Package ‘rnassqs’

August 19, 2019

Type Package

Title Access the NASS 'Quick Stats' API

Version 0.5.0

Maintainer Nicholas Potter <potter.nicholas@gmail.com>

Description Interface to access data via the United States Department of Agriculture's National Agricultural Statistical Service (NASS) 'Quick Stats' web API <https://quickstats.nass.usda.gov/api>. Convenience functions facilitate building queries based on available parameters and valid parameter values.

URL https://github.com/ropensci/rnassqs

BugReports http://www.github.com/ropensci/rnassqs/issues

License MIT + file LICENSE

LazyData TRUE

Language en-US

Imports httr, jsonlite, stats, utils

Suggests assertthat, httptest, knitr, rmarkdown, testthat

RoxygenNote 6.1.1

Encoding UTF-8

VignetteBuilder knitr

NeedsCompilation no

Author Nicholas Potter [aut, cre],
Jonathan Adams [ctb],
Joseph Stachelek [ctb],
Julia Piaskowski [ctb],
Adam Sparks [rev],
Neal Richardson [ctb, rev]

Repository CRAN

Date/Publication 2019-08-19 19:10:02 UTC

R topics documented:

`rnassqs-package` *rnassqs-package: Access the NASS 'Quick Stats' API*

Description

rnassqs is a wrapper for the United States Department of Agriculture's National Agricultural Statistical Service (NASS) 'Quick Stats' API to enable getting NASS 'Quick Stats' data directly from R. Based on the httr API package guide.

Details

The functions in this package facilitate getting data from NASS 'Quick Stats'. It handles the API key checking and storage, authorization, and fetching of data.

Author(s)

Nicholas Potter

References

`http://quickstats.nass.usda.gov`

See Also

`http://quickstats.nass.usda.gov/api`

`nassqs` *Get data and return a data frame*

Description

The primary function in the `rnassqs` package, `nassqs` makes a HTTP GET request to the USDA-NASS Quick Stats API and returns the data parsed as a data.frame, plain text, or list. Various other functions make use of `nassqs` to make specific queries. For a data request the Quick Stats API returns JSON that when parsed to a data.frame contains 39 columns and a varying number of rows depending on the query. Unfortunately there is not a way to restrict the number of columns.

Usage

```
nassqs(..., as = c("data.frame", "text", "list"))
```

Arguments

`...`	either a named list of parameters or a series of parameters to form the query
`as`	whether to return a data.frame, list, or text string `nassqs_GET()`

Value

a data frame, list, or text string of requested data.

See Also

`nassqs_GET()`, `nassqs_yields()`, `nassqs_acres()`

Examples

```
# Get corn yields in Virginia in 2012
params <- list(commodity_name = "CORN",
               year = 2012,
               agg_level_desc = "COUNTY",
               state_alpha = "VA",
               statisticcat_desc = "YIELD")
yields <- nassqs(params)
head(yields)
```

nassqs_acres *Get NASS Area given a set of parameters.*

Description

Get NASS Area given a set of parameters.

Usage

```
nassqs_acres(..., area = c("AREA", "AREA PLANTED", "AREA BEARING",
  "AREA BEARING & NON-BEARING", "AREA GROWN", "AREA HARVESTED",
  "AREA IRRIGATED", "AREA NON-BEARING", "AREA PLANTED",
  "AREA PLANTED, NET"))
```

Arguments

`...`	either a named list of parameters or a series of parameters to form the query
`area`	the type of area to return. Default is all types.

Value

a data.frame of acres data

Examples

```
# Get Area bearing for Apples in Washington, 2012.
params <- list(
  commodity_desc = "APPLES",
  year = "2012",
  state_name = "WASHINGTON",
  agg_level_desc = "STATE"
)
area <- nassqs_area(params, area = "AREA BEARING")
head(area)
```

nassqs_auth *Get/Set the environmental variable NASSQS_TOKEN to the API key*

Description

If the API key is provided, sets the environmental variable. You can set your API key in four ways:

Usage

```
nassqs_auth(key)
```

Arguments

`key`	the API key (obtained from `https://quickstats.nass.usda.gov/api`)

Details

1. directly or as a variable from your `R` program: `nassqs_auth(key = "<your api key>"`
2. by setting `NASSQS_TOKEN` in your `R` environment file (you'll never have to enter it again).
3. by entering it into the console when asked (it will be stored for the rest of the session.)

Examples

```
# Set the API key
nassqs_auth(key = "<your api key>")
Sys.getenv("NASSQS_TOKEN")
```

`nassqs_check` *Check the response.*

Description

Check that the response is valid, i.e. that it doesn't exceed 50,000 records and that all the parameter values are valid. This is used to ensure that the query is valid before querying to reduce wait times before receiving an error.

Usage

```
nassqs_check(response)
```

Arguments

`response`	a `httr::GET()` request result returned from the API.

Value

nothing if check is passed, or an informative error if not passed.

`nassqs_fields` *Deprecated: Return list of NASS QS parameters.*

Description

Deprecated. Use `nassqs_params()` instead.

Usage

```
nassqs_fields(...)
```

Arguments

`...`	a parameter, series of parameters, or a list of parameters that you would like a description of. If missing, a list of all available parameters is returned.

`nassqs_GET` *Issue a GET request to the NASS 'Quick Stats' API*

Description

This is the workhorse of the package that provides the core request functionality to the NASS 'Quick Stats' API: `https://quickstats.nass.usda.gov/api`. In most cases `nassqs()` or other high-level functions should be used. `nassqs_GET()` uses `httr::GET()` to make a HTTP GET request, which returns a request object which must then be parsed to a data.frame, list, or other `R` object. Higher-level functions will do that parsing automatically. However, if you need access to the request object directly, `nassqs_GET()` provides that.

Usage

```
nassqs_GET(..., api_path = c("api_GET", "get_param_values",
  "get_counts"))
```

Arguments

`...`	either a named list of parameters or a series of parameters to use in the query
`api_path`	the API path that determines the type of request being made

Value

a `httr::GET()` response object

Examples

```
# Yields for corn in 2012 in Washington
params <- list(commodity_name = "CORN",
               year = 2012,
               agg_level_desc = "STATE",
               state_alpha = "WA",
               statisticcat_desc = "YIELD")

# Returns a request object that must be parsed either manually or
# by using nassqs_parse()
response <- nassqs_GET(params)
yields <- nassqs_parse(response)
head(yields)

# Get the number of records that would be returned for a given request
# Equivalent to 'nassqs_record_count(params)'
response <- nassqs_GET(params, api_path = "get_counts")
records <- nassqs_parse(response)
records

# Get the list of allowable values for the parameters 'statisticcat_desc'
# Equivalent to 'nassqs_param_values("statisticcat_desc")'
req <- nassqs_GET(list(param = "statisticcat_desc"),
                  api_path = "get_param_values")
statisticcat_desc_values <- nassqs_parse(req, as = "list")
head(statisticcat_desc_values)
```

`nassqs_params` *Return list of NASS QS parameters.*

Description

Contains a simple hard-coded list of all available parameters. If no parameter name is provided, returns a list of all parameters. More information can be found in the API documentation on parameters found at `https://quickstats.nass.usda.gov/api#param_define`.

Usage

```
nassqs_params(...)
```

Arguments

`...`	a parameter, series of parameters, or a list of parameters that you would like a description of. If missing, a list of all available parameters is returned.

Value

a list of all available parameters or a description of a subset

Examples

```
# Get a list of all available parameters
nassqs_params()

# Get information about specific parameters
nassqs_params("source_desc", "group_desc")
```

`nassqs_param_values` *Get all values for a specific parameter.*

Description

Returns a list of all possible values for a given parameter.

Usage

```
nassqs_param_values(param)
```

Arguments

`param`	the name of a NASS quickstats parameter

Value

a list containing all valid values for that parameter

Examples

```

  # See all values available for the statisticcat_desc field. Values may not
  # be available in the context of other parameters you set, for example
  # a given state may not have any 'YIELD' in blueberries if they don't grow
  # blueberries in that state.
  # Requires an API key:

  nassqs_param_values("source_desc")
```

`nassqs_parse` *Parse a response object from* `nassqs_GET()`.

Description

Returns a data frame, list, or text string. If a data.frame, all columns except `year` strings because the 'Quick Stats' data returns suppressed data as '(D)', '(Z)', or other character indicators which mean different things. Converting the value to a numerical results in NA, which loses that information.

Usage

```
nassqs_parse(req, as = c("data.frame", "list", "text"), ...)
```

Arguments

`req`	the GET response from `nassqs_GET()`
`as`	whether to return a data.frame, list, or text string
`...`	additional parameters passed to `jsonlite::fromJSON()` or `utils::read.csv()`

Value

a data frame, list, or text string of the content from the response.

Examples

```
  # Set parameters and make the request
  params <- list(commodity_name = "CORN",
                 year = 2012,
                 agg_level_desc = "STATE",
                 state_alpha = "WA",
                 statisticcat_desc = "YIELD")
  response <- nassqs_GET(params)

  # Parse the response to a data frame
  corn <- nassqs_parse(response, as = "data.frame")
  head(corn)

  # Parse the response into a raw character string.
  corn_text<- nassqs_parse(response, as = "text")
  head(corn_text)

  # Get a list of parameter values and parse as a list
  response <- nassqs_GET(list(param = "statisticcat_desc"),
                    api_path = "get_param_values")
  statisticcat_desc_values <- nassqs_parse(response, as = "list")
  head(statisticcat_desc_values)
```

`nassqs_record_count` *Get a count of number of records for given parameters.*

Description

Returns the number of records that fit a set of parameters. Useful if your current parameter set returns more than the 50,000 record limit.

Usage

```
nassqs_record_count(...)
```

Arguments

`...`	either a named list of parameters or a series of parameters to form the query

Value

integer that is the number of records that are returned from the API in response to the query

Examples

```
# Check the number of records returned for corn in 1995, Washington state
params <- list(
  commodity_desc = "CORN",
  year = "2005",
  agg_level_desc = "STATE",
  state_name = "WASHINGTON"
)

records <- nassqs_record_count(params)
records  # returns 17
```

`nassqs_yields` *Get yield records for a specified crop.*

Description

Returns yields for other specified parameters. This function is intended to simplify common requests.

Usage

```
nassqs_yields(...)
```

Arguments

`...`	either a named list of parameters or a series of parameters to form the query

Value

a data.frame of yields data

Examples

```
# Get yields for wheat in 2012, all geographies
params <- list(
  commodity_desc = "WHEAT",
  year = "2012",
  agg_level_desc = "STATE",
  state_alpha = "WA")

yields <- nassqs_yield(params)
head(yields)
```

Index