Yandex.Maps API

Geocoding

15.05.2015

Yandex

Yandex.Maps API. Geocoding. Version 1.0
Document build date: 15.05.2015.
This volume is a part of Yandex technical documentation.
Yandex helpdesk site: http://help.yandex.ru

Copyright Disclaimer

Contact information

Yandex LLC
http://www.yandex.com
Phone: +7 495 739 7000
Email: pr@yandex-team.ru
Headquarters: 16 L'va Tolstogo St., Moscow, Russia 119021

Contents

Geocoding

The Yandex.Maps service provides a geocoding service for its users. This service can determine geographical coordinates and other information about an object using its name or address, as well as the opposite, using the coordinates of an object to determine its address (reverse geocoding).

For example, the geocoder receives the request “Türkiye, İstanbul, Kartal, Esentepe, Aydos Sokak, 32” and returns the geographical coordinates of this building: “29.198184 40.900640” (longitude, latitude). Conversely, if the request contains the geographical coordinates of the building “29.198184 40.900640”, the geocoder will return the address.

The geocoder can be accessed either over the HTTP protocol or using the JavaScript API. When accessing the geocoder over HTTP, the response may be formed either as an XML document in YMapsML format, or in JSON format.

This document describes parameters for sending an HTTP request to the geocoder and the response, and also provides examples.

HTTP request parameters

A request for the geocoder consists of an HTTP request sent to the URL http://geocode-maps.yandex.ru/1.x/.

For example, to find out the coordinates of the building at the address "Kabasakal Caddesi, Istanbul, Turkey", we could send the following request to the geocoder:

```
http://geocode-maps.yandex.ru/1.x/?geocode=Kabasakal+Caddesi,+Istanbul,
+Turkey&lang=en-US
```

In response, the geocoder returns the geographical coordinates of the building, along with any additional information about the found object (see the request response).

For reverse geocoding, the request specifies the coordinates of the object you are looking for, and the response returns its address.

The table below shows a complete list of parameters for the HTTP request:

Parameter	Description	Example
Mandatory parameters		

`geocode`	The address you want to geocode, or the geographical coordinates.
	Coordinates can be set in one of the formats described below.
		Forward geocoding:
		`geocode=Kabasakal +Caddesi,+Istanbul, +Turkey`
		Reverse geocoding:
		`geocode=28.98017,41 .008434`

Optional parameters		
`kind`	Type of toponym (only for reverse geocoding). List of accepted values: • `house` — house or building • `street` — street • `metro` — subway station • `district` — city district • `locality` — locality (city, town, village, etc.)	Reverse geocoding: `kind=street`

Parameter	Description	Example
format	The format for the geocoder's response: • xml — Results are returned as a YMapsML document. • json — Results are returned in JSON format. Default value: xml.	format=json
callback	The name of the JavaScript function that the geocoder's response is returned to (in accordance with JSONP conventions). This parameter is accepted only if the response is returned in JSON format.	callback=my_response_handler
ll,spn	Geographical area for the object search. The search can be limited to this area, or unlimited (depending on the value of the rspn parameter). If it is not limited, using the ll and spn parameters affects the order of results output; objects that are inside this area are given priority. **Record format** The ll parameter defines the longitude and latitude of the center of the area (in degrees), while spn defines its range (in degrees). The span of the area is defined by two numbers; the first is the difference between the maximum and minimum longitude of the area, and the second is the difference between the maximum and minimum latitude. **Reverse geocoding** For reverse geocoding, the ll parameter is ignored, and spn is taken into account only if the kind parameter takes one of the following values: house, street, locality or metro.	Forward geocoding: ll=28.980017,41.008434&spn=0.552069,0.400552 Reverse geocoding: spn=0.552069,0.400552&kind=street
rspn	Restricts the search scope to the area defined using ll and spn parameters. Possible values: • 0 — Do not restrict the search (by default). • 1 — Restrict.	rspn=1
results	Maximum number of objects to be returned. Default value: 10.	results=5
skip	Number of objects in the response that should be skipped (starting from the first one). Default value: 0.	skip=3
lang	The preferred language for object descriptions. **Record format** lang=language-region, where • language — Two-letter language code. Specified in ISO 639-1 format. • region — Two-letter country code. Specified in ISO 3166-1 format.	lang=en-US

Parameter	Description	Example
	List of supported values: • `ru-RU` — Russian (by default) • `uk-UA` — Ukrainian • `be-BY` — Belarusian • `en-US` — American English • `en-BR` — British English • `tr-TR` — Turkish (only for maps of Turkey)	

Formats for geographical coordinates

In the HTTP request to the geocoder, geographical coordinates are set using the `geocode` parameter. The coordinate values are listed sequentially and separated by a space, comma, or semicolon. In addition, any number of spaces is allowed on either side of the separators.

Note:
The ";" character should be encoded as "%3B".

To indicate the sign for coordinates, you can use the "+" and "-" symbols, or letters corresponding to the hemisphere (using the "+" symbol is optional). For example, N (Northern latitude) and E (Eastern longitude) indicate positive coordinates, while W (Western longitude) and S (Southern latitude) indicate negative coordinates.

Letters can be placed either before or after the coordinates: "N39.889847, E32.810152" or "39.889847N, 32.810152E". The letters can also be separated by a space: "N 39.889847, E 32.810152".

Request example for reverse geocoding:

```
http://geocode-maps.yandex.ru/1.x/?geocode=E134.854,S25.828&lang=en-US
```

The list below shows acceptable formats for expressing geographical coordinates:

Record format	Order of coordinates	Example
+-float, +-float	Longitude, latitude	134.854, -25.828
float [direction] * , float [direction]	Any	E134.854, S25.828 134.854E, 25.828S
+-deg° mm' ss", +-deg° mm' ss"	Latitude, longitude	-25°49′41.1″, 134°51′15.88″
deg° mm' ss" [direction], deg° mm' ss" [direction]	Any	25°49′41.1″S, 134°51′15.88″E
NMEA	Any	2549.67,S, 13451.26,E

* [direction] — The letter designation for one of the four directions: N, E, W, S.

The geocoder returns the given coordinates in the `metaDataProperty/GeocoderResponseMetaData/Point/pos` element in the format "[longitude] [latitude]":

```
<metaDataProperty>
    <GeocoderResponseMetaData>
        <request>E134.854,S25.828</request>
        <found>1</found>
        <results>10</results>
        <Point>
            <pos>134.854412 -25.828084</pos>
        </Point>
    </GeocoderResponseMetaData>
</metaDataProperty>
```

The geocoder response

The geocoder response can be returned in the following formats:

- XML.
- JSON (may be returned as an argument of a function — JSONP).

Both formats have the same structure, and the results returned are identical. For XML output, the results are shown as XML elements. For JSON, key:value pairs are used, contained in curly brackets, and groups using arrays are in square brackets. The name of the XML element is the same as the matching key in the JSON output.

If forward geocoding was performed (the coordinates were determined using the address and/or name), the results are sorted according to their similarity to the address or name that was specified in the request, and not by their geographical proximity to the first object found.

For reverse geocoding (the address was determined from the coordinates), results are sorted according to the size of the geometric area that the object belongs to, in reverse order (house number, street, district, city, and so on).

The response in XML format

By default, the geocoder response is returned in XML format. Let's look at the XML response from the service for a request containing the address of the Sultanahmet Camii in Istanbul:

```
http://geocode-maps.yandex.ru/1.x/?geocode=Sultanahmet+Camii+İç+Yolları&lang=en-US
```

Response:

```
<?xml version="1.0" encoding="utf-8"?>
<ymaps xmlns="http://maps.yandex.ru/ymaps/1.x">
  <GeoObjectCollection>
    <metaDataProperty xmlns="http://www.opengis.net/gml">
      <GeocoderResponseMetaData xmlns="http://maps.yandex.ru/geocoder/1.x">
        <geo:request xmlns:geo="http://maps.yandex.ru/geocoder/1.x">Sultanahmet
Camii İç Yolları</geo:request>
        <found>1</found>
        <results>10</results>
      </GeocoderResponseMetaData>
    </metaDataProperty>
    <featureMember xmlns="http://www.opengis.net/gml">
      <GeoObject xmlns="http://maps.yandex.ru/ymaps/1.x">
        <metaDataProperty xmlns="http://www.opengis.net/gml">
          <GeocoderMetaData xmlns="http://maps.yandex.ru/geocoder/1.x">
            <kind>street</kind>
            <text>Turkey, Istanbul, Eminönü, Sultanahmet, Sultanahmet Camii Iç
Yollari</text>
            <precision>street</precision>
            <AddressDetails xmlns="urn:oasis:names:tc:ciq:xsdschema:xAL:2.0">
              <Country>
                <AddressLine xml:lang="tr">Istanbul, Eminönü, Sultanahmet,
Sultanahmet Camii Iç Yollari</AddressLine>
                <CountryNameCode>TR</CountryNameCode>
                <CountryName xml:lang="tr">Turkey</CountryName>
                <AdministrativeArea>
                  <AdministrativeAreaName xml:lang="tr">Istanbul</
AdministrativeAreaName>
                  <SubAdministrativeArea>
                    <SubAdministrativeAreaName xml:lang="tr">Eminönü</
SubAdministrativeAreaName>
                    <Locality>
                      <LocalityName xml:lang="tr">Sultanahmet</LocalityName>
                      <Thoroughfare>
                        <ThoroughfareName xml:lang="tr">Sultanahmet Camii Iç
Yollari</ThoroughfareName>
                      </Thoroughfare>
                    </Locality>
```

```
                  </SubAdministrativeArea>
                </AdministrativeArea>
              </Country>
            </AddressDetails>
          </GeocoderMetaData>
        </metaDataProperty>
        <description xmlns="http://www.opengis.net/gml">Turkey, Istanbul,
Eminönü, Sultanahmet</description>
        <name xmlns="http://www.opengis.net/gml">Sultanahmet Camii Iç Yollari</
name>
        <boundedBy xmlns="http://www.opengis.net/gml">
          <Envelope>
            <lowerCorner>28.975779 41.004937</lowerCorner>
            <upperCorner>28.977881 41.006761</upperCorner>
          </Envelope>
        </boundedBy>
        <Point xmlns="http://www.opengis.net/gml">
          <pos>28.976893 41.005808</pos>
        </Point>
      </GeoObject>
    </featureMember>
  </GeoObjectCollection>
</ymaps>
```

The geocoder response consists of a document in YMapsML format, which conforms to the XML schema http://maps.yandex.ru/geocoder/1.x.

The `GeocoderResponseMetaData` element contains information about the request and the number of results found, while the `GeocoderMetaData` element contains the text of the request, the type of object found, information on how accurately the result matches the request, and a detailed mailing address of the object.

The `Point` element contains the coordinates of the center of the object, while the `boundedBy` element contains the boundary within which it is recommended to display the object on the map (a viewport). For most objects, the viewport is a rectangle around the object.

The response in JSON format

To get a response in JSON format, it is necessary to add the `format` parameter with the value `json`:

```
http://geocode-maps.yandex.ru/1.x/?format=json&geocode=Sultanahmet+Camii+İç
+Yolları&lang=en-US
```

In this case, the geocoder response will look like this:

Response:

```
{
  "response": {
    "GeoObjectCollection":{
      "metaDataProperty": {
        "GeocoderResponseMetaData": {
          "request": "Sultanahmet Camii İç Yolları",
          "found": "1",
          "results": "10"
        }
      },
      "featureMember": [{
        "GeoObject": {
          "metaDataProperty": {
            "GeocoderMetaData": {
              "kind":"street",
               "text":"Turkey, Istanbul, Eminönü, Sultanahmet, Sultanahmet Camii
Iç Yollari",
              "precision":"street",
              "AddressDetails": {
                "Country": {
                    "AddressLine": "Istanbul, Eminönü, Sultanahmet, Sultanahmet
Camii Iç Yollari",
```

```
                  "CountryNameCode": "TR",
                  "CountryName": "Turkey",
                  "AdministrativeArea": {
                    "AdministrativeAreaName": "Istanbul",
                    "SubAdministrativeArea": {
                      "SubAdministrativeAreaName": "Eminönü",
                      "Locality": {
                        "LocalityName": "Sultanahmet",
                        "Thoroughfare": {
                          "ThoroughfareName": "Sultanahmet Camii Iç Yollari"
                        }
                      }
                    }
                  }
                }
              }
            },
            "description": "Turkey, Istanbul, Eminönü, Sultanahmet",
            "name":"Sultanahmet Camii Iç Yollari",
            "boundedBy": {
              "Envelope": {
                "lowerCorner": "28.975779 41.004937",
                "upperCorner":"28.977881 41.006761"
              }
            },
            "Point": {
              "pos":"28.976893 41.005808"
            }
          }
        }]
    }
  }
}
```


JSONP

The security policies of modern browsers do not allow web pages to load data from outside servers. "Outside" means servers whose domain name does not match the domain name of the server that the page is located on. If you need to make a page that queries geocoding results, you should use JSONP technology.

When using JSONP, the server that the data is being returned to is passed the name of a function, and the results are returned as a JSON object, but as a parameter of the function with the specified name.

In order to get search results in JSONP format, it is necessary to add to the `callback` parameter the name of the function that will process results that are returned in JSON format.

So for this request

```
http://geocode-maps.yandex.ru/1.x/?
format=json&callback=my_function&geocode=Sultanahmet+Camii+İç+Yolları&lang=en-US
```

the JSON object from the previous example will be returned, but as an argument of the function `my_function`:

```
my_function({
  "response": {
    "GeoObjectCollection": {
      "metaDataProperty": {
        "GeocoderResponseMetaData": {
          "request": "Sultanahmet Camii İç Yolları",
          "found": "1",
          "results": "10"

    ...

});
```

See also
yg:GeocoderMetaData
yg:GeocoderResponseMetaData

Overview of the geocoder response XML schema

Namespaces used

The geocoder response consists of elements which belong to the following four XML namespaces:

Namespace	Schema URL	Documentation	Elements	Definition
ymaps	http://maps.yandex.ru/ymaps/1.x	YMapsML reference	`ymaps`, `GeoObjectCollection`, `GeoObject`	The language used to represent geographical data on Yandex maps.
gml	http://www.opengis.net/gml	The GML 3.1 specification	`featureMember`, `metaDataProperty`, `boundedBy`, `Point`	GML – the geography description standard.
yg	http://maps.yandex.ru/geocoder/1.x	Description of the XML schema of the geocoder response	`GeocoderResponseMetaData`, `GeocoderMetaData`	The XML schema of the geocoder response.
xal	urn:oasis:names:tc:ciq:xsdschema:xAL:2.0	The xAL 2.0 specification	`AddressDetails`	xAL (eXtensible Address Language) is used for structuring mailing addresses in the geocoder response.

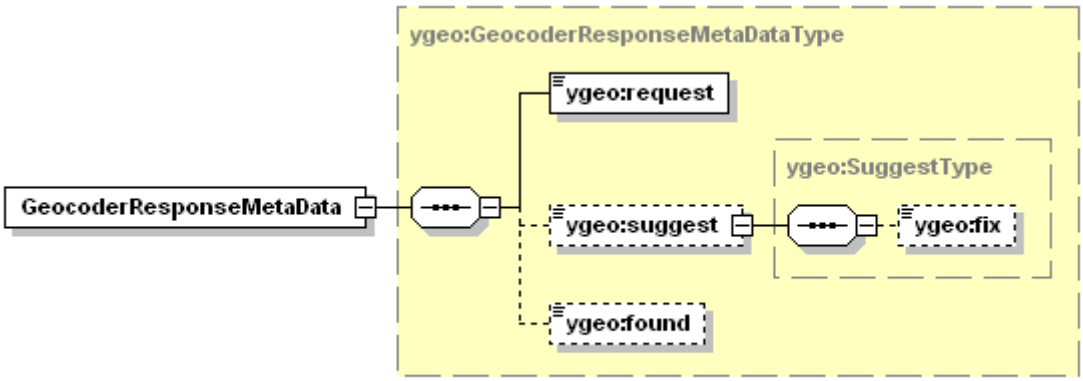
Since the geocoder response is a YMapsML document, the root element belongs to the `ymaps` namespace. In addition, this namespace includes ymaps:GeoObjectCollection and ymaps:GeoObject.

The designated geocoder namespace `yg` includes metadata elements `GeocoderResponseMetaData`, `GeocoderMetaData` and all their child elements, except `AddressDetails`. The following standard **GML** elements are used as well: gml:featureMember, gml:metaDataProperty, gml:Point and gml:boundedBy.

yg:GeocoderResponseMetaData

The <yg:**GeocoderResponseMetaData**> element contains data related to the request.

The element contains information about the request and the number of toponyms found.



Contains:

yg:request, yg:suggest, yg:found, yg:results, yg:skip

Attributes:

This tag contains no attributes.

Examples:

```
<GeocoderResponseMetaData>
  <geo:request>Yalova,Turkie</geo:request>
  <geo:suggest>Yalova,T
    <geo:fix>u</geo:fix>
  rkey</geo:suggest>
  <found>7</found>
  <results>10</results>
</GeocoderResponseMetaData>
```

yg:fix

The <yg:**fix**> element contains a character corrected by the spelling service.

A character corrected by the spelling service, see the example.

Contains:

This tag does not contain other tags.

Contained by:

yg:suggest

Attributes:

This tag contains no attributes.

Examples:

See the example for the `yg:GeocoderResponseMetaData` element.

yg:found

The <**yg:found**> element contains the number of toponyms found.

The number of toponyms found.

Contains:

This tag does not contain other tags.

Contained by:

yg:GeocoderResponseMetaData

Attributes:

This tag contains no attributes.

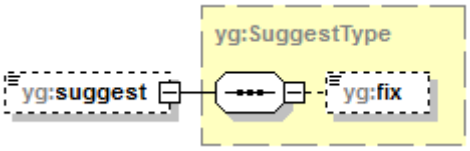
Examples:

See the example for the `yg:GeocoderResponseMetaData` element.

yg:suggest

The <**yg:suggest**> element contains a version of the request, corrected by the spelling service.

If the address was corrected by the spelling service during input, the element will contain the correct version (see the example).



Contains:

`yg:fix`

Contained by:

yg:GeocoderResponseMetaData

Attributes:

This tag contains no attributes.

Examples:

See the example for the `yg:GeocoderResponseMetaData` element.

yg:request

The <**yg:request**> element contains the requested address.

The address specified in the request.

Contains:

This tag does not contain other tags.

Contained by:

yg:GeocoderResponseMetaData

Attributes:

This tag contains no attributes.

Examples:

See the example for the `yg:GeocoderResponseMetaData` element.

yg:results

The <**yg:results**> element contains the number of required search results.

Contained by:

yg:GeocoderResponseMetaData

Attributes:

This tag contains no attributes.

Examples

See the example for the GeocoderResponseMetaData element.

yg:skip

The <**yg:skip**> element specifies how many results should be skipped (starting from the beginning of the list) in the response from the service.

Contained by:

yg:GeocoderResponseMetaData

Attributes:

This tag contains no attributes.

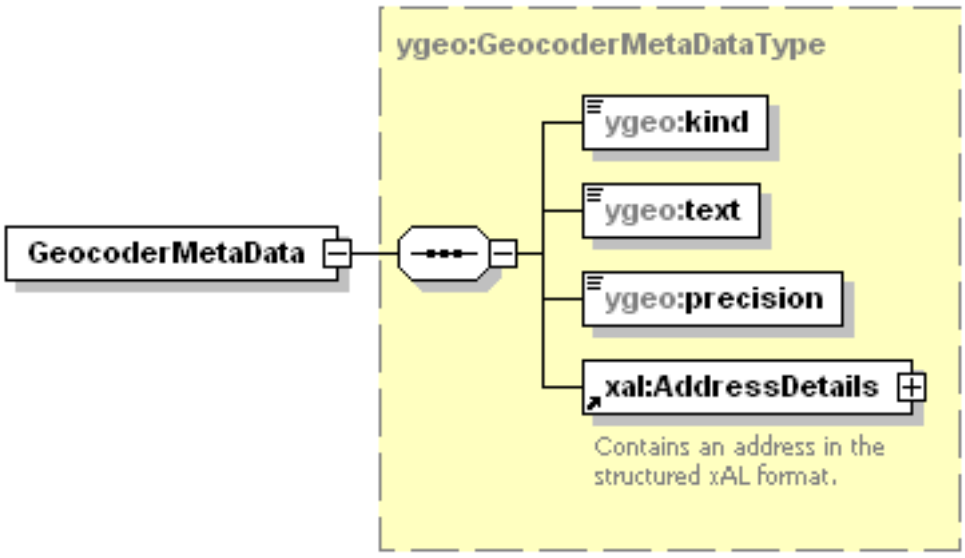
Examples

See the example for the GeocoderResponseMetaData element.

yg:GeocoderMetaData

The <**yg:GeocoderMetaData**> element contains detailed information about the toponym that was found.

Diagram



Contains:

yg:kind, yg:text, yg:precision, xal:AddressDetails

Attributes:

This tag contains no attributes.

Example

```
<GeocoderMetaData>
  <kind>locality</kind>
  <text>Türkiye, Çanakkale, Eceabat, Yalova</text>
  <precision>other</precision>
  <AddressDetails>
    <Country>
      <AddressLine>Çanakkale, Eceabat, Yalova</AddressLine>
      <CountryNameCode>TR</CountryNameCode>
      <CountryName>Türkiye</CountryName>
      <AdministrativeArea>
        <AdministrativeAreaName>Çanakkale</AdministrativeAreaName>
        <SubAdministrativeArea>
            <SubAdministrativeAreaName>Eceabat</SubAdministrativeAreaName>
          <Locality>
            <LocalityName>Yalova</LocalityName>
          </Locality>
        </SubAdministrativeArea>
      </AdministrativeArea>
    </Country>
  </AddressDetails>
</GeocoderMetaData>
```

yg:kind

The <**yg:kind**> element contains the type of toponym found.

Contains the type of toponym found (see the table of possible values).

Possible values for the `kind` element:

Value of yg:kind	Type of toponym found	Value of yg:text (example)
`house`	a detached house	house Türkiye, İstanbul, Kartal, Esentepe, Aydos Sokak, 32
`street`	street	Türkiye, İstanbul, Kartal, Esentepe, Aydos Sokak
`metro`	subway station	Türkiye, İstanbul, Metro Fuar Merkezi
`district`	city district	Türkiye, İstanbul, Kadıköy, Bostancı Mh
`locality`	locality: city, town, village, etc.	Türkiye, Antalya, Antalya
`area`	regional district (oblast)	Türkiye, İstanbul, Şişli
`province`	oblast, region	Türkiye, İstanbul, Şişli
`country`	country	Türkiye
`hydro`	bodies of water: river, lake, stream, reservoir, etc.	Türkiye, İstanbul, Ömerli Barajı
`vegetation`	forest, park, etc.	Türkiye, İstanbul, Saip Molla Özel Ormanı
`airport`	airport	United States, Norfolk, Airport-Norfolk International
`other`	any other	stanbul, Pendik, Havaalanı İç Yolları

Contains:

This tag does not contain other tags.

Contained by:

yg:GeocoderMetaData

Attributes:

This tag contains no attributes.

Examples:

See the example for the `GeocoderMetaData` element.

yg:precision

The <**yg:precision**> element contains an assessment of how accurately the toponym matches the request.

The `precision` element is used when it is necessary to geocode a request with pinpoint accuracy, such as a house number.

It is not uncommon for the address in the request to be invalid or incomplete, for a house with a specified number not to exist, or for a house that has been built recently not to be marked on the map. In such cases, the geocoder returns the closest result (by address) — namely, a house on the same side of the street that has the number that is closest to the requested number. The `precision` element will contain information about the degree of discrepancy between the request and the result (see the table of possible values).

If the difference between the house numbers exceeds 10, then instead of the nearest house, the geocoder returns an object that describes the street.

Possible values of the `precision` element:

yg:precision	The house number as provided in the request	The house number in the best matching result	Comment
`exact`	27, unit 1	27c1	Exact match.
`number`	31, building 4	31 building 2	Only the house number matches.
`near`	16/3	18	The house number does not match, but a house nearby is found (as 18–16 < 10).
`street`	18	4	Only the street name matches (as 18–4 > 10).
`other`	22	–	The street name does not match but a village, district, etc. matches.

Contains:

This tag does not contain other tags.

Contained by:

yg:GeocoderMetaData

Examples:

See the example for the `GeocoderMetaData` element.

yg:text

The <**yg:text**> element contains the full mailing address of a toponym in a single text string.

The full mailing address of a toponym in a single text string.

Contains:

This tag does not contain other tags.

Contained by:

yg:GeocoderMetaData

Attributes:

This tag contains no attributes.

Examples:

See the example for the `GeocoderMetaData` element.

Examples

Basic search by name

The request “Sultanahmet Camii İç Yolları”: http://geocode-maps.yandex.ru/1.x/?geocode=Sultanahmet+Camii+İç+Yolları&lang=en-US.

The same request, but for getting results as JSON output: http://geocode-maps.yandex.ru/1.x/?format=json&geocode=Sultanahmet+Camii+İç+Yolları&lang=en-US.

The request “Cinci Meydanı Sokak, Küçük Ayasofya, Eminönü,İstanbul,Türkiye”: http://geocode-maps.yandex.ru/1.x/?geocode=Cinci+Meydanı+Sokak,Küçük+Ayasofya,Eminönü,İstanbul,Türkiye&lang=en-US.

The request “Yalova”: http://geocode-maps.yandex.ru/1.x/?geocode=Yalova&lang=en-US.

A request with a spelling mistake “Stanbul”, and the correction of the typo in the response: http://geocode-maps.yandex.ru/1.x/?geocode=Stanbul&lang=en-US.

Searching for objects in a specific region

If a region to search in is specified in the request, then the first results shown will be those objects that are closest to this region, for example: http://geocode-maps.yandex.ru/1.x/?geocode=Yalova&ll=29.270485,40.65731&spn=2.5,2.5&lang=en-US.

Restricting the number of results in the response

Some requests may correspond to several objects. In the request for the geocoder, the desired amount of objects to output can be specified, as well as the number of the first one of them.

The request “Yalova”, first five results: http://geocode-maps.yandex.ru/1.x/?geocode=Yalova&results=1&lang=en-US.

The request “Yalova”, two results, starting from the 6th: http://geocode-maps.yandex.ru/1.x/?geocode=Yalova&results=2&skip=5&lang=en-US.

Index

Yandex

Yandex.Maps API

Geocoding

15.05.2015