

Package: rinat (via r-universe)

August 30, 2026

Type Package

Title Access 'iNaturalist' Data Through APIs

Version 0.1.10.9000

Date 2025-08-31

Description A programmatic interface to the API provided by the 'iNaturalist' website <<https://www.inaturalist.org/>> to download species occurrence data submitted by citizen scientists.

License MIT + file LICENSE

Imports httr, curl, plyr, jsonlite, ggplot2, maps, sp

RoxygenNote 7.3.2

URL <https://docs.ropensci.org/rinat/> (website)

<https://github.com/ropensci/rinat> (devel)

Suggests knitr, rmarkdown, testthat, sf, dplyr, lubridate, ggforce, stringr, purrr

VignetteBuilder knitr

Encoding UTF-8

Config/pak/sysreqs libssl-dev

Repository <https://ropensci.r-universe.dev>

Date/Publication 2025-08-30 16:29:03 UTC

RemoteUrl <https://github.com/ropensci/rinat>

RemoteRef master

RemoteSha 44c9f1c892765e40948b2a286a53f302328ccf03

Contents

get_inat_obs	2
get_inat_obs_id	4
get_inat_obs_project	4

get_inat_obs_user	5
get_inat_taxon_stats	6
get_inat_user_stats	7
inat_map	8

Index	9
--------------	----------

get_inat_obs	<i>Download iNaturalist data</i>
--------------	----------------------------------

Description

Primary function to retrieve observations from iNaturalist, allows users to search for data, or just filter results by a subset of what is offered by the API.

Usage

```
get_inat_obs(  
  query = NULL,  
  taxon_name = NULL,  
  taxon_id = NULL,  
  place_id = NULL,  
  quality = NULL,  
  geo = NULL,  
  annotation = NULL,  
  year = NULL,  
  month = NULL,  
  day = NULL,  
  bounds = NULL,  
  photo_license = NULL,  
  maxresults = 100,  
  meta = FALSE  
)
```

Arguments

query	Query string for a general search.
taxon_name	Filter by iNat taxon name. Note that this will also select observations of descendant taxa. Note that names are not unique, so if the name matches multiple taxa, no observations may be returned.
taxon_id	Filter by iNat taxon ID. Note that this will also select observations of descendant taxa.
place_id	Filter by iNat place ID.
quality	The quality grade to be used. Must be either "casual" or "research". If left blank both will be returned.
geo	Flag for returning only results that are georeferenced, TRUE will exclude non-georeferenced results, but they cannot be excluded.

annotation	Filter by annotation. Vector of length 2, the first element being the term ID (e.g. "1" for life stage) and the second element being the value ID (e.g. "2" for adult). Refer to the annotation documentation to know which values to use.
year	Return observations only in that year (can only be one year, not a range of years).
month	Return observations only by month, must be numeric, 1...12
day	Return observations only on a given day of the month, 1...31
bounds	A bounding box of longitude (-180 to 180) and latitude (-90 to 90) to search within. It is a vector in the form of southern latitude, western longitude, northern latitude, and eastern longitude. Alternatively supply an sf or sp object from which the bounding box will be derived.
photo_license	Filter by photo license. Values can be CC variations ("CC0", "CC-BY", "CC-BY-NC", "CC-BY-SA", "CC-BY-ND", "CC-BY-NC-SA", "CC-BY-NC-ND"), "any" (any CC licence), or "none" (all rights reserved). Note that an observation and its photos don't necessarily have the same licence.
maxresults	The maximum number of results to return. Should not be a number higher than 10000.
meta	(logical) If TRUE, the output of this function is a list with metadata on the output and a data.frame of the data. If FALSE (default), just the data.frame.

Value

A dataframe of the number of observations requested.

Note

Filtering doesn't always work with the query parameter for some reason (a problem on the API end). If you want to filter by time, it's best to use the scientific name and put it in the 'taxa' field, and not in the query field. Another issue is that the query parameter will search the entire entry, so it is possible to get unintended results. Depending on your use case it may be advisable to use the "taxon" field instead of the query field.

Examples

```
## Not run:
## Make a standard query
get_inat_obs(query = "Monarch Butterfly")

## Restrict to juveniles thanks to annotations
get_inat_obs(query = "possum", annotation = c(1, 8))

## Filter by a bounding box of Northern California
bounds <- c(38.44047, -125, 40.86652, -121.837)
get_inat_obs(query = "Mule Deer", bounds = bounds)

## Filter by taxon, allows higher order filtering,
## Here we can search for just stone flies (order Plecoptera)
get_inat_obs(taxon_name = "Plecoptera")
```

```
## Get metadata (the number of results found on the server)
out <- get_inat_obs(query = "Monarch Butterfly", meta = TRUE)
out$meta

## End(Not run)
```

get_inat_obs_id	<i>Get information on a specific observation</i>
-----------------	--

Description

Get information on a specific observation

Usage

```
get_inat_obs_id(id)
```

Arguments

`id` a single id for an iNaturalist observation record

Value

a list with full details on a given record

Examples

```
## Not run:
m_obs <- get_inat_obs(query="Monarch Butterfly")
get_inat_obs_id(m_obs$id[1])

## End(Not run)
```

get_inat_obs_project	<i>Download observations or info from a project</i>
----------------------	---

Description

Retrieve observations from a particular iNaturalist project. This function can be used to get either observations or information from a project by project name or ID.

Usage

```
get_inat_obs_project(grpid, type = c("observations", "info"), raw = FALSE)
```

Arguments

groupid	Name of the group as an iNaturalist slug or group ID.
type	Character, either "observations" or "info". "observations" returns all observations, and "info" returns project details similar to what you can find on a project's page.
raw	Logical. If TRUE and searching for project info, returns the raw output of parsed JSON for that project. Otherwise just some basic information is returned as a list.

Details

An iNaturalist slug is usually the project name as a single string with words separated by hyphens. For instance, the project "World Oceans Week 2022" has a slug of "world-oceans-week-2022", which you can find by searching projects on iNaturalist and looking at the [project's page's URL](#).

Examples

```
## Not run:
get_inat_obs_project(354, type = "observations")
get_inat_obs_project("crows-in-vermont", type="info",raw=FALSE)

## End(Not run)
```

get_inat_obs_user	<i>Download observations for a user</i>
-------------------	---

Description

Get all the observations of a specific iNaturalist user.

Usage

```
get_inat_obs_user(username, maxresults = 100)
```

Arguments

username	username of the iNaturalist user to fetch records
maxresults	the maximum number of results to return

Value

a list with full details on a given record

Examples

```
## Not run:
m_obs <- get_inat_obs(query="Monarch Butterfly")
get_inat_obs_user(as.character(m_obs$user_login[1]))

## End(Not run)
```

get_inat_taxon_stats *Get stats on taxon counts*

Description

Get stats on taxa within a constrained range. This range can be by user, place, project, day or date range. Output will be a count of the total number of taxa observed at each taxonomic level.

Usage

```
get_inat_taxon_stats(
  date = NULL,
  date_range = NULL,
  place = NULL,
  project = NULL,
  uid = NULL
)
```

Arguments

date	Retrieve observations on a specific date. Must be a string in the form "YYYY-MM-DD".
date_range	A vector of two dates defining a date range, each in the form "YYYY-MM-DD".
place	Get taxon stats by place. You can find place IDs on the iNaturalist website . Must be a numeric ID.
project	Get taxon stats by a project's numeric ID.
uid	Get taxon stats by user ID (string).

Value

A vector listing counts of observations at each level of identification possible (species, genus, etc.)

Examples

```
## Not run:
counts <- get_inat_taxon_stats(date = "2010-06-14")

## End(Not run)
```

get_inat_user_stats *Get stats on users*

Description

Get stats on which users reported the most species or had the most observations within a given range. This range can be by user, place, project, day or date range. Output will be a count of the total number of taxa observed at each taxonomic level.

Usage

```
get_inat_user_stats(  
  date = NULL,  
  date_range = NULL,  
  place = NULL,  
  project = NULL,  
  uid = NULL  
)
```

Arguments

date	Retrieve observations on a specific date. Must be a string in the form "YYYY-MM-DD".
date_range	A vector of dates defining a range, each in the form "YYYY-MM-DD".
place	Get taxon stats by place. You can find place IDs on the iNaturalist website . Must be a numeric ID.
project	Get taxon stats by a project's numeric ID.
uid	Get taxon stats by user ID (string).

Value

A list containing two data.frames, with users sorted by either number of observations or number of species observed.

Examples

```
## Not run:  
counts <- get_inat_user_stats(date = "2010-06-14")  
  
## End(Not run)
```

inat_map	<i>Plot iNaturalist observations</i>
----------	--------------------------------------

Description

Plot observations from iNaturalist. You have the option of automatically plotting, or returning a ggplot map object that you can add layers onto.

Usage

```
inat_map(data, map = "usa", subregion = ".", plot = TRUE)
```

Arguments

data	data frame of iNaturalist observations.
map	the map area to plot, default is USA. See the same map argument in map_data for available areas.
subregion	the name of the subregion to plot. See the region argument in map_data for more details.
plot	a logical value. TRUE plots the map object and returns it, and FALSE returns a ggplot object that you can modify and plot later.

Value

A ggplot map object.

Examples

```
## Not run:
m_obs <- get_inat_obs(taxon_name = "Ambystoma maculatum")
salamander_map <- inat_map(m_obs, plot = FALSE)
### Now we can modify the returned map
salamander_map + borders("state") + theme_bw()

## End(Not run)
```

Index

`get_inat_obs`, [2](#)
`get_inat_obs_id`, [4](#)
`get_inat_obs_project`, [4](#)
`get_inat_obs_user`, [5](#)
`get_inat_taxon_stats`, [6](#)
`get_inat_user_stats`, [7](#)

`inat_map`, [8](#)

`map_data`, [8](#)