GBIF Data
with rgbif
Cheat Sheet

RopenSci

rgbif Package Interface

Taxonomic names

rgbif::name_lookup(…)
rgbif::name_backbone(…)
rgbif::name_usage(…)
rgbif::name_suggest(…)

Data providers and datasets

rgbif::datasets(…)
rgbif::dataset_metrics(…)
rgbif::dataset_search(…)
rgbif::dataset_suggest(…)
rgbif::installations(…)
rgbif::networks(…)
rgbif::nodes(…)
rgbif::organizations(…)

Occurrence data

rgbif::occ_count(…)
rgbif::count_facet(…)
rgbif::occ_get(…)
rgbif::occ_search(…)
rgbif::occ_data(…)
rgbif::occ_metadata(…)
rgbif::occ_issues(…)
rgbif::occ_issues_lookup(…)

Occurrence Downloads

rgbif::occ_download(…)
rgbif::occ_download_cancel(…)
rgbif::occ_download_cancel_staged(…)
rgbif::occ_download_get(…)
rgbif::occ_download_import(…)
rgbif::occ_download_list(…)
rgbif::occ_download_meta(…)

Names - Search

Using GBIF often starts by searching for names

rgbif::name_suggest()
Meant for rapid name suggestions, very coarse

rgbif::name_lookup()
Fuzzy search and extensive metadata returned - similar to dataset_search()

rgbif::name_usage()
Get particular metadata parts of a name - similar to datasets()

Names - GBIF Backbone

GBIF occurrences use their **backbone** taxonomy

We recommend getting a **taxonKey** via name_backbone()
Then use the key to get occurrence data

A basic search

```
rgbif::name_backbone(name = 'Helianthus')$usageKey
#> 3119134
```

Some taxa don't have a single match

```
rgbif::name_backbone(name='Satyrium')
#> "Multiple equal matches for Satyrium"
```

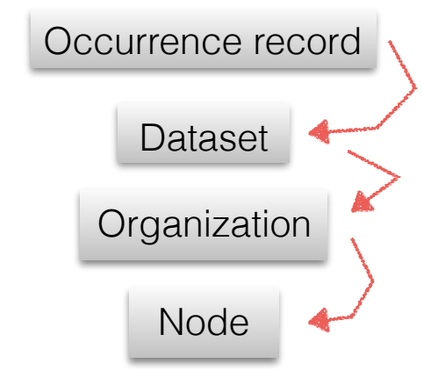
Think about filtering by rank

```
rgbif::name_backbone(name='Satyrium', kingdom="Plantae")$usageKey
#> 5307264
```

verbose = TRUE gives alternative matches

```
rgbif::name_backbone(name='Heliantus annuus', verbose=TRUE)
#> $data …
#> $alternatives …
```

Data Providers



Network group of datasets

Institution datasets are publshed through one of these

rgbif::datasets()
Get particular metadata parts of a dataset

rgbif::dataset_search()
Comprehensive dataset search, including facets

rgbif::dataset_suggest()
Fuzzy search - meant for rapid suggestions of datasets

rgbif::dataset_metrics()
Dataset metrics

rgbif::organizations()
Search organizations

rgbif::nodes()
Search nodes

rgbif::networks()
Search networks

rgbif::installations()
Search installations

Installation: install.packages("rgbif"); Develeopment version: devtools::install_github("ropensci/rgbif")

Occurrences - Count

Search by occurrence key

```
occ_count()
```

Search by many keys

```
occ_count()
```

Return data, meta, hierarchy, or all

```
occ_count()
```

Return only certain data fields

```
occ_count()
```

Occurrences - Search

```
occ_search(key = 3119195, limit = 2)
```

Search for occurrences. A compact tbl_df is returned, with metadata on top

```
Records found [31234]
Records returned [2]
No. unique hierarchies [1]
No. media records [2]
Args [taxonKey=3119195, limit=2, offset=0, fields=all]
First 10 rows of data

                name        key decimalLatitude decimalLongitude         issues
1 Helianthus annuus 1249279611        34.04810       -117.79884 cdround,gass84
2 Helianthus annuus 1248872560        37.81227         -8.82959
Variables not shown: datasetKey (chr), publishingOrgKey (chr), publishingCountry
     (chr), protocol (chr), lastCrawled (chr), lastParsed (chr), extensions
     (chr), basisOfRecord (chr), taxonKey (int), kingdomKey (int), phylumKey
     (int), classKey (int), orderKey (int), familyKey (int), genusKey (int),
     speciesKey (int), scientificName (chr), kingdom (chr), phylum (chr), order
```

```
occ_search(scientificname = 'Ursus')
```

You can search directly on scientific names as well

```
occ_search(datasetKey='7b5d6a48-f762-11e1-a439-00145eb45e9a')
```

Search on a particular dataset key

Or search on:

- country
- continent
- type status
- date
- elevation
- depth
- location
- institution
- and more …

```
occ_search(geometry='<WKT String>')
```

Geometry based searches with Well Known Text (WKT) strings is a powerful approach - With WKT you can specify incredibly complex polygons defining any country, water body, park, etc.

If occ_search is too slow for you…

```
rgbif::occ_data()
```

Has an almost identical interface as occ_search(), but is simplified for speed:

- we only return occurrence data, whereas occ_search returns data, taxonomic hierarchies, and media entries
- does minimal data parsing

Cleaning data

```
res <- occ_search(scientificname = 'Ursus')

library('magrittr')
```

Keep only records w/ gass84

```
res %>% occ_issues(gass84)
```

Split issues into separate columns

```
res %>% occ_issues(mutate = "split")
```

Expand to more descriptive names

```
res %>% occ_issues(mutate = "expand")
```

Split and expand

```
res %>% occ_issues(mutate = "split_expand")
```

Split, expand, and remove an issue

```
res %>% occ_issues(-cudc, mutate = "split_expand")
```

Occurrences - Downloads

Sometimes you want all the data … or at least lots of data

With occ_search() / occ_data(), GBIF limits you to 200,000 records

occ_download() gives a way to do what you can do in the GBIF website, but in R - with no limit on number of records requested

Usage

1. Query: occ_download('country = US')
2. Wait for your request to be processed
3. Check on request process with occ_meta('<request key>')
4. When ready, fetch your data with occ_get('<request key>') (you get a zip file on your machine)
5. Import the data with like occ_download_import(occ_get('<key>'))

Example queries

Search by taxon key

```
occ_download('taxonKey = 3119195')
```

Records with latitude > 50 degrees

```
occ_download('decimalLatitude > 50')
```

Records from the US

```
occ_download('country = US')
```

Occurrences
Get occurrences by occurrence key

Search by occurrence key

```
occ_get(key=766766824)
```

Search by many keys

```
occ_get(key=c(766766824, 620594291, 766420684))
```

Return data, meta, hierarchy, or all

```
occ_get(key=766766824, return='data')
```

Return only certain data fields

```
occ_get(key=766766824, fields=c('scientificName', 'lastCrawled', 'county'))
```


Installation: install.packages("rgbif"); Develeopment version: devtools::install_github("ropensci/rgbif")