

Package: epair (via r-universe)

July 31, 2026

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/aqswweb/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 7.3.3

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

Config/pak/sysreqs libicu-dev libssl-dev

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

Date/Publication 2025-12-01 19:01:14 UTC

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

RemoteRef master

RemoteSha e8f1f1d4ba115f8189e76ddea2f9206d2ab8cc11

Contents

add.variables	4
clear.all.cached	5
clear.cached	6
create.authentication	6
create.base.call	7
endpoints	7
get.transpose	8
get_all_mas	8
get_all_pqaos	9
get_annual_summary_in_bbox	9
get_annual_summary_in_cbsa	11
get_annual_summary_in_county	12
get_annual_summary_in_site	13
get_annual_summary_in_state	15
get_aqs_key	16
get_cbsas	16
get_counties_in_state	17
get_daily_summary_in_bbox	17
get_daily_summary_in_cbsa	19
get_daily_summary_in_county	20
get_daily_summary_in_site	21
get_daily_summary_in_state	23
get_fields_by_service	24
get_known_issues	24
get_monitors_in_bbox	25
get_monitors_in_cbsa	26
get_monitors_in_county	27
get_monitors_in_site	28
get_monitors_in_state	29
get_parameter_classes	30
get_parameters_in_class	31
get_qa_ape_in_agency	31
get_qa_ape_in_county	32
get_qa_ape_in_pqao	33
get_qa_ape_in_site	34
get_qa_ape_in_state	36
get_qa_blanks_in_agency	37
get_qa_blanks_in_county	38
get_qa_blanks_in_pqao	39
get_qa_blanks_in_site	40
get_qa_blanks_in_state	41
get_qa_ca_in_agency	42
get_qa_ca_in_county	43
get_qa_ca_in_pqao	44
get_qa_ca_in_site	45
get_qa_ca_in_state	46

get_qa_fra_in_agency	47
get_qa_fra_in_county	48
get_qa_fra_in_pqao	50
get_qa_fra_in_site	51
get_qa_fra_in_state	52
get_qa_frv_in_agency	53
get_qa_frv_in_county	54
get_qa_frv_in_pqao	55
get_qa_frv_in_site	56
get_qa_frv_in_state	57
get_qa_pep_in_agency	58
get_qa_pep_in_county	59
get_qa_pep_in_pqao	60
get_qa_pep_in_site	61
get_qa_pep_in_state	62
get_qa_qc_in_agency	63
get_qa_qc_in_county	64
get_qa_qc_in_pqao	65
get_qa_qc_in_site	66
get_qa_qc_in_state	68
get_quarterly_summary_in_bbox	69
get_quarterly_summary_in_cbsa	70
get_quarterly_summary_in_county	71
get_quarterly_summary_in_site	72
get_quarterly_summary_in_state	73
get_revision_history	74
get_samples_in_bbox	74
get_samples_in_cbsa	76
get_samples_in_county	77
get_samples_in_site	78
get_samples_in_state	80
get_sites_in_county	81
get_state_fips	82
get_tf_qa_ape_in_agency	82
get_tf_qa_ape_in_county	83
get_tf_qa_ape_in_pqao	85
get_tf_qa_ape_in_site	86
get_tf_qa_ape_in_state	87
get_tf_sample_in_agency	88
get_tf_sample_in_county	89
get_tf_sample_in_site	90
get_tf_sample_in_state	91
is_API_running	92
list.cached.data	93
list.remove.escapes.spaces	93
list.string.replacer	94
lookup_by_bbox	95
lookup_by_cbsa	96

lookup_by_county	97
lookup_by_ma	99
lookup_by_pqao	100
lookup_by_site	101
lookup_by_state	102
non.cached.perform.call	103
perform.call	104
perform.call.raw	105
place.call	105
place.call.raw	106
remove.escapes.spaces	106
retrieve.cached.call	107
save.new.cached.call	108
service.names	108
services	108
string.replacer	109
variable.types	109
variables	109
Index	110

add.variables	<i>Add variables to a query</i>
---------------	---------------------------------

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	<i>Removes all cached memory of perform.call</i>
------------------	--

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	<i>Removes memory of cached perform.call data for specific parameters</i>
--------------	---

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	<i>Generate the string authentication needed for EPA API</i>
-----------------------	--

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/data

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	<i>Make the first call when forming a query.</i>
------------------	--

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	<i>Endpoints available in the EPA API</i>
-----------	---

Description

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

get.transpose	<i>Transpose a data frame</i>
---------------	-------------------------------

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 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	<i>Get Monitoring Agencies.</i>
-------------	---------------------------------

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	<i>Get Primary Quality Assurance Organizations.</i>
---------------	---

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	<i>Get annual summary data in a bounding box (lat, long).</i>
----------------------------	---

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 Statistical 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",
```

```

    cdate = NULL,
    edate = 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 <code>get_state_fips()</code> if unsure.
county	County code. Use <code>get_counties_in_state()</code> if unsure.
site	Measurement site code. Use <code>get_sites_by_county()</code> 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: <code>"/cache"</code> . (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	<i>Get an API key from the AQS API.</i>
-------------	---

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	<i>Get all Core Based Statistical Areas.</i>
-----------	--

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 Statistical 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",
```

`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_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_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_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	<i>Get quality assurance annual performance evaluations for a primary quality assurance organization.</i>
--------------------	---

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	<i>Get quality assurance annual performance evaluations at a site.</i>
--------------------	--

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	<i>Get quality assurance annual performance evaluations in a state.</i>
---------------------	---

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)
```



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

get_qa_blanks_in_county
<i>Get quality assurance blank data in a county.</i>

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	<i>Get quality assurance collocated assessment data for a monitoring agency.</i>
---------------------	--

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	<i>Get quality assurance collocated assessment data for a primary quality assurance organization.</i>
-------------------	---

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	<i>Get quality assurance collocated assessment data at a site.</i>
-------------------	--

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	<i>Get quality assurance collocated assessment data in a state.</i>
--------------------	---

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	<i>Get quality assurance flow rate audit data for a primary quality assurance organization.</i>
--------------------	---

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	<i>Get quality assurance flow rate audit data at a site.</i>
--------------------	--

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	<i>Get quality assurance flow rate audit data in a state.</i>
---------------------	---

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	<i>Get quality assurance flow rate verification data for a primary quality assurance organization.</i>
--------------------	--

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	<i>Get quality assurance flow rate verification data at a site.</i>
--------------------	---

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	<i>Get quality assurance flow rate verification data in a state.</i>
---------------------	--

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	<i>Get quality assurance PEP audit data for a monitoring agency.</i>
----------------------	--

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	<i>Get quality assurance PEP audit data for a primary quality assurance organization.</i>
--------------------	---

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	<i>Get quality assurance PEP audit data at a site.</i>
--------------------	--

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	<i>Get quality assurance one point quality control data for a monitoring agency.</i>
---------------------	--

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	<i>Get quality assurance annual one point quality control data in a county.</i>
---------------------	---

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	<i>Get quality assurance one point quality control data for a primary quality assurance organization.</i>
-------------------	---

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	<i>Get quality assurance one point quality control data at a site.</i>
-------------------	--

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	<i>Get quality assurance one point quality control data in a state.</i>
--------------------	---

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 Statistical 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 Statistical 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	<i>Get samples (finest grained data) for a state.</i>
----------------------	---

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	<i>Get all measurement sites within a county.</i>
---------------------	---

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 <code>get_state_fips()</code> to find the appropriate FIPS code.
county.code	A county code. Use <code>get_counties_in_state()</code> 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	<i>Get all FIPS codes for each US state.</i>
----------------	--

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	<i>Get quality assurance annual performance evaluations for a monitoring agency in transaction format.</i>
-------------------------	--

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 <code>get_state_fips()</code> if unsure.
county	County code. Use <code>get_counties_in_state()</code> 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: <code>"/cache"</code> . (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

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

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	<i>Shows contents of cache directory</i>
------------------	--

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	<i>Remove tabs, new lines, and empty spaces from entries in a list</i>
----------------------------	--

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 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	<i>Replace every string entry in a list</i>
----------------------	---

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	<i>Internal function to perform geospatial lookup by bounding box (lat, long).</i>
----------------	--

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	<i>Internal function to perform geospatial lookup by Core Based Statistical Area.</i>
----------------	---

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 Statistical 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	<i>Internal function to perform geospatial lookup by county.</i>
------------------	--

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	<i>Internal function to perform geospatial lookup by monitoring agency.</i>
--------------	---

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	<i>Internal function to perform geospatial lookup by primary quality assurance organization.</i>
----------------	--

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	<i>Internal function to perform geospatial lookup by site.</i>
----------------	--

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	<i>Internal function to perform geospatial lookup by state.</i>
-----------------	---

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	<i>Perform call and keep original result</i>
------------------	--

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	<i>Place the URL as a call to the EPA API</i>
------------	---

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	<i>Perform call and maintain JSON Lite structure</i>
----------------	--

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	<i>Remove tabs, new lines, and empty spaces from entries in a data frame</i>
-----------------------	--

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 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
```

<code>retrieve.cached.call</code>	<i>Retrieves memory of previously cached call.</i>
-----------------------------------	--

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)
```

<code>save.new.cached.call</code>	<i>Saves a new call that has not been previously cached yet.</i>
-----------------------------------	--

Description

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

Usage

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

Arguments

<code>endpoint</code>	An endpoint from the available EPA API endpoints
<code>variables</code>	A list of variables or a single variable to filter the EPA API endpoint.
<code>directory</code>	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)
```

<code>service.names</code>	<i>Names of services offered by the EPA API</i>
----------------------------	---

Description

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

<code>services</code>	<i>Services offered by the EPA API</i>
-----------------------	--

Description

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

string.replacer	<i>Replace all characters entries in data frame</i>
-----------------	---

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	<i>Variable parameter names to use</i>
----------------	--

Description

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

variables	<i>Variables used for querying in EPA API</i>
-----------	---

Description

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

Index

* datasets

- endpoints, [7](#)
- service.names, [108](#)
- services, [108](#)
- variable.types, [109](#)
- variables, [109](#)

add.variables, [4](#)

clear.all.cached, [5](#)

clear.cached, [6](#)

create.authentication, [6](#)

create.base.call, [7](#)

endpoints, [7](#)

get.transpose, [8](#)

get_all_mas, [8](#)

get_all_pqaos, [9](#)

get_annual_summary_in_bbox, [9](#)

get_annual_summary_in_cbsa, [11](#)

get_annual_summary_in_county, [12](#)

get_annual_summary_in_site, [13](#)

get_annual_summary_in_state, [15](#)

get_aqs_key, [16](#)

get_cbsas, [16](#)

get_counties_in_state, [17](#)

get_daily_summary_in_bbox, [17](#)

get_daily_summary_in_cbsa, [19](#)

get_daily_summary_in_county, [20](#)

get_daily_summary_in_site, [21](#)

get_daily_summary_in_state, [23](#)

get_fields_by_service, [24](#)

get_known_issues, [24](#)

get_monitors_in_bbox, [25](#)

get_monitors_in_cbsa, [26](#)

get_monitors_in_county, [27](#)

get_monitors_in_site, [28](#)

get_monitors_in_state, [29](#)

get_parameter_classes, [30](#)

get_parameters_in_class, [31](#)

get_qa_ape_in_agency, [31](#)

get_qa_ape_in_county, [32](#)

get_qa_ape_in_pqao, [33](#)

get_qa_ape_in_site, [34](#)

get_qa_ape_in_state, [36](#)

get_qa_blanks_in_agency, [37](#)

get_qa_blanks_in_county, [38](#)

get_qa_blanks_in_pqao, [39](#)

get_qa_blanks_in_site, [40](#)

get_qa_blanks_in_state, [41](#)

get_qa_ca_in_agency, [42](#)

get_qa_ca_in_county, [43](#)

get_qa_ca_in_pqao, [44](#)

get_qa_ca_in_site, [45](#)

get_qa_ca_in_state, [46](#)

get_qa_fra_in_agency, [47](#)

get_qa_fra_in_county, [48](#)

get_qa_fra_in_pqao, [50](#)

get_qa_fra_in_site, [51](#)

get_qa_fra_in_state, [52](#)

get_qa_frv_in_agency, [53](#)

get_qa_frv_in_county, [54](#)

get_qa_frv_in_pqao, [55](#)

get_qa_frv_in_site, [56](#)

get_qa_frv_in_state, [57](#)

get_qa_pep_in_agency, [58](#)

get_qa_pep_in_county, [59](#)

get_qa_pep_in_pqao, [60](#)

get_qa_pep_in_site, [61](#)

get_qa_pep_in_state, [62](#)

get_qa_qc_in_agency, [63](#)

get_qa_qc_in_county, [64](#)

get_qa_qc_in_pqao, [65](#)

get_qa_qc_in_site, [66](#)

get_qa_qc_in_state, [68](#)

get_quarterly_summary_in_bbox, [69](#)

get_quarterly_summary_in_cbsa, [70](#)

get_quarterly_summary_in_county, [71](#)

- get_quarterly_summary_in_site, [72](#)
- get_quarterly_summary_in_state, [73](#)
- get_revision_history, [74](#)
- get_samples_in_bbox, [74](#)
- get_samples_in_cbsa, [76](#)
- get_samples_in_county, [77](#)
- get_samples_in_site, [78](#)
- get_samples_in_state, [80](#)
- get_sites_in_county, [81](#)
- get_state_fips, [82](#)
- get_tf_qa_ape_in_agency, [82](#)
- get_tf_qa_ape_in_county, [83](#)
- get_tf_qa_ape_in_pqao, [85](#)
- get_tf_qa_ape_in_site, [86](#)
- get_tf_qa_ape_in_state, [87](#)
- get_tf_sample_in_agency, [88](#)
- get_tf_sample_in_county, [89](#)
- get_tf_sample_in_site, [90](#)
- get_tf_sample_in_state, [91](#)

- is_API_running, [92](#)

- list.cached.data, [93](#)
- list.remove.escapes.spaces, [93](#)
- list.string.replacer, [94](#)
- lookup_by_bbox, [95](#)
- lookup_by_cbsa, [96](#)
- lookup_by_county, [97](#)
- lookup_by_ma, [99](#)
- lookup_by_pqao, [100](#)
- lookup_by_site, [101](#)
- lookup_by_state, [102](#)

- non.cached.perform.call, [103](#)

- perform.call, [104](#)
- perform.call.raw, [105](#)
- place.call, [105](#)
- place.call.raw, [106](#)

- remove.escapes.spaces, [106](#)
- retrieve.cached.call, [107](#)

- save.new.cached.call, [108](#)
- service.names, [108](#)
- services, [108](#)
- string.replacer, [109](#)

- variable.types, [109](#)
- variables, [109](#)