Package ‘rfieldclimate’

March 28, 2023

Type Package

Title Client for the 'FieldClimate' API

Version 0.1.1

Maintainer Eduard Szöcs <eduard.szoecs@basf.com>

Description Provides functionality to interact with the 'FieldClimate' API <https://api.fieldclimate.com/v2/docs/>.

License GPL-3

Imports digest, dplyr, httr, jsonlite, lubridate, magrittr, purrr, tidyr

Suggests covr, testthat

Encoding UTF-8

RoxygenNote 7.2.1

NeedsCompilation no

Author Eduard Szöcs [aut, cre], BASF SE [cph]

Repository CRAN

Date/Publication 2023-03-28 10:30:02 UTC

R topics documented:

fc_get_user 2
fc_headers 3
fc_parse_data 4
fc_ping 5
parse_sensor 6
parse_station 6
parse_timepoint 7

Index **8**

`fc_get_user` *Read user information*

Description

Read user information

List of user devices.

Get station information

Get min and max date of device data availability

Getdata between specified time periods.

Usage

```
fc_get_user(...)

fc_get_user_stations(...)

fc_get_station(station_id = NULL, ...)

fc_get_data(station_id = NULL, ...)

fc_get_data_range(
  station_id = NULL,
  data_group = c("raw", "hourly", "daily", "monthly"),
  from = NULL,
  to = NULL,
  ...
)
```

Arguments

`...`	additional arguments passed to `fc_request()`
`station_id`	station id to query
`data_group`	how to group data
`from`	time in unix timestamps since UTC, e.g. via as.integer(as.POSIXct(Sys.time()))
`to`	time in unix timestamps since UTC as.integer(as.POSIXct(Sys.time()))

Value

a list with user information.

a list with user stations information.

a list with station details.

a list with station metadata.

a list with station data.

Examples

```
## Not run:
  fc_get_user()

## End(Not run)
## Not run:
stations <- fc_get_user_stations()
stations

## End(Not run)
## Not run:
stations <- fc_get_user_stations()
fc_get_station(stations[[1]]$station_name)

## End(Not run)
## Not run:
stations <- fc_get_user_stations()
fc_get_data(stations[[1]]$station_name)

## End(Not run)
## Not run:
stations <- fc_get_user_stations()
fc_get_data_range(
  station_id = stations[[1]]$station_name,
  data_group = "raw",
  from = as.integer(as.POSIXct(Sys.time() - 60*60*24)),
  to =  as.integer(as.POSIXct(Sys.time())))

## End(Not run)
```

`fc_headers` *Create authentication headers*

Description

authentication is done via hmac, see `fc_headers()`.

Usage

```
fc_headers(
  method = c("GET", "PUT", "POST", "DELETE"),
  path = NULL,
  public_key = Sys.getenv("FC_PUBLIC_KEY"),
  private_key = Sys.getenv("FC_PRIVATE_KEY")
)

fc_request(
  method = c("GET", "PUT", "POST", "DELETE"),
  path = NULL,
  body = NULL,
  public_key = Sys.getenv("FC_PUBLIC_KEY"),
  private_key = Sys.getenv("FC_PRIVATE_KEY"),
  verbose = FALSE,
  timeout = 10
)
```

Arguments

`method`	request method
`path`	request path (required)
`public_key`	public key. Read by default from env variable `FC_PUBLIC_KEY`
`private_key`	private key. Read by default from env variable `FC_PRIVATE_KEY`
`body`	request body named list. Will be passed to `httr::VERB()` and form-encoded.
`verbose`	logical, should the request be printed?
`timeout`	number of seconds to wait for a response before giving up.

Value

an object of type "request" as returned by `httr::add_headers()`.

a list with the parsed response.

See Also

https://api.fieldclimate.com/v2/docs/#authentication-hmac

Examples

```
fc_headers(path = "/user", public_key = "invalid", private_key = "invalid")
## Not run:
fc_request("GET", "/user")

## End(Not run)
```

`fc_parse_data` *parse data into long data.frame*

Description

parse data into long data.frame

parse stations into data.frame

Usage

```
fc_parse_data(obj)

fc_parse_stations(obj)
```

Arguments

`obj`	stations object as returned by e.g. `fc_get_user_stations()`

Value

a data.frame with parsed data.

a data.frame with parsed station data.

Examples

```
## Not run:
stations <- fc_get_user_stations()
obj <- fc_get_data_range(
  station_id = stations[[1]]$station_name,
  data_group = "raw",
  from = as.integer(as.POSIXct(Sys.time() - 60*60*24)),
  to =  as.integer(as.POSIXct(Sys.time())))
fc_parse_data(obj)

## End(Not run)
## Not run:
stations <- fc_get_user_stations()
fc_parse_stations(stations)

## End(Not run)
```

`fc_ping` *Ping fieldclimate API*

Description

Ping fieldclimate API

Usage

```
fc_ping(timeout = 2)
```

Arguments

`timeout`	number of seconds to wait for a response before giving up.

Value

a logical whether the API is reachable or not.

Examples

```
## Not run:
fc_ping()

## End(Not run)
```

parse_sensor *parse a sensor*

Description

parse a sensor

Usage

```
parse_sensor(sensor)
```

Arguments

sensor	a sensor

parse_station *parse a station*

Description

parse a station

Usage

```
parse_station(station)
```

Arguments

station	a stations

`parse_timepoint` *parse a timepoint into a long data.frame*

Description

parse a timepoint into a long data.frame

Usage

```
parse_timepoint(timepoint)
```

Arguments

`timepoint`	a timepoint

Index

fc_get_data *(*fc_get_user*)*, 2
fc_get_data_range *(*fc_get_user*)*, 2
fc_get_station *(*fc_get_user*)*, 2
fc_get_user, 2
fc_get_user_stations *(*fc_get_user*)*, 2
fc_get_user_stations(), *5*
fc_headers, 3
fc_headers(), *3*
fc_parse_data, 4
fc_parse_stations *(*fc_parse_data*)*, 4
fc_ping, 5
fc_request *(*fc_headers*)*, 3
fc_request(), *2*

httr::add_headers(), *4*
httr::VERB(), *4*

parse_sensor, 6
parse_station, 6
parse_timepoint, 7