Search queries

The search API allows you to look up a location from a textual description or address. Nominatim supports structured and free-form search queries.

The search query may also contain special phrases which are translated into specific OpenStreetMap (OSM) tags (e.g. Pub => `amenity=pub`). This can be used to narrow down the kind of objects to be returned.

> **Note**
>
> Special phrases are not suitable to query all objects of a certain type in an area. Nominatim will always just return a collection of the best matches. To download OSM data by object type, use the Overpass API.

Endpoint

The search API has the following format:

```
https://nominatim.openstreetmap.org/search?<params>
```

> **Deprecation warning**
>
> The API can also be used with the URL `https://nominatim.openstreetmap.org/search.php`. This is now deprecated and will be removed in future versions.

The query term can be given in two different forms: free-form or structured.

Free-form query

Parameter	Value
q	Free-form query string to search for

In this form, the query can be unstructured. Free-form queries are processed first left-to-right and then right-to-left if that fails. So you may search for pilkington avenue, birmingham as well as for birmingham, pilkington avenue. Commas are optional, but improve performance by reducing the complexity of the search.

The free-form may also contain special phrases to describe the type of place to be returned or a coordinate to search close to a position.

Structured query

Parameter	Value
amenity	name and/or type of POI
street	housenumber and streetname
city	city
county	county
state	state
country	country
postalcode	postal code

The structured form of the search query allows to lookup up an address that is already split into its components. Each parameter represents a field of the address. All parameters are optional. You should only use the ones that are relevant for the address you want to geocode.

Attention

Cannot be combined with the `q=<query>` parameter. Newer versions of the API will return an error if you do so. Older versions simply return unexpected results.

Parameters

The following parameters can be used to further restrict the search and change the output. They are usable for both forms of the search query.

Output format

Parameter	Value	Default
format	one of: `xml`, `json`, `jsonv2`, `geojson`, `geocodejson`	`jsonv2`

See Place Output Formats for details on each format.

Note

The Nominatim service at https://nominatim.openstreetmap.org has a different default behaviour for historical reasons. When the `format` parameter is omitted, the request will be forwarded to the Web UI.

Parameter	Value	Default
json_callback	function name	*unset*

When given, then JSON output will be wrapped in a callback function with the given name. See JSONP for more information.

Only has an effect for JSON output formats.

Parameter	Value	Default
limit	number	10

Limit the maximum number of returned results. Cannot be more than 40. Nominatim may decide to return less results than given, if additional results do not sufficiently match the query.

Output details

Parameter	Value	Default
addressdetails	0 or 1	0

When set to 1, include a breakdown of the address into elements. The exact content of the address breakdown depends on the output format.

Tip

If you are interested in a stable classification of address categories (suburb, city, state, etc), have a look at the `geocodejson` format. All other formats return classifications according to OSM tagging. There is a much larger set of categories and they are not always consistent, which makes them very hard to work with.

Parameter	Value	Default
extratags	0 or 1	0

When set to 1, the response include any additional information in the result that is available in the database, e.g. wikipedia link, opening hours.

Parameter	Value	Default
namedetails	0 or 1	0

When set to 1, include a full list of names for the result. These may include language variants, older names, references and brand.

Language of results

Parameter	Value	Default
accept-language	browser language string	content of "Accept-Language" HTTP header

Preferred language order for showing search results. This may either be a simple comma-separated list of language codes or have the same format as the "Accept-Language" HTTP header.

Tip

First-time users of Nominatim tend to be confused that they get different results when using Nominatim in the browser versus in a command-line tool like wget or curl. The command-line tools usually don't send any Accept-Language header, prompting Nominatim to show results in the local language. Browsers on the contrary always send the currently chosen browser language.

Result restriction

There are two ways to influence the results. *Filters* exclude certain kinds of results completely. *Boost parameters* only change the order of the results and thus give a preference to some results over others.

Parameter	Value	Default
countrycodes	comma-separated list of country codes	*unset*

Filter that limits the search results to one or more countries. The country code must be the ISO 3166-1alpha2 code of the country, e.g. `gb` for the United Kingdom, `de` for Germany.

Each place in Nominatim is assigned to one country code based on OSM country boundaries. In rare cases a place may not be in any country at all, for example, when it is in international waters. These places are also excluded when the filter is set.

Note

This parameter should not be confused with the 'country' parameter of the structured query. The 'country' parameter contains a search term and will be handled with some fuzziness. The `countrycodes` parameter is a hard filter and as such should be preferred. Having both parameters in the same query will work. If the parameters contradict each other, the search will come up empty.

Parameter	Value	Default
layer	comma-separated list of: `address`, `poi`, `railway`, `natural`, `manmade`	*unset* (no restriction)

The layer filter allows to select places by themes.

The `address` layer contains all places that make up an address: address points with house numbers, streets, inhabited places (suburbs, villages, cities, states tec.) and administrative boundaries.

The `poi` layer selects all point of interest. This includes classic POIs like restaurants, shops, hotels but also less obvious features like recycling bins, guideposts or benches.

The `railway` layer includes railway infrastructure like tracks. Note that in Nominatim's standard configuration, only very few railway features are imported into the database.

The `natural` layer collects features like rivers, lakes and mountains while the `manmade` layer functions as a catch-all for features not covered by the other layers.

Parameter	Value	Default
featureType	one of: `country`, `state`, `city`, `settlement`	*unset*

The featureType allows to have a more fine-grained selection for places from the address layer. Results can be restricted to places that make up the 'state', 'country' or 'city' part of an address. A featureType of settlement selects any human inhabited feature from 'state' down to 'neighbourhood'.

When featureType is set, then results are automatically restricted to the address layer (see above).

Tip

Instead of using the featureType filters `country`, `state` or `city`, you can also use a structured query without the finer-grained parameters amenity or street.

Parameter	Value	Default
exclude_place_ids	comma-separated list of place ids	

If you do not want certain OSM objects to appear in the search result, give a comma separated list of the `place_id`s you want to skip. This can be used to retrieve additional search results. For example, if a previous query only returned a few results, then including those here would cause the search to return other, less accurate, matches (if possible).

Parameter	Value	Default
viewbox	`<x1>,<y1>,<x2>,<y2>`	*unset*

Boost parameter which focuses the search on the given area. Any two corner points of the box are accepted as long as they make a proper box. `x` is longitude, `y` is latitude.

Parameter	Value	Default
bounded	0 or 1	0

When set to 1, then it turns the 'viewbox' parameter (see above) into a filter parameter, excluding any results outside the viewbox.

When `bounded=1` is given and the viewbox is small enough, then an amenity-only search is allowed. Give the special keyword for the amenity in square brackets, e.g. `[pub]` and a selection of objects of this type is returned. There is no guarantee that the result returns all objects in the area.

Polygon output

Parameter	Value	Default
polygon_geojson	0 or 1	0
polygon_kml	0 or 1	0
polygon_svg	0 or 1	0
polygon_text	0 or 1	0

Add the full geometry of the place to the result output. Output formats in GeoJSON, KML, SVG or WKT are supported. Only one of these options can be used at a time.

Parameter	Value	Default
polygon_threshold	floating-point number	0.0

When one of the polygon_* outputs is chosen, return a simplified version of the output geometry. The parameter describes the tolerance in degrees with which the geometry may differ from the original geometry. Topology is preserved in the geometry.

Other

Parameter	Value	Default
email	valid email address	*unset*

If you are making large numbers of request please include an appropriate email address to identify your requests. See Nominatim's Usage Policy for more details.

Parameter	Value	Default
dedupe	0 or 1	1

Sometimes you have several objects in OSM identifying the same place or object in reality. The simplest case is a street being split into many different OSM ways due to different characteristics. Nominatim will attempt to detect such duplicates and only return one match. Setting this parameter to 0 disables this deduplication mechanism and ensures that all results are returned.

Parameter	Value	Default
debug	0 or 1	0

Output assorted developer debug information. Data on internals of Nominatim's "search loop" logic, and SQL queries. The output is HTML format. This overrides the specified machine readable format.

Examples

XML WITH KML POLYGON

- https://nominatim.openstreetmap.org/search?q=135+pilkington+avenue,+birmingham&format=xml&polygon_kml=1&addressdetails=1

```
<?xml version="1.0" encoding="UTF-8" ?>
<searchresults timestamp="Tue, 08 Aug 2023 15:45:41 +00:00"
               attribution="Data © OpenStreetMap contributors, ODbL 1.0. http://osm.org/copyright"
               querystring="135 pilkington avenue, birmingham"
               more_url="https://nominatim.openstreetmap.org/search?
q=135+pilkington+avenue%2C+birmingham&amp;polygon_kml=1&amp;addressdetails=1&amp;limit=20&amp;exclude_place_
               exclude_place_ids="125279639">
  <place place_id="125279639"
         osm_type="way"
         osm_id="90394480"
         lat="52.5487921"
         lon="-1.8164308"
         boundingbox="52.5487473,52.5488481,-1.8165130,-1.8163464"
         place_rank="30"
         address_rank="30"
         display_name="135, Pilkington Avenue, Maney, Sutton Coldfield, Wylde Green, Birmingham, West
Midlands Combined Authority, England, B72 1LH, United Kingdom"
         class="building"
         type="residential"
         importance="9.999999994736442e-08">
    <geokml>
      <Polygon>
        <outerBoundaryIs>
          <LinearRing>
            <coordinates>-1.816513,52.5487566 -1.816434,52.5487473 -1.816429,52.5487629
-1.8163717,52.5487561 -1.8163464,52.5488346 -1.8164599,52.5488481 -1.8164685,52.5488213
-1.8164913,52.548824 -1.816513,52.5487566</coordinates>
          </LinearRing>
```

```
        </outerBoundaryIs>
      </Polygon>
    </geokml>
    <house_number>135</house_number>
    <road>Pilkington Avenue</road>
    <hamlet>Maney</hamlet>
    <town>Sutton Coldfield</town>
    <village>Wylde Green</village>
    <city>Birmingham</city>
    <ISO3166-2-lvl8>GB-BIR</ISO3166-2-lvl8>
    <state_district>West Midlands Combined Authority</state_district>
    <state>England</state>
    <ISO3166-2-lvl4>GB-ENG</ISO3166-2-lvl4>
    <postcode>B72 1LH</postcode>
    <country>United Kingdom</country>
    <country_code>gb</country_code>
  </place>
</searchresults>
```


JSON WITH SVG POLYGON

https://nominatim.openstreetmap.org/search?q=Unter%20den%20Linden%201%20Berlin&format=json&addressdetails=1&limit=1&polygon_svg=1


```
[
  {
    "address": {
      "ISO3166-2-lvl4": "DE-BE",
      "borough": "Mitte",
      "city": "Berlin",
      "country": "Deutschland",
      "country_code": "de",
      "historic": "Kommandantenhaus",
      "house_number": "1",
      "neighbourhood": "Friedrichswerder",
      "postcode": "10117",
      "road": "Unter den Linden",
      "suburb": "Mitte"
    },
    "boundingbox": [
      "52.5170798",
      "52.5173311",
      "13.3975116",
      "13.3981577"
    ],
    "class": "historic",
    "display_name": "Kommandantenhaus, 1, Unter den Linden, Friedrichswerder, Mitte, Berlin, 10117,
Deutschland",
    "importance": 0.8135042058306902,
    "lat": "52.51720765",
    "licence": "Data © OpenStreetMap contributors, ODbL 1.0. https://osm.org/copyright",
    "lon": "13.397834399325466",
    "osm_id": 15976890,
    "osm_type": "way",
    "place_id": 108681845,
    "svg": "M 13.3975116 -52.5172905 L 13.397549 -52.5170798 13.397715 -52.5170906 13.3977122 -52.5171064
13.3977392 -52.5171086 13.3977417 -52.5170924 13.3979655 -52.5171069 13.3979623 -52.5171233 13.3979893
-52.5171248 13.3979922 -52.5171093 13.3981577 -52.5171203 13.398121 -52.5173311 13.3978115 -52.5173103 Z",
    "type": "house"
  }
]
```


JSON WITH ADDRESS DETAILS

https://nominatim.openstreetmap.org/search?addressdetails=1&q=bakery+in+berlin+wedding&format=jsonv2&limit=1


```
[
  {
    "address": {
      "ISO3166-2-lvl4": "DE-BE",
      "borough": "Mitte",
      "city": "Berlin",
      "country": "Deutschland",
      "country_code": "de",
      "neighbourhood": "Sprengelkiez",
      "postcode": "13347",
      "road": "Lindower Straße",
      "shop": "Ditsch",
      "suburb": "Wedding"
    },
    "addresstype": "shop",
    "boundingbox": [
      "52.5427201",
      "52.5427654",
      "13.3668619",
      "13.3669442"
    ],
    "category": "shop",
    "display_name": "Ditsch, Lindower Straße, Sprengelkiez, Wedding, Mitte, Berlin, 13347, Deutschland",
    "importance": 9.99999999995449e-06,
    "lat": "52.54274275",
    "licence": "Data © OpenStreetMap contributors, ODbL 1.0. http://osm.org/copyright",
    "lon": "13.36690305710228",
    "name": "Ditsch",
    "osm_id": 437595031,
    "osm_type": "way",
    "place_id": 204751033,
    "place_rank": 30,
    "type": "bakery"
  }
]
```


GEOJSON

https://nominatim.openstreetmap.org/search?q=17+Strada+Pictor+Alexandru+Romano%2C+Bukarest&format=geojson


```
{
  "type": "FeatureCollection",
  "licence": "Data © OpenStreetMap contributors, ODbL 1.0. https://osm.org/copyright",
  "features": [
    {
      "type": "Feature",
      "properties": {
        "place_id": "35811445",
        "osm_type": "node",
        "osm_id": "2846295644",
        "display_name": "17, Strada Pictor Alexandru Romano, Bukarest, Bucharest, Sector 2, Bucharest,
023964, Romania",
        "place_rank": "30",
        "category": "place",
        "type": "house",
        "importance": 0.62025
      },
      "bbox": [
        26.1156689,
        44.4354754,
        26.1157689,
```

```
        44.4355754
      ],
      "geometry": {
        "type": "Point",
        "coordinates": [
          26.1157189,
          44.4355254
        ]
      }
    }
  ]
}
```


GEOCODEJSON

https://nominatim.openstreetmap.org/search?q=%CE%91%CE%B3%CE%AF%CE%B1+%CE%A4%CF%81%CE%B9%CE%AC%CE%B4%CE%B1%2C+%CE%91%CE%B4%CF%89%CE%BD%CE%B9%CE%B4%CE%BF%CF%82%2C+Athens%2C+Greece&format=geocodejson


```
{
  "type": "FeatureCollection",
  "geocoding": {
    "version": "0.1.0",
    "attribution": "Data © OpenStreetMap contributors, ODbL 1.0. https://osm.org/copyright",
    "licence": "ODbL",
    "query": "Αγία Τριάδα, Αδωνιδος, Athens, Greece"
  },
  "features": [
    {
      "type": "Feature",
      "properties": {
        "geocoding": {
          "type": "place_of_worship",
          "label": "Αγία Τριάδα, Αδωνιδος, Άγιος Νικόλαος, 5ο Δημοτικό Διαμέρισμα Αθηνών, Athens,
Municipality of Athens, Regional Unit of Central Athens, Region of Attica, Attica, 11472, Greece",
          "name": "Αγία Τριάδα",
          "admin": null
        }
      },
      "geometry": {
        "type": "Point",
        "coordinates": [
          23.72949633941,
          38.0051697
        ]
      }
    }
  ]
}
```