



NAME

v.in.pygbif - Search and import GBIF species distribution data

KEYWORDS

[vector](#), [geometry](#)

SYNOPSIS

v.in.pygbif

v.in.pygbif --help

v.in.pygbif [-brpigotns] output=*name* taxa=*string* [mask=*name*]

[date_from=*string*] [date_to=*string*]

[basisofrecord=*string*] rank=*string* [recordedby=*string*]

[institutioncode=*string*] [country=*string*] [continent=*string*] [--overwrite] [-

-help] [--verbose] [--quiet] [--ui]

Flags:

-b

Do not build topology

-r

Do not limit import to current region (works only in lat/lon)

-p

Print result from matching taxa names and exit

-i

Produce individual map for each taxon

-g

Print result from matching taxon names in shell script style and exit

-o

Print number of matching occurrences per taxon and exit

-t

Print result of taxon matching in table format and exit

-n

Do not limit search to records with coordinates

- s** Do also import occurrences with spatial issues
- overwrite** Allow output files to overwrite existing files
- help** Print usage summary
- verbose** Verbose module output
- quiet** Quiet module output
- ui** Force launching GUI dialog

Parameters:

output=*name* [required]

Name of resulting vector map with occurrences

taxa=*string* [required]

Comma separated list of taxon names or keys to fetch data for

mask=*name*

Name of input vector map

Vector map that delimits region of interest

date_from=*string*

Lower bound of acceptable dates (format: yyyy, yyyy-MM, yyyy-MM-dd, or MM-dd)

date_to=*string*

Upper bound of acceptable dates (format: yyyy, yyyy-MM, yyyy-MM-dd, or MM-dd)

basisofrecord=*string*

Accepted basis of records

Options: *ALL, FOSSIL_SPECIMEN, HUMAN_OBSERVATION, LITERATURE, LIVING_SPECIMEN, MACHINE_OBSERVATION, OBSERVATION, PRESERVED_SPECIMEN, UNKNOWN*

Default: *ALL*

rank=*string* [required]

Rank of the taxon to search for

Options: *class, cultivar, cultivar_group, domain, family, form, genus, informal, infrageneric_name, infraorder, infraspecific_name, infrasubspecific_name, kingdom, order, phylum, section, series, species, strain, subclass, subfamily, subform, subgenus, subkingdom, suborder, subphylum, subsection, subseries, subspecies, subtribe, subvariety, superclass, superfamily, superorder, superphylum, suprageneric_name, tribe, unranked, variety*

Default: *species*

recordedby=string

The person who recorded the occurrence.

institutioncode=string

An identifier of any form assigned by the source to identify the institution the record belongs to.

country=string

The 2-letter country code (as per ISO-3166-1) of the country in which the occurrence was recorded

continent=string

The continent in which the occurrence was recorded

Options: *africa*, *antarctica*, *asia*, *europa*, *north_america*, *oceania*, *south_america*

Table of contents

DESCRIPTION:

The module *v.in.pygbif* is a wrapper around the [pygbif](#) package.

Thus, *pygbif* is a dependency of *v.in.pygbif*. *pygbif* can be installed like this:

```
pip install pygbif
```

Through *pygbif*, the module allows to download data from the Global Biodiversity Information Facility ([GBIF](#)) using different search/filter criteria.

Since some of the Darwin Core attribute columns represent SQL keywords, the prefix "g_" was added to all attribute columns. The names of taxa provided at input to the search are written to the column "g_search".

The point data is downloaded and projected into the current location. By default import is limited to the current computational region in order to avoid possible projection errors, e.g. when projecting global data into UTM locations. However, in *latlon* location this limitation can be skipped using the *-r* flag.

Providing a mask automatically overrides the limitation of the search to the current computational region.

Terminology in *v.in.pygbif* is oriented on the *Darwin Core* standard: <http://rs.tdwg.org/dwc/>.

Please note that the GBIF Search API has a hard limit of 200,000 occurrences per request. If you want to fetch more records, either subdivide your area of interest or split up your search by using different search criteria.

When a list of taxa is given as input, *v.in.pygbif* issues a search for each

taxon individually. Thus, in order to split up a search it is recommended to either use different filters on time or space.

EXAMPLES:

```
# Check matching taxon names and alternatives in GBIF:
v.in.pygbif taxa="Poa,Plantago" rank=genus -p

# Check matching taxon names and alternatives in GBIF and print output in
table:
v.in.pygbif taxa="Poa pratensis,Plantago media,Acer negundo" rank=species -
t

# Get number of occurrences for two geni:
v.in.pygbif taxa="Poa,Plantago" rank=genus -o

# Get number of occurrences for two species:
v.in.pygbif taxa="Poa pratensis,Plantago media" rank=species -o

# Fetch occurrences for two species into a map for each species:
v.in.pygbif taxa="Poa pratensis,Plantago media" rank=species output=gbif -i
```

SEE ALSO:

[v.in.gbif](#)

REFERENCES:

<http://pygbif.readthedocs.io/en/latest/index.html>

<http://www.gbif.org>

<http://www.gbif.org/developer/summary>

AUTHOR:

Stefan Blumentrath, Norwegian Institute for Nature Research, Oslo, Norway
Helmut Kudrnovsky

SOURCE CODE

Available at: [v.in.pygbif source code](#) ([history](#))

[Main index](#) | [Vector index](#) | [Topics index](#) | [Keywords index](#) | [Graphical index](#) | [Full index](#)

© 2003-2017 [GRASS Development Team](#), GRASS GIS 7.2.1svn Reference Manual