Package: epair (via r-universe)

November 4, 2024

Title EPA Data Helper for R

Version 1.1.0

Description Aid the user in making queries to the EPA API site found at https://aqs.epa.gov/aqsweb/documents/data_api. This package combines API calling methods from various web scraping packages with specific strings to retrieve data from the EPA API. It also contains easy to use loaded variables that help a user navigate services offered by the API and aid the user in determining the appropriate way to make a an API call.

Depends R (>= 3.3.3)

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.2.0

Imports httr (>= 1.4.1), jsonlite (>= 1.6.1), R.cache (>= 0.15.0), stringr

Suggests httptest (>= 4.1.0), knitr (>= 1.33), rmarkdown (>= 2.8), testthat (>= 3.0.2)

BugReports https://github.com/ropensci/epair/issues

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

VignetteBuilder knitr

SystemRequirements pandoc

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

RemoteUrl https://github.com/ropensci/epair

RemoteRef master

RemoteSha 6eedb899c8e774d0c1251173fd60e81d79b89788

Contents

`add.variables` *Add variables to a query*

Description

Add variables to a query

Usage

```
add.variables(query, variables)
```

Arguments

`query`	A URL containing authentication for the EPA API site.
`variables`	A list of variables. Each variable should be declared with the appropriate name. Consult VARIABLE.TYPES for the right names.

Value

A URL consisting of query + variables.

Examples

```
## Not run:
endpoint <- "dailyData/byState"
variable.list <- list("state" = '37',
                      "bdate" = '20200101',
                      "edate" = '20200102',
                      "param" = '44201')
call <- epair::create.base.call(endpoint)
call <- add.variables(call, variable.list)
call

## End(Not run)
```

`clear.all.cached` *Removes all cached memory of perform.call*

Description

Removes all cached memory of perform.call

Usage

```
clear.all.cached(directory = "/cache")
```

Arguments

`directory`	Place inside user-level cache directory that was used to store the cached data previously. Default: "/cache".

Value

'Done' if data was successfully forgotten, error message if cache directory was not found

Examples

```
## Not run:
clear.all.cached()

## End(Not run)
```

clear.cached *Removes memory of cached perform.call data for specific parameters*

Description

Removes memory of cached perform.call data for specific parameters

Usage

```
clear.cached(endpoint, variables = list(), directory = "/cache")
```

Arguments

endpoint	An endpoint from the available EPA API endpoints
variables	A list of variables or a single variable to filter the EPA API endpoint.
directory	Place inside user-level cache directory that was used to store the cached data previously. Default: "/cache".

Value

TRUE if data was successfully forgotten, error message if cached data was not found

Examples

```
## Not run:
endpoint <- 'list/states'
clear.cached(endpoint)

## End(Not run)
```

create.authentication *Generate the string authentication needed for EPA API*

Description

Generate the string authentication needed for EPA API

Usage

```
create.authentication(email, key)
```

Arguments

email	Email registered with EPA API
key	Key obtained from EPA API. Register your email for a key here https://aqs.epa.gov/aqsweb/documents/dat

Value

A string with authentication info. It looks like '&email=user_email&key=user_key'.

Examples

```
auth <- create.authentication("myemail@domain.com", "myapikey")
auth
```

`create.base.call` *Make the first call when forming a query.*

Description

Make the first call when forming a query.

Usage

```
create.base.call(endpoint)
```

Arguments

`endpoint`	Endpoint for forming a query. See ENDPOINTS for all available endpoints. See SERVICES if you know the service but not the endpoint.

Value

A URL string containing authentication for the call.

Examples

```
## Not run:
endpoint <- "list/states"
call <- epair:::create.base.call(endpoint)
call

## End(Not run)
```

`endpoints` *Endpoints available in the EPA API*

Description

The endpoints vector contains all endpoints available in the EPA API. To get endpoints directly from the site, use get.endpoints().

get.transpose *Transpose a data frame*

Description

Transpose a data frame

Usage

```
get.transpose(df)
```

Arguments

df	Data frame to be transposed

Value

The transposed data frame. First variable entries become column names.

Examples

```
service <- c("Sign up")
description <- c("Email will\r\n\t\t\t\t\t\t\t be sent to the registered
address from aqsdatamart@epa.gov.")
df <- data.frame(service, description)
t.df <- epair:::get.transpose(df)
t.df
```

get_all_mas *Get Monitoring Agencies.*

Description

Get Monitoring Agencies.

Usage

```
get_all_mas()
```

Value

API response containing all Monitoring Agencies.

Examples

```
## Not run:
mas <- get_all_mas()
mas$Data

## End(Not run)
```

`get_all_pqaos` *Get Primary Quality Assurance Organizations.*

Description

Get Primary Quality Assurance Organizations.

Usage

```
get_all_pqaos()
```

Value

API response containing all Primary Quality Assurance Organizations.

Examples

```
## Not run:
pqaos <- get_all_pqaos()
pqaos$Data

## End(Not run)
```

`get_annual_summary_in_bbox`
Get annual summary data in a bounding box (lat, long).

Description

Get annual summary data in a bounding box (lat, long).

Usage

```
get_annual_summary_in_bbox(
  bdate,
  edate,
  param,
  minlat,
  maxlat,
  minlong,
  maxlong,
  cached = TRUE,
  cache_directory = "/cache",
  cbdate = NULL,
  cedate = NULL
)
```

Arguments

bdate	Beginning date to check. Year, month, day format.
edate	Ending date to check. Year, month, day format.
param	Pollutant parameter that site is measuring.
minlat	Minimum latitude coordinate.
maxlat	Maximum latitude coordinate.
minlong	Minimum longitude coordinate.
maxlong	Maximum longitude coordinate.
cached	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
cache_directory	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)
cbdate	Beginning date of last change to DB. (Optional)
cedate	Ending date of last change to DB. (Optional)

Value

API response containing annual summary data in a bounding box.

Examples

```
## Not run:
bdate <- "20200101"
edate <- "20200102"
param <- "42401"
minlat <- 33.3
maxlat <- 33.6
minlong <- -87
maxlong <- -86.7
```

```
result <- get_annual_summary_in_bbox(bdate,
                                     edate,
                                     param,
                                     minlat,
                                     maxlat,
                                     minlong,
                                     maxlong)
result$Data

## End(Not run)
```

`get_annual_summary_in_cbsa` *Get annual summary data in a Core Based Statistical Area.*

Description

Get annual summary data in a Core Based Statistical Area.

Usage

```
get_annual_summary_in_cbsa(
  bdate,
  edate,
  param,
  cbsa,
  cached = TRUE,
  cache_directory = "/cache",
  cbdate = NULL,
  cedate = NULL
)
```

Arguments

`bdate`	Beginning date to check. Year, month, day format.
`edate`	Ending date to check. Year, month, day format.
`param`	Pollutant parameter that site is measuring.
`cbsa`	An encoding for a Core Base Statiscal Area. If unsure, use get_cbsas().
`cached`	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
`cache_directory`	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)
`cbdate`	Beginning date of last change to DB. (Optional)
`cedate`	Ending date of last change to DB. (Optional)

Value

API response containing quarterly summary data at the cbsa level.

Examples

```
## Not run:
bdate <- "20190101"
edate <- "20190601"
cbsa <- "16740"
param <- "42401"
result <- get_annual_summary_in_cbsa(bdate,
                                     edate,
                                     param,
                                     cbsa)
result$Data

## End(Not run)
```

---

get_annual_summary_in_county

Get annual summary data in a county.

Description

Get annual summary data in a county.

Usage

```
get_annual_summary_in_county(
  bdate,
  edate,
  state.fips,
  county,
  param,
  cached = TRUE,
  cache_directory = "/cache",
  cbdate = NULL,
  cedate = NULL
)
```

Arguments

bdate	Beginning date to check. Year, month, day format.
edate	Ending date to check. Year, month, day format.
state.fips	State FIPS code. Use get_state_fips() if unsure.
county	County code. Use get_counties_in_state() if unsure.
param	Pollutant parameter that site is measuring.

cached	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
cache_directory	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)
cbdate	Beginning date of last change to DB. (Optional)
cedate	Ending date of last change to DB. (Optional)

Value

API response containing annual summary data in a county.

Examples

```
## Not run:
bdate <- "20200101"
edate <- "20200102"
state.fips <- "37"
county <- "001"
param <- "42401"
result <- get_annual_summary_in_county(bdate,
                                       edate,
                                       state.fips,
                                       county,
                                       param)
result$Data

## End(Not run)
```

---

get_annual_summary_in_site

Get annual summary data at a measurement site.

Description

Get annual summary data at a measurement site.

Usage

```
get_annual_summary_in_site(
  bdate,
  edate,
  param,
  state.fips,
  county,
  site,
  cached = TRUE,
  cache_directory = "/cache",
```

```
  cbdate = NULL,
  cedate = NULL
)
```

Arguments

bdate	Beginning date to check. Year, month, day format.
edate	Ending date to check. Year, month, day format.
param	Pollutant parameter that site is measuring.
state.fips	State FIPS code. Use get_state_fips() if unsure.
county	County code. Use get_counties_in_state() if unsure.
site	Measurement site code. Use get_sites_by_county() if unsure.
cached	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
cache_directory	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)
cbdate	Beginning date of last change to DB. (Optional)
cedate	Ending date of last change to DB. (Optional)

Value

API response containing annual summary data for a site.

Examples

```
## Not run:
bdate <- "20170618"
edate <- "20170618"
state.fips <- "37"
county <- "183"
site <- "0014"
param <- "44201"
result <- get_annual_summary_in_site(bdate,
                                     edate,
                                     param,
                                     state.fips,
                                     county,
                                     site)
result$Data

## End(Not run)
```

`get_annual_summary_in_state` *Get annual summary data in a state.*

Description

Get annual summary data in a state.

Usage

```
get_annual_summary_in_state(
  bdate,
  edate,
  state.fips,
  param,
  cached = TRUE,
  cache_directory = "/cache",
  cbdate = NULL,
  cedate = NULL
)
```

Arguments

`bdate`	Beginning date to check. Year, month, day format.
`edate`	Ending date to check. Year, month, day format.
`state.fips`	State FIPS code. Use get_state_fips() if unsure.
`param`	Pollutant parameter that site is measuring.
`cached`	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
`cache_directory`	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)
`cbdate`	Beginning date of last change to DB. (Optional)
`cedate`	Ending date of last change to DB. (Optional)

Value

API response containing annual summary data for a state.

Examples

```
## Not run:
bdate <- "20200101"
edate <- "20200102"
state.fips <- "37"
param <- "42401"
```

```
result <- get_annual_summary_in_state(bdate,
                                      edate,
                                      state.fips,
                                      param)
result$Data

## End(Not run)
```

`get_aqs_key` *Get an API key from the AQS API.*

Description

Get an API key from the AQS API.

Usage

```
get_aqs_key(user.email)
```

Arguments

`user.email`	Email provided by the user to get an API key for.

Examples

```
## Not run:
email <- "an.example.email@domain.com"
get_aqs_key(email)

## End(Not run)
```

`get_cbsas` *Get all Core Based Statistical Areas.*

Description

Get all Core Based Statistical Areas.

Usage

```
get_cbsas()
```

Value

API response containing a list of all Core Based Statistical Areas.

Examples

```
## Not run:
cbsas <- get_cbsas()
cbsas$Data

## End(Not run)
```

`get_counties_in_state` *Get all counties within a state.*

Description

Get all counties within a state.

Usage

```
get_counties_in_state(state.fips)
```

Arguments

`state.fips`	A state FIPS code. Use get_state_fips() to find the appropriate FIPS code.

Value

API response containing all counties and county codes within a given state.

Examples

```
## Not run:
state <- "37"
counties.in.state <- get_state_fips(state)
counties.in.state$Data

## End(Not run)
```

`get_daily_summary_in_bbox`
Returns daily summary data given a bounding box (lat, long).

Description

Returns daily summary data given a bounding box (lat, long).

Usage

```
get_daily_summary_in_bbox(
  bdate,
  edate,
  param,
  minlat,
  maxlat,
  minlong,
  maxlong,
  cached = TRUE,
  cache_directory = "/cache",
  cbdate = NULL,
  cedate = NULL
)
```

Arguments

bdate	Beginning date to check. Year, month, day format.
edate	Ending date to check. Year, month, day format.
param	Pollutant parameter that site is measuring.
minlat	Minimum latitude coordinate.
maxlat	Maximum latitude coordinate.
minlong	Minimum longitude coordinate.
maxlong	Maximum longitude coordinate.
cached	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
cache_directory	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)
cbdate	Beginning date of last change to DB. (Optional)
cedate	Ending date of last change to DB. (Optional)

Value

API response containing daily summary data bounded by lat long coords.

Examples

```
## Not run:
bdate <- "20200101"
edate <- "20200102"
param <- "42401"
minlat <- 33.3
maxlat <- 33.6
minlong <- -87
maxlong <- -86.7
```

```
result <- get_daily_summary_in_bbox(bdate,
                                    edate,
                                    param,
                                    minlat,
                                    maxlat,
                                    minlong,
                                    maxlong)
result$Data

## End(Not run)
```

`get_daily_summary_in_cbsa` *Get daily summary data in a Core Based Statistical Area.*

Description

Get daily summary data in a Core Based Statistical Area.

Usage

```
get_daily_summary_in_cbsa(
  bdate,
  edate,
  param,
  cbsa,
  cached = TRUE,
  cache_directory = "/cache",
  cbdate = NULL,
  cedate = NULL
)
```

Arguments

`bdate`	Beginning date to check. Year, month, day format.
`edate`	Ending date to check. Year, month, day format.
`param`	Pollutant parameter that site is measuring.
`cbsa`	An encoding for a Core Base Statiscal Area. If unsure, use get_cbsas().
`cached`	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
`cache_directory`	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)
`cbdate`	Beginning date of last change to DB. (Optional)
`cedate`	Ending date of last change to DB. (Optional)

Value

API response containing daily summary data at the CBSA level.

Examples

```
## Not run:
bdate <- 20170101
edate <- 20170101
cbsa <- 16740
param <- 42602
result <- get_daily_summary_in_cbsa(bdate,
                                    edate,
                                    cbsa,
                                    param)
result$Data

## End(Not run)
```

---

get_daily_summary_in_county

Returns data summarized by day at the county level.

Description

Returns data summarized by day at the county level.

Usage

```
get_daily_summary_in_county(
  bdate,
  edate,
  state.fips,
  county,
  param,
  cached = TRUE,
  cache_directory = "/cache",
  cbdate = NULL,
  cedate = NULL
)
```

Arguments

bdate	Beginning date to check. Year, month, day format.
edate	Ending date to check. Year, month, day format.
state.fips	State FIPS code. Use get_state_fips() if unsure.
county	County code. Use get_counties_in_state() if unsure.
param	Pollutant parameter that site is measuring.

cached	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
cache_directory	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)
cbdate	Change begin date. (Optional)
cedate	Change end date. (Optional)

Value

API response containing daily data.

Examples

```
## Not run:
param <- 44201
bdate <- 20170618
edate <- 20170618
state <- 37
county <- 183
result <- get_daily_summary_in_county(bdate,
                                      edate,
                                      state,
                                      county,
                                      param)
result$Data

## End(Not run)
```

---

get_daily_summary_in_site

Returns data summarized by day at measurement site level.

Description

Returns data summarized by day at measurement site level.

Usage

```
get_daily_summary_in_site(
  bdate,
  edate,
  state.fips,
  county,
  site,
  param,
  cached = TRUE,
  cache_directory = "/cache",
```

```
  cbdate = NULL,
  cedate = NULL
)
```

Arguments

bdate	Beginning date to check. Year, month, day format.
edate	Ending date to check. Year, month, day format.
state.fips	State FIPS code. Use get_state_fips() if unsure.
county	County code. Use get_counties_in_state() if unsure.
site	Measurement site code. Use get_sites_by_county() if unsure.
param	Pollutant parameter that site is measuring.
cached	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
cache_directory	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)
cbdate	Change begin date. (Optional)
cedate	Change end date. (Optional)

Value

API response containing daily data.

Examples

```
## Not run:
param <- 44201
bdate <- 20170618
edate <- 20170618
state <- 37
county <- 183
site <- 0014
result <- get_daily_summary_in_site(bdate,
                                    edate,
                                    state,
                                    county,
                                    site,
                                    param)
result$Data

## End(Not run)
```

`get_daily_summary_in_state` *Returns daily data at the state level.*

Description

Returns daily data at the state level.

Usage

```
get_daily_summary_in_state(
  bdate,
  edate,
  state.fips,
  param,
  cached = TRUE,
  cache_directory = "/cache",
  cbdate = NULL,
  cedate = NULL
)
```

Arguments

bdate	Beginning date to check. Year, month, day format.
edate	Ending date to check. Year, month, day format.
state.fips	State FIPS code. Use get_state_fips() if unsure.
param	Pollutant parameter that site is measuring.
cached	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
cache_directory	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)
cbdate	Change begin date. (Optional)
cedate	Change end date. (Optional)

Value

API response containing daily data.

Examples

```
## Not run:
param <- 44201
bdate <- 20170618
edate <- 20170618
state <- 37
```

```
result <- get_daily_summary_in_state(bdate,
                                     edate,
                                     state,
                                     param)
result$Data

## End(Not run)
```

`get_fields_by_service` *Get fields required per service.*

Description

Get fields required per service.

Usage

```
get_fields_by_service(service)
```

Arguments

`service`	A service provided by EPA's AQS system.

Examples

```
## Not run:
result <- get_fields_by_service()
result$Data

## End(Not run)
```

`get_known_issues` *Get any known issues within the API.*

Description

Get any known issues within the API.

Usage

```
get_known_issues()
```

Examples

```
## Not run:
result <- get_known_issues()
result$Data

## End(Not run)
```

`get_monitors_in_bbox` *Get all monitoring sites within a bounding box (lat, long).*

Description

Get all monitoring sites within a bounding box (lat, long).

Usage

```
get_monitors_in_bbox(
  bdate,
  edate,
  param,
  minlat,
  maxlat,
  minlong,
  maxlong,
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

`bdate`	Beginning date to check. Year, month, day format.
`edate`	Ending date to check. Year, month, day format.
`param`	Pollutant parameter that site is measuring.
`minlat`	Minimum latitude coordinate.
`maxlat`	Maximum latitude coordinate.
`minlong`	Minimum longitude coordinate.
`maxlong`	Maximum longitude coordinate.
`cached`	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
`cache_directory`	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Value

API response containing operational information about the monitor.

Examples

```
## Not run:
bdate <- "20200101"
edate <- "20200102"
param <- "42401"
minlat <- 33.3
maxlat <- 33.6
minlong <- -87
maxlong <- -86.7
result <- get_monitors_in_bbox(bdate,
                               edate,
                               param,
                               minlat,
                               maxlat,
                               minlong,
                               maxlong)
result$Data

## End(Not run)
```

`get_monitors_in_cbsa` *Get monitors within a Core Based Statistical Area.*

Description

Get monitors within a Core Based Statistical Area.

Usage

```
get_monitors_in_cbsa(
  bdate,
  edate,
  param,
  cbsa,
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

`bdate`	Beginning date to check. Year, month, day format.
`edate`	Ending date to check. Year, month, day format.
`param`	Pollutant parameter that site is measuring.
`cbsa`	An encoding for a Core Base Statiscal Area. If unsure, use get_cbsas().
`cached`	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
`cache_directory`	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Value

API response containing operational information about the monitor.

Examples

```
## Not run:
bdate <- "20200101"
edate <- "20200102"
cbsa <- "16740"
param <- "42401"
result <- get_monitors_in_cbsa(bdate, edate, param, cbsa)
result$Data

## End(Not run)
```

`get_monitors_in_county` *Get all monitors in a county.*

Description

Get all monitors in a county.

Usage

```
get_monitors_in_county(
  bdate,
  edate,
  state.fips,
  county,
  param,
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

`bdate`	Beginning date to check. Year, month, day format.
`edate`	Ending date to check. Year, month, day format.
`state.fips`	State FIPS code. Use get_state_fips() if unsure.
`county`	County code. Use get_counties_in_state() if unsure.
`param`	Pollutant parameter that site is measuring.
`cached`	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
`cache_directory`	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Value

API response containing operational information about the monitor.

Examples

```
## Not run:
bdate <- "20200101"
edate <- "20200102"
state.fips <- "37"
county <- "001"
param <- "42401"
result <- get_monitors_in_county(bdate, edate, state.fips, county, param)
result$Data

## End(Not run)
```

---

get_monitors_in_site *Get all monitors at a site.*

Description

Get all monitors at a site.

Usage

```
get_monitors_in_site(
  bdate,
  edate,
  state.fips,
  county,
  param,
  site,
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

bdate	Beginning date to check. Year, month, day format.
edate	Ending date to check. Year, month, day format.
state.fips	State FIPS code. Use get_state_fips() if unsure.
county	County code. Use get_counties_in_state() if unsure.
param	Pollutant parameter that site is measuring.
site	Measurement site code. Use get_sites_by_county() if unsure.
cached	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)

cache_directory
: Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Value

API response containing operational information about the monitor.

Examples

```
## Not run:
bdate <- "20200101"
edate <- "20200102"
state.fips <- "37"
county <- "001"
param <- "42401"
site <- "001"
result <- get_monitors_in_site(bdate, edate, state.fips, county, param, site)
result$Data

## End(Not run)
```

---

get_monitors_in_state *Get monitors in state.*

Description

Get monitors in state.

Usage

```
get_monitors_in_state(
  bdate,
  edate,
  state.fips,
  param,
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

bdate	Beginning date to check. Year, month, day format.
edate	Ending date to check. Year, month, day format.
state.fips	State FIPS code. Use get_state_fips() if unsure.
param	Pollutant parameter that site is measuring.
cached	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)

cache_directory
: Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Value

API response containing operational information about the monitor.

Examples

```
## Not run:
bdate <- "20200101"
edate <- "20200102"
state.fips <- "37"
param <- "42401"
result <- get_monitors_in_state(bdate, edate, state.fips, param)
result$Data

## End(Not run)
```

---

get_parameters_in_class

Get all parameters available within a particular parameter class.

Description

Get all parameters available within a particular parameter class.

Usage

```
get_parameters_in_class(class)
```

Arguments

class	A type of pollutant. Find types of pollutants with get_parameter_classes().

Value

API response containing parameters found within a class/group of like parameters.

Examples

```
## Not run:
class <- "AQI POLLUTANTS"
parameters <- get_parameters_in_class(class)
parameters$Data

## End(Not run)
```

get_parameter_classes *Get all types of parameters.*

Description

Get all types of parameters.

Usage

```
get_parameter_classes()
```

Value

API response containing types of parameters and their respective code.

Examples

```
## Not run:
classes <- get_parameter_classes()
classes$Data

## End(Not run)
```

get_qa_ape_in_agency *Get quality assurance annual performance evaluations for a monitoring agency.*

Description

Get quality assurance annual performance evaluations for a monitoring agency.

Usage

```
get_qa_ape_in_agency(
  bdate,
  edate,
  param,
  agency,
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

bdate	Beginning date to check. Year, month, day format.
edate	Ending date to check. Year, month, day format.
param	Pollutant parameter that site is measuring.
agency	The monitoring agency.
cached	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
cache_directory	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Value

API response containing operational information about the quality assurance annual performance evaluations.

Examples

```
## Not run:
bdate <- "20170101"
edate <- "20171231"
param <- "44201"
agency <- "0013"
result <- get_qa_ape_in_agency(bdate,
                               edate,
                               param,
                               agency)
result$Data

## End(Not run)
```

---

get_qa_ape_in_county *Get quality assurance annual performance evaluations in a county.*

Description

Get quality assurance annual performance evaluations in a county.

Usage

```
get_qa_ape_in_county(
  bdate,
  edate,
  state.fips,
  county,
  param,
```

```
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

bdate	Beginning date to check. Year, month, day format.
edate	Ending date to check. Year, month, day format.
state.fips	State FIPS code. Use get_state_fips() if unsure.
county	County code. Use get_counties_in_state() if unsure.
param	Pollutant parameter that site is measuring.
cached	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
cache_directory	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Value

API response containing operational information about the quality assurance annual performance evaluations.

Examples

```
## Not run:
bdate <- "20170101"
edate <- "20171231"
state.fips <- "01"
county <- "003"
param <- "44201"
result <- get_qa_ape_in_county(bdate, edate, state.fips, county, param)
result$Data

## End(Not run)
```

---

get_qa_ape_in_pqao *Get quality assurance annual performance evaluations for a primary quality assurance organization.*

Description

Get quality assurance annual performance evaluations for a primary quality assurance organization.

Usage

```
get_qa_ape_in_pqao(
  bdate,
  edate,
  param,
  pqao,
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

bdate	Beginning date to check. Year, month, day format.
edate	Ending date to check. Year, month, day format.
param	Pollutant parameter that site is measuring.
pqao	An encoding for a primary quality assurance organization. If unsure, use get_all_pqaos().
cached	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
cache_directory	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Value

API response containing operational information about the quality assurance annual performance evaluations.

Examples

```
## Not run:
bdate <- "20170101"
edate <- "20171231"
pqao <- "0013"
param <- "44201"
result <- get_qa_ape_in_pqao(bdate, edate, param, pqao)
result$Data

## End(Not run)
```

---

get_qa_ape_in_site *Get quality assurance annual performance evaluations at a site.*

Description

Get quality assurance annual performance evaluations at a site.

Usage

```
get_qa_ape_in_site(
  bdate,
  edate,
  state.fips,
  county,
  param,
  site,
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

bdate	Beginning date to check. Year, month, day format.
edate	Ending date to check. Year, month, day format.
state.fips	State FIPS code. Use get_state_fips() if unsure.
county	County code. Use get_counties_in_state() if unsure.
param	Pollutant parameter that site is measuring.
site	Measurement site code. Use get_sites_by_county() if unsure.
cached	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
cache_directory	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Value

API response containing operational information about the quality assurance annual performance evaluations.

Examples

```
## Not run:
bdate <- "20200101"
edate <- "20201231"
state.fips <- "01"
county <- "003"
param <- "44201"
site <- "0010"
result <- get_qa_ape_in_site(bdate, edate, state.fips, county, param, site)
result$Data

## End(Not run)
```

`get_qa_ape_in_state` *Get quality assurance annual performance evaluations in a state.*

Description

Get quality assurance annual performance evaluations in a state.

Usage

```
get_qa_ape_in_state(
  bdate,
  edate,
  state.fips,
  param,
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

`bdate`	Beginning date to check. Year, month, day format.
`edate`	Ending date to check. Year, month, day format.
`state.fips`	State FIPS code. Use get_state_fips() if unsure.
`param`	Pollutant parameter that site is measuring.
`cached`	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
`cache_directory`	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Value

API response containing operational information about the quality assurance annual performance evaluations.

Examples

```
## Not run:
bdate <- "20170101"
edate <- "20171231"
state.fips <- "01"
param <- "44201"
result <- get_qa_ape_in_state(bdate, edate, state.fips, param)
result$Data

## End(Not run)
```

get_qa_blanks_in_agency *Get quality assurance blank data for a monitoring agency.*

Description

Get quality assurance blank data for a monitoring agency.

Usage

```
get_qa_blanks_in_agency(
  bdate,
  edate,
  param,
  agency,
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

bdate	Beginning date to check. Year, month, day format.
edate	Ending date to check. Year, month, day format.
param	Pollutant parameter that site is measuring.
agency	The monitoring agency.
cached	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
cache_directory	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Value

API response containing operational information about the quality assurance blank data.

Examples

```
## Not run:
bdate <- "20180101"
edate <- "20180131"
param <- "88101"
agency <- "0013"
result <- get_qa_blanks_in_agency(bdate,
                                  edate,
                                  param,
                                  agency)
result$Data
```

```
## End(Not run)
```

`get_qa_blanks_in_county` *Get quality assurance blank data in a county.*

Description

Get quality assurance blank data in a county.

Usage

```
get_qa_blanks_in_county(
  bdate,
  edate,
  state.fips,
  county,
  param,
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

`bdate`	Beginning date to check. Year, month, day format.
`edate`	Ending date to check. Year, month, day format.
`state.fips`	State FIPS code. Use get_state_fips() if unsure.
`county`	County code. Use get_counties_in_state() if unsure.
`param`	Pollutant parameter that site is measuring.
`cached`	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
`cache_directory`	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Value

API response containing operational information about the quality assurance blank data.

Examples

```
## Not run:
bdate <- "20180101"
edate <- "20180131"
state.fips <- "01"
county <- "033"
param <- "88101"
result <- get_qa_blanks_in_county(bdate, edate, state.fips, county, param)
result$Data

## End(Not run)
```

`get_qa_blanks_in_pqao` *Get quality assurance blank data for a primary quality assurance organization.*

Description

Get quality assurance blank data for a primary quality assurance organization.

Usage

```
get_qa_blanks_in_pqao(
  bdate,
  edate,
  param,
  pqao,
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

`bdate`	Beginning date to check. Year, month, day format.
`edate`	Ending date to check. Year, month, day format.
`param`	Pollutant parameter that site is measuring.
`pqao`	An encoding for a primary quality assurance organization. If unsure, use get_all_pqaos().
`cached`	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
`cache_directory`	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Value

API response containing operational information about the quality assurance blank data.

Examples

```
## Not run:
bdate <- "20180101"
edate <- "20180131"
param <- "88101"
pqao <- "0013"
result <- get_qa_blanks_in_pqao(bdate, edate, param, pqao)
result$Data

## End(Not run)
```

`get_qa_blanks_in_site` *Get quality assurance blank data at a site.*

Description

Get quality assurance blank data at a site.

Usage

```
get_qa_blanks_in_site(
  bdate,
  edate,
  state.fips,
  county,
  param,
  site,
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

`bdate`	Beginning date to check. Year, month, day format.
`edate`	Ending date to check. Year, month, day format.
`state.fips`	State FIPS code. Use get_state_fips() if unsure.
`county`	County code. Use get_counties_in_state() if unsure.
`param`	Pollutant parameter that site is measuring.
`site`	Measurement site code. Use get_sites_by_county() if unsure.
`cached`	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
`cache_directory`	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Value

API response containing operational information about the quality assurance blank data.

Examples

```
## Not run:
bdate <- "20180101"
edate <- "20180131"
state.fips <- "01"
county <- "033"
param <- "88101"
site <- "1002"
result <- get_qa_blanks_in_site(bdate,
                                edate,
                                state.fips,
                                county,
                                param,
                                site)
result$Data

## End(Not run)
```

`get_qa_blanks_in_state` *Get quality assurance blank data in a state.*

Description

Get quality assurance blank data in a state.

Usage

```
get_qa_blanks_in_state(
  bdate,
  edate,
  state.fips,
  param,
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

`bdate`	Beginning date to check. Year, month, day format.
`edate`	Ending date to check. Year, month, day format.
`state.fips`	State FIPS code. Use get_state_fips() if unsure.
`param`	Pollutant parameter that site is measuring.

cached	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
cache_directory	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Value

API response containing operational information about the quality assurance blank data.

Examples

```
## Not run:
bdate <- "20180101"
edate <- "20180131"
state.fips <- "01"
param <- "88101"
result <- get_qa_blanks_in_state(bdate, edate, state.fips, param)
result$Data

## End(Not run)
```

---

get_qa_ca_in_agency *Get quality assurance collocated assessment data for a monitoring agency.*

Description

Get quality assurance collocated assessment data for a monitoring agency.

Usage

```
get_qa_ca_in_agency(
  bdate,
  edate,
  param,
  agency,
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

bdate	Beginning date to check. Year, month, day format.
edate	Ending date to check. Year, month, day format.
param	Pollutant parameter that site is measuring.
agency	The monitoring agency.

cached TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)

cache_directory Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Value

API response containing operational information about the quality assurance annual performance evaluations.

Examples

```
## Not run:
bdate <- "20190101"
edate <- "20190131"
param <- "88101"
agency <- "0013"
result <- get_qa_ca_in_agency(bdate,
                              edate,
                              param,
                              agency)
result$Data

## End(Not run)
```

---

get_qa_ca_in_county *Get quality assurance collocated assessment data in a county.*

Description

Get quality assurance collocated assessment data in a county.

Usage

```
get_qa_ca_in_county(
  bdate,
  edate,
  state.fips,
  county,
  param,
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

bdate	Beginning date to check. Year, month, day format.
edate	Ending date to check. Year, month, day format.
state.fips	State FIPS code. Use get_state_fips() if unsure.
county	County code. Use get_counties_in_state() if unsure.
param	Pollutant parameter that site is measuring.
cached	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
cache_directory	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Value

API response containing operational information about the quality assurance annual performance evaluations.

Examples

```
## Not run:
bdate <- "20190101"
edate <- "20190131"
state.fips <- "01"
county <- "089"
param <- "88101"
result <- get_qa_ca_in_county(bdate, edate, state.fips, county, param)
result$Data

## End(Not run)
```

---

get_qa_ca_in_pqao *Get quality assurance collocated assessment data for a primary quality assurance organization.*

Description

Get quality assurance collocated assessment data for a primary quality assurance organization.

Usage

```
get_qa_ca_in_pqao(
  bdate,
  edate,
  param,
  pqao,
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

bdate	Beginning date to check. Year, month, day format.
edate	Ending date to check. Year, month, day format.
param	Pollutant parameter that site is measuring.
pqao	An encoding for a primary quality assurance organization. If unsure, use get_all_pqaos().
cached	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
cache_directory	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Value

API response containing operational information about the quality assurance annual performance evaluations.

Examples

```
## Not run:
bdate <- "20190101"
edate <- "20190131"
param <- "88101"
pqao <- "0013"
result <- get_qa_ca_in_pqao(bdate, edate, param, pqao)
result$Data

## End(Not run)
```

---

get_qa_ca_in_site *Get quality assurance collocated assessment data at a site.*

Description

Get quality assurance collocated assessment data at a site.

Usage

```
get_qa_ca_in_site(
  bdate,
  edate,
  state.fips,
  county,
  param,
  site,
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

bdate	Beginning date to check. Year, month, day format.
edate	Ending date to check. Year, month, day format.
state.fips	State FIPS code. Use get_state_fips() if unsure.
county	County code. Use get_counties_in_state() if unsure.
param	Pollutant parameter that site is measuring.
site	Measurement site code. Use get_sites_by_county() if unsure.
cached	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
cache_directory	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Value

API response containing operational information about the quality assurance annual performance evaluations.

Examples

```
## Not run:
bdate <- "20190101"
edate <- "20190131"
state.fips <- "01"
county <- "089"
param <- "88101"
site <- "0014"
result <- get_qa_ca_in_site(bdate, edate, state.fips, county, param, site)
result$Data

## End(Not run)
```

---

get_qa_ca_in_state *Get quality assurance collocated assessment data in a state.*

Description

Get quality assurance collocated assessment data in a state.

Usage

```
get_qa_ca_in_state(
  bdate,
  edate,
  state.fips,
  param,
```

```
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

bdate	Beginning date to check. Year, month, day format.
edate	Ending date to check. Year, month, day format.
state.fips	State FIPS code. Use get_state_fips() if unsure.
param	Pollutant parameter that site is measuring.
cached	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
cache_directory	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Value

API response containing operational information about the quality assurance annual performance evaluations.

Examples

```
## Not run:
bdate <- "20190101"
edate <- "20190131"
state.fips <- "01"
param <- "88101"
result <- get_qa_ca_in_state(bdate, edate, state.fips, param)
result$Data

## End(Not run)
```

---

get_qa_fra_in_agency *Get quality assurance flow rate audit data for a monitoring agency.*

Description

Get quality assurance flow rate audit data for a monitoring agency.

Usage

```
get_qa_fra_in_agency(
  bdate,
  edate,
  param,
  agency,
```

```
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

bdate	Beginning date to check. Year, month, day format.
edate	Ending date to check. Year, month, day format.
param	Pollutant parameter that site is measuring.
agency	The monitoring agency.
cached	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
cache_directory	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Value

API response containing operational information about the quality assurance flow rate audit data.

Examples

```
## Not run:
bdate <- "20200101"
edate <- "20200131"
param <- "88101"
agency <- "0013"
result <- get_qa_fra_in_agency(bdate,
                               edate,
                               param,
                               agency)
result$Data

## End(Not run)
```

---

get_qa_fra_in_county *Get quality assurance flow rate audit data in a county.*

Description

Get quality assurance flow rate audit data in a county.

Usage

```
get_qa_fra_in_county(
  bdate,
  edate,
  state.fips,
  county,
  param,
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

bdate	Beginning date to check. Year, month, day format.
edate	Ending date to check. Year, month, day format.
state.fips	State FIPS code. Use get_state_fips() if unsure.
county	County code. Use get_counties_in_state() if unsure.
param	Pollutant parameter that site is measuring.
cached	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
cache_directory	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Value

API response containing operational information about the quality assurance flow rate audit data.

Examples

```
## Not run:
bdate <- "20200101"
edate <- "20200131"
state.fips <- "01"
county <- "003"
param <- "88101"
result <- get_qa_fra_in_county(bdate, edate, state.fips, county, param)
result$Data

## End(Not run)
```

`get_qa_fra_in_pqao` *Get quality assurance flow rate audit data for a primary quality assurance organization.*

Description

Get quality assurance flow rate audit data for a primary quality assurance organization.

Usage

```
get_qa_fra_in_pqao(
  bdate,
  edate,
  param,
  pqao,
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

`bdate`	Beginning date to check. Year, month, day format.
`edate`	Ending date to check. Year, month, day format.
`param`	Pollutant parameter that site is measuring.
`pqao`	An encoding for a primary quality assurance organization. If unsure, use get_all_pqaos().
`cached`	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
`cache_directory`	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Value

API response containing operational information about the quality assurance flow rate audit data.

Examples

```
## Not run:
bdate <- "20200101"
edate <- "20200131"
param <- "88101"
pqao <- "0013"
result <- get_qa_fra_in_pqao(bdate, edate, param, pqao)
result$Data

## End(Not run)
```

`get_qa_fra_in_site` *Get quality assurance flow rate audit data at a site.*

Description

Get quality assurance flow rate audit data at a site.

Usage

```
get_qa_fra_in_site(
  bdate,
  edate,
  state.fips,
  county,
  param,
  site,
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

`bdate`	Beginning date to check. Year, month, day format.
`edate`	Ending date to check. Year, month, day format.
`state.fips`	State FIPS code. Use get_state_fips() if unsure.
`county`	County code. Use get_counties_in_state() if unsure.
`param`	Pollutant parameter that site is measuring.
`site`	Measurement site code. Use get_sites_by_county() if unsure.
`cached`	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
`cache_directory`	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Value

API response containing operational information about the quality assurance flow rate audit data.

Examples

```
## Not run:
bdate <- "20200101"
edate <- "20200131"
state.fips <- "01"
county <- "003"
param <- "88101"
```

```
site <- "0010"
result <- get_qa_fra_in_site(bdate, edate, state.fips, county, param, site)
result$Data

## End(Not run)
```

`get_qa_fra_in_state` *Get quality assurance flow rate audit data in a state.*

Description

Get quality assurance flow rate audit data in a state.

Usage

```
get_qa_fra_in_state(
  bdate,
  edate,
  state.fips,
  param,
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

`bdate`	Beginning date to check. Year, month, day format.
`edate`	Ending date to check. Year, month, day format.
`state.fips`	State FIPS code. Use get_state_fips() if unsure.
`param`	Pollutant parameter that site is measuring.
`cached`	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
`cache_directory`	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Value

API response containing operational information about the quality assurance flow rate audit data.

Examples

```
## Not run:
bdate <- "20200101"
edate <- "20200131"
state.fips <- "01"
param <- "88101"
```

```
result <- get_qa_fra_in_state(bdate, edate, state.fips, param)
result$Data

## End(Not run)
```

`get_qa_frv_in_agency` *Get quality assurance flow rate verification data for a monitoring agency.*

Description

Get quality assurance flow rate verification data for a monitoring agency.

Usage

```
get_qa_frv_in_agency(
  bdate,
  edate,
  param,
  agency,
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

`bdate`	Beginning date to check. Year, month, day format.
`edate`	Ending date to check. Year, month, day format.
`param`	Pollutant parameter that site is measuring.
`agency`	The monitoring agency.
`cached`	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
`cache_directory`	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Value

API response containing operational information about the quality assurance flow rate verification data.

Examples

```
## Not run:
bdate <- "20200101"
edate <- "20200131"
param <- "88101"
agency <- "0013"
result <- get_qa_frv_in_agency(bdate,
                               edate,
                               param,
                               agency)
result$Data

## End(Not run)
```

`get_qa_frv_in_county` *Get quality assurance flow rate verification data in a county.*

Description

Get quality assurance flow rate verification data in a county.

Usage

```
get_qa_frv_in_county(
  bdate,
  edate,
  state.fips,
  county,
  param,
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

`bdate`	Beginning date to check. Year, month, day format.
`edate`	Ending date to check. Year, month, day format.
`state.fips`	State FIPS code. Use get_state_fips() if unsure.
`county`	County code. Use get_counties_in_state() if unsure.
`param`	Pollutant parameter that site is measuring.
`cached`	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
`cache_directory`	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Value

API response containing operational information about the quality assurance flow rate verification data.

Examples

```
## Not run:
bdate <- "20200101"
edate <- "20200131"
state.fips <- "01"
county <- "003"
param <- "88101"
result <- get_qa_frv_in_county(bdate, edate, state.fips, county, param)
result$Data

## End(Not run)
```

---

get_qa_frv_in_pqao *Get quality assurance flow rate verification data for a primary quality assurance organization.*

Description

Get quality assurance flow rate verification data for a primary quality assurance organization.

Usage

```
get_qa_frv_in_pqao(
  bdate,
  edate,
  param,
  pqao,
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

bdate	Beginning date to check. Year, month, day format.
edate	Ending date to check. Year, month, day format.
param	Pollutant parameter that site is measuring.
pqao	An encoding for a primary quality assurance organization. If unsure, use get_all_pqaos().
cached	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
cache_directory	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Value

API response containing operational information about the quality assurance flow rate verification data.

Examples

```
## Not run:
bdate <- "20200101"
edate <- "20200131"
param <- "88101"
pqao <- "0013"
result <- get_qa_frv_in_pqao(bdate, edate, param, pqao)
result$Data

## End(Not run)
```

`get_qa_frv_in_site` *Get quality assurance flow rate verification data at a site.*

Description

Get quality assurance flow rate verification data at a site.

Usage

```
get_qa_frv_in_site(
  bdate,
  edate,
  state.fips,
  county,
  param,
  site,
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

`bdate`	Beginning date to check. Year, month, day format.
`edate`	Ending date to check. Year, month, day format.
`state.fips`	State FIPS code. Use get_state_fips() if unsure.
`county`	County code. Use get_counties_in_state() if unsure.
`param`	Pollutant parameter that site is measuring.
`site`	Measurement site code. Use get_sites_by_county() if unsure.
`cached`	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)

cache_directory
: Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Value

API response containing operational information about the quality assurance flow rate verification data.

Examples

```
## Not run:
bdate <- "20200101"
edate <- "20200131"
state.fips <- "01"
county <- "003"
param <- "88101"
site <- "0010"
result <- get_qa_frv_in_site(bdate, edate, state.fips, county, param, site)
result$Data

## End(Not run)
```

---

get_qa_frv_in_state *Get quality assurance flow rate verification data in a state.*

Description

Get quality assurance flow rate verification data in a state.

Usage

```
get_qa_frv_in_state(
  bdate,
  edate,
  state.fips,
  param,
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

bdate	Beginning date to check. Year, month, day format.
edate	Ending date to check. Year, month, day format.
state.fips	State FIPS code. Use get_state_fips() if unsure.
param	Pollutant parameter that site is measuring.

cached	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
cache_directory	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Value

API response containing operational information about the quality assurance flow rate verification data.

Examples

```
## Not run:
bdate <- "20200101"
edate <- "20200131"
state.fips <- "01"
param <- "88101"
result <- get_qa_frv_in_state(bdate, edate, state.fips, param)
result$Data

## End(Not run)
```

---

get_qa_pep_in_agency *Get quality assurance PEP audit data for a monitoring agency.*

Description

Get quality assurance PEP audit data for a monitoring agency.

Usage

```
get_qa_pep_in_agency(
  bdate,
  edate,
  param,
  agency,
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

bdate	Beginning date to check. Year, month, day format.
edate	Ending date to check. Year, month, day format.
param	Pollutant parameter that site is measuring.
agency	The monitoring agency.

cached TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)

cache_directory
Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Value

API response containing operational information about the quality assurance PEP audit data.

Examples

```
## Not run:
bdate <- "20180101"
edate <- "20181231"
param <- "88101"
agency <- "0013"
result <- get_qa_pep_in_agency(bdate,
                               edate,
                               param,
                               agency)
result$Data

## End(Not run)
```

---

get_qa_pep_in_county *Get quality assurance PEP audit data in a county.*

Description

Get quality assurance PEP audit data in a county.

Usage

```
get_qa_pep_in_county(
  bdate,
  edate,
  state.fips,
  county,
  param,
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

bdate	Beginning date to check. Year, month, day format.
edate	Ending date to check. Year, month, day format.
state.fips	State FIPS code. Use get_state_fips() if unsure.
county	County code. Use get_counties_in_state() if unsure.
param	Pollutant parameter that site is measuring.
cached	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
cache_directory	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Value

API response containing operational information about the quality assurance PEP audit data.

Examples

```
## Not run:
bdate <- "20180101"
edate <- "20181231"
state.fips <- "01"
county <- "089"
param <- "88101"
result <- get_qa_pep_in_county(bdate, edate, state.fips, county, param)
result$Data

## End(Not run)
```

---

get_qa_pep_in_pqao *Get quality assurance PEP audit data for a primary quality assurance organization.*

Description

Get quality assurance PEP audit data for a primary quality assurance organization.

Usage

```
get_qa_pep_in_pqao(
  bdate,
  edate,
  param,
  pqao,
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

bdate	Beginning date to check. Year, month, day format.
edate	Ending date to check. Year, month, day format.
param	Pollutant parameter that site is measuring.
pqao	An encoding for a primary quality assurance organization. If unsure, use get_all_pqaos().
cached	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
cache_directory	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Value

API response containing operational information about the quality assurance PEP audit data.

Examples

```
## Not run:
bdate <- "20180101"
edate <- "20181231"
param <- "88101"
pqao <- "0013"
result <- get_qa_pep_in_pqao(bdate, edate, param, pqao)
result$Data

## End(Not run)
```

---

get_qa_pep_in_site *Get quality assurance PEP audit data at a site.*

Description

Get quality assurance PEP audit data at a site.

Usage

```
get_qa_pep_in_site(
  bdate,
  edate,
  state.fips,
  county,
  param,
  site,
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

bdate	Beginning date to check. Year, month, day format.
edate	Ending date to check. Year, month, day format.
state.fips	State FIPS code. Use get_state_fips() if unsure.
county	County code. Use get_counties_in_state() if unsure.
param	Pollutant parameter that site is measuring.
site	Measurement site code. Use get_sites_by_county() if unsure.
cached	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
cache_directory	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Value

API response containing operational information about the quality assurance PEP audit data.

Examples

```
## Not run:
bdate <- "20180101"
edate <- "20181231"
state.fips <- "01"
county <- "089"
param <- "88101"
site <- "0014"
result <- get_qa_pep_in_site(bdate, edate, state.fips, county, param, site)
result$Data

## End(Not run)
```

---

get_qa_pep_in_state *Get quality assurance PEP audit data in a state.*

Description

Get quality assurance PEP audit data in a state.

Usage

```
get_qa_pep_in_state(
  bdate,
  edate,
  state.fips,
  param,
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

bdate	Beginning date to check. Year, month, day format.
edate	Ending date to check. Year, month, day format.
state.fips	State FIPS code. Use get_state_fips() if unsure.
param	Pollutant parameter that site is measuring.
cached	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
cache_directory	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Value

API response containing operational information about the quality assurance PEP audit data.

Examples

```
## Not run:
bdate <- "20180101"
edate <- "20181231"
state.fips <- "01"
param <- "88101"
result <- get_qa_pep_in_state(bdate, edate, state.fips, param)
result$Data

## End(Not run)
```

---

get_qa_qc_in_agency *Get quality assurance one point quality control data for a monitoring agency.*

Description

Get quality assurance one point quality control data for a monitoring agency.

Usage

```
get_qa_qc_in_agency(
  bdate,
  edate,
  param,
  agency,
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

bdate	Beginning date to check. Year, month, day format.
edate	Ending date to check. Year, month, day format.
param	Pollutant parameter that site is measuring.
agency	The monitoring agency.
cached	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
cache_directory	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Value

API response containing operational information about the quality assurance one point quality control data.

Examples

```
## Not run:
bdate <- "20170101"
edate <- "20171231"
param <- "44201"
agency <- "0013"
result <- get_qa_qc_in_agency(bdate,
                              edate,
                              param,
                              agency)
result$Data

## End(Not run)
```

---

get_qa_qc_in_county *Get quality assurance annual one point quality control data in a county.*

Description

Get quality assurance annual one point quality control data in a county.

Usage

```
get_qa_qc_in_county(
  bdate,
  edate,
  state.fips,
  county,
```

```
  param,
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

bdate	Beginning date to check. Year, month, day format.
edate	Ending date to check. Year, month, day format.
state.fips	State FIPS code. Use get_state_fips() if unsure.
county	County code. Use get_counties_in_state() if unsure.
param	Pollutant parameter that site is measuring.
cached	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
cache_directory	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Value

API response containing operational information about the quality assurance one point quality control data.

Examples

```
## Not run:
bdate <- "20170101"
edate <- "20171231"
state.fips <- "01"
county <- "003"
param <- "44201"
result <- get_qa_qc_in_county(bdate, edate, state.fips, county, param)
result$Data

## End(Not run)
```

---

get_qa_qc_in_pqao *Get quality assurance one point quality control data for a primary quality assurance organization.*

Description

Get quality assurance one point quality control data for a primary quality assurance organization.

Usage

```
get_qa_qc_in_pqao(
  bdate,
  edate,
  param,
  pqao,
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

bdate	Beginning date to check. Year, month, day format.
edate	Ending date to check. Year, month, day format.
param	Pollutant parameter that site is measuring.
pqao	An encoding for a primary quality assurance organization. If unsure, use get_all_pqaos().
cached	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
cache_directory	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Value

API response containing operational information about the quality assurance one point quality control data.

Examples

```
## Not run:
bdate <- "20170101"
edate <- "20171231"
pqao <- "0013"
param <- "44201"
result <- get_qa_qc_in_pqao(bdate, edate, param, pqao)
result$Data

## End(Not run)
```

---

get_qa_qc_in_site *Get quality assurance one point quality control data at a site.*

Description

Get quality assurance one point quality control data at a site.

Usage

```
get_qa_qc_in_site(
  bdate,
  edate,
  state.fips,
  county,
  param,
  site,
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

bdate	Beginning date to check. Year, month, day format.
edate	Ending date to check. Year, month, day format.
state.fips	State FIPS code. Use get_state_fips() if unsure.
county	County code. Use get_counties_in_state() if unsure.
param	Pollutant parameter that site is measuring.
site	Measurement site code. Use get_sites_by_county() if unsure.
cached	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
cache_directory	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Value

API response containing operational information about the quality assurance one point quality control data.

Examples

```
## Not run:
bdate <- "20200101"
edate <- "20201231"
state.fips <- "01"
county <- "003"
param <- "44201"
site <- "0010"
result <- get_qa_qc_in_site(bdate, edate, state.fips, county, param, site)
result$Data

## End(Not run)
```

`get_qa_qc_in_state` *Get quality assurance one point quality control data in a state.*

Description

Get quality assurance one point quality control data in a state.

Usage

```
get_qa_qc_in_state(
  bdate,
  edate,
  state.fips,
  param,
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

`bdate`	Beginning date to check. Year, month, day format.
`edate`	Ending date to check. Year, month, day format.
`state.fips`	State FIPS code. Use get_state_fips() if unsure.
`param`	Pollutant parameter that site is measuring.
`cached`	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
`cache_directory`	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Value

API response containing operational information about the quality assurance one point quality control data.

Examples

```
## Not run:
bdate <- "20170101"
edate <- "20171231"
state.fips <- "01"
param <- "44201"
result <- get_qa_qc_in_state(bdate, edate, state.fips, param)
result$Data

## End(Not run)
```

`get_quarterly_summary_in_bbox` *Get quarterly summary data in a bounding box (lat, long).*

Description

Get quarterly summary data in a bounding box (lat, long).

Usage

```
get_quarterly_summary_in_bbox(
  bdate,
  edate,
  param,
  minlat,
  maxlat,
  minlong,
  maxlong,
  cbdate = NULL,
  cedate = NULL
)
```

Arguments

`bdate`	Beginning date to check. Year, month, day format.
`edate`	Ending date to check. Year, month, day format.
`param`	Pollutant parameter that site is measuring.
`minlat`	Minimum latitude coordinate.
`maxlat`	Maximum latitude coordinate.
`minlong`	Minimum longitude coordinate.
`maxlong`	Maximum longitude coordinate.
`cbdate`	Beginning date of last change to DB. (Optional)
`cedate`	Ending date of last change to DB. (Optional)

Value

API response containing quarterly summary data in a bounding box.

Examples

```
## Not run:
bdate <- "20200101"
edate <- "20200102"
param <- "42401"
minlat <- 33.3
```

```
maxlat <- 33.6
minlong <- -87
maxlong <- -86.7
result <- get_quarterly_summary_in_bbox(bdate,
                                        edate,
                                        param,
                                        minlat,
                                        maxlat,
                                        minlong,
                                        maxlong)
result$Data

## End(Not run)
```

`get_quarterly_summary_in_cbsa` *Get quarterly summary data in a Core Based Statistical Area.*

Description

Get quarterly summary data in a Core Based Statistical Area.

Usage

```
get_quarterly_summary_in_cbsa(
  bdate,
  edate,
  param,
  cbsa,
  cbdate = NULL,
  cedate = NULL
)
```

Arguments

`bdate`	Beginning date to check. Year, month, day format.
`edate`	Ending date to check. Year, month, day format.
`param`	Pollutant parameter that site is measuring.
`cbsa`	An encoding for a Core Base Statiscal Area. If unsure, use get_cbsas().
`cbdate`	Beginning date of last change to DB. (Optional)
`cedate`	Ending date of last change to DB. (Optional)

Value

API response containing quarterly summary data at the cbsa level.

Examples

```
## Not run:
bdate <- "20190101"
edate <- "20190601"
cbsa <- "16740"
param <- "42401"
result <- get_quarterly_summary_in_cbsa(bdate,
                                        edate,
                                        param,
                                        cbsa)
result$Data

## End(Not run)
```

`get_quarterly_summary_in_county` *Get quarterly summary data in a county.*

Description

Get quarterly summary data in a county.

Usage

```
get_quarterly_summary_in_county(
  bdate,
  edate,
  state.fips,
  county,
  param,
  cbdate = NULL,
  cedate = NULL
)
```

Arguments

`bdate`	Beginning date to check. Year, month, day format.
`edate`	Ending date to check. Year, month, day format.
`state.fips`	State FIPS code. Use get_state_fips() if unsure.
`county`	County code. Use get_counties_in_state() if unsure.
`param`	Pollutant parameter that site is measuring.
`cbdate`	Beginning date of last change to DB. (Optional)
`cedate`	Ending date of last change to DB. (Optional)

Value

API response containing quarterly summary data in a county.

Examples

```
## Not run:
bdate <- "20200101"
edate <- "20200102"
state.fips <- "37"
county <- "001"
param <- "42401"
result <- get_quarterly_summary_in_county(bdate,
                                          edate,
                                          state.fips,
                                          county,
                                          param)
result$Data

## End(Not run)
```

`get_quarterly_summary_in_site` *Returns data summaries by yearly quarter.*

Description

Returns data summaries by yearly quarter.

Usage

```
get_quarterly_summary_in_site(
  bdate,
  edate,
  state.fips,
  county,
  param,
  site,
  cbdate = NULL,
  cedate = NULL
)
```

Arguments

`bdate`	Beginning date to check. Year, month, day format.
`edate`	Ending date to check. Year, month, day format.
`state.fips`	State FIPS code. Use get_state_fips() if unsure.
`county`	County code. Use get_counties_in_state() if unsure.
`param`	Pollutant parameter that site is measuring.
`site`	Measurement site code. Use get_sites_by_county() if unsure.
`cbdate`	Beginning date of last change to DB. (Optional)
`cedate`	Ending date of last change to DB. (Optional)

Value

API response containing quarterly summary data at a site.

Examples

```
## Not run:
bdate <- "20200101"
edate <- "20200102"
state.fips <- "37"
county <- "001"
site <- "001"
param <- "42401"
result <- get_quarterly_summary_in_site(bdate,
                                        edate,
                                        state.fips,
                                        county,
                                        param,
                                        site)
result$Data

## End(Not run)
```

`get_quarterly_summary_in_state` *Get quarterly summary data in a state.*

Description

Get quarterly summary data in a state.

Usage

```
get_quarterly_summary_in_state(
  bdate,
  edate,
  state.fips,
  param,
  cbdate = NULL,
  cedate = NULL
)
```

Arguments

`bdate`	Beginning date to check. Year, month, day format.
`edate`	Ending date to check. Year, month, day format.
`state.fips`	State FIPS code. Use get_state_fips() if unsure.
`param`	Pollutant parameter that site is measuring.
`cbdate`	Beginning date of last change to DB. (Optional)
`cedate`	Ending date of last change to DB. (Optional)

Value

API response containing quarterly summary data for a state.

Examples

```
## Not run:
bdate <- "20200101"
edate <- "20200102"
state.fips <- "37"
param <- "42401"
result <- get_quarterly_summary_in_state(bdate,
                                         edate,
                                         state.fips,
                                         param)
result$Data

## End(Not run)
```

---

get_revision_history *Get the API's revision history.*

Description

Get the API's revision history.

Usage

```
get_revision_history()
```

Examples

```
## Not run:
result <- get_revision_history()
result$Data

## End(Not run)
```

---

get_samples_in_bbox *Get samples (finest grained data) for a bounding box (lat, long).*

Description

Get samples (finest grained data) for a bounding box (lat, long).

Usage

```
get_samples_in_bbox(
  bdate,
  edate,
  minlat,
  maxlat,
  minlong,
  maxlong,
  param,
  cached = TRUE,
  cache_directory = "/cache",
  duration = NULL,
  cbdate = NULL,
  cedate = NULL
)
```

Arguments

bdate	Beginning date to check. Year, month, day format.
edate	Ending date to check. Year, month, day format.
minlat	Minimum latitude coordinate.
maxlat	Maximum latitude coordinate.
minlong	Minimum longitude coordinate.
maxlong	Maximum longitude coordinate.
param	Pollutant parameter that site is measuring.
cached	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
cache_directory	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)
duration	The 1-character AQS sample duration code. (Optional)
cbdate	Beginning date of last change to DB. (Optional)
cedate	Ending date of last change to DB. (Optional)

Value

API response containing operational information about the monitor.

Examples

```
## Not run:
bdate <- "20200101"
edate <- "20200102"
param <- "42401"
minlat <- 33.3
maxlat <- 33.6
```

```
minlong <- -87
maxlong <- -86.7
result <- get_samples_in_bbox(bdate = bdate,
                              edate = edate,
                              param = param,
                              minlat = minlat,
                              maxlat = maxlat,
                              minlong = minlong,
                              maxlong = maxlong)
result$Data

## End(Not run)
```

`get_samples_in_cbsa` *Get samples (finest grained data) for a Core Based Statistical Area.*

Description

Get samples (finest grained data) for a Core Based Statistical Area.

Usage

```
get_samples_in_cbsa(
  bdate,
  edate,
  param,
  cbsa,
  cached = TRUE,
  cache_directory = "/cache",
  duration = NULL,
  cbdate = NULL,
  cedate = NULL
)
```

Arguments

`bdate`	Beginning date to check. Year, month, day format.
`edate`	Ending date to check. Year, month, day format.
`param`	Pollutant parameter that site is measuring.
`cbsa`	An encoding for a Core Base Statiscal Area. If unsure, use get_cbsas().
`cached`	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
`cache_directory`	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)
`duration`	The 1-character AQS sample duration code. (Optional)
`cbdate`	Change begin date. (Optional)
`cedate`	Change end date. (Optional)

Value

API response containing sample measurements in a CBSA.

Examples

```
## Not run:
bdate <- "20200101"
edate <- "20200102"
cbsa <- "16740"
param <- "42401"
result <- get_samples_in_cbsa(bdate, edate, cbsa, param)
result$Data

## End(Not run)
```

`get_samples_in_county` *Get samples (finest grained data) for a county.*

Description

Get samples (finest grained data) for a county.

Usage

```
get_samples_in_county(
  bdate,
  edate,
  state.fips,
  param,
  county,
  cached = TRUE,
  cache_directory = "/cache",
  duration = NULL,
  cbdate = NULL,
  cedate = NULL
)
```

Arguments

`bdate`	Beginning date to check. Year, month, day format.
`edate`	Ending date to check. Year, month, day format.
`state.fips`	State FIPS code. Use get_state_fips() if unsure.
`param`	Pollutant parameter that site is measuring.
`county`	County code.
`cached`	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)

cache_directory	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)
duration	The 1-character AQS sample duration code. (Optional)
cbdate	Change begin date. (Optional)
cedate	Change end date. (Optional)

Value

API response containing sample measurements in a state.

Examples

```
## Not run:
bdate <- "20160101"
edate <- "20160102"
state.fips <- "15"
county <- "001"
param <- "42401"
result <- get_samples_in_county(bdate = bdate,
                                edate = edate,
                                param = param,
                                state.fips = state.fips,
                                county = county)
result$Data

## End(Not run)
```

---

get_samples_in_site *Get samples (finest grained data) for a measurement site.*

Description

Get samples (finest grained data) for a measurement site.

Usage

```
get_samples_in_site(
  bdate,
  edate,
  state.fips,
  county,
  site,
  param,
  cached = TRUE,
  cache_directory = "/cache",
  duration = NULL,
  cbdate = NULL,
  cedate = NULL
)
```

Arguments

bdate	Beginning date to check. Year, month, day format.
edate	Ending date to check. Year, month, day format.
state.fips	State FIPS code. Use get_state_fips() if unsure.
county	County code. Use get_counties_in_state() if unsure.
site	Measurement site code. Use get_sites_by_county() if unsure.
param	Pollutant parameter that site is measuring.
cached	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
cache_directory	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)
duration	The 1-character AQS sample duration code. (Optional)
cbdate	Change begin date. (Optional)
cedate	Change end date. (Optional)

Value

API response containing samples at given site.

Examples

```
## Not run:
bdate <- "20160101"
edate <- "20160102"
state.fips <- "15"
county <- "001"
param <- "42401"
site <- "0007"
cbdate <- "20200101"
cedate <- "20201231"
result <- get_samples_in_site(bdate = bdate,
                              edate = edate,
                              param = param,
                              state.fips = state.fips,
                              county = county,
                              site = site)
result$Data

## End(Not run)
```

`get_samples_in_state` *Get samples (finest grained data) for a state.*

Description

Get samples (finest grained data) for a state.

Usage

```
get_samples_in_state(
  bdate = bdate,
  edate = edate,
  state.fips = state.fips,
  param = param,
  cached = TRUE,
  cache_directory = "/cache",
  duration = NULL,
  cbdate = NULL,
  cedate = NULL
)
```

Arguments

`bdate`	Beginning date to check. Year, month, day format.
`edate`	Ending date to check. Year, month, day format.
`state.fips`	State FIPS code. Use get_state_fips() if unsure.
`param`	Pollutant parameter that site is measuring.
`cached`	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
`cache_directory`	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)
`duration`	The 1-character AQS sample duration code. (Optional)
`cbdate`	Change begin date. (Optional)
`cedate`	Change end date. (Optional)

Value

API response containing sample measurements in a state.

Examples

```
## Not run:
bdate <- "20160101"
edate <- "20160102"
state.fips <- "15"
param <- "42401"
result <- get_samples_in_state(bdate = bdate,
                               edate = edate,
                               param = param,
                               state.fips = state.fips)
result$Data

## End(Not run)
```

`get_sites_in_county` *Get all measurement sites within a county.*

Description

Get all measurement sites within a county.

Usage

```
get_sites_in_county(state.fips, county.code)
```

Arguments

`state.fips`	A state FIPS code. Use get_state_fips() to find the appropriate FIPS code.
`county.code`	A county code. Use get_counties_in_state() to find the appropriate code for a given county.

Value

API response containing all measurement sites within a given county.

Examples

```
## Not run:
state <- "37"
county.code <- "001"
measurement.sites <- get_sites_in_county(state, county.code)
measurement.sites$Data

## End(Not run)
```

`get_state_fips` *Get all FIPS codes for each US state.*

Description

Get all FIPS codes for each US state.

Usage

```
get_state_fips()
```

Value

API response containing states and their respective FIPS codes.

Examples

```
## Not run:
state.fips <- get_state_fips()
state.fips$Data

## End(Not run)
```

`get_tf_qa_ape_in_agency` *Get quality assurance annual performance evaluations for a monitoring agency in transaction format.*

Description

Get quality assurance annual performance evaluations for a monitoring agency in transaction format.

Usage

```
get_tf_qa_ape_in_agency(
  bdate,
  edate,
  param,
  agency,
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

bdate	Beginning date to check. Year, month, day format.
edate	Ending date to check. Year, month, day format.
param	Pollutant parameter that site is measuring.
agency	The monitoring agency.
cached	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
cache_directory	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Value

API response containing operational information about the quality assurance annual performance evaluations in transaction format.

Examples

```
## Not run:
bdate <- "20170101"
edate <- "20171231"
param <- "44201"
agency <- "0013"
result <- get_tf_qa_ape_in_agency(bdate,
                                  edate,
                                  param,
                                  agency)
result$Data

## End(Not run)
```

---

get_tf_qa_ape_in_county

Get quality assurance annual performance evaluations in a county in transaction format.

Description

Get quality assurance annual performance evaluations in a county in transaction format.

Usage

```
get_tf_qa_ape_in_county(
  bdate,
  edate,
  state.fips,
```

```
  county,
  param,
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

bdate	Beginning date to check. Year, month, day format.
edate	Ending date to check. Year, month, day format.
state.fips	State FIPS code. Use get_state_fips() if unsure.
county	County code. Use get_counties_in_state() if unsure.
param	Pollutant parameter that site is measuring.
cached	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
cache_directory	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Value

API response containing operational information about the quality assurance annual performance evaluations in transaction format.

Examples

```
## Not run:
bdate <- "20170101"
edate <- "20171231"
state.fips <- "01"
county <- "003"
param <- "44201"
result <- get_tf_qa_ape_in_county(bdate,
                                  edate,
                                  state.fips,
                                  county,
                                  param)
result$Data

## End(Not run)
```

`get_tf_qa_ape_in_pqao` *Get quality assurance annual performance evaluations for a primary quality assurance organization in transaction format.*

Description

Get quality assurance annual performance evaluations for a primary quality assurance organization in transaction format.

Usage

```
get_tf_qa_ape_in_pqao(
  bdate,
  edate,
  param,
  pqao,
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

`bdate`	Beginning date to check. Year, month, day format.
`edate`	Ending date to check. Year, month, day format.
`param`	Pollutant parameter that site is measuring.
`pqao`	An encoding for a primary quality assurance organization. If unsure, use get_all_pqaos().
`cached`	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
`cache_directory`	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Value

API response containing operational information about the quality assurance annual performance evaluations in transaction format.

Examples

```
## Not run:
bdate <- "20170101"
edate <- "20171231"
pqao <- "0013"
param <- "44201"
result <- get_tf_qa_ape_in_pqao(bdate, edate, param, pqao)
result$Data

## End(Not run)
```

`get_tf_qa_ape_in_site` *Get quality assurance annual performance evaluations at a site in transaction format.*

Description

Get quality assurance annual performance evaluations at a site in transaction format.

Usage

```
get_tf_qa_ape_in_site(
  bdate,
  edate,
  state.fips,
  county,
  param,
  site,
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

`bdate`	Beginning date to check. Year, month, day format.
`edate`	Ending date to check. Year, month, day format.
`state.fips`	State FIPS code. Use get_state_fips() if unsure.
`county`	County code. Use get_counties_in_state() if unsure.
`param`	Pollutant parameter that site is measuring.
`site`	Measurement site code. Use get_sites_by_county() if unsure.
`cached`	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
`cache_directory`	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Value

API response containing operational information about the quality assurance annual performance evaluations in transaction format.

Examples

```
## Not run:
bdate <- "20200101"
edate <- "20201231"
state.fips <- "01"
```

```
county <- "003"
param <- "44201"
site <- "0010"
result <- get_tf_qa_ape_in_site(bdate,
                                edate,
                                state.fips,
                                county,
                                param,
                                site)
result$Data

## End(Not run)
```

`get_tf_qa_ape_in_state` *Get quality assurance annual performance evaluations in a state in transaction format.*

Description

Get quality assurance annual performance evaluations in a state in transaction format.

Usage

```
get_tf_qa_ape_in_state(
  bdate,
  edate,
  state.fips,
  param,
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

`bdate`	Beginning date to check. Year, month, day format.
`edate`	Ending date to check. Year, month, day format.
`state.fips`	State FIPS code. Use get_state_fips() if unsure.
`param`	Pollutant parameter that site is measuring.
`cached`	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
`cache_directory`	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Value

API response containing operational information about the quality assurance annual performance evaluations in transaction format.

Examples

```
## Not run:
bdate <- "20170101"
edate <- "20171231"
state.fips <- "01"
param <- "44201"
result <- get_tf_qa_ape_in_state(bdate, edate, state.fips, param)
result$Data

## End(Not run)
```

---

get_tf_sample_in_agency

Get sample data in the transaction format for a monitoring agency.

Description

Get sample data in the transaction format for a monitoring agency.

Usage

```
get_tf_sample_in_agency(
  bdate,
  edate,
  param,
  agency,
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

bdate	Beginning date to check. Year, month, day format.
edate	Ending date to check. Year, month, day format.
param	Pollutant parameter that site is measuring.
agency	The monitoring agency.
cached	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
cache_directory	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Value

API response containing all sample data in submission format.

Examples

```
## Not run:
bdate <- "20200101"
edate <- "20200331"
state.fips <- "01"
param <- "44201"
agency <- "0013"
result <- get_tf_sample_in_agency(bdate,
                                  edate,
                                  param,
                                  agency)
result$Data

## End(Not run)
```

`get_tf_sample_in_county` *Get sample data in the transaction format in a county.*

Description

Get sample data in the transaction format in a county.

Usage

```
get_tf_sample_in_county(
  bdate,
  edate,
  state.fips,
  county,
  param,
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

`bdate`	Beginning date to check. Year, month, day format.
`edate`	Ending date to check. Year, month, day format.
`state.fips`	State FIPS code. Use get_state_fips() if unsure.
`county`	County code. Use get_counties_in_state() if unsure.
`param`	Pollutant parameter that site is measuring.

cached	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
cache_directory	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Value

API response containing all sample data in submission format.

Examples

```
## Not run:
bdate <- "20200101"
edate <- "20200331"
state.fips <- "01"
county <- "003"
param <- "44201"
result <- get_tf_sample_in_county(bdate, edate, state.fips, county, param)
result$Data

## End(Not run)
```

---

get_tf_sample_in_site *Get sample data in the transaction format at a site.*

Description

Get sample data in the transaction format at a site.

Usage

```
get_tf_sample_in_site(
  bdate,
  edate,
  state.fips,
  county,
  param,
  site,
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

bdate	Beginning date to check. Year, month, day format.
edate	Ending date to check. Year, month, day format.
state.fips	State FIPS code. Use get_state_fips() if unsure.

county	County code. Use get_counties_in_state() if unsure.
param	Pollutant parameter that site is measuring.
site	Measurement site code. Use get_sites_by_county() if unsure.
cached	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
cache_directory	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Value

API response containing all sample data in submission format.

Examples

```
## Not run:
bdate <- "20200101"
edate <- "20200331"
state.fips <- "01"
county <- "003"
param <- "44201"
site <- "0010"
result <- get_tf_sample_in_site(bdate,
                                edate,
                                state.fips,
                                county,
                                param,
                                site)
result$Data

## End(Not run)
```

`get_tf_sample_in_state` *Get sample data in the transaction format in a state.*

Description

Get sample data in the transaction format in a state.

Usage

```
get_tf_sample_in_state(
  bdate,
  edate,
  state.fips,
  param,
```

```
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

bdate	Beginning date to check. Year, month, day format.
edate	Ending date to check. Year, month, day format.
state.fips	State FIPS code. Use get_state_fips() if unsure.
param	Pollutant parameter that site is measuring.
cached	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
cache_directory	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Value

API response containing all sample data in submission format.

Examples

```
## Not run:
bdate <- "20200101"
edate <- "20200331"
state.fips <- "01"
param <- "44201"
result <- get_tf_sample_in_state(bdate, edate, state.fips, param)
result$Data

## End(Not run)
```

---

is_API_running *Check if the API is up and running*

Description

Check if the API is up and running

Usage

```
is_API_running()
```

Examples

```
## Not run:
is_API_running()

## End(Not run)
```

list.cached.data *Shows contents of cache directory*

Description

Shows contents of cache directory

Usage

```
list.cached.data(directory = "/cache")
```

Arguments

directory	Place inside user-level cache directory that was used to store the cached data previously. Default: "/cache".

Value

Character vector of file names currently in cache directory.

Examples

```
## Not run:
my.files <- list.cached.data()
my.files

## End(Not run)
```

list.remove.escapes.spaces *Remove tabs, new lines, and empty spaces from entries in a list*

Description

Remove tabs, new lines, and empty spaces from entries in a list

Usage

```
list.remove.escapes.spaces(a.list)
```

Arguments

a.list	List to remove entries from.

Value

A list without tabs, new lines, and empty spaces

Examples

```
service <- c("Sign up")
description <- c("Email will\r\n\t\t\t\t\t\t\t be sent to the registered
address from aqsdatamart@epa.gov.")

og_list <- list("service" = service, "description" = description)
clean <- epair:::remove.escapes.spaces(og_list)
clean
```

`list.string.replacer` *Replace every string entry in a list*

Description

Replace every string entry in a list

Usage

```
list.string.replacer(entry.list, pattern, replacement)
```

Arguments

`entry.list`	List containing character entries
`pattern`	Pattern to replace
`replacement`	Replacement for entries following the pattern

Value

A list with entries matching the pattern replaced by replacement

Examples

```
services <- list("\t Some text from the table")
services <- epair:::list.string.replacer(services, "\t", "")
services
```

lookup_by_bbox *Internal function to perform geospatial lookup by bounding box (lat, long).*

Description

Internal function to perform geospatial lookup by bounding box (lat, long).

Usage

```
lookup_by_bbox(
  endpoint,
  bdate,
  edate,
  param,
  minlat,
  maxlat,
  minlong,
  maxlong,
  cached = TRUE,
  cache_directory = "/cache",
  duration = NULL,
  cbdate = NULL,
  cedate = NULL
)
```

Arguments

endpoint	Base url to make call.
bdate	Beginning date to check. Year, month, day format.
edate	Ending date to check. Year, month, day format.
param	Pollutant parameter that site is measuring.
minlat	Minimum latitude coordinate.
maxlat	Maximum latitude coordinate.
minlong	Minimum longitude coordinate.
maxlong	Maximum longitude coordinate.
cached	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
cache_directory	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)
duration	The 1-character AQS sample duration code. (Optional)
cbdate	Beginning date of last change to DB. (Optional)
cedate	Ending date of last change to DB. (Optional)

Value

API response containing bbox bounded data.

Examples

```
## Not run:
bdate <- "20200101"
edate <- "20200102"
param <- "42401"
minlat <- 33.3
maxlat <- 33.6
minlong <- -87
maxlong <- -86.7
result <- lookup_by_bbox(MONITORS, bdate,
                                   edate,
                                   param,
                                   minlat,
                                   maxlat,
                                   minlong,
                                   maxlong)
result$Data

## End(Not run)
```

---

lookup_by_cbsa *Internal function to perform geospatial lookup by Core Based Statistical Area.*

Description

Internal function to perform geospatial lookup by Core Based Statistical Area.

Usage

```
lookup_by_cbsa(
  endpoint,
  bdate,
  edate,
  param,
  cbsa,
  cached = TRUE,
  cache_directory = "/cache",
  duration = NULL,
  cbdate = NULL,
  cedate = NULL
)
```

Arguments

endpoint	Base url to make call.
bdate	Beginning date to check. Year, month, day format.
edate	Ending date to check. Year, month, day format.
param	Pollutant parameter that site is measuring.
cbsa	An encoding for a Core Base Statiscal Area. If unsure, use get_cbsas().
cached	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
cache_directory	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)
duration	The 1-character AQS sample duration code. (Optional)
cbdate	Beginning date of last change to DB. (Optional)
cedate	Ending date of last change to DB. (Optional)

Value

API response containing data at the cbsa level.

Examples

```
## Not run:
bdate <- "20200101"
edate <- "20200102"
cbsa <- "16740"
param <- "42401"
result <- lookup_by_cbsa(MONITORS, bdate, edate, param, cbsa)
result$Data

## End(Not run)
```

---

lookup_by_county *Internal function to perform geospatial lookup by county.*

Description

Internal function to perform geospatial lookup by county.

Usage

```
lookup_by_county(
  endpoint,
  bdate,
  edate,
  state.fips,
```

```
  county,
  param,
  cached = TRUE,
  cache_directory = "/cache",
  duration = NULL,
  cbdate = NULL,
  cedate = NULL
)
```

Arguments

endpoint	Base url to make call.
bdate	Beginning date to check. Year, month, day format.
edate	Ending date to check. Year, month, day format.
state.fips	State FIPS code. Use get_state_fips() if unsure.
county	County code. Use get_counties_in_state() if unsure.
param	Pollutant parameter that site is measuring.
cached	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
cache_directory	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)
duration	The 1-character AQS sample duration code. (Optional)
cbdate	Beginning date of last change to DB. (Optional)
cedate	Ending date of last change to DB. (Optional)

Value

API response containing operational information about the monitor.

Examples

```
## Not run:
bdate <- "20200101"
edate <- "20200102"
state.fips <- "37"
county <- "001"
param <- "42401"
result <- lookup_by_county(MONITORS, bdate, edate, state.fips, county, param)
result$Data

## End(Not run)
```

lookup_by_ma *Internal function to perform geospatial lookup by monitoring agency.*

Description

Internal function to perform geospatial lookup by monitoring agency.

Usage

```
lookup_by_ma(
  endpoint,
  bdate,
  edate,
  param,
  agency,
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

endpoint	Base url to make call.
bdate	Beginning date to check. Year, month, day format.
edate	Ending date to check. Year, month, day format.
param	Pollutant parameter that site is measuring.
agency	The monitoring agency.
cached	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
cache_directory	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Examples

```
## Not run:
bdate <- "20200101"
edate <- "20201231"
param <- "44201"
agency <- "0013"
result <- lookup_by_ma(QA_APE, bdate, edate, param, agency)
result$Data

## End(Not run)
```

lookup_by_pqao *Internal function to perform geospatial lookup by primary quality assurance organization.*

Description

Internal function to perform geospatial lookup by primary quality assurance organization.

Usage

```
lookup_by_pqao(
  endpoint,
  bdate,
  edate,
  param,
  pqao,
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

endpoint	Base url to make call.
bdate	Beginning date to check. Year, month, day format.
edate	Ending date to check. Year, month, day format.
param	Pollutant parameter that site is measuring.
pqao	An encoding for a Primary Quality Assurance Organization. If unsure, use get_all_pqaos().
cached	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
cache_directory	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)

Examples

```
## Not run:
bdate <- "20200101"
edate <- "20201231"
param <- "44201"
pqao <- "0013"
result <- lookup_by_pqao(QA_APE, bdate, edate, param, pqao)
result$Data

## End(Not run)
```

lookup_by_site *Internal function to perform geospatial lookup by site.*

Description

Internal function to perform geospatial lookup by site.

Usage

```
lookup_by_site(
  endpoint,
  bdate,
  edate,
  state.fips,
  county,
  param,
  site,
  cached = TRUE,
  cache_directory = "/cache",
  duration = NULL,
  cbdate = NULL,
  cedate = NULL
)
```

Arguments

endpoint	Base url to make call.
bdate	Beginning date to check. Year, month, day format.
edate	Ending date to check. Year, month, day format.
state.fips	State FIPS code. Use get_state_fips() if unsure.
county	County code. Use get_counties_in_state() if unsure.
param	Pollutant parameter that site is measuring.
site	Measurement site code. Use get_sites_by_county() if unsure.
cached	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
cache_directory	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)
duration	The 1-character AQS sample duration code. (Optional)
cbdate	Beginning date of last change to DB. (Optional)
cedate	Ending date of last change to DB. (Optional)

Value

API response containing operational information about the monitor.

Examples

```
## Not run:
bdate <- "20200101"
edate <- "20200102"
state.fips <- "37"
county <- "001"
site <- "001"
param <- "42401"
result <- lookup_by_site(MONITORS,
                         bdate,
                         edate,
                         state.fips,
                         county,
                         param,
                         site)
result$Data

## End(Not run)
```

---

lookup_by_state *Internal function to perform geospatial lookup by state.*

Description

Internal function to perform geospatial lookup by state.

Usage

```
lookup_by_state(
  endpoint,
  bdate,
  edate,
  state.fips,
  param,
  cached = TRUE,
  cache_directory = "/cache",
  duration = NULL,
  cbdate = NULL,
  cedate = NULL
)
```

Arguments

endpoint	Base url to make call.
bdate	Beginning date to check. Year, month, day format.
edate	Ending date to check. Year, month, day format.
state.fips	State FIPS code. Use get_state_fips() if unsure.

param	Pollutant parameter that site is measuring.
cached	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE. (Optional)
cache_directory	Place inside user-level cache directory to store the cached data. Default: "/cache". (Optional)
duration	The 1-character AQS sample duration code. (Optional)
cbdate	Beginning date of last change to DB. (Optional)
cedate	Ending date of last change to DB. (Optional)

Value

API response containing operational information about the monitor.

Examples

```
## Not run:
bdate <- "20200101"
edate <- "20200102"
state.fips <- "37"
param <- "42401"
result <- lookup_by_state(MONITORS, bdate, edate, state.fips, param)
result$Data

## End(Not run)
```

---

non.cached.perform.call

Perform call and convert data into list

Description

Perform call and convert data into list

Usage

```
non.cached.perform.call(endpoint, variables = list())
```

Arguments

endpoint	An endpoint from the available EPA API endpoints
variables	A list of variables or a single variable to filter the EPA API endpoint.

Value

A list containing requested data

Examples

```
## Not run:
endpoint <- 'list/states'
result <- non.cached.perform.call(endpoint)

## End(Not run)
```

`perform.call` *Cached version of the perform.call function*

Description

Cached version of the perform.call function

Usage

```
perform.call(
  endpoint,
  variables = list(),
  cached = TRUE,
  cache_directory = "/cache"
)
```

Arguments

`endpoint`	An endpoint from the available EPA API endpoints
`variables`	A list of variables or a single variable to filter the EPA API endpoint.
`cached`	TRUE or FALSE specifying if the data from the call is to be cached. Default: TRUE.
`cache_directory`	Place inside user-level cache directory to store the cached data. Default: "/cache".

Value

A list containing requested data

Examples

```
## Not run:
endpoint <- 'list/states'
result <- perform.call(endpoint)

## End(Not run)
```

perform.call.raw *Perform call and keep original result*

Description

Perform call and keep original result

Usage

```
perform.call.raw(endpoint, variables = list())
```

Arguments

endpoint	An endpoint from the available EPA API endpoints
variables	A list of variables or a single variable to filter the EPA API endpoint.

Value

A list containing result from query to EPA API

Examples

```
## Not run:
endpoint <- 'list/states'
result <- perform.call.raw(endpoint)

## End(Not run)
```

place.call *Place the URL as a call to the EPA API*

Description

Place the URL as a call to the EPA API

Usage

```
place.call(url)
```

Arguments

url	A string with a valid URL for the EPA API

Value

Result of query from the API

Examples

```
## Not run:
url <- "user_url"
result <- place.call(url)

## End(Not run)
```

`place.call.raw` *Perform call and maintain JSON Lite structure*

Description

Perform call and maintain JSON Lite structure

Usage

```
place.call.raw(url)
```

Arguments

`url`	URL following structure from EPA API

Value

Results of data request in JSON format

Examples

```
## Not run:
endpoint <- 'list/states'
call <- create.base.call(endpoint)
raw.call <- place.call.raw(call)
raw.call

## End(Not run)
```

`remove.escapes.spaces` *Remove tabs, new lines, and empty spaces from entries in a data frame*

Description

Remove tabs, new lines, and empty spaces from entries in a data frame

Usage

```
remove.escapes.spaces(df)
```

Arguments

df	Data frame to remove tabs, new lines, and empty spaces from

Value

Data frame without tabs, new lines, and empty spaces

Examples

```
service <- c("Sign up")
description <- c("Email will\r\n\t\t\t\t\t\t\t be sent to the registered
address from aqsdatamart@epa.gov.")

og.df <- data.frame(service, description)
clean.df <- epair:::remove.escapes.spaces(og.df)
clean.df
```

---

retrieve.cached.call *Retrieves memory of previously cached call.*

Description

Retrieves memory of previously cached call.

Usage

```
retrieve.cached.call(endpoint, variables = list(), directory = "/cache")
```

Arguments

endpoint	An endpoint from the available EPA API endpoints
variables	A list of variables or a single variable to filter the EPA API endpoint.
directory	Place inside user-level cache directory that was used to store the cached data previously. Default: "/cache".

Value

A list containing requested data

Examples

```
endpoint <- 'list/states'
retrieve.cached.call(endpoint)
```

save.new.cached.call *Saves a new call that has not been previously cached yet.*

Description

Saves a new call that has not been previously cached yet.

Usage

```
save.new.cached.call(endpoint, variables = list(), directory = "/cache")
```

Arguments

endpoint	An endpoint from the available EPA API endpoints
variables	A list of variables or a single variable to filter the EPA API endpoint.
directory	Place inside user-level cache directory that was used to store the cached data previously. Default: "/cache".

Value

A list containing requested data

Examples

```
## Not run:
endpoint <- 'list/states'
save.new.cached.call(endpoint)

## End(Not run)
```

service.names *Names of services offered by the EPA API*

Description

The service.names list contains names of all services offered by the EPA API along with a description of each service.

services *Services offered by the EPA API*

Description

The services list contains comprehensive information about all services provided by the EPA API site.

string.replacer *Replace all characters entries in data frame*

Description

Replace all characters entries in data frame

Usage

```
string.replacer(df, pattern, replacement)
```

Arguments

df	Data frame containing character entries
pattern	Pattern to use for matching
replacement	Replacement of entries matching pattern

Value

A data frame with entries following the pattern being replaced by replacement

Examples

```
df <- data.frame(c("1", "2", "3", "4"))
modified.df <- epair:::string.replacer(df, "1", "One")
modified.df
```

variable.types *Variable parameter names to use*

Description

The variable.types list contains the listing endpoints for finding out more information in making calls requiring more variables.

variables *Variables used for querying in EPA API*

Description

The variables data frame contains information about what variables can be used to build queries in the EPA API.

Index