Package: ArctosR (via r-universe)

May 17, 2026

Title An Interface to the 'Arctos' Database

Version 0.1.4

Maintainer Harlan R. Williams <`harlanrhwilliams@gmail.com`>

Date 2026-4-17

Description Performs requests to the 'Arctos' API to download data. Provides a set of builder classes for performing complex requests, as well as a set of simple functions for automating many common requests and workflows. More information about 'Arctos' can be found in Cicero et al. (2024) <doi:10.1371/journal.pone.0296478> or on their website <`https://arctosdb.org/`>.

URL `https://github.com/hrhwilliams/arctosr`, `https://hrhwilliams.github.io/ArctosR/`

BugReports `https://github.com/hrhwilliams/arctosr/issues`

Depends R (>= 4.1.0)

License GPL-3

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Imports R6 (>= 2.5.1), curl (>= 5.0.0), jsonlite (>= 1.8.0)

Suggests geodata, ggplot2, ggspatial, ggtext, knitr, kuenm2, maps, patchwork, rmarkdown, sf, terra, testthat (>= 3.0.0)

Config/testthat/edition 3

VignetteBuilder knitr

Config/pak/sysreqs libssl-dev

Repository https://hrhwilliams.r-universe.dev

Date/Publication 2026-04-17 17:41:56 UTC

RemoteUrl https://github.com/hrhwilliams/arctosr

RemoteRef HEAD

RemoteSha 88edd34306dcf29fb679449c993b9008691cec1d

Contents

`ArctosR-package` *ArctosR: An Interface to the Arctos Database*

Description

The `ArctosR` package provides a set of functions to help users perform requests to the Arctos API to download data. It provides a set of builder classes for performing complex requests, as well as a set of simple functions for automating many common requests and workflows.

About Arctos

Arctos is a collection management information system serving over 5 million records from natural and cultural history collections. Arctos integrates access to collections from disciplines such as anthropology, botany, entomology, ethnology, herpetology, geology, ichthyology, mammalogy, mineralogy, ornithology, paleontology, parasitology, as well as archival and cultural collections. The Arctos database is accessible through a web interface at `https://arctos.database.museum/` More information about Arctos, can be found at `https://arctosdb.org/about/`, and in Cicero et al. (2024) `doi:10.1371/journal.pone.0296478`.

Functions in ArctosR

`get_query_parameters`, `get_result_parameters`, `get_record_count`, `get_records`, `check_for_status`, `get_error_response`, `get_last_response_url`, `response_data`, `save_response_rds`, `read_response_rds`, `save_response_csv`, `expand_column`

Author(s)

Maintainer: Harlan R. Williams <`harlanrhwilliams@gmail.com`>

Authors:

- Marlon E. Cobos <`manubio13@gmail.com`>
- Jocelyn P. Colella <`colella@ku.edu`>
- Michelle S. Koo <`mkoo@berkeley.edu`>
- Vijay Barve <`vijay.barve@gmail.com`>

See Also

Useful links:

- `https://github.com/hrhwilliams/arctosr`
- Report bugs at `https://github.com/hrhwilliams/arctosr/issues`

`CatalogRequestBuilder` *CatalogRequestBuilder*

Description

CatalogRequestBuilder

CatalogRequestBuilder

Super class

`ArctosR::RequestBuilder` -> `CatalogRequestBuilder`

Methods

Public methods:

- `CatalogRequestBuilder$set_limit()`
- `CatalogRequestBuilder$set_query()`
- `CatalogRequestBuilder$set_filter()`
- `CatalogRequestBuilder$set_parts()`
- `CatalogRequestBuilder$set_attributes()`
- `CatalogRequestBuilder$set_components()`
- `CatalogRequestBuilder$set_columns()`

- `CatalogRequestBuilder$set_columns_list()`
- `CatalogRequestBuilder$from_previous_response()`
- `CatalogRequestBuilder$build_request()`
- `CatalogRequestBuilder$clone()`

Method **`set_limit()`:** Sets the limit on how many records to initially request from Arctos.

Usage:

`CatalogRequestBuilder$set_limit(limit)`

Arguments:

`limit` (`integer(1)`).

Returns: CatalogRequestBuilder.

Method **`set_query()`:** Sets the query parameters to use to search Arctos.

Usage:

`CatalogRequestBuilder$set_query(...)`

Arguments:

`query` (`list`).

Returns: CatalogRequestBuilder.

Method **`set_filter()`:** Sets the result parameters to use to filter out results.

Usage:

`CatalogRequestBuilder$set_filter(...)`

Arguments:

`query` (`list`).

Returns: CatalogRequestBuilder.

Method **`set_parts()`:** Set parts to query over.

Usage:

`CatalogRequestBuilder$set_parts(...)`

Arguments:

`parts` (`list`).

Returns: CatalogRequestBuilder.

Method **`set_attributes()`:** Set attributes to query over.

Usage:

`CatalogRequestBuilder$set_attributes(...)`

Arguments:

`attributes` (`list`).

Returns: CatalogRequestBuilder.

Method **`set_components()`:** Set components to query over.

Usage:

`CatalogRequestBuilder$set_components(...)`

Arguments:

`components` (`list`).

Returns: CatalogRequestBuilder.

Method **`set_columns()`:** Sets the columns in the dataframe returned by Arctos.

Usage:

`CatalogRequestBuilder$set_columns(...)`

Arguments:

`cols` (`list`).

Returns: CatalogRequestBuilder.

Method **`set_columns_list()`:** Sets the columns in the dataframe returned by Arctos.

Usage:

`CatalogRequestBuilder$set_columns_list(l)`

Arguments:

`cols` (`list`).

Returns: CatalogRequestBuilder.

Method **`from_previous_response()`:** Sets the columns in the dataframe returned by Arctos.

Usage:

`CatalogRequestBuilder$from_previous_response(response)`

Arguments:

`response` a response object from a previous request

Returns: FromResponseRequestBuilder.

Method **`build_request()`:** Send a request for data to Arctos with parameters specified by the other methods called on this class.

Usage:

`CatalogRequestBuilder$build_request()`

Returns: Response.

Method **`clone()`:** The objects of this class are cloneable with this method.

Usage:

`CatalogRequestBuilder$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

`check_for_status` *Check if the query object ends with a successful response*

Description

Checks if a response failed as part of a query.

Usage

```
check_for_status(query)
```

Arguments

`query`	A query object to check the return status of

Value

TRUE if the query succeeded, FALSE otherwise

Examples

```
library(ArctosR)

if (interactive()) {
  # query with an invalid column name 'paarts'
  query <- get_records(
    scientific_name = "Canis lupus", guid_prefix = "MSB:Mamm",
    columns = list("guid", "paarts", "partdetail")
  )

  check_for_status(query)
}
```

`expand_column` *Expand information of columns in JSON format*

Description

Expand all information contained in a JSON formatted column in a query object. Information is presented as nested data frames if needed.

Usage

```
expand_column(query, column_name)
```

Arguments

`query`	The query object with a JSON formatted column to be expanded.
`column_name`	(character) The name of the column to be expanded.

Value

Nothing.

Examples

```
library(ArctosR)

if (interactive()) {
  # Request to download all available data
  query <- get_records(
    scientific_name = "Canis lupus", guid_prefix = "MSB:Mamm",
    columns = list("guid", "parts", "partdetail")
  )

  # The partdetail column is a JSON list of parts and their attributes
  # This will convert the column to dataframes:
  expand_column(query, "partdetail")
}
```

`FromResponseRequestBuilder`

FromResponseRequestBuilder

Description

Builder for the case where a request is made with the context of a previous response.

Super class

`ArctosR::RequestBuilder` -> `FromResponseRequestBuilder`

Methods

Public methods:

- `FromResponseRequestBuilder$new()`
- `FromResponseRequestBuilder$request_more()`
- `FromResponseRequestBuilder$build_request()`
- `FromResponseRequestBuilder$clone()`

Method `new()`**:**

Usage:

`FromResponseRequestBuilder$new(response, records)`

Method `request_more()`: Request at most `count` more records from this response's original query

Usage:

`FromResponseRequestBuilder$request_more(count)`

Arguments:

`count` number of additional records to request

Returns: FromResponseRequestBuilder

Method `build_request()`: Perform the request.

Usage:

`FromResponseRequestBuilder$build_request()`

Returns: Request

Method `clone()`: The objects of this class are cloneable with this method.

Usage:

`FromResponseRequestBuilder$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

`get_error_response` *Get the last error message of a query object*

Description

Returns the error string returned from Arctos if the last response in a query object returned a status code other than HTTP 200 for debugging purposes.

Usage

```
get_error_response(query)
```

Arguments

`query`	A query object to return the error string of

Value

The error string of a failing response object, or "No error" if the query didn't fail

Examples

```
library(ArctosR)

if (interactive()) {
  # query with an invalid column name 'paarts'
  query <- get_records(
    scientific_name = "Canis lupus", guid_prefix = "MSB:Mamm",
    columns = list("guid", "paarts", "partdetail")
  )

  get_error_response(query)
}
```

`get_last_response_url` *Get the last URL used by a request in a query object*

Description

Returns the last URL used by a request in a query object

Usage

```
get_last_response_url(query)
```

Arguments

`query`	A query object to return the URL for

Value

The URL of the last performed request in a query object

Examples

```
library(ArctosR)

if (interactive()) {
  query <- get_records(
    scientific_name = "Canis lupus", guid_prefix = "MSB:Mamm",
    columns = list("guid", "parts", "partdetail")
  )

  get_last_response_url(query)
}
```

`get_query_parameters` *Get parameters to perform queries*

Description

Request information about all valid query parameters for querying Arctos.

Usage

```
get_query_parameters()
```

Value

Data frame listing valid query parameters and associated description and category. The returned columns are: `display`, `obj_name`, `category`, `subcategory`, `description`. All entries in `obj_name` are valid parameters to pass to `get_records` as keys.

Examples

```
library(ArctosR)

if (interactive()) {
  q <- get_query_parameters()
}
```

`get_record_count` *Count number of records in a query*

Description

Request from Arctos the total number of records that match a specific query. A list of possible query keys can be obtained from the output of `get_query_parameters`.

Usage

```
get_record_count(..., api_key)
```

Arguments

`...`	Query parameters and their values to pass to Arctos to search. For example, ‘scientific_name = "Canis lupus"“
`api_key`	(character) The API key to use for this request.

Value

The number of records matching the given query, as an integer.

Examples

```
library(ArctosR)

if (interactive()) {
  count <- get_record_count(
    scientific_name = "Canis lupus", guid_prefix = "MSB:Mamm"
  )
}
```

`get_records` *Get records from Arctos based on a query*

Description

Make a request to Arctos to return data based on a query. The columns (fields) returned are specified in the list defined in `columns`. A list of possible query keys can be obtained from the output of `get_query_parameters`.

Usage

```
get_records(..., api_key = NULL, columns = NULL, limit = NULL,
            filter_by = NULL, all_records = FALSE)
```

Arguments

`...`	Query parameters and their values to pass to Arctos to search. For example, `scientific_name = "Canis lupus"`
`api_key`	(character) The API key to use for this request.
`columns`	A list of columns to be returned in the table of records to be downloaded from Arctos.
`limit`	(numeric) The maximum number of records to download at once. Default is 100.
`filter_by`	An optional list of record attributes to filter results by.
`all_records`	(logical) If true, the request is performed multiple times to obtain data from Arctos until all records matching the query have been downloaded.

Value

A query object consisting of metadata for each request sent to Arctos to fulfill the user's query, and a data frame of records.

Examples

```
library(ArctosR)

if (interactive()) {
  # Request to download all available data
  query <- get_records(
    scientific_name = "Canis lupus", guid_prefix = "MSB:Mamm",
    columns = list("guid", "parts", "partdetail")
  )
}

if (interactive()) {
  # Request to download data about rodents examined for Orthohantavirus
  orthohantavirus_MSB <- get_records(guid_prefix="MSB:Mamm", taxon_name=Rodentia,
                                     filter_by=list("detected"="Orthohantavirus")
  )
}
```

`get_relationships` *Get the relationships (e.g. "host of") a cataloged item has*

Description

A cataloged item in Arctos can be related to any other number of items by relationships defined in the code table `ctid_references`. This function will return all items related by any such relationship in the table in a data frame.

Usage

```
get_relationships(guid, api_key = NULL)
```

Arguments

`guid`	The Arctos GUID of the item to query relationships over
`api_key`	(character) The API key to use for this request.

Value

a data frame of all related items. This contains URLs

Examples

```
library(ArctosR)

if (interactive()) {
  r <- get_relationships("MSB:Mamm:140026")
}
```

`get_request_url` *Get the first URL in a completed query*

Description

Returns the first URL used by a completed query which can be shared. The API key is automatically stripped from the URL for security.

Usage

```
get_request_url(query)
```

Arguments

`query`	A completed query returned from `get_records`

Value

A URL as a string

Examples

```
library(ArctosR)

if (interactive()) {
  q <- get_records(guid_prefix="MSB:Mamm")
  url <- get_request_url(q)
}
```

`get_result_parameters` *Get parameters to define valid results in queries*

Description

Request information about all valid result columns to request from Arctos.

Usage

```
get_result_parameters()
```

Value

Data frame listing valid result columns and associated description and category. The returned columns are: `display`, `obj_name`, `query_cost`, `category`, `description`, `default_order`. The names in `obj_name` are passed to `get_records` in the `columns` parameter as a `list`.

Examples

```
library(ArctosR)

if (interactive()) {
  r <- get_result_parameters()
}
```

`InfoRequestBuilder` *InfoRequestBuilder*

Description

Builder for a request for query parameter or result parameter documentation from Arctos. For a valid request, only one method to specify the type of request can be called.

Super class

`ArctosR::RequestBuilder` -> `InfoRequestBuilder`

Methods

Public methods:

- `InfoRequestBuilder$build_request()`
- `InfoRequestBuilder$clone()`

Method `build_request()`:

Usage:

`InfoRequestBuilder$build_request()`

Method `clone()`: The objects of this class are cloneable with this method.

Usage:

`InfoRequestBuilder$clone(deep = FALSE)`

Arguments:

`deep`	Whether to make a deep clone.

`Metadata` *Metadata*

Description

Metadata for a specific HTTP response from Arctos.

Methods

Public methods:

- `Metadata$to_list()`
- `Metadata$clone()`

Method `to_list()`**:**

Usage:

```
Metadata$to_list()
```

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
Metadata$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

`Query` *Query*

Description

The results of a user query. Able to accept multiple responses to increase the record count, or to add columns.

Methods

Public methods:

- `Query$catalog_request()`
- `Query$from_response_request()`
- `Query$info_request()`
- `Query$perform()`
- `Query$save_metadata_json()`
- `Query$save_records_csv()`
- `Query$expand_col()`
- `Query$get_responses()`

- `Query$clone()`

Method `catalog_request()`**:**

Usage:

`Query$catalog_request()`

Method `from_response_request()`**:**

Usage:

`Query$from_response_request()`

Method `info_request()`**:**

Usage:

`Query$info_request()`

Method `perform()`**:**

Usage:

`Query$perform(api_key = NULL)`

Method `save_metadata_json()`**:**

Usage:

`Query$save_metadata_json(file_path)`

Method `save_records_csv()`**:**

Usage:

`Query$save_records_csv(file_path, expanded = FALSE)`

Method `expand_col()`**:**

Usage:

`Query$expand_col(column_name)`

Method `get_responses()`**:**

Usage:

`Query$get_responses()`

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

`Query$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

`read_response_rds` *Read query records previously saved as an RDS file*

Description

Load in a query object saved to an RDS file.

Usage

```
read_response_rds(filename)
```

Arguments

`filename`	(character) The name of the file to load in.

Value

A query object

Examples

```
library(ArctosR)

if (interactive()) {
  # Request to download all available data
  query <- get_records(
    scientific_name = "Canis lupus", guid_prefix = "MSB:Mamm",
    columns = list("guid", "parts", "partdetail")
  )

  # Save the data in a .RDS file
  save_response_rds(query, "wolves.RDS")

  # Load the data from the .RDS just saved
  read_response_rds("wolves.RDS")


}
```

`Records` *Records*

Description

A (possibly nested) data frame of records returned by a static set of query and result parameters

Methods

Public methods:

- `Records$new()`
- `Records$append()`
- `Records$save_flat_csv()`
- `Records$save_nested_csvs()`
- `Records$expand_col()`
- `Records$clone()`

Method `new()`**:**

Usage:

`Records$new(df, tbl)`

Method `append()`**:**

Usage:

`Records$append(other)`

Method `save_flat_csv()`**:** Writes the data in the response object to a CSV file.

Usage:

`Records$save_flat_csv(file_path)`

Method `save_nested_csvs()`**:**

Usage:

`Records$save_nested_csvs(file_path)`

Method `expand_col()`**:** Expand a column of nested JSON tables in the response to a list of dataframes.

Usage:

`Records$expand_col(column)`

Arguments:

`col (string)`

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

`Records$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

`Request` *Request*

Description

A generic Arctos request. Not intended to be used directly. See InfoRequestBuilder and CatalogRequestBuilder.

Methods

Public methods:

- `Request$with_endpoint()`
- `Request$add_param()`
- `Request$add_params()`
- `Request$serialize()`
- `Request$perform()`
- `Request$from_raw_response()`
- `Request$clone()`

Method `with_endpoint()`**:**

Usage:

```
Request$with_endpoint(endpoint)
```

Method `add_param()`**:**

Usage:

```
Request$add_param(...)
```

Method `add_params()`**:**

Usage:

```
Request$add_params(l)
```

Method `serialize()`**:**

Usage:

```
Request$serialize()
```

Method `perform()`**:**

Usage:

```
Request$perform(api_key = NULL)
```

Method `from_raw_response()`**:**

Usage:

```
Request$from_raw_response(raw_response)
```

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

`Request$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

`RequestBuilder` *RequestBuilder*

Description

A builder for a generic Arctos request. Not to be used directly.

Methods

Public methods:

- `RequestBuilder$debug()`
- `RequestBuilder$build_request()`
- `RequestBuilder$clone()`

Method `debug()`**:** Turn on printing of debug information.

Usage:

`RequestBuilder$debug()`

Method `build_request()`**:**

Usage:

`RequestBuilder$build_request()`

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

`RequestBuilder$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

`Response` *Response*

Description

Response returned from Arctos.

Methods

Public methods:

- `Response$new()`
- `Response$set_start_index()`
- `Response$was_success()`
- `Response$is_empty()`
- `Response$has_json_content()`
- `Response$to_list()`
- `Response$to_records()`
- `Response$clone()`

Method `new()`**:**

Usage:

```
Response$new(request, raw_response)
```

Method `set_start_index()`**:**

Usage:

```
Response$set_start_index(start)
```

Method `was_success()`**:**

Usage:

```
Response$was_success()
```

Method `is_empty()`**:**

Usage:

```
Response$is_empty()
```

Method `has_json_content()`**:**

Usage:

```
Response$has_json_content()
```

Method `to_list()`**:**

Usage:

```
Response$to_list()
```

Method `to_records()`**:**

Usage:

```
Response$to_records(start = 0)
```

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
Response$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

`response_data` *Get query records as a data frame*

Description

Obtain the data frame with the records from a successful query.

Usage

```
response_data(query)
```

Arguments

`query`	The query object to extract the data frame from.

Value

A data frame with the information requested in the query.

Examples

```
library(ArctosR)

if (interactive()) {
  # Request to download all available data
  query <- get_records(
    scientific_name = "Canis lupus", guid_prefix = "MSB:Mamm",
    columns = list("guid", "parts", "partdetail")
  )

  # Grab the dataframe of records from the query
  df <- response_data(query)
}
```

`save_response_csv` *save_response_csv*

Description

Save the records inside the query object as a CSV file, optionally alongside metadata relating to the requests made to download the data.

Usage

```
save_response_csv(query, filename, expanded = FALSE, with_metadata = TRUE)
```

Arguments

`query`	The query object to be saved
`filename`	(character) Name of the file to be saved.
`expanded`	(logical) Setting this option to TRUE will create a folder of CSVs representing hierarchical data. See details.
`with_metadata`	Whether to save the metadata of the response as a JSON file along side the CSV or folder of CSVs.

Details

Some columns from Arctos are themselves tables, so to accurately represent the structure of the data, these inner tables can be saved as separate CSVs that are named according to which record they belong.

Value

Nothing.

Examples

```
library(ArctosR)

if (interactive()) {
  # Request to download all available data
  query <- get_records(
    scientific_name = "Canis lupus", guid_prefix = "MSB:Mamm",
    columns = list("guid", "parts", "partdetail")
  )

  # Save the response in a flat CSV with an additional metadata file in JSON
  save_response_csv(query, "msb-wolves.csv", with_metadata = TRUE)


}
```

`save_response_rds` *Write query records as an RDS file*

Description

Save the query object as an RDS file, which stores the entire state of the query and can be loaded at a later time.

Usage

```
save_response_rds(query, filename)
```

Arguments

`query`	The query object to be saved.
`filename`	(character) Name of the file to be saved.

Value

Nothing.

Examples

```
library(ArctosR)

if (interactive()) {
  # Request to download all available data
  query <- get_records(
    scientific_name = "Canis lupus", guid_prefix = "MSB:Mamm",
    columns = list("guid", "parts", "partdetail")
  )

  # Save the data in a .RDS file
  save_response_rds(query, "wolves.RDS")



}
```

Index