Package 'hubeau'

October 13, 2022

Type Package

Title Get Data from the French National Database on Water 'Hub'Eau'

Version 0.3.1

Date 2022-08-12

Description Collection of functions to help retrieving data from 'Hub'Eau' the
free and public French National APIs on water <https://hubeau.eaufrance.fr/>.

License MIT + file LICENSE

URL <https://inrae.github.io/hubeau/>,

<https://github.com/inrae/hubeau#readme>

BugReports <https://github.com/inrae/hubeau/issues>

Depends R (>= 2.10)

Imports httr, purrr, tibble, urltools

Suggests dplyr, spelling, testthat (>= 3.0.0)

Config/testthat/edition 3

Encoding UTF-8

RoxygenNote 7.2.0

Language en-US

NeedsCompilation no

Author David Dorchies [aut, cre] (<https://orcid.org/0000-0002-6595-7984>),
Pascal Irz [ctb]

Maintainer David Dorchies <david.dorchies@inrae.fr>

Repository CRAN

Date/Publication 2022-08-16 09:20:08 UTC

R topics documented:

---

convert_list_to_tibble

Convert list provided by the APIs into a tibble

Description

Convert list provided by the APIs into a tibble

Usage

```
convert_list_to_tibble(l)
```

Arguments

l   a list provided by the API (See doApiQuery)

Details

This function is used internally by all the retrieving data functions for converting data after the call
to doApiQuery.

Value

A tibble::tibble with one row by record and one column by field.

Examples

```
# To get the available APIs in the package
list_apis()

# To get the available endpoints in an API
list_endpoints("prelevements")

# To get available parameters in endpoint "chroniques" of the API "prelevements"
```

```
list_params(api = "prelevements", endpoint = "chroniques")

# To query the endpoint "chroniques" of the API "prelevements"
# on all devices in the commune of Romilly-sur-Seine in 2018
if(interactive()) {
resp <- doApiQuery(api = "prelevements",
                   endpoint = "chroniques",
                   params = list(code_commune_insee = "10323", annee = "2018"))
convert_list_to_tibble(resp)
}
```

---

doApiQuery   *Main internal functions for querying the Hub'Eau API endpoints*

Description

The function doQueryApi is called by all the function querying the API endpoints and return the raw data sent by the endpoint.

Usage

```
doApiQuery(api, endpoint, params)
```

Arguments

api	a [character](#) name of the API (e.g.: "indicateurs_services", "prelevements"...), see example for available APIs
endpoint	a [character](#) name of the endpoint, see example for available endpoints in an API
params	a [list](#) the list of parameters of the queries and their values in the format list(ParamName = "Param value", ...), use the function [list_params](#) for a list of the available filter parameters for a given API endpoint and see the API documentation for their description

Details

Pagination of the queries is handled automatically and the returned [list](#) is the concatenation of all the results sent by the API.

The functions get_[api]_[endpoint] call the function doQueryApi and parse the response in a [tibble::tibble](#) format for the user (See [convert_list_to_tibble](#)).

By default the user agent used for the query is "https://github.com/inrae/hubeau". You can redefined the user agent with the global option "hubeau.user_agent": options(hubeau.user_agent = "My user agent").

Value

A [list](#) with the concatenated results returned by the API.

Examples

```
# To get the available APIs in the package
list_apis()

# To get the available endpoints in an API
list_endpoints("prelevements")

# To get available parameters in endpoint "chroniques" of the API "prelevements"
list_params(api = "prelevements", endpoint = "chroniques")

# To query the endpoint "chroniques" of the API "prelevements"
# on all devices in the commune of Romilly-sur-Seine in 2018
if(interactive()) {
resp <- doApiQuery(api = "prelevements",
                   endpoint = "chroniques",
                   params = list(code_commune_insee = "10323", annee = "2018"))
convert_list_to_tibble(resp)
}
```

---

get_ecoulement_stations

Retrieve data from API "Ecoulement des cours d'eau"

Description

The data originate from the "ONDE" river low waters monitoring network. Available endpoints are:

- get_ecoulement_stations retrieves site data and locations
- get_ecoulement_observations retrieves flow information
- get_ecoulement_campagnes retrieves annual surveys

See the API documentation for available filter parameters: https://hubeau.eaufrance.fr/page/api-ecoulement

Usage

```
get_ecoulement_stations(params)

get_ecoulement_observations(params)

get_ecoulement_campagnes(params)
```

Arguments

params   a list the list of parameters of the queries and their values in the format list(ParamName
                = "Param value", ...), use the function list_params for a list of the available
                filter parameters for a given API endpoint and see the API documentation for
                their description

Value

A [tibble::tibble](#) with one row by record and one column by field.

Examples

```
if(interactive()) {
# Retrieve 2022 observation campaigns in the Jura French department
get_ecoulement_campagnes(
  list(code_departement = "39",
       date_campagne_min = "2022-01-01",
       date_campagne_max = "2022-12-31")
)

# Retrieve river stations
stations_39 <- get_ecoulement_stations(
  list(code_departement = "39",
       fields = "code_station,libelle_cours_eau,libelle_commune")
)
stations_39

# Get the query parameters for the requested API/endpoint
list_params(api = "ecoulement",
                      endpoint = "observations")

# Retrieve the river flow data in the Jura departement in 2022 with
# a selection of the fields
onde_39 <- get_ecoulement_observations(
  list(code_departement = "39",
       date_observation_min = "2022-01-01",
       date_observation_max = "2022-12-31",
       fields = "code_station,libelle_station,date_observation,libelle_ecoulement")
)
onde_39
}
```

get_hydrometrie_obs_elab

Retrieve data from API "Hydrométrie"

Description

Available endpoints are:

- `get_hydrometrie_obs_elab` retrieves hydrometric elaborate observations (daily/monthly mean flow)

- `get_hydrometrie_observations_tr` retrieves hydrometric "real time" observations ()

- `get_hydrometrie_sites` retrieves hydrometric sites

- `get_hydrometrie_stations` retrieves hydrometric stations

See the API documentation of each endpoint for available filter parameters: `https://hubeau.`
`eaufrance.fr/page/api-hydrometrie`

Usage

```
get_hydrometrie_obs_elab(params)

get_hydrometrie_observations_tr(params, entities = "station")

get_hydrometrie_sites(params, unique_site = TRUE)

get_hydrometrie_stations(params, code_sandre_reseau_station = FALSE)
```

Arguments

`params`	a list the list of parameters of the queries and their values in the format `list(ParamName = "Param value", ...)`, use the function list_params for a list of the available filter parameters for a given API endpoint and see the API documentation for their description
`entities`	1-length character string filtering the rows of the returned value, possible values are: "station" for filtering on station rows, "site" for filtering on site rows, "both" for keeping all the rows
`unique_site`	optional logical, if set to `FALSE` sites with several different locations produce one row by different location otherwise the first location found is used for fields `code_commune_site`, `libelle_commune`, `code_departement`, `code_region`, `libelle_region`, `libelle_departement`
`code_sandre_reseau_station`	
	optional logical indicating if `code_sandre_reseau_station` field is included in the result; if so, one line is added by item and other fields are repeated

Value

A tibble::tibble with one row by record and one column by field.

Examples

```
if(interactive()) {
# Retrieve the hydrometric sites in the department of Aube
get_hydrometrie_sites(list(code_departement = "10"))

# The same operation returning 2 rows for the site 'H0203020' which has 2 different locations
get_hydrometrie_sites(list(code_departement = "10"), unique_site = FALSE)

# Retrieve the hydrometric stations in the department of Aube
get_hydrometrie_stations(list(code_departement = "10"))

# Which parameters are available for endpoint "obs_elab" of API "hydrometrie"?
list_params("hydrometrie", "obs_elab")

# Retrieve the hydrometric monthly mean flow at site 'H0203020'
```

```
get_hydrometrie_obs_elab(list(code_entite = "H0203020", grandeur_hydro_elab = "QmM"))

# Retrieve the hydrometric daily mean flow at site 'H0203020' of the last 30 days
get_hydrometrie_obs_elab(
  list(code_entite = "H0203020",
  date_debut_obs_elab = format(Sys.Date() -30, "%Y-%m-%d"),
  grandeur_hydro_elab = "QmJ"))
}
```

---

get_indicateurs_services_communes

Retrieve data from API "Indicateurs des services"

Description

Retrieve performance indicators collected in drinking water and sanitation services in France.

See list_params and the API documentation for available filter parameters: `https://hubeau.eaufrance.fr/page/api-indicateurs-services`

Usage

```
get_indicateurs_services_communes(params)

get_indicateurs_services_indicateurs(params)

get_indicateurs_services_services(params)
```

Arguments

params   a list the list of parameters of the queries and their values in the format list(ParamName
                = "Param value", ...), use the function list_params for a list of the available
                filter parameters for a given API endpoint and see the API documentation for
                their description

Value

get_indicateurs_services_communes returns a tibble::tibble with one row by commune, by service and by year and the following columns:

- "code_commune_insee": character identifier of the commune
- "nom_commune": character name of the commune
- "codes_service": integer identifier of the drinking water supply and/or sanitation service
- "annee": integer year of the data
- The following columns are the performance indicators flagged by their respective codes. The documentation of these codes can be found at this URL: `https://www.services.eaufrance.fr/indicateurs/indicateurs`.

get_indicateurs_services_indicateurs returns a tibble::tibble with one row by service and by year and the following columns:

- "code_service": character identifier of the service

- "nom_service": character name of the service

- "numero_siren ": character SIREN identifier of the service

- "type_collectivite": character kind of community

- "mode_gestion": character management mechanism of the service

- "annee": integer year of the data

- "indicateur": value of the indicator

- "uri_indicateur": the link to the indicator documentation

get_indicateurs_services_services returns a tibble::tibble with one row by commune, by service and by year and the following columns:

- "code_service": character identifier of the service

- "nom_service": character name of the service

- "code_commune_insee": character identifier of the commune

- "nom_commune": character name of the commune

- "numero_siren ": character SIREN identifier of the service

- "type_collectivite": character kind of community

- "mode_gestion": character management mechanism of the service

- "annee": integer year of the data

- The following columns are the performance indicators flagged by their respective codes. The documentation of these codes can be found at this URL: https://www.services.eaufrance.fr/indicateurs/indicateurs.

Examples

```
if(interactive()) {
# Retrieve performance indicator time series in the commune of Romilly-sur-Seine
get_indicateurs_services_communes(list(code_commune = "10323"))

# Retrieve the drinking water withdrawal indicators of the year 2012 for all services
get_indicateurs_services_indicateurs(list(code_indicateur = "D102.0", annee = "2012"))

# Retrieve performance indicator time series of Romilly-sur-Seine with service details
get_indicateurs_services_services(list(code_commune = "10323"))
}
```

get_niveaux_nappes_stations
Retrieve data from API "Piézométrie"

Description

The available endpoints are:

- get_niveaux_nappes_stations retrieves list of piezometric stations
- get_nappes_chroniques retrieves piezometric archived time series
- get_nappes_chroniques_tr retrieves piezometric "real time" data

See the API documentation for available filter parameters: https://hubeau.eaufrance.fr/page/api-piezometrie

Usage

```
get_niveaux_nappes_stations(params)

get_niveaux_nappes_chroniques(params)

get_niveaux_nappes_chroniques_tr(params)
```

Arguments

params   a list the list of parameters of the queries and their values in the format list(ParamName = "Param value", ...), use the function list_params for a list of the available filter parameters for a given API endpoint and see the API documentation for their description

Value

A tibble::tibble with one row by record and one column by field.

Examples

```
if(interactive()) {
# Retrieve the hydrometric stations in the department of Aube
get_niveaux_nappes_stations(list(code_departement = "10"))

# Retrieve the archived observed piezometric level at station '07548X0009/F' (old BSS identifier)
# for the year 2020
df <- get_niveaux_nappes_chroniques(list(code_bss = "07548X0009/F",
                                         date_debut_mesure = "2020-01-01",
                                         date_fin_mesure = "2020-12-31"))

# Plot the water elevation (NGF)
plot(as.POSIXct(df$date_mesure), df$niveau_nappe_eau, type = "l")
```

```
# For retrieving the last real time observed piezometric level
# at station 'BSS001VZGZ' (new BSS identifier)
df <- get_niveaux_nappes_chroniques_tr(list(bss_id = "BSS001VZGZ"))

# Plot the water elevation (NGF)
plot(as.POSIXct(df$date_mesure), df$niveau_eau_ngf, type = "l")
}
```

get_poisson_observations

Retrieve data from API "Poisson"

Description

Available endpoint:

- get_poisson_observations retrieves data of scientific fishery operations

See the API documentation for available filter parameters: https://hubeau.eaufrance.fr/page/api-poisson

Usage

```
get_poisson_observations(params)
```

Arguments

params   a [list](#) the list of parameters of the queries and their values in the format list(ParamName
                  = "Param value", ...), use the function [list_params](#) for a list of the available
                  filter parameters for a given API endpoint and see the API documentation for
                  their description

Value

A [tibble::tibble](#) with one row by record and one column by field.

Examples

```
if(interactive()) {
# Get the query parameters for the requested API/endpoint
list_params(api = "poisson",
                      endpoint = "observations")

# Retrieve selected fields on a river fish sampled in Brest
library(dplyr)
fields <- paste("code_operation",
                "date_operation",
                "libelle_point_prelevement_aspe",
                "effectif_lot",
```

```
                 "code_alternatif_taxon",
                 sep = ",")
brest_fishes <- get_poisson_observations(
  list(
    libelle_commune = "Brest",
    fields = fields
    )
 ) %>%
 group_by_at(vars(-effectif_lot)) %>%
   summarise(nb_individals = sum(effectif_lot))

brest_fishes
}
```

get_prelevements_points_prelevement
Retrieve data from API "Prélèvements en eau"

Description

Available endpoints are:

- `get_prelevements_points_prelevement` retrieves withdrawal points

- `get_prelevements_ouvrages` retrieves withdrawal devices

- `get_prelevements_chroniques` retrieves time series of withdrawals

See the API documentation for available filter parameters: `https://hubeau.eaufrance.fr/page/api-prelevements-eau`

Usage

```
get_prelevements_points_prelevement(params)

get_prelevements_ouvrages(params)

get_prelevements_chroniques(params)
```

Arguments

params   a list the list of parameters of the queries and their values in the format `list(ParamName = "Param value", ...)`, use the function list_params for a list of the available filter parameters for a given API endpoint and see the API documentation for their description

Value

A tibble::tibble with one row by record and one column by field.

Examples

```
if(interactive()) {
# Retrieve the withdrawal points located in Romilly-sur-Seine
get_prelevements_points_prelevement(list(code_commune_insee = "10323"))

# Retrieve the withdrawal devices located in Romilly-sur-Seine
get_prelevements_ouvrages(list(code_commune_insee = "10323"))

# Retrieve the withdrawal time series of the devices located in Romilly-sur-Seine
get_prelevements_chroniques(list(code_commune_insee = "10323"))
}
```

---

hubeau | *hubeau: A package for retrieving data on the French databases on water 'Hub'Eau'*

Description

The 'hubeau' package provides functions for 'Hub'Eau' APIs and their related endpoints. These functions are named as follow: hubeau::get_[API]_[endpoint].

Currently available APIs and related endpoints are listed below.

API "Ecoulement des cours d'eau":

API documentation: https://hubeau.eaufrance.fr/page/api-ecoulement

Available functions:

- get_ecoulement_stations(): site data and locations
- get_ecoulement_observations(): flow observations collected during campaigns
- get_ecoulement_campagnes(): observation campaign information

API "Hydrométrie":

API documentation: https://hubeau.eaufrance.fr/page/api-hydrometrie

Available functions:

- get_hydrometrie_sites(): hydrometry sites (can contain several stations)
- get_hydrometrie_stations(): hydrometry stations
- get_hydrometrie_observations_tr(): hydrometry water level and discharge time series
- get_hydrometrie_obs_elab(): hydrometric elaborate observations (daily/monthly mean flow)

API "Indicateurs des services":

API documentation: https://hubeau.eaufrance.fr/page/api-indicateurs-services

Available functions:

- get_indicateurs_services_communes(): performance indicators by commune
- get_indicateurs_services_indicateurs(): performance indicators by indicator
- get_indicateurs_services_services(): performance indicators by commune for each service

API "Piézométrie":

API documentation: <https://hubeau.eaufrance.fr/page/api-piezometrie>

Available functions:

- `get_niveaux_nappes_chroniques()`: archived time series of piezometric stations
- `get_niveaux_nappes_chroniques_tr()`: real-time time series of piezometric stations
- `get_niveaux_nappes_stations()`: piezometric stations

API "Poisson":

API documentation: <https://hubeau.eaufrance.fr/page/api-poisson>

Available function:

- `get_poisson_observations()`: scientific fishery observations

API "Prélèvements en eau":

API documentation: <https://hubeau.eaufrance.fr/page/api-prelevements-eau>

Available functions:

- `get_prelevements_chroniques()`: time series of annual withdrawn volumes by device
- `get_prelevements_ouvrages()`: withdrawal devices (can contain several withdrawal points)
- `get_prelevements_points_prelevement()`: withdrawal points

---

list_apis   *List available Hub'Eau APIs, endpoints and filter parameters*

Description

`list_apis()` returns the list of available APIs in the package.

`list_endpoints()` returns the list of available endpoints for an API.

`list_params()` returns the list of available parameters for an API endpoint.

Usage

```
list_apis()

list_endpoints(api)

list_params(api, endpoint)
```

Arguments

api	a character name of the API (e.g.: "indicateurs_services", "prelevements"...), see example for available APIs
endpoint	a character name of the endpoint, see example for available endpoints in an API

Details

The listed APIs correspond to the term `[API]` used in the name of the functions `get_[API]_[endpoint]` used for querying the APIs.

Value

[character vector](#) of APIs, endpoints or filter parameters

Examples

```
# To get the available APIs in the package
list_apis()

# To get the available endpoints in an API
list_endpoints("prelevements")

# To get available parameters in endpoint "chroniques" of the API "prelevements"
list_params(api = "prelevements", endpoint = "chroniques")
```

Index