

# Package: nlgeocoder (via r-universe)

July 1, 2024

**Title** Geocoding for the Netherlands

**Version** 0.1.3.9000

**Description** R interface to the open location server API of 'Publieke Diensten Op de Kaart' (<<http://www.pdok.nl>>). It offers geocoding, address suggestions and lookup of geographical objects. Included is an utility function for displaying leaflet tiles restricted to the Netherlands.

**Depends** R (>= 3.4)

**License** GPL-2

**BugReports** <https://github.com/uRosConf/nlgeocoder/issues>

**URL** <https://github.com/uRosConf/nlgeocoder>

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 6.1.1

**Suggests** testthat, knitr, rmarkdown, sf, shiny, leaflet

**VignetteBuilder** knitr

**Imports** jsonlite

**Repository** <https://edwindj.r-universe.dev>

**RemoteUrl** <https://github.com/uRosConf/nlgeocoder>

**RemoteRef** HEAD

**RemoteSha** 70ca713b7aaf7b355158783b29c4eafc4cf9e24f

## Contents

nlgeocoder-package . . . . .	2
addPdokTiles . . . . .	3
addresses . . . . .	3
nl_free . . . . .	4
nl_geocode . . . . .	5
nl_lookup . . . . .	6
nl_reverse . . . . .	6
nl_suggest . . . . .	7

---

nlgeocoder-package     *nlgeocoder package*

---

## Description

The nlgeocoder package provides a R wrapper around the pdok webservice.

## Details

it allows to geocode data using the following functions:

- [nl\\_geocode](#): geocode a vector of addresses.
- [nl\\_free](#): find possible matches for one address.
- [nl\\_suggest](#): find possible suggestions for one address.
- [nl\\_lookup](#): lookup information of object found in [nl\\_suggest](#).

## Author(s)

**Maintainer:** Edwin de Jonge <edwindjonge@gmail.com> (0000-0002-6580-4718)

Authors:

- Willy Tadema
- Egge-Jan Pollé

Other contributors:

- Juris Breidaks [contributor]

## See Also

Useful links:

- <https://github.com/uRos2018/nlgeocoder>
- Report bugs at <https://github.com/uRos2018/nlgeocoder/issues>

---

addPdokTiles                      *Leaflet function to add pdok tiles*

---

### Description

This function adds PDOK tile layers to [leaflet](#). It is a small wrapper around [addTiles](#) that sets the correct urls for the layers.

### Usage

```
addPdokTiles(map, type = c("brt", "aerial", "pastel", "gray"),
             group = match.arg(type), options, ...)
```

### Arguments

map	leaflet object
type	one of the PDOK tiles: currently: "brt", "aerial", "pastel" or "gray"
group	group is set equal to type
options	passed to <a href="#">addTiles</a>
...	Passed to <a href="#">addTiles</a> .

### Details

PDOK provides tiles that can be used as a tile layer in several geovisualization tools including [leaflet](#). The layers differ from other tile layers such as [Openstreetmap](#), [Google](#) and [cartomap](#) in the following manner:

- The tiles are restricted to the Netherlands: tiles accross the Dutch border are "grayed out". This is useful for cartographic applications that are restricted to Dutch geo-information.
- The tiles are open source and based on open sourced geo-information.

---

addresses                      *Several Points of Interest (POI) that are included for test purposes.*

---

### Description

- POI, points of interest
- Address, Uncleaned address information.

### Usage

```
addresses
```

### Format

An object of class `data.frame` with 5 rows and 2 columns.

---

nl_free	<i>Free geocoding search</i>
---------	------------------------------

---

### Description

This function wraps the "free" pdok service, and allows for free search. For syntax and examples see the documentation of pdok. A more easy/convenient but simpler function for geocoding is [nl\\_geocode](#).

### Usage

```
nl_free(q, rows = NULL, start = NULL, fq = NULL, lat = NULL,
        lon = NULL, type = NULL, fl = NULL, df = NULL, ...,
        verbose = FALSE)
```

### Arguments

q	query to geocoding service.
rows	maximum number of rows to be returned. Default is 10.
start	start index. Default is 0.
fq	apply a filter to the query, e.g. fq=bron:BAG. The default filter is 'type:(municipality OR town OR neighborhood OR road OR postcode OR address)'. Use fq=* to remove the default filter.
lat	enter a decimal degree (latitude, in WGS84). Together with lon this will define a point to search from. Search results will be ordered by distance from this point. For example: 'lat=52.09&lon=5.12'
lon	enter a decimal degree (longitude, in WGS84). Together with lat this will define a point to search from. Search results will be ordered by distance from this point. For example: 'lat=52.09&lon=5.12'
type	restrict the results on a specific type.
fl	fields to return.
df	field that should be search in.
...	parameters passed to the pdok webservice
verbose	logical should the function print out messages.

### Value

The result of the pdok free webservice converted to a R list object.

### See Also

[nl\\_geocode](#)

**Examples**

```
l <- nl_free("Henri Faasdreef 312")
l$response$numFound
l$response$docs["weergavenaam"]
```

---

nl_geocode	<i>Geocode addresses</i>
------------	--------------------------

---

**Description**

nl\_geocode returns for a vector of addresses the most probable object/location. This function is more user friendly than the barebones webservices ([nl\\_free](#)), and uses the same function signature as `ggmap::geocode`.

**Usage**

```
nl_geocode(location, output = c("wgs84", "rd", "data.frame"),
  messaging = FALSE, type = "adres", ..., verbose = messaging)

nl_geocode_rd(location, messaging = FALSE, type = "adres", ...,
  verbose = messaging)

nl_geocode_df(location, messaging = FALSE, type = "adres", ...,
  verbose = messaging)
```

**Arguments**

location	string with location to be found
output	Should the output be a <code>data.frame</code> or <code>sf</code> object in wgs84 or Rijksdriehoekstelsel format?
messaging	Print the urls fired to the webserver (consistent with <code>'ggmap::geocode'</code> )
type	restrict the type of object that is returned from the service, see details for possible types.
...	will be passed to <a href="#">nl_free</a> .
verbose	identical to messaging (consistent with other nlgeoder functions)

**Details**

type can be one or more of the following: "provincie", "gemeente", "woonplaats", "weg", "post-code", "adres", "perceel", "hectometerpaal", "wijk", "buurt", "waterschapsgrens", "appartementsrecht".

**Value**

The return type can be specified and can be of type "sf" or "data.frame", depending on the value of output.

**Examples**

```
data("addresses")

r <- nl_geocode(addresses$Address)
r["weergavenaam"]
names(r)
```

---

nl\_lookup                      *Look up a geo object*

---

**Description**

Retrieve detailed properties of a geo object found with [nl\\_suggest](#) or [nl\\_free](#).

**Usage**

```
nl_lookup(id, ..., output = c("list", "raw"), verbose = FALSE)
```

**Arguments**

id	of object found in <a href="#">nl_suggest</a> or <a href="#">nl_free</a>
...	extra parameters are passed to the lookup service of pdok.
output	What type of output should be returned
verbose	should the function print message while retrieving the data?

**Value**

Depending on the value of output the raw search results in R format or the properties of the specific object as a R list object.

**Examples**

```
obj <- nl_lookup("weg-f633e85f07eda4e68a00fb13f9d128f5")
names(obj)
obj$weergavenaam
```

---

nl\_reverse                      *experimental reverse api*

---

**Description**

experimental reverse api

**Usage**

```
nl_reverse()
```

---

`nl_suggest`*Get a list of suggestions for geolocations in NL*

---

## Description

`nl_suggest` returns a list of suggestions for a location description.

## Usage

```
nl_suggest(q, ..., rows = NULL, type = NULL, verbose = FALSE,
           fl = NULL, sort = NULL, qf = NULL, bq = NULL)
```

## Arguments

<code>q</code>	search terms that should be geolocated
<code>...</code>	parameters passed to geolocation service
<code>rows</code>	maximum number of rows to be returned. Default is 10.
<code>type</code>	restrict type of geolocation to a type (see details for possible types)
<code>verbose</code>	Should the functions print messages on what it is retrieving.
<code>fl</code>	the columns that should be returned (aka select on columns of result)
<code>sort</code>	how the data should be sorted
<code>qf</code>	the fields that should be queried
<code>bq</code>	the boosting of the query.

## Details

`type` can be one or more of the following: "provincie", "gemeente", "woonplaats", "weg", "post-code", "adres", "perceel", "hectometerpaal", "wijk", "buurt", "waterschapsgrens", "appartementrecht".

## Value

The result of the pdok suggest webservice converted to a R list object.

## Examples

```
sug <- nl_suggest("Henri Faasdreef")

# how many objects have a score?
sug$response$numFound

# get suggestions
sug$response$docs
```

# Index

## \* datasets

addresses, [3](#)

addPdokTiles, [3](#)

addresses, [3](#)

addTiles, [3](#)

data.frame, [5](#)

leaflet, [3](#)

nl\_free, [2](#), [4](#), [5](#), [6](#)

nl\_geocode, [2](#), [4](#), [5](#)

nl\_geocode\_df (nl\_geocode), [5](#)

nl\_geocode\_rd (nl\_geocode), [5](#)

nl\_lookup, [2](#), [6](#)

nl\_reverse, [6](#)

nl\_suggest, [2](#), [6](#), [7](#)

nlgeocoder (nlgeocoder-package), [2](#)

nlgeocoder-package, [2](#)

sf, [5](#)