameters

- **location** (*geopy.point.Point*, list or tuple of (latitude, longitude), or string as "%(latitude)s, %(longitude)s") – The coordinates for which you want a timezone.
- **at_time** (*int or float or datetime*) – The time at which you want the timezone of this location. This is optional, and defaults to the time that the function is called in UTC.
- **timeout** (*int*) – Time, in seconds, to wait for the geocoding service to respond before raising a *geopy.exc.GeocoderTimedOut* exception. Set this only if you wish to override, on this call only, the value set during the geocoder's initialization.

Return type pytz timezone. See `pytz.timezone()`.

class geopy.geocoders.**IGNFrance**(*api_key*, *username=None*, *password=None*, *referer=None*, *domain='wxs.ign.fr'*, *scheme=None*, *timeout=DEFAULT_SENTINEL*, *proxies=DEFAULT_SENTINEL*, *user_agent=None*, *format_string=None*, *ssl_context=DEFAULT_SENTINEL*)

Geocoder using the IGN France GeoCoder OpenLS API.

Documentation at: https://geoservices.ign.fr/documentation/geoservices/index.html

__init__(*api_key*, *username=None*, *password=None*, *referer=None*, *domain='wxs.ign.fr'*, *scheme=None*, *timeout=DEFAULT_SENTINEL*, *proxies=DEFAULT_SENTINEL*, *user_agent=None*, *format_string=None*, *ssl_context=DEFAULT_SENTINEL*)

Parameters

- **api_key** (*str*) – The API key required by IGN France API to perform geocoding requests. You can get your key here: https://geoservices.ign.fr/documentation/services-acces.html. Mandatory. For authentication with referer and with username/password, the api key always differ.
- **username** (*str*) – When making a call need HTTP simple authentication username. Mandatory if no referer set
- **password** (*str*) – When making a call need HTTP simple authentication password. Mandatory if no referer set
- **referer** (*str*) – When making a call need HTTP referer. Mandatory if no password and username
- **domain** (*str*) – Currently it is `'wxs.ign.fr'`, can be changed for testing purposes for developer API e.g `'gpp3-wxs.ign.fr'` at the moment.
- **scheme** (*str*) – See *geopy.geocoders.options.default_scheme*.
- **timeout** (*int*) – See *geopy.geocoders.options.default_timeout*.

- **proxies** (*dict*) – See *geopy.geocoders.options.default_proxies*.
- **user_agent** (*str*) – See *geopy.geocoders.options.default_user_agent*.

  New in version 1.12.0.
- **format_string** (*str*) – See *geopy.geocoders.options.default_format_string*.

  New in version 1.14.0.
- **ssl_context** (ssl.SSLContext) – See *geopy.geocoders.options.default_ssl_context*.

  New in version 1.14.0.

geocode(*query*, *query_type='StreetAddress'*, *maximum_responses=25*, *is_freeform=False*, *filtering=None*, *exactly_one=True*, *timeout=DEFAULT_SENTINEL*)

Return a location point by address.

Parameters

- **query** (*str*) – The query string to be geocoded.
- **query_type** (*str*) – The type to provide for geocoding. It can be *PositionOfInterest*, *StreetAddress* or *CadastralParcel*. *StreetAddress* is the default choice if none provided.
- **maximum_responses** (*int*) – The maximum number of responses to ask to the API in the query body.
- **is_freeform** (*str*) – Set if return is structured with freeform structure or a more structured returned. By default, value is False.
- **filtering** (*str*) – Provide string that help setting geocoder filter. It contains an XML string. See examples in documentation and ignfrance.py file in directory tests.
- **exactly_one** (*bool*) – Return one result or a list of results, if available.
- **timeout** (*int*) – Time, in seconds, to wait for the geocoding service to respond before raising a *geopy.exc.GeocoderTimedOut* exception. Set this only if you wish to override, on this call only, the value set during the geocoder's initialization.

Return type None, *geopy.location.Location* or a list of them, if `exactly_one=False`.

reverse(*query*, *reverse_geocode_preference=('StreetAddress',)*, *maximum_responses=25*, *filtering=''*, *exactly_one=DEFAULT_SENTINEL*, *timeout=DEFAULT_SENTINEL*)

Return an address by location point.

Parameters

- **query** (*geopy.point.Point*, list or tuple of `(latitude, longitude)`, or string as `"%(latitude)s, %(longitude)s"`.) – The coordinates for which you wish to obtain the closest human-readable addresses.
- **reverse_geocode_preference** (*list*) – Enable to set expected results type. It can be *StreetAddress* or *PositionOfInterest*. Default is set to *StreetAddress*.
- **maximum_responses** (*int*) – The maximum number of responses to ask to the API in the query body.
- **filtering** (*str*) – Provide string that help setting geocoder filter. It contains an XML string. See examples in documentation and ignfrance.py file in directory tests.

- **exactly_one** (*bool*) – Return one result or a list of results, if available.

  Changed in version 1.14.0: Default value for `exactly_one` was `False`, which differs from the conventional default across geopy. Please always pass this argument explicitly, otherwise you would get a warning. In geopy 2.0 the default value will become `True`.

- **timeout** (*int*) – Time, in seconds, to wait for the geocoding service to respond before raising a *geopy.exc.GeocoderTimedOut* exception. Set this only if you wish to override, on this call only, the value set during the geocoder's initialization.

Return type `None`, *geopy.location.Location* or a list of them, if `exactly_one=False`.

class geopy.geocoders.**Mapzen**(*api_key=None*, *format_string=None*, *boundary_rect=None*, *country_bias=None*, *timeout=DEFAULT_SENTINEL*, *proxies=DEFAULT_SENTINEL*, *user_agent=None*, *domain='search.mapzen.com'*, *scheme=None*, *ssl_context=DEFAULT_SENTINEL*)

Mapzen Search geocoder.

Documentation at: https://mapzen.com/documentation/search/

Warning: Please note that Mapzen has shut down their API so this geocoder class might be removed in future releases.

__init__(*api_key=None*, *format_string=None*, *boundary_rect=None*, *country_bias=None*, *timeout=DEFAULT_SENTINEL*, *proxies=DEFAULT_SENTINEL*, *user_agent=None*, *domain='search.mapzen.com'*, *scheme=None*, *ssl_context=DEFAULT_SENTINEL*)

Parameters

- **api_key** (*str*) – Mapzen API key, optional.
- **format_string** (*str*) – See *geopy.geocoders.options.default_format_string*.
- **boundary_rect** (*tuple*) – Coordinates to restrict search within, given as (west, south, east, north) coordinate tuple.
- **country_bias** (*str*) – Bias results to this country (ISO alpha-3).
- **timeout** (*int*) – See *geopy.geocoders.options.default_timeout*.
- **proxies** (*dict*) – See *geopy.geocoders.options.default_proxies*.
- **user_agent** (*str*) – See *geopy.geocoders.options.default_user_agent*.

  New in version 1.12.0.

- **domain** (*str*) – Specify a custom domain for Mapzen API.
- **scheme** (*str*) – See *geopy.geocoders.options.default_scheme*.
- **ssl_context** (`ssl.SSLContext`) – See *geopy.geocoders.options.default_ssl_context*.

  New in version 1.14.0.

geocode(*query*, *exactly_one=True*, *timeout=DEFAULT_SENTINEL*)

Return a location point by address.

Parameters

- **query** (*str*) – The address, query or structured query to geocode you wish to geocode.
- **exactly_one** (*bool*) – Return one result or a list of results, if available.
- **timeout** (*int*) – Time, in seconds, to wait for the geocoding service to respond before raising a *geopy.exc.GeocoderTimedOut* exception. Set this only if you wish to override, on this call only, the value set during the geocoder's initialization.

Return type None, *geopy.location.Location* or a list of them, if `exactly_one=False`.

reverse(*query*, *exactly_one=True*, *timeout=DEFAULT_SENTINEL*)

Return an address by location point.

Parameters

- **query** (*geopy.point.Point*, list or tuple of `(latitude, longitude)`, or string as `"%(latitude)s, %(longitude)s"`.) – The coordinates for which you wish to obtain the closest human-readable addresses.
- **exactly_one** (*bool*) – Return one result or a list of results, if available.
- **timeout** (*int*) – Time, in seconds, to wait for the geocoding service to respond before raising a *geopy.exc.GeocoderTimedOut* exception. Set this only if you wish to override, on this call only, the value set during the geocoder's initialization.

Return type None, *geopy.location.Location* or a list of them, if `exactly_one=False`.

class geopy.geocoders.**OpenCage**(*api_key*, *domain='api.opencagedata.com'*, *scheme=None*, *timeout=DEFAULT_SENTINEL*, *proxies=DEFAULT_SENTINEL*, *user_agent=None*, *format_string=None*, *ssl_context=DEFAULT_SENTINEL*)

Geocoder using the OpenCageData API.

Documentation at: https://geocoder.opencagedata.com/api

New in version 1.1.0.

__init__(*api_key*, *domain='api.opencagedata.com'*, *scheme=None*, *timeout=DEFAULT_SENTINEL*, *proxies=DEFAULT_SENTINEL*, *user_agent=None*, *format_string=None*, *ssl_context=DEFAULT_SENTINEL*)

Parameters

- **api_key** (*str*) – The API key required by OpenCageData to perform geocoding requests. You can get your key here: https://geocoder.opencagedata.com/
- **domain** (*str*) – Currently it is `'api.opencagedata.com'`, can be changed for testing purposes.
- **scheme** (*str*) – See *geopy.geocoders.options.default_scheme*.
- **timeout** (*int*) – See *geopy.geocoders.options.default_timeout*.
- **proxies** (*dict*) – See *geopy.geocoders.options.default_proxies*.
- **user_agent** (*str*) – See *geopy.geocoders.options.default_user_agent*.

  New in version 1.12.0.
- **format_string** (*str*) – See *geopy.geocoders.options.default_format_string*.

  New in version 1.14.0.

- **ssl_context** (ssl.SSLContext) – See *geopy.geocoders.options.default_ssl_context*.

  New in version 1.14.0.

geocode(*query*, *bounds=None*, *country=None*, *language=None*, *exactly_one=True*, *timeout=DEFAULT_SENTINEL*)
Return a location point by address.

Parameters

- **query** (*str*) – The address or query you wish to geocode.
- **language** (*str*) – an IETF format language code (such as *es* for Spanish or pt-BR for Brazilian Portuguese); if this is omitted a code of *en* (English) will be assumed by the remote service.
- **bounds** (*str*) – Provides the geocoder with a hint to the region that the query resides in. This value will help the geocoder but will not restrict the possible results to the supplied region. The bounds parameter should be specified as 4 coordinate points forming the south-west and north-east corners of a bounding box. The order of the coordinates is *longitude,latitude,longitude,latitude*. For example, `bounds=-0.563160,51.280430,0.278970,51.683979`.
- **country** (*str*) – Provides the geocoder with a hint to the country that the query resides in. This value will help the geocoder but will not restrict the possible results to the supplied country. The country code is a 3 character code as defined by the ISO 3166-1 Alpha 3 standard.
- **exactly_one** (*bool*) – Return one result or a list of results, if available.
- **timeout** (*int*) – Time, in seconds, to wait for the geocoding service to respond before raising a *geopy.exc.GeocoderTimedOut* exception. Set this only if you wish to override, on this call only, the value set during the geocoder's initialization.

Return type `None`, *geopy.location.Location* or a list of them, if `exactly_one=False`.

reverse(*query*, *language=None*, *exactly_one=DEFAULT_SENTINEL*, *timeout=DEFAULT_SENTINEL*)
Return an address by location point.

Parameters

- **query** (*geopy.point.Point*, list or tuple of `(latitude, longitude)`, or string as `"%(latitude)s, %(longitude)s"`.) – The coordinates for which you wish to obtain the closest human-readable addresses.
- **language** (*str*) – The language in which to return results.
- **exactly_one** (*bool*) – Return one result or a list of results, if available.

  Changed in version 1.14.0: Default value for `exactly_one` was `False`, which differs from the conventional default across geopy. Please always pass this argument explicitly, otherwise you would get a warning. In geopy 2.0 the default value will become `True`.
- **timeout** (*int*) – Time, in seconds, to wait for the geocoding service to respond before raising a *geopy.exc.GeocoderTimedOut* exception. Set this only if you wish to override, on this call only, the value set during the geocoder's initialization.

Return type `None`, *geopy.location.Location* or a list of them, if `exactly_one=False`.

class geopy.geocoders.**OpenMapQuest**(*api_key=None*, *format_string=None*, *scheme=None*, *timeout=DEFAULT_SENTINEL*, *proxies=DEFAULT_SENTINEL*, *user_agent=None*, *ssl_context=DEFAULT_SENTINEL*)

Geocoder using MapQuest Open Platform Web Services.

Documentation at: https://developer.mapquest.com/documentation/open/

__init__(*api_key=None*, *format_string=None*, *scheme=None*, *timeout=DEFAULT_SENTINEL*, *proxies=DEFAULT_SENTINEL*, *user_agent=None*, *ssl_context=DEFAULT_SENTINEL*)

Parameters

- **api_key** (*str*) – API key provided by MapQuest, required.

  Changed in version 1.12.0: OpenMapQuest now requires an API key. Using an empty key will result in a *geopy.exc.ConfigurationError*.

- **format_string** (*str*) – See *geopy.geocoders.options.default_format_string*.
- **scheme** (*str*) – See *geopy.geocoders.options.default_scheme*.
- **timeout** (*int*) – See *geopy.geocoders.options.default_timeout*.
- **proxies** (*dict*) – See *geopy.geocoders.options.default_proxies*.
- **user_agent** (*str*) – See *geopy.geocoders.options.default_user_agent*.

  New in version 1.12.0.

- **ssl_context** (ssl.SSLContext) – See *geopy.geocoders.options.default_ssl_context*.

  New in version 1.14.0.

geocode(*query*, *exactly_one=True*, *timeout=DEFAULT_SENTINEL*)

Return a location point by address.

Parameters

- **query** (*str*) – The address or query you wish to geocode.
- **exactly_one** (*bool*) – Return one result or a list of results, if available.
- **timeout** (*int*) – Time, in seconds, to wait for the geocoding service to respond before raising a *geopy.exc.GeocoderTimedOut* exception. Set this only if you wish to override, on this call only, the value set during the geocoder's initialization.

Return type None, *geopy.location.Location* or a list of them, if exactly_one=False.

class geopy.geocoders.**Nominatim**(*format_string=None*, *view_box=None*, *country_bias=None*, *timeout=DEFAULT_SENTINEL*, *proxies=DEFAULT_SENTINEL*, *domain='nominatim.openstreetmap.org'*, *scheme=None*, *user_agent=None*, *ssl_context=DEFAULT_SENTINEL*)

Nominatim geocoder for OpenStreetMap servers.

Documentation at: https://wiki.openstreetmap.org/wiki/Nominatim

Attention: Nominatim requires each application to provide their own custom user-agent: `geolocator = Nominatim(user_agent="my-application")`. Nominatim usage policy: https://operations.osmfoundation.org/policies/nominatim/

__init__(*format_string=None*, *view_box=None*, *country_bias=None*, *timeout=DEFAULT_SENTINEL*, *proxies=DEFAULT_SENTINEL*, *domain='nominatim.openstreetmap.org'*, *scheme=None*, *user_agent=None*, *ssl_context=DEFAULT_SENTINEL*)

Parameters

- **format_string** (*str*) – See *geopy.geocoders.options.default_format_string*.
- **view_box** (*tuple*) – Coordinates to restrict search within.
- **country_bias** (*str*) – Bias results to this country.
- **timeout** (*int*) – See *geopy.geocoders.options.default_timeout*.
- **proxies** (*dict*) – See *geopy.geocoders.options.default_proxies*.
- **domain** (*str*) – Should be the localized Openstreetmap domain to connect to. The default is `'nominatim.openstreetmap.org'`, but you can change it to a domain of your own.

  New in version 1.8.2.
- **scheme** (*str*) – See *geopy.geocoders.options.default_scheme*.

  New in version 1.8.2.
- **user_agent** (*str*) – See *geopy.geocoders.options.default_user_agent*.

  New in version 1.12.0.
- **ssl_context** (`ssl.SSLContext`) – See *geopy.geocoders.options.default_ssl_context*.

  New in version 1.14.0.

geocode(*query*, *exactly_one=True*, *timeout=DEFAULT_SENTINEL*, *limit=None*, *addressdetails=False*, *language=False*, *geometry=None*)

Return a location point by address.

Parameters

- **query** (*dict or str*) – The address, query or a structured query you wish to geocode.

  Changed in version 1.0.0: For a structured query, provide a dictionary whose keys are one of: *street*, *city*, *county*, *state*, *country*, or *postalcode*. For more information, see Nominatim's documentation for *structured requests*:

  https://wiki.openstreetmap.org/wiki/Nominatim
- **exactly_one** (*bool*) – Return one result or a list of results, if available.
- **timeout** (*int*) – Time, in seconds, to wait for the geocoding service to respond before raising a *geopy.exc.GeocoderTimedOut* exception. Set this only if you wish to override, on this call only, the value set during the geocoder's initialization.
- **limit** (*int*) – Maximum amount of results to return from Nominatim. Unless exactly_one is set to False, limit will always be 1.

New in version 1.13.0.

- **addressdetails** (*bool*) – If you want in *Location.raw* to include addressdetails such as city_district, etc set it to True
- **language** (*str*) – Preferred language in which to return results. Either uses standard RFC2616 accept-language string or a simple comma-separated list of language codes.

  New in version 1.0.0.
- **geometry** (*str*) – If present, specifies whether the geocoding service should return the result's geometry in *wkt*, *svg*, *kml*, or *geojson* formats. This is available via the *raw* attribute on the returned *geopy.location.Location* object.

  New in version 1.3.0.

Return type None, *geopy.location.Location* or a list of them, if `exactly_one=False`.

reverse(*query*, *exactly_one=True*, *timeout=DEFAULT_SENTINEL*, *language=False*, *addressdetails=True*)

Return an address by location point.

Parameters

- **query** (*geopy.point.Point*, list or tuple of `(latitude, longitude)`, or string as `"%(latitude)s, %(longitude)s"`.) – The coordinates for which you wish to obtain the closest human-readable addresses.
- **exactly_one** (*bool*) – Return one result or a list of results, if available.
- **timeout** (*int*) – Time, in seconds, to wait for the geocoding service to respond before raising a *geopy.exc.GeocoderTimedOut* exception. Set this only if you wish to override, on this call only, the value set during the geocoder's initialization.
- **language** (*str*) – Preferred language in which to return results. Either uses standard RFC2616 accept-language string or a simple comma-separated list of language codes.

  New in version 1.0.0.
- **addressdetails** (*bool*) – Whether or not to include address details, such as city, county, state, etc. in *Location.raw*

  New in version 1.14.0.

Return type None, *geopy.location.Location* or a list of them, if `exactly_one=False`.

class geopy.geocoders.**Photon**(*format_string=None*, *scheme=None*, *timeout=DEFAULT_SENTINEL*, *proxies=DEFAULT_SENTINEL*, *domain='photon.komoot.de'*, *user_agent=None*, *ssl_context=DEFAULT_SENTINEL*)

Geocoder using Photon geocoding service (data based on OpenStreetMap and service provided by Komoot on https://photon.komoot.de).

Documentation at: https://github.com/komoot/photon

Photon/Komoot geocoder aims to let you *search as you type with OpenStreetMap*. No API Key is needed by this platform.

__init__(*format_string=None*, *scheme=None*, *timeout=DEFAULT_SENTINEL*, *proxies=DEFAULT_SENTINEL*, *domain='photon.komoot.de'*, *user_agent=None*, *ssl_context=DEFAULT_SENTINEL*)

Parameters

- **format_string** (*str*) – See *geopy.geocoders.options.default_format_string*.
- **scheme** (*str*) – See *geopy.geocoders.options.default_scheme*.
- **timeout** (*int*) – See *geopy.geocoders.options.default_timeout*.
- **proxies** (*dict*) – See *geopy.geocoders.options.default_proxies*.
- **domain** (*str*) – Should be the localized Photon domain to connect to. The default is `'photon.komoot.de'`, but you can change it to a domain of your own.
- **user_agent** (*str*) – See *geopy.geocoders.options.default_user_agent*.

  New in version 1.12.0.
- **ssl_context** (`ssl.SSLContext`) – See *geopy.geocoders.options.default_ssl_context*.

  New in version 1.14.0.

geocode(*query*, *exactly_one=True*, *timeout=DEFAULT_SENTINEL*, *location_bias=None*, *language=False*, *limit=None*, *osm_tag=None*)

Return a location point by address.

Parameters

- **query** (*str*) – The address or query you wish to geocode.
- **exactly_one** (*bool*) – Return one result or a list of results, if available.
- **timeout** (*int*) – Time, in seconds, to wait for the geocoding service to respond before raising a *geopy.exc.GeocoderTimedOut* exception. Set this only if you wish to override, on this call only, the value set during the geocoder's initialization.
- **location_bias** – The coordinates to used as location bias.
- **language** (*str*) – Preferred language in which to return results.
- **limit** (*int*) – Limit the number of returned results, defaults to no limit.

  New in version 1.12.0.
- **osm_tag** (*str or list or set*) – The expression to filter (include/exclude) by key and/ or value, str as `'key:value'` or list/set of str if multiple filters are required as `['key:!val', '!key', ':!value']`.

Return type `None`, *geopy.location.Location* or a list of them, if `exactly_one=False`.

reverse(*query*, *exactly_one=True*, *timeout=DEFAULT_SENTINEL*, *language=False*, *limit=None*)

Return an address by location point.

Parameters

- **query** (*geopy.point.Point*, list or tuple of `(latitude, longitude)`, or string as `"%(latitude)s, %(longitude)s"`.) – The coordinates for which you wish to obtain the closest human-readable addresses.
- **exactly_one** (*bool*) – Return one result or a list of results, if available.
- **timeout** (*int*) – Time, in seconds, to wait for the geocoding service to respond before raising a *geopy.exc.GeocoderTimedOut* exception. Set this only if you wish to override, on this call only, the value set during the geocoder's initialization.
- **language** (*str*) – Preferred language in which to return results.

- **limit** (*int*) – Limit the number of returned results, defaults to no limit.

  New in version 1.12.0.

Return type None, *geopy.location.Location* or a list of them, if `exactly_one=False`.

class geopy.geocoders.**PickPoint**(*api_key*, *format_string=None*, *view_box=None*, *country_bias=None*, *timeout=DEFAULT_SENTINEL*, *proxies=DEFAULT_SENTINEL*, *domain='api.pickpoint.io'*, *scheme=None*, *user_agent=None*, *ssl_context=DEFAULT_SENTINEL*)

Bases: geopy.geocoders.osm.Nominatim

PickPoint geocoder is a commercial version of Nominatim.

Documentation at: https://pickpoint.io/api-reference

New in version 1.13.0.

__init__(*api_key*, *format_string=None*, *view_box=None*, *country_bias=None*, *timeout=DEFAULT_SENTINEL*, *proxies=DEFAULT_SENTINEL*, *domain='api.pickpoint.io'*, *scheme=None*, *user_agent=None*, *ssl_context=DEFAULT_SENTINEL*)

Parameters

- **api_key** (*string*) – PickPoint API key obtained at https://pickpoint.io.
- **format_string** (*str*) – See *geopy.geocoders.options.default_format_string*.
- **view_box** (*tuple*) – Coordinates to restrict search within.
- **country_bias** (*string*) – Bias results to this country.
- **timeout** (*int*) – See *geopy.geocoders.options.default_timeout*.
- **proxies** (*dict*) – See *geopy.geocoders.options.default_proxies*.
- **domain** (*str*) – Should be the localized Openstreetmap domain to connect to. The default is `'api.pickpoint.io'`, but you can change it to a domain of your own.
- **scheme** (*str*) – See *geopy.geocoders.options.default_scheme*.
- **user_agent** (*str*) – See *geopy.geocoders.options.default_user_agent*.
- **ssl_context** (ssl.SSLContext) – See *geopy.geocoders.options.default_ssl_context*.

  New in version 1.14.0.

geocode(*query*, *exactly_one=True*, *timeout=DEFAULT_SENTINEL*, *limit=None*, *addressdetails=False*, *language=False*, *geometry=None*)

Return a location point by address.

Parameters

- **query** (*dict or str*) – The address, query or a structured query you wish to geocode.

  Changed in version 1.0.0: For a structured query, provide a dictionary whose keys are one of: *street*, *city*, *county*, *state*, *country*, or *postalcode*. For more information, see Nominatim's documentation for *structured requests*:

  https://wiki.openstreetmap.org/wiki/Nominatim

- **exactly_one** (*bool*) – Return one result or a list of results, if available.

- **timeout** (*int*) – Time, in seconds, to wait for the geocoding service to respond before raising a *geopy.exc.GeocoderTimedOut* exception. Set this only if you wish to override, on this call only, the value set during the geocoder's initialization.
- **limit** (*int*) – Maximum amount of results to return from Nominatim. Unless exactly_one is set to False, limit will always be 1.

  New in version 1.13.0.
- **addressdetails** (*bool*) – If you want in *Location.raw* to include addressdetails such as city_district, etc set it to True
- **language** (*str*) – Preferred language in which to return results. Either uses standard RFC2616 accept-language string or a simple comma-separated list of language codes.

  New in version 1.0.0.
- **geometry** (*str*) – If present, specifies whether the geocoding service should return the result's geometry in *wkt*, *svg*, *kml*, or *geojson* formats. This is available via the *raw* attribute on the returned *geopy.location.Location* object.

  New in version 1.3.0.

Return type None, *geopy.location.Location* or a list of them, if `exactly_one=False`.

reverse(*query*, *exactly_one=True*, *timeout=DEFAULT_SENTINEL*, *language=False*, *addressdetails=True*)

Return an address by location point.

Parameters

- **query** (*geopy.point.Point*, list or tuple of `(latitude, longitude)`, or string as `"%(latitude)s, %(longitude)s"`.) – The coordinates for which you wish to obtain the closest human-readable addresses.
- **exactly_one** (*bool*) – Return one result or a list of results, if available.
- **timeout** (*int*) – Time, in seconds, to wait for the geocoding service to respond before raising a *geopy.exc.GeocoderTimedOut* exception. Set this only if you wish to override, on this call only, the value set during the geocoder's initialization.
- **language** (*str*) – Preferred language in which to return results. Either uses standard RFC2616 accept-language string or a simple comma-separated list of language codes.

  New in version 1.0.0.
- **addressdetails** (*bool*) – Whether or not to include address details, such as city, county, state, etc. in *Location.raw*

  New in version 1.14.0.

Return type None, *geopy.location.Location* or a list of them, if `exactly_one=False`.

class geopy.geocoders.**LiveAddress**(*auth_id*, *auth_token*, *candidates=1*, *scheme='https'*, *timeout=DEFAULT_SENTINEL*, *proxies=DEFAULT_SENTINEL*, *user_agent=None*, *format_string=None*, *ssl_context=DEFAULT_SENTINEL*)

Geocoder using the LiveAddress API provided by SmartyStreets.

Documentation at: https://smartystreets.com/docs/cloud/us-street-api

__init__(*auth_id*, *auth_token*, *candidates=1*, *scheme='https'*, *timeout=DEFAULT_SENTINEL*, *proxies=DEFAULT_SENTINEL*, *user_agent=None*, *format_string=None*, *ssl_context=DEFAULT_SENTINEL*)

Parameters

- **auth_id** (*str*) – Valid *Auth ID* from SmartyStreets.

  New in version 1.5.0.

- **auth_token** (*str*) – Valid *Auth Token* from SmartyStreets.

- **candidates** (*int*) – An integer between 1 and 10 indicating the max number of candidate addresses to return if a valid address could be found.

- **scheme** (*str*) – Must be `https`.

  Deprecated since version 1.14.0: Don't use this parameter, it's going to be removed in the future versions of geopy.

  Changed in version 1.8.0: LiveAddress now requires *https*. Specifying *scheme=http* will result in a *geopy.exc.ConfigurationError*.

- **timeout** (*int*) – See *geopy.geocoders.options.default_timeout*.

- **proxies** (*dict*) – See *geopy.geocoders.options.default_proxies*.

- **user_agent** (*str*) – See *geopy.geocoders.options.default_user_agent*.

  New in version 1.12.0.

- **format_string** (*str*) – See *geopy.geocoders.options.default_format_string*.

  New in version 1.14.0.

- **ssl_context** (`ssl.SSLContext`) – See *geopy.geocoders.options.default_ssl_context*.

  New in version 1.14.0.

geocode(*query*, *exactly_one=True*, *timeout=DEFAULT_SENTINEL*)

Return a location point by address.

Parameters

- **query** (*str*) – The address or query you wish to geocode.

- **exactly_one** (*bool*) – Return one result or a list of results, if available.

- **timeout** (*int*) – Time, in seconds, to wait for the geocoding service to respond before raising a *geopy.exc.GeocoderTimedOut* exception. Set this only if you wish to override, on this call only, the value set during the geocoder's initialization.

Return type `None`, *geopy.location.Location* or a list of them, if `exactly_one=False`.

class geopy.geocoders.**What3Words**(*api_key*, *format_string=None*, *scheme=None*, *timeout=DEFAULT_SENTINEL*, *proxies=DEFAULT_SENTINEL*, *user_agent=None*, *ssl_context=DEFAULT_SENTINEL*)

What3Words geocoder.

Documentation at: https://what3words.com/api/reference

New in version 1.5.0.

__init__(*api_key*, *format_string=None*, *scheme=None*, *timeout=DEFAULT_SENTINEL*, *proxies=DEFAULT_SENTINEL*, *user_agent=None*, *ssl_context=DEFAULT_SENTINEL*)

Parameters

- **api_key** (*str*) – Key provided by What3Words (https://accounts.what3words.com/register).
- **format_string** (*str*) – See *geopy.geocoders.options.default_format_string*.
- **scheme** (*str*) – See *geopy.geocoders.options.default_scheme*.
- **timeout** (*int*) – See *geopy.geocoders.options.default_timeout*.
- **proxies** (*dict*) – See *geopy.geocoders.options.default_proxies*.
- **user_agent** (*str*) – See *geopy.geocoders.options.default_user_agent*.

  New in version 1.12.0.
- **ssl_context** (ssl.SSLContext) – See *geopy.geocoders.options.default_ssl_context*.

  New in version 1.14.0.

geocode(*query*, *lang='en'*, *exactly_one=True*, *timeout=DEFAULT_SENTINEL*)

Return a location point for a *3 words* query. If the *3 words* address doesn't exist, a *geopy.exc.GeocoderQueryError* exception will be thrown.

Parameters

- **query** (*str*) – The 3-word address you wish to geocode.
- **lang** (*str*) – two character language codes as supported by the API (https://docs.what3words.com/api/v2/#lang).
- **exactly_one** (*bool*) – Return one result or a list of results, if available. Due to the address scheme there is always exactly one result for each *3 words* address, so this parameter is rather useless for this geocoder.

  Changed in version 1.14.0: `exactly_one=False` now returns a list of a single location. This option wasn't respected before.
- **timeout** (*int*) – Time, in seconds, to wait for the geocoding service to respond before raising a *geopy.exc.GeocoderTimedOut* exception. Set this only if you wish to override, on this call only, the value set during the geocoder's initialization.

Return type *geopy.location.Location* or a list of them, if `exactly_one=False`.

reverse(*query*, *lang='en'*, *exactly_one=True*, *timeout=DEFAULT_SENTINEL*)

Return a *3 words* address by location point. Each point on surface has a *3 words* address, so there's always a non-empty response.

Parameters

- **query** (*geopy.point.Point*, list or tuple of `(latitude, longitude)`, or string as `"%(latitude)s, %(longitude)s"`.) – The coordinates for which you wish to obtain the 3 word address.
- **lang** (*str*) – two character language codes as supported by the API (https://docs.what3words.com/api/v2/#lang).

- **exactly_one** (*bool*) – Return one result or a list of results, if available. Due to the address scheme there is always exactly one result for each *3 words* address, so this parameter is rather useless for this geocoder.

  Changed in version 1.14.0: `exactly_one=False` now returns a list of a single location. This option wasn't respected before.

- **timeout** (*int*) – Time, in seconds, to wait for the geocoding service to respond before raising a *geopy.exc.GeocoderTimedOut* exception. Set this only if you wish to override, on this call only, the value set during the geocoder's initialization.

Return type *geopy.location.Location* or a list of them, if `exactly_one=False`.

class geopy.geocoders.**Yandex**(*api_key=None*, *lang=None*, *timeout=DEFAULT_SENTINEL*, *proxies=DEFAULT_SENTINEL*, *user_agent=None*, *scheme=None*, *format_string=None*, *ssl_context=DEFAULT_SENTINEL*)

Yandex geocoder.

Documentation at: https://tech.yandex.com/maps/doc/geocoder/desc/concepts/input_params-docpage/

New in version 1.5.0.

__init__(*api_key=None*, *lang=None*, *timeout=DEFAULT_SENTINEL*, *proxies=DEFAULT_SENTINEL*, *user_agent=None*, *scheme=None*, *format_string=None*, *ssl_context=DEFAULT_SENTINEL*)

Changed in version 1.14.0: Default scheme has been changed from `http` to `https`.

Parameters

- **api_key** (*str*) – Yandex API key (not obligatory) http://api.yandex.ru/maps/form.xml
- **lang** (*str*) – response locale, the following locales are supported: `"ru_RU"` (default), `"uk_UA"`, `"be_BY"`, `"en_US"`, `"tr_TR"`.
- **timeout** (*int*) – See *geopy.geocoders.options.default_timeout*.
- **proxies** (*dict*) – See *geopy.geocoders.options.default_proxies*.
- **user_agent** (*str*) – See *geopy.geocoders.options.default_user_agent*.

  New in version 1.12.0.

- **scheme** (*str*) – See *geopy.geocoders.options.default_scheme*.

  New in version 1.14.0.

- **format_string** (*str*) – See *geopy.geocoders.options.default_format_string*.

  New in version 1.14.0.

- **ssl_context** (`ssl.SSLContext`) – See *geopy.geocoders.options.default_ssl_context*.

  New in version 1.14.0.

geocode(*query*, *exactly_one=True*, *timeout=DEFAULT_SENTINEL*)

Return a location point by address.

Parameters

- **query** (*str*) – The address or query you wish to geocode.
- **exactly_one** (*bool*) – Return one result or a list of results, if available.

- **timeout** (*int*) – Time, in seconds, to wait for the geocoding service to respond before raising a *geopy.exc.GeocoderTimedOut* exception. Set this only if you wish to override, on this call only, the value set during the geocoder's initialization.

Return type None, *geopy.location.Location* or a list of them, if `exactly_one=False`.

reverse(*query*, *exactly_one=DEFAULT_SENTINEL*, *timeout=DEFAULT_SENTINEL*, *kind=None*)

Return an address by location point.

Parameters

- **query** (*geopy.point.Point*, list or tuple of `(latitude, longitude)`, or string as `"%(latitude)s, %(longitude)s"`.) – The coordinates for which you wish to obtain the closest human-readable addresses.
- **exactly_one** (*bool*) – Return one result or a list of results, if available.

  Changed in version 1.14.0: Default value for `exactly_one` was `False`, which differs from the conventional default across geopy. Please always pass this argument explicitly, otherwise you would get a warning. In geopy 2.0 the default value will become `True`.
- **timeout** (*int*) – Time, in seconds, to wait for the geocoding service to respond before raising a *geopy.exc.GeocoderTimedOut* exception. Set this only if you wish to override, on this call only, the value set during the geocoder's initialization.
- **kind** (*str*) – Type of toponym. Allowed values: *house*, *street*, *metro*, *district*, *locality*.

  New in version 1.14.0.

Return type None, *geopy.location.Location* or a list of them, if `exactly_one=False`.

CHAPTER 2

Calculating Distance

Geopy can calculate geodesic distance between two points using the geodesic distance or the great-circle distance, with a default of the geodesic distance available as the function `geopy.distance.distance`.

Great-circle distance (*`great_circle`*) uses a spherical model of the earth, using the mean earth radius as defined by the International Union of Geodesy and Geophysics, $(2a + b)/3$ = 6371.0087714150598 kilometers approx 6371.009 km (for WGS-84), resulting in an error of up to about 0.5%. The radius value is stored in `distance.EARTH_RADIUS`, so it can be customized (it should always be in kilometers, however).

The geodesic distance is the shortest distance on the surface of an ellipsoidal model of the earth. The default algorithm uses the method is given by Karney (2013) (*`geodesic`*); this is accurate to round-off and always converges. An older *deprecated* method due to Vincenty (1975) (*`vincenty`*) is also available; this is only accurate to 0.2 mm and the distance calculation fails to converge for nearly antipodal points.

`geopy.distance.distance` currently uses *`geodesic`*.

There are multiple popular ellipsoidal models, and which one will be the most accurate depends on where your points are located on the earth. The default is the WGS-84 ellipsoid, which is the most globally accurate. geopy includes a few other models in the `distance.ELLIPSOIDS` dictionary:

```
             model             major (km)   minor (km)     flattening
ELLIPSOIDS = {'WGS-84':        (6378.137,    6356.7523142,  1 / 298.257223563),
              'GRS-80':        (6378.137,    6356.7523141,  1 / 298.257222101),
              'Airy (1830)':   (6377.563396, 6356.256909,   1 / 299.3249646),
              'Intl 1924':     (6378.388,    6356.911946,   1 / 297.0),
              'Clarke (1880)': (6378.249145, 6356.51486955, 1 / 293.465),
              'GRS-67':        (6378.1600,   6356.774719,   1 / 298.25),
              }
```

Here are examples of `distance.distance` usage:

```
>>> from geopy import distance
>>> newport_ri = (41.49008, -71.312796)
>>> cleveland_oh = (41.499498, -81.695391)
>>> print(distance.distance(newport_ri, cleveland_oh).miles)
538.39044536
```

(continues on next page)

(continued from previous page)

```
>>> wellington = (-41.32, 174.81)
>>> salamanca = (40.96, -5.50)
>>> print(distance.distance(wellington, salamanca).km)
19959.6792674
```

The second example above fails with *vincenty*.

Using *great_circle* distance:

```
>>> print(distance.great_circle(newport_ri, cleveland_oh).miles)
536.997990696
```

You can change the ellipsoid model used by the geodesic formulas like so:

```
>>> ne, cl = newport_ri, cleveland_oh
>>> print(distance.geodesic(ne, cl, ellipsoid='GRS-80').miles)
```

The above model name will automatically be retrieved from the `distance.ELLIPSOIDS` dictionary. Alternatively, you can specify the model values directly:

```
>>> distance.geodesic(ne, cl, ellipsoid=(6377., 6356., 1 / 297.)).miles
```

Distances support simple arithmetic, making it easy to do things like calculate the length of a path:

```
>>> from geopy import Nominatim
>>> d = distance.distance
>>> g = Nominatim()
>>> _, wa = g.geocode('Washington, DC')
>>> _, pa = g.geocode('Palo Alto, CA')
>>> print((d(ne, cl) + d(cl, wa) + d(wa, pa)).miles)
3277.30439191
```

geopy.distance.**lonlat**(*x*, *y*, *z=0*)

`geopy.distance.distance` accepts coordinates in `(y, x)/(lat, lon)` order, while some other libraries and systems might use `(x, y)/(lon, lat)`.

This function provides a convenient way to convert coordinates of the `(x, y)/(lon, lat)` format to a *geopy.point.Point* instance.

Example:

```
>>> from geopy.distance import lonlat, distance
>>> newport_ri_xy = (-71.312796, 41.49008)
>>> cleveland_oh_xy = (-81.695391, 41.499498)
>>> print(distance(lonlat(*newport_ri_xy), lonlat(*cleveland_oh_xy)).miles)
538.3904453677203
```

Parameters

- **x** – longitude
- **y** – latitude
- **z** – (optional) altitude

Returns Point(latitude, longitude, altitude)

class geopy.distance.**geodesic**(**args*, ***kwargs*)

Calculate the geodesic distance between two points.

Set which ellipsoidal model of the earth to use by specifying an `ellipsoid` keyword argument. The default is 'WGS-84', which is the most globally accurate model. If `ellipsoid` is a string, it is looked up in the *ELLIPSOIDS* dictionary to obtain the major and minor semiaxes and the flattening. Otherwise, it should be a tuple with those values. See the comments above the *ELLIPSOIDS* dictionary for more information.

Example:

```
>>> from geopy.distance import geodesic
>>> newport_ri = (41.49008, -71.312796)
>>> cleveland_oh = (41.499498, -81.695391)
>>> print(geodesic(newport_ri, cleveland_oh).miles)
538.390445368
```

New in version 1.13.0.

__init__(**args*, ***kwargs*)

Initialize self. See help(type(self)) for accurate signature.

class geopy.distance.**vincenty**(**args*, ***kwargs*)

Deprecated since version 1.13: Use *geodesic* instead.

Calculate the geodesic distance between two points using the Vincenty's method.

Set which ellipsoidal model of the earth to use by specifying an `ellipsoid` keyword argument. The default is 'WGS-84', which is the most globally accurate model. If `ellipsoid` is a string, it is looked up in the *ELLIPSOIDS* dictionary to obtain the major and minor semiaxes and the flattening. Otherwise, it should be a tuple with those values. See the comments above the *ELLIPSOIDS* dictionary for more information.

Example:

```
>>> from geopy.distance import vincenty
>>> newport_ri = (41.49008, -71.312796)
>>> cleveland_oh = (41.499498, -81.695391)
>>> print(vincenty(newport_ri, cleveland_oh).miles)
538.390445362
```

Note: Vincenty's method for distance fails to converge for some valid (nearly antipodal) points. In such cases, use *geodesic* which always produces an accurate result.

__init__(**args*, ***kwargs*)

Initialize self. See help(type(self)) for accurate signature.

class geopy.distance.**great_circle**(**args*, ***kwargs*)

Use spherical geometry to calculate the surface distance between two points.

Set which radius of the earth to use by specifying a `radius` keyword argument. It must be in kilometers. The default is to use the module constant *EARTH_RADIUS*, which uses the average great-circle radius.

Example:

```
>>> from geopy.distance import great_circle
>>> newport_ri = (41.49008, -71.312796)
>>> cleveland_oh = (41.499498, -81.695391)
>>> print(great_circle(newport_ri, cleveland_oh).miles)
536.997990696
```

__init__(**args*, ***kwargs*)

Initialize self. See help(type(self)) for accurate signature.

CHAPTER 3

Data

class geopy.location.**Location**(*address=''*, *point=None*, *raw=None*)

Contains a parsed geocoder response. Can be iterated over as `(location<String>, (latitude<float>, longitude<Float))`. Or one can access the properties `address`, `latitude`, `longitude`, or `raw`. The last is a dictionary of the geocoder's response for this item.

__init__(*address=''*, *point=None*, *raw=None*)

Initialize self. See help(type(self)) for accurate signature.

address

Location as a formatted string returned by the geocoder or constructed by geopy, depending on the service.

Return type unicode

altitude

Location's altitude.

Return type float or None

latitude

Location's latitude.

Return type float or None

longitude

Location's longitude.

Return type float or None

raw

Location's raw, unparsed geocoder response. For details on this, consult the service's documentation.

Return type dict or None

class geopy.point.**Point**

A geodetic point with latitude, longitude, and altitude.

Latitude and longitude are floating point values in degrees. Altitude is a floating point value in kilometers. The reference level is never considered and is thus application dependent, so be consistent! The default for all values is 0.

Points can be created in a number of ways...

With latitude, longitude, and altitude:

```
>>> p1 = Point(41.5, -81, 0)
>>> p2 = Point(latitude=41.5, longitude=-81)
```

With a sequence of 0 to 3 values (latitude, longitude, altitude):

```
>>> p1 = Point([41.5, -81, 0])
>>> p2 = Point((41.5, -81))
```

Copy another *Point* instance:

```
>>> p2 = Point(p1)
>>> p2 == p1
True
>>> p2 is p1
False
```

Give a string containing at least latitude and longitude:

```
>>> p1 = Point('41.5,-81.0')
>>> p2 = Point('41.5 N -81.0 W')
>>> p3 = Point('-41.5 S, 81.0 E, 2.5km')
>>> p4 = Point('23 26m 22s N 23 27m 30s E 21.0mi')
>>> p5 = Point('''3 26' 22" N 23 27' 30" E''')
```

Point values can be accessed by name or by index:

```
>>> p = Point(41.5, -81.0, 0)
>>> p.latitude == p[0]
True
>>> p.longitude == p[1]
True
>>> p.altitude == p[2]
True
```

When unpacking (or iterating), a `(latitude, longitude, altitude)` tuple is returned:

```
>>> latitude, longitude, altitude = p
```

static __new__(*cls*, *latitude=None*, *longitude=None*, *altitude=None*)

Parameters

- **latitude** (*float*) – Latitude of point.
- **longitude** (*float*) – Longitude of point.
- **altitude** (*float*) – Altitude of point.

classmethod from_point(*point*)

Create and return a new `Point` instance from another `Point` instance.

classmethod from_sequence(*seq*)

Create and return a new `Point` instance from any iterable with 0 to 3 elements. The elements, if present, must be latitude, longitude, and altitude, respectively.

classmethod from_string(*string*)

Create and return a `Point` instance from a string containing latitude and longitude, and optionally, altitude.

Latitude and longitude must be in degrees and may be in decimal form or indicate arcminutes and arcseconds (labeled with Unicode prime and double prime, ASCII quote and double quote or 'm' and 's'). The degree symbol is optional and may be included after the decimal places (in decimal form) and before the arcminutes and arcseconds otherwise. Coordinates given from south and west (indicated by S and W suffixes) will be converted to north and east by switching their signs. If no (or partial) cardinal directions are given, north and east are the assumed directions. Latitude and longitude must be separated by at least whitespace, a comma, or a semicolon (each with optional surrounding whitespace).

Altitude, if supplied, must be a decimal number with given units. The following unit abbrevations (case-insensitive) are supported:

- `km` (kilometers)
- `m` (meters)
- `mi` (miles)
- `ft` (feet)
- `nm`, `nmi` (nautical miles)

Some example strings that will work include:

- `41.5;-81.0`
- `41.5,-81.0`
- `41.5 -81.0`
- `41.5 N -81.0 W`
- `-41.5 S;81.0 E`
- `23 26m 22s N 23 27m 30s E`
- `23 26' 22" N 23 27' 30" E`
- `UT: N 39°20' 0'' / W 74°35' 0''`

CHAPTER 4

Exceptions

class geopy.exc.**GeopyError**
Bases: Exception

Geopy-specific exceptions are all inherited from GeopyError.

class geopy.exc.**ConfigurationError**
Bases: *geopy.exc.GeopyError*

When instantiating a geocoder, the arguments given were invalid. See the documentation of each geocoder's `__init__` for more details.

class geopy.exc.**GeocoderServiceError**
Bases: *geopy.exc.GeopyError*

There was an exception caused when calling the remote geocoding service, and no more specific exception could be raised by geopy. When calling geocoders' `geocode` or *reverse* methods, this is the most generic exception that can be raised, and any non-geopy exception will be caught and turned into this. The exception's message will be that of the original exception.

class geopy.exc.**GeocoderQueryError**
Bases: *geopy.exc.GeocoderServiceError*

Either geopy detected input that would cause a request to fail, or a request was made and the remote geocoding service responded that the request was bad.

class geopy.exc.**GeocoderQuotaExceeded**
Bases: *geopy.exc.GeocoderServiceError*

The remote geocoding service refused to fulfill the request because the client has used its quota.

class geopy.exc.**GeocoderAuthenticationFailure**
Bases: *geopy.exc.GeocoderServiceError*

The remote geocoding service rejected the API key or account credentials this geocoder was instantiated with.

class geopy.exc.**GeocoderInsufficientPrivileges**
Bases: *geopy.exc.GeocoderServiceError*

The remote geocoding service refused to fulfill a request using the account credentials given.

class geopy.exc.**GeocoderTimedOut**
Bases: *geopy.exc.GeocoderServiceError*

The call to the geocoding service was aborted because no response has been received within the `timeout` argument of either the geocoding class or, if specified, the method call. Some services are just consistently slow, and a higher timeout may be needed to use them.

class geopy.exc.**GeocoderUnavailable**
Bases: *geopy.exc.GeocoderServiceError*

Either it was not possible to establish a connection to the remote geocoding service, or the service responded with a code indicating it was unavailable.

class geopy.exc.**GeocoderParseError**
Bases: *geopy.exc.GeocoderServiceError*

Geopy could not parse the service's response. This is probably due to a bug in geopy.

class geopy.exc.**GeocoderNotFound**
Bases: *geopy.exc.GeopyError*

Caller requested the geocoder matching a string, e.g., `"google"` > `GoogleV3`, but no geocoder could be found.

CHAPTER 5

Logging

geopy will log geocoding URLs with a logger name `geopy` at level *DEBUG*, and for some geocoders, these URLs will include authentication information. Default logging level is *NOTSET*, which delegates the messages processing to the root logger. See docs for `logging.Logger.setLevel()` for more information. geopy does no logging above *DEBUG*.

CHAPTER 6

Semver

geopy attempts to follow semantic versioning, however some breaking changes are still being made in minor releases, such as:

- Backwards-incompatible changes of the undocumented API. This shouldn't affect anyone, unless they extend geocoder classes or use undocumented features or monkey-patch anything. If you believe that something is missing in geopy, please consider opening an issue or providing a patch or a PR instead of hacking around geopy.
- Geocoder classes which simply don't work (usually because their service has been discontinued) might get removed. They don't work anyway, so that's hardly a breaking change, right? :)
- Geocoding services sometimes introduce new APIs and deprecate the previous ones. We try to upgrade without breaking the geocoder's API interface, but the *geopy.location.Location.raw* value might change in a backwards-incompatible way.
- Behavior for invalid input and peculiar edge cases might be altered. For example, *geopy.point.Point* instances did coordinate values normalization, though it's not documented, and it was completely wrong for the latitudes outside the *[-90; 90]* range. So instead of using an incorrectly normalized value for latitude, an `ValueError` exception is now thrown (#294).

To make the upgrade less painful, please read the changelog before upgrading.

CHAPTER 7

Changelog

Changelog for 1.x.x series.

For changes in the 0.9 series, see the 0.9x changelog.

CHAPTER 8

Indices and search

- genindex
- search

Python Module Index

g

Index

Symbols

A

B

C

D

F

G

I

L

M

N

O

P

R

T

V

W

Y