Package 'epidatr'

September 19, 2023

Type Package

Title Client for Delphi's 'Epidata' API

Version 1.0.0

Date 2023-09-11

URL https://cmu-delphi.github.io/epidatr/,
https://cmu-delphi.github.io/delphi-epidata/,
https://github.com/cmu-delphi/epidatr

BugReports https://github.com/cmu-delphi/epidatr/issues

Description The Delphi 'Epidata' API provides real-time access to epidemiological surveillance data for influenza, 'COVID-19', and other diseases for the USA at various geographical resolutions, both from official government sources such as the Center for Disease Control (CDC) and Google Trends and private partners such as Facebook and Change 'Healthcare'. It is built and maintained by the Carnegie Mellon University Delphi research group. To cite this API: David C. Farrow, Logan C. Brooks, Aaron 'Rumack', Ryan J. 'Tibshirani', 'Roni' 'Rosenfeld' (2015). Delphi 'Epidata' API. <https://github.com/cmu-delphi/delphi-epidata>.

Depends R (>= 3.5.0)

License MIT + file LICENSE

Encoding UTF-8

Imports cachem, checkmate, cli, httr, glue, jsonlite, magrittr, MMWRweek, purrr, openssl, readr, tibble, xml2

RoxygenNote 7.2.3

Suggests dplyr, ggplot2, knitr, rmarkdown, rlang, testthat (>= 3.1.5), withr

VignetteBuilder knitr

Language en-US

Config/testthat/edition 3

Collate 'auth.R' 'avail_endpoints.R' 'cache.R' 'check.R' 'constants.R' 'covidcast.R' 'endpoints.R' 'epidatacall.R' 'epidatr-package.R' 'model.R' 'request.R' 'utils-pipe.R' 'utils.R'

NeedsCompilation no

Author Logan Brooks [aut],
Dmitry Shemetov [aut],
Samuel Gratzl [aut],
David Weber [ctb, cre],
Alex Reinhart [ctb],
Daniel McDonald [ctb],
Kean Ming Tan [ctb],
Will Townes [ctb],
George Haff [ctb],
Kathryn Mazaitis [ctb]

Maintainer David Weber <davidweb@andrew.cmu.edu>

Repository CRAN

Date/Publication 2023-09-19 14:10:02 UTC

R topics documented:

`avail_endpoints` *List all available endpoints.*

Description

A function that prints a tibble with two columns: `Endpoint` contains the function for accessing the Delphi Epidata API endpoint along with a `Description`.

Usage

```
avail_endpoints()
```

Value

A `tibble::tibble`.

Examples

```
avail_endpoints()
```

`cache_info` *Describe current cache*

Description

Print out the information about the cache (as would be returned by cachem's `info()` method).

Usage

```
cache_info()
```

Value

`list` containing the info result as created by cachem

See Also

set_cache to start a new cache (and general caching info), clear_cache to delete the cache and set a new one, and disable_cache to disable without deleting

---

clear_cache *Manually reset the cache, deleting all currently saved data and starting afresh*

Description

Deletes the current cache and resets a new cache. Deletes local data! If you are using a session unique cache, you will have to pass the arguments you used for set_cache earlier, otherwise the system-wide .Renviron-based defaults will be used.

Usage

```
clear_cache(disable = FALSE, ...)
```

Arguments

disable	instead of setting a new cache, disable caching entirely; defaults to FALSE
...	Arguments passed on to set_cache
	cache_dir the directory in which the cache is stored. By default, this is tools::R_user_dir() if on R 4.0+, but must be specified for earlier versions of R. The path can be either relative or absolute. The environmental variable is EPIDATR_CACHE_DIR.
	days the maximum length of time in days to keep any particular cached call. By default this is 1. The environmental variable is EPIDATR_CACHE_MAX_AGE_DAYS.
	max_size the size of the entire cache, in MB, at which to start pruning entries. By default this is 1024, or 1GB. The environmental variable is EPIDATR_CACHE_MAX_SIZE_MB.
	logfile where cachem's log of transactions is stored, relative to the cache directory. By default, it is "logfile.txt". The environmental variable is EPIDATR_CACHE_LOGFILE.
	confirm whether to confirm directory creation. default is TRUE; should only be set in non-interactive scripts

Value

NULL no return value, all effects are stored in the package environment

See Also

set_cache to start a new cache (and general caching info), disable_cache to only disable without deleting, and cache_info

`covidcast_epidata` *Creates the COVIDcast Epidata autocomplete helper*

Description

Creates a helper object that can use auto-complete to help find COVIDcast sources and signals. The COVIDcast endpoint of the Epidata API contains many separate data sources and signals. It can be difficult to find the name of the signal you're looking for, so you can use `covidcast_epidata` to get help with finding sources and functions without leaving R.

The `covidcast_epidata()` function fetches a list of all signals, and returns an object containing fields for every signal:

```
epidata <- covidcast_epidata()
epidata$signals
#> # A tibble: 443 x 3
#>    source        signal                        short_description
#>    <chr>         <chr>                         <chr>
#>  1 chng         smoothed_outpatient_cli      Estimated percentage of outpatie~
#>  2 chng         smoothed_adj_outpatient_cli  Estimated percentage of outpatie~
#>  3 chng            smoothed_outpatient_covid     COVID-Confirmed Doctor Visits
#>  4 chng            smoothed_adj_outpatient_covid COVID-Confirmed Doctor Visits
#>  5 chng         smoothed_outpatient_flu      Estimated percentage of outpatie~
#>  6 chng         smoothed_adj_outpatient_flu  Estimated percentage of outpatie~
#>  7 covid-act-now pcr_specimen_positivity_rate  Proportion of PCR specimens test~
#>  8 covid-act-now pcr_specimen_total_tests     Total number of PCR specimens te~
#>  9 doctor-visits smoothed_cli                 Percentage of daily doctor visit~
#> 10 doctor-visits smoothed_adj_cli             Percentage of daily doctor visit~
#> # i 433 more rows
```

If you use an editor that supports tab completion, such as RStudio, type `epidata$signals$` and wait for the tab completion popup. You will be able to type the name of signals and have the autocomplete feature select them from the list for you. Note that some signal names have dashes in them, so to access them we rely on the backtick operator:

```
epidata$signals$`fb-survey:smoothed_cli`
#> [1] "COVID-Like Symptoms (Unweighted 7-day average)"
#> [1] "fb-survey:smoothed_cli"
#> [1] "Estimated percentage of people with COVID-like illness "
```

These objects can be used directly to fetch data, without requiring us to use the `covidcast()` function. Simply use the `$call` attribute of the object:

```
epidata$signals$`fb-survey:smoothed_cli`$call("state", "pa",
                                              epirange(20210405, 20210410))
#> # A tibble: 6 x 15
#>   geo_value signal     source geo_type time_type time_value direction issue
```

```
#>   <chr>     <chr>      <chr>  <fct>    <fct>     <date>         <dbl> <date>
#> 1 pa       smoothed_~ fb-su~ state    day       2021-04-05       NA 2021-04-10
#> 2 pa       smoothed_~ fb-su~ state    day       2021-04-06       NA 2021-04-11
#> 3 pa       smoothed_~ fb-su~ state    day       2021-04-07       NA 2021-04-12
#> 4 pa       smoothed_~ fb-su~ state    day       2021-04-08       NA 2021-04-13
#> 5 pa       smoothed_~ fb-su~ state    day       2021-04-09       NA 2021-04-14
#> 6 pa       smoothed_~ fb-su~ state    day       2021-04-10       NA 2021-04-15
#> # i 7 more variables: lag <int>, missing_value <int>, missing_stderr <int>,
#> #   missing_sample_size <int>, value <dbl>, stderr <dbl>, sample_size <dbl>
```

Usage

```
covidcast_epidata(base_url = global_base_url, timeout_seconds = 30)
```

Arguments

base_url	optional alternative API base url
timeout_seconds	the maximum amount of time to wait for a response

Value

An instance of covidcast_epidata

---

create_epidata_call *An abstraction that holds information needed to make an epidata request*

Description

epidata_call objects are generated internally by endpoint functions like pub_covidcast; by default, they are piped directly into the fetch function to fetch and format the data. For most endpoints this will return a tibble, but a few non-COVIDCAST endpoints only support will return a JSON-like list instead.

Usage

```
create_epidata_call(
  endpoint,
  params,
  meta = NULL,
  only_supports_classic = FALSE
)

fetch(epidata_call, fetch_args = fetch_args_list())
```

Arguments

endpoint	the epidata endpoint to call
params	the parameters to pass to the epidata endpoint
meta	meta data to attach to the epidata call
only_supports_classic	if true only classic format is supported
epidata_call	an instance of epidata_call
fetch_args	a fetch_args object

Details

create_epidata_call is the constructor for epidata_call objects, but you should not need to use it directly; instead, use an endpoint function, e.g., pub_covidcast, to generate an epidata_call for the data of interest.

There are some other functions available for debugging and advanced usage: - request_url (for debugging): outputs the request URL from which data would be fetched (note additional parameters below)

fetch usually returns the data in tibble format, but a few of the endpoints only support the JSON classic format (delphi, pvt_meta_norostat, and meta). In that case a JSON-like nested list structure is returned instead.

Value

- For create_epidata_call: an epidata_call object
- For fetch: a tibble or a JSON-like list

Examples

```
## Not run:
call <- pub_covidcast(
  source = "jhu-csse",
  signals = "confirmed_7dav_incidence_prop",
  time_type = "day",
  geo_type = "state",
  time_values = epirange(20200601, 20200801),
  geo_values = c("ca", "fl"),
  fetch_args = fetch_args_list(dry_run = TRUE)
)
call %>% fetch()

## End(Not run)
```

disable_cache *Turn off the caching for this session*

Description

Disable caching until you call `set_cache` or restart R. The files defining the cache are untouched. If you are looking to disable the caching more permanently, set `EPIDATR_USE_CACHE=FALSE` as environmental variable in your `.Renviron`.

Usage

```
disable_cache()
```

Value

`NULL` no return value, all effects are stored in the package environment

See Also

`set_cache` to start a new cache (and general caching info), `clear_cache` to delete the cache and set a new one, and `cache_info`

epirange *EpiRange*

Description

Specify a date range (in days or epiweeks) for an API request.

Usage

```
epirange(from, to)
```

Arguments

`from`	A `Date`, integer-like value, or integer-like string that takes the form YYYYMMDD for dates or YYYYWW for epiweeks.
`to`	A `Date`, integer-like value, or integer-like string that takes the form YYYYMMDD for dates or YYYYWW for epiweeks.

Details

Epiweeks, also known as MMWR weeks number the weeks of the year from 1 to 53, each week spanning from Sunday to Saturday. The numbering is defined by the CDC.

Value

An EpiRange object.

Examples

```
# Represents 2021-01-01 to 2021-01-07, inclusive
epirange(20210101, 20210107)

# The same, but using Date objects
epirange(as.Date("2021-01-01"), as.Date("2021-01-07"))

# Represents epiweeks 2 through 4 of 2022, inclusive
epirange(202202, 202204)
```

---

fetch_args_list *Customize fetch settings*

Description

A constructor for fetch_args objects, which are used to pass arguments to the fetch function.

Usage

```
fetch_args_list(
  ...,
  fields = NULL,
  disable_date_parsing = FALSE,
  disable_data_frame_parsing = FALSE,
  return_empty = FALSE,
  timeout_seconds = 30,
  base_url = NULL,
  dry_run = FALSE,
  debug = FALSE,
  format_type = "json"
)
```

Arguments

...	not used for values, forces later arguments to bind by name
fields	a list of epidata fields to return, or NULL to return all fields (default) e.g. c("time_value", "value") to return only the time_value and value fields or c("-direction") to return everything except the direction field
disable_date_parsing	disable automatic date parsing; by default FALSE
disable_data_frame_parsing	disable automatic conversion to data frame; this is only supported by endpoints that only support the 'classic' format (non-tabular). by default FALSE

return_empty	boolean that allows returning an empty tibble if there is no data; by default FALSE
timeout_seconds	the maximum amount of time to wait for a response; by default FALSE
base_url	base URL to use; by default NULL, which means the global base url "https://api.delphi.cmu.edu/epi
dry_run	boolean that allows skipping the call to the API and instead returns the epidata_call object (useful for debugging); by default TRUE
debug	boolean that allows returning the raw response from the API; by default FALSE
format_type	the format to request from the API, one of classic, json, csv; this is only used by fetch_debug, and by default is "json"

Value

- For fetch_args_list: a fetch_args object

---

get_auth_key *Getting the API key*

Description

Get the API key from the environment variable DELPHI_EPIDATA_KEY or getOption("delphi.epidata.key").

Usage

```
get_auth_key()
```

Value

The API key as a string or "".

---

pub_covidcast *COVID data via the covidcast endpoint*

Description

API docs: https://cmu-delphi.github.io/delphi-epidata/api/covidcast_signals.html

The primary endpoint for fetching COVID-19 data, providing access to a wide variety of signals from a wide variety of sources. See the API documentation link above for more. Delphi's COVIDcast public dashboard is powered by this endpoint.

Usage

```
pub_covidcast(
  source,
  signals,
  geo_type,
  time_type,
  geo_values,
  time_values,
  ...,
  as_of = NULL,
  issues = NULL,
  lag = NULL,
  fetch_args = fetch_args_list()
)
```

Arguments

source	string. The data source to query (see: https://cmu-delphi.github.io/delphi-epidata/api/covidcast_signals.html).
signals	string. The signals to query from a specific source (see: https://cmu-delphi.github.io/delphi-epidata/api/covidcast_signals.html).
geo_type	string. The geographic resolution of the data (see: https://cmu-delphi.github.io/delphi-epidata/api/covidcast_geography.html).
time_type	string. The temporal resolution of the data (either "day" or "week", depending on signal).
geo_values	character. The geographies to return. "*" fetches all. (See: https://cmu-delphi.github.io/delphi-epidata/api/covidcast_geography.html.)
time_values	timeset. Dates to fetch.
...	not used for values, forces later arguments to bind by name
as_of	Date. Optionally, the as of date for the issues to fetch. If not specified, the most recent data is returned. Mutually exclusive with issues or lag.
issues	timeset. Optionally, the issue of the data to fetch. If not specified, the most recent issue is returned. Mutually exclusive with as_of or lag.
lag	integer. Optionally, the lag of the issues to fetch. If not set, the most recent issue is returned. Mutually exclusive with as_of or issues.
fetch_args	fetch_args. Additional arguments to pass to fetch().

Value

tibble::tibble

See Also

pub_covidcast_meta(), covidcast_epidata(), epirange()

Examples

```
## Not run:
pub_covidcast(
  source = "jhu-csse",
  signals = "confirmed_7dav_incidence_prop",
  geo_type = "state",
  time_type = "day",
  geo_values = c("ca", "fl"),
  time_values = epirange(20200601, 20200801)
)
pub_covidcast(
  source = "jhu-csse",
  signals = "confirmed_7dav_incidence_prop",
  geo_type = "state",
  time_type = "day",
  geo_values = "*",
  time_values = epirange(20200601, 20200801)
)

## End(Not run)
```

`pub_covidcast_meta` *Metadata for the COVIDcast endpoint*

Description

API docs: https://cmu-delphi.github.io/delphi-epidata/api/covidcast_meta.html.

Fetch a summary of metadata for all sources and signals that are available in the API, along with basic summary statistics such as the dates they are available, the geographic levels at which they are reported, and etc.

Usage

```
pub_covidcast_meta(fetch_args = fetch_args_list())
```

Arguments

`fetch_args`	`fetch_args`. Additional arguments to pass to `fetch()`.

Value

`tibble::tibble`

See Also

`pub_covidcast()`,`covidcast_epidata()`

Examples

```
## Not run:
pub_covidcast_meta()

## End(Not run)
```

`pub_covid_hosp_facility` *COVID hospitalization data for specific facilities*

Description

API docs: https://cmu-delphi.github.io/delphi-epidata/api/covid_hosp_facility.html

Obtains the COVID-19 reported patient impact and hospital capacity data by facility. This dataset is provided by the US Department of Health & Human Services. The companion function `pub_covid_hosp_facility_lookup` can be used to look up facility identifiers in a variety of ways.

Usage

```
pub_covid_hosp_facility(
  hospital_pks,
  collection_weeks,
  ...,
  publication_dates = NULL,
  fetch_args = fetch_args_list()
)
```

Arguments

`hospital_pks`	character. Facility identifiers.
`collection_weeks`	`timeset`. Epiweeks to fetch.
`...`	not used for values, forces later arguments to bind by name
`publication_dates`	`timeset`. Publication dates to fetch.
`fetch_args`	`fetch_args`. Additional arguments to pass to `fetch()`.

Details

Starting October 1, 2022, some facilities are only required to report annually.

Value

`tibble::tibble`

See Also

pub_covid_hosp_facility(), epirange()

Examples

```
## Not run:
pub_covid_hosp_facility(
  hospital_pks = "100075",
  collection_weeks = epirange(20200101, 20200501)
)

## End(Not run)
```

---

pub_covid_hosp_facility_lookup

COVID hospitalization facility identifiers

Description

API docs: https://cmu-delphi.github.io/delphi-epidata/api/covid_hosp_facility_lookup.html

Obtains unique identifiers and other metadata for COVID hospitalization facilities of interest. This is a companion endpoint to the pub_covid_hosp_facility() endpoint.

Usage

```
pub_covid_hosp_facility_lookup(
  ...,
  state = NULL,
  ccn = NULL,
  city = NULL,
  zip = NULL,
  fips_code = NULL,
  fetch_args = fetch_args_list()
)
```

Arguments

...	not used for values, forces later arguments to bind by name
state	string. A two-letter character state abbreviation.
ccn	string. A facility CMS certification number.
city	string. A city name.
zip	string. A 5-digit zip code.
fips_code	string. A 5-digit fips county code, zero-padded.
fetch_args	fetch_args. Additional arguments to pass to fetch().

Details

Only one location argument needs to be specified. Combinations of the arguments are not currently supported.

Value

tibble::tibble

See Also

pub_covid_hosp_facility()

Examples

```
## Not run:
pub_covid_hosp_facility_lookup(state = "fl")
pub_covid_hosp_facility_lookup(city = "southlake")

## End(Not run)
```

---

pub_covid_hosp_state_timeseries

COVID Hospitalization Data by State

Description

API docs: https://cmu-delphi.github.io/delphi-epidata/api/covid_hosp.html.

Obtains the COVID-19 reported patient impact and hospital capacity data by state. This dataset is provided by the US Department of Health & Human Services.

Usage

```
pub_covid_hosp_state_timeseries(
  states,
  dates,
  ...,
  issues = NULL,
  fetch_args = fetch_args_list()
)
```

Arguments

states	character. Two letter state abbreviations.
dates	timeset. Dates to fetch.
...	not used for values, forces later arguments to bind by name
issues	timeset. Optionally, the issues to fetch. If not set, the most recent issue is returned.
fetch_args	fetch_args. Additional arguments to pass to fetch().

Details

Starting October 1, 2022, some facilities are only required to report annually.

Value

tibble::tibble

Examples

```
## Not run:
pub_covid_hosp_state_timeseries(
  states = "fl",
  dates = epirange(20200101, 20200501)
)

## End(Not run)
```

---

pub_delphi *Delphi's ILINet forecasts*

Description

API docs: https://cmu-delphi.github.io/delphi-epidata/api/delphi.html

Usage

```
pub_delphi(system, epiweek, fetch_args = fetch_args_list())
```

Arguments

system	character. System name to fetch.
epiweek	timeset. Epiweeks to fetch.
fetch_args	fetch_args. Additional arguments to pass to fetch().

Value

list

Examples

```
## Not run:
pub_delphi(system = "ec", epiweek = 201501)

## End(Not run)
```

pub_dengue_nowcast *Delphi's PAHO Dengue nowcast*

Description

API docs: https://cmu-delphi.github.io/delphi-epidata/api/dengue_nowcast.html

Usage

```
pub_dengue_nowcast(locations, epiweeks, fetch_args = fetch_args_list())
```

Arguments

locations	character. Locations to fetch.
epiweeks	timeset. Epiweeks to fetch.
fetch_args	fetch_args. Additional arguments to pass to fetch().

Value

tibble::tibble

Examples

```
## Not run:
pub_dengue_nowcast(
  locations = "pr",
  epiweeks = epirange(201401, 202301)
)

## End(Not run)
```

pub_ecdc_ili *ECDC ILI data*

Description

API docs: https://cmu-delphi.github.io/delphi-epidata/api/ecdc_ili.html.

Obtain information on influenza-like-illness from the European Centre for Disease Prevention and Control.

Usage

```
pub_ecdc_ili(
  regions,
  epiweeks,
  ...,
  issues = NULL,
  lag = NULL,
  fetch_args = fetch_args_list()
)
```

Arguments

regions	character. Regions to fetch.
epiweeks	timeset. Epiweeks to fetch.
...	not used for values, forces later arguments to bind by name
issues	timeset. Optionally, the issues to fetch. If not set, the most recent issue is returned. Mutually exclusive with lag.
lag	integer. Optionally, the lag of the issues to fetch. If not set, the most recent issue is returned. Mutually exclusive with issues.
fetch_args	fetch_args. Additional arguments to pass to fetch().

Details

The list of location argument can be found in https://github.com/cmu-delphi/delphi-epidata/blob/main/labels/ecdc_regions.txt.

Value

tibble::tibble

Examples

```
## Not run:
pub_ecdc_ili(regions = "austria", epiweeks = epirange(201901, 202001))

## End(Not run)
```

---

pub_flusurv *FluSurv hospitalization data*

Description

API docs: https://cmu-delphi.github.io/delphi-epidata/api/flusurv.html.

Obtain information on flu hospitalization rates from the Center of Disease Control.

See also https://gis.cdc.gov/GRASP/Fluview/FluHospRates.html.

Usage

```
pub_flusurv(
  locations,
  epiweeks,
  ...,
  issues = NULL,
  lag = NULL,
  fetch_args = fetch_args_list()
)
```

Arguments

locations	character. Character vector indicating location.
epiweeks	timeset. Epiweeks to fetch.
...	not used for values, forces later arguments to bind by name
issues	timeset. Optionally, the issues to fetch. If not set, the most recent issue is returned. Mutually exclusive with lag.
lag	integer. Optionally, the lag of the issues to fetch. If not set, the most recent issue is returned. Mutually exclusive with issues.
fetch_args	fetch_args. Additional arguments to pass to fetch().

Details

The list of location argument can be found in https://github.com/cmu-delphi/delphi-epidata/blob/main/labels/flusurv_locations.txt.

Value

tibble::tibble

Examples

```
## Not run:
pub_flusurv(locations = "CA", epiweeks = epirange(201701, 201801))

## End(Not run)
```

---

pub_fluview *FluView ILINet data*

Description

API docs: https://cmu-delphi.github.io/delphi-epidata/api/fluview.html. For

Obtains information on outpatient inluenza-like-illness (ILI) from U.S. Outpatient Influenza-like Illness Surveillance Network (ILINet).

more information on ILINet, see https://gis.cdc.gov/grasp/fluview/fluportaldashboard.html.

Usage

```
pub_fluview(
  regions,
  epiweeks,
  ...,
  issues = NULL,
  lag = NULL,
  auth = NULL,
  fetch_args = fetch_args_list()
)
```

Arguments

regions	character. Locations to fetch. Can be any string IDs in national, HHS region, census division, most states and territories, and so on. Full list link below.
epiweeks	timeset. Epiweeks to fetch in the form epirange(startweek, endweek), where startweek and endweek are of the form YYYYWW (string or numeric).
...	not used for values, forces later arguments to bind by name
issues	timeset. Optionally, the issues to fetch. If not set, the most recent issue is returned. Mutually exclusive with lag.
lag	integer. Optionally, the lag of the issues to fetch. If not set, the most recent issue is returned. Mutually exclusive with issues.
auth	string. Optionally, restricted access key (not the same as API key).
fetch_args	fetch_args. Additional arguments to pass to fetch().

Details

The full list of location inputs can be accessed at https://github.com/cmu-delphi/delphi-epidata/blob/main/src/acquisition/fluview/fluview_locations.py.

Value

tibble::tibble

Examples

```
## Not run:
pub_fluview(regions = "nat", epiweeks = epirange(201201, 202005))

## End(Not run)
```

pub_fluview_clinical *FluView virological data from clinical labs*

Description

API docs: https://cmu-delphi.github.io/delphi-epidata/api/fluview_clinical.html

Usage

```
pub_fluview_clinical(
  regions,
  epiweeks,
  ...,
  issues = NULL,
  lag = NULL,
  fetch_args = fetch_args_list()
)
```

Arguments

regions	character. Regions to fetch.
epiweeks	timeset. Epiweeks to fetch in the form epirange(startweek,endweek), where startweek and endweek are of the form YYYYWW (string or numeric).
...	not used for values, forces later arguments to bind by name
issues	timeset. Optionally, the issues to fetch. If not set, the most recent issue is returned. Mutually exclusive with lag.
lag	integer. Optionally, the lag of the issues to fetch. If not set, the most recent issue is returned. Mutually exclusive with issues.
fetch_args	fetch_args. Additional arguments to pass to fetch().

Value

tibble::tibble

Examples

```
## Not run:
pub_fluview_clinical(regions = "nat", epiweeks = epirange(201601, 201701))

## End(Not run)
```

pub_fluview_meta *FluView metadata*

Description

API docs: https://cmu-delphi.github.io/delphi-epidata/api/fluview_meta.html Returns information about the fluview endpoint.

Usage

```
pub_fluview_meta(fetch_args = fetch_args_list())
```

Arguments

fetch_args	fetch_args. Additional arguments to pass to fetch().

Value

tibble::tibble

Examples

```
## Not run:
pub_fluview_meta()

## End(Not run)
```

pub_gft *Google Flu Trends data*

Description

API docs: https://cmu-delphi.github.io/delphi-epidata/api/gft.html

Obtains estimates of inluenza activity based on volume of certain search queries from Google.

Usage

```
pub_gft(locations, epiweeks, fetch_args = fetch_args_list())
```

Arguments

locations	character. Locations to fetch.
epiweeks	timeset Epiweeks to fetch.
fetch_args	fetch_args. Additional arguments to pass to fetch().

Details

Google has discontinued Flu Trends and this is now a static endpoint. Possibile input for locations can be found in https://github.com/cmu-delphi/delphi-epidata/blob/main/labels/regions.txt, https://github.com/cmu-delphi/delphi-epidata/blob/main/labels/states.txt, and https://github.com/cmu-delphi/delphi-epidata/blob/main/labels/cities.txt.

Value

tibble::tibble

Examples

```
## Not run:
pub_gft(locations = "hhs1", epiweeks = epirange(201201, 202001))

## End(Not run)
```

---

pub_kcdc_ili *KCDC ILI data*

Description

API docs: https://cmu-delphi.github.io/delphi-epidata/api/kcdc_ili.html

Usage

```
pub_kcdc_ili(
  regions,
  epiweeks,
  ...,
  issues = NULL,
  lag = NULL,
  fetch_args = fetch_args_list()
)
```

Arguments

regions	character. Regions to fetch.
epiweeks	timeset. Epiweeks to fetch.
...	not used for values, forces later arguments to bind by name
issues	timeset. Optionally, the issues to fetch. If not set, the most recent issue is returned. Mutually exclusive with lag.
lag	integer. Optionally, the lag of the issues to fetch. If not set, the most recent issue is returned. Mutually exclusive with issues.
fetch_args	fetch_args. Additional arguments to pass to fetch().

Value

tibble::tibble

Examples

```
## Not run:
pub_kcdc_ili(regions = "ROK", epiweeks = 200436)

## End(Not run)
```

---

pub_meta *API metadata*

Description

API docs: https://cmu-delphi.github.io/delphi-epidata/api/meta.html

Usage

```
pub_meta(fetch_args = fetch_args_list())
```

Arguments

fetch_args	fetch_args. Additional arguments to pass to fetch().

Value

list

---

pub_nidss_dengue *NIDSS dengue data*

Description

API docs: https://cmu-delphi.github.io/delphi-epidata/api/nidss_dengue.html

Obtains counts of confirmed dengue cases in Taiwan from Taiwan National Infectious Disease Statistical System.

Usage

```
pub_nidss_dengue(locations, epiweeks, fetch_args = fetch_args_list())
```

Arguments

locations	character. Locations to fetch.
epiweeks	timeset. Epiweeks to fetch.
fetch_args	fetch_args. Additional arguments to pass to fetch().

Details

Possible location inputs can be found in https://github.com/cmu-delphi/delphi-epidata/blob/main/labels/nidss_regions.txt and https://github.com/cmu-delphi/delphi-epidata/blob/main/labels/nidss_locations.txt.

Value

tibble::tibble

Examples

```
## Not run:
pub_nidss_dengue(locations = "taipei", epiweeks = epirange(201201, 201301))

## End(Not run)
```

---

pub_nidss_flu *NIDSS flu data*

Description

API docs: https://cmu-delphi.github.io/delphi-epidata/api/nidss_flu.html

Obtains information on outpatient inluenza-like-illness from Taiwan National Infectious Disease Statistical System.

Usage

```
pub_nidss_flu(
  regions,
  epiweeks,
  ...,
  issues = NULL,
  lag = NULL,
  fetch_args = fetch_args_list()
)
```

Arguments

regions	character. Regions to fetch.
epiweeks	timeset. Epiweeks to fetch.
...	not used for values, forces later arguments to bind by name
issues	timeset. Optionally, the issues to fetch. If not set, the most recent issue is returned. Mutually exclusive with lag.
lag	integer. Optionally, the lag of the issues to fetch. If not set, the most recent issue is returned. Mutually exclusive with issues.
fetch_args	fetch_args. Additional arguments to pass to fetch().

Value

tibble::tibble

Examples

```
## Not run:
pub_nidss_flu(regions = "taipei", epiweeks = epirange(201501, 201601))

## End(Not run)
```

---

pub_nowcast *Delphi's ILI nowcast*

Description

API docs: https://cmu-delphi.github.io/delphi-epidata/api/nowcast.html.

Obtains information on outpatient inluenza-like-illness (ILI) from Delphi's

Usage

```
pub_nowcast(locations, epiweeks, fetch_args = fetch_args_list())
```

Arguments

locations	character. Locations to fetch.
epiweeks	timeset. Epiweeks to fetch.
fetch_args	fetch_args. Additional arguments to pass to fetch().

Details

The full list of location inputs can be accessed at https://github.com/cmu-delphi/delphi-epidata/blob/main/src/acquisition/fluview/fluview_locations.py.

Value

tibble::tibble

Examples

```
## Not run:
pub_nowcast(locations = "ca", epiweeks = epirange(201201, 201301))

## End(Not run)
```

pub_paho_dengue *PAHO Dengue data*

Description

API docs: https://cmu-delphi.github.io/delphi-epidata/api/paho_dengue.html

Usage

```
pub_paho_dengue(
  regions,
  epiweeks,
  ...,
  issues = NULL,
  lag = NULL,
  fetch_args = fetch_args_list()
)
```

Arguments

regions	character. Regions to fetch.
epiweeks	timeset. Epiweeks to fetch.
...	not used for values, forces later arguments to bind by name
issues	timeset. Optionally, the issues to fetch. If not set, the most recent issue is returned. Mutually exclusive with lag.
lag	integer. Optionally, the lag of the issues to fetch. If not set, the most recent issue is returned. Mutually exclusive with issues.
fetch_args	fetch_args. Additional arguments to pass to fetch().

Value

tibble::tibble

Examples

```
## Not run:
pub_paho_dengue(regions = "ca", epiweeks = epirange(201401, 201501))

## End(Not run)
```

pub_wiki *Wikipedia access data*

Description

API docs: https://cmu-delphi.github.io/delphi-epidata/api/wiki.html Number of page visits for selected English, Influenza-related wikipedia articles.

- Source: Wikimedia
- Temporal Resolution: Hourly, daily, and weekly from 2007-12-09 (2007w50)
- Spatial Resolution: N/A
- Other resolution: By article (54)
- Open access

Usage

```
pub_wiki(
  articles,
  ...,
  dates = NULL,
  epiweeks = NULL,
  hours = NULL,
  language = "en",
  fetch_args = fetch_args_list()
)
```

Arguments

articles	character. Articles to fetch.
...	not used for values, forces later arguments to bind by name
dates	timeset. Dates to fetch. Mutually exclusive with epiweeks.
epiweeks	timeset. Epiweeks to fetch. Mutually exclusive with dates.
hours	integer. Optionally, the hours to fetch.
language	string. Language to fetch.
fetch_args	fetch_args. Additional arguments to pass to fetch().

Value

tibble::tibble

Examples

```
## Not run:
pub_wiki(articles = "avian_influenza", epiweeks = epirange(201501, 201601))

## End(Not run)
```

pvt_cdc *CDC page hits*

Description

API docs: https://cmu-delphi.github.io/delphi-epidata/api/cdc.html

Usage

```
pvt_cdc(auth, locations, epiweeks, fetch_args = fetch_args_list())
```

Arguments

auth	string. Restricted access key (not the same as API key).
locations	character. Locations to fetch.
epiweeks	timeset. Epiweeks to fetch.
fetch_args	fetch_args. Additional arguments to pass to fetch(). See fetch_args_list() for details.

Value

tibble::tibble

Examples

```
## Not run:
pvt_cdc(
  auth = Sys.getenv("SECRET_API_AUTH_CDC"),
  locations = "fl,ca",
  epirange(201501, 201601)
)

## End(Not run)
```

pvt_dengue_sensors *Dengue digital surveillance sensors in PAHO member countries*

Description

API docs: https://cmu-delphi.github.io/delphi-epidata/api/dengue_sensors.html

Usage

```
pvt_dengue_sensors(
  auth,
  names,
  locations,
  epiweeks,
  fetch_args = fetch_args_list()
)
```

Arguments

auth	string. Restricted access key (not the same as API key).
names	character. Names to fetch.
locations	character. Locations to fetch.
epiweeks	timeset. Epiweeks to fetch.
fetch_args	fetch_args. Additional arguments to pass to fetch().

Value

tibble::tibble

Examples

```
## Not run:
pvt_dengue_sensors(
  auth = Sys.getenv("SECRET_API_AUTH_SENSORS"),
  names = "ght",
  locations = "ag",
  epiweeks = epirange(201501, 202001)
)

## End(Not run)
```

---

pvt_ght *Google Health Trends data*

Description

API docs: https://cmu-delphi.github.io/delphi-epidata/api/ght.html

Estimate of influenza activity based on volume of certain search queries. ...

Usage

```
pvt_ght(auth, locations, epiweeks, query, fetch_args = fetch_args_list())
```

Arguments

auth	string. Restricted access key (not the same as API key).
locations	character. Locations to fetch.
epiweeks	timeset. Epiweeks to fetch.
query	string. The query to be fetched.
fetch_args	fetch_args. Additional arguments to pass to fetch().

Value

tibble::tibble

Examples

```
## Not run:
pvt_ght(
  auth = Sys.getenv("SECRET_API_AUTH_GHT"),
  locations = "ma",
  epiweeks = epirange(199301, 202304),
  query = "how to get over the flu"
)

## End(Not run)
```

---

pvt_meta_norostat *NoroSTAT metadata*

Description

API docs: https://cmu-delphi.github.io/delphi-epidata/api/meta_norostat.html

Usage

```
pvt_meta_norostat(auth, fetch_args = fetch_args_list())
```

Arguments

auth	string. Restricted access key (not the same as API key).
fetch_args	fetch_args. Additional arguments to pass to fetch().

Value

list

Examples

```
## Not run:
pvt_meta_norostat(auth = Sys.getenv("SECRET_API_AUTH_NOROSTAT"))

## End(Not run)
```

pvt_norostat *NoroSTAT data (point data, no min/max)*

Description

API docs: https://cmu-delphi.github.io/delphi-epidata/api/norostat.html

This is the documentation of the API for accessing the NoroSTAT (norostat) endpoint of the Delphi's epidemiological data.

Usage

```
pvt_norostat(auth, locations, epiweeks, fetch_args = fetch_args_list())
```

Arguments

auth	string. Your authentication key.
locations	character. Locations to fetch.
epiweeks	timeset. Epiweeks to fetch.
fetch_args	fetch_args. Additional arguments to pass to fetch().

Value

tibble::tibble

Examples

```
## Not run:
pvt_norostat(
  auth = Sys.getenv("SECRET_API_AUTH_NOROSTAT"),
  locations = "1",
  epiweeks = 201233
)

## End(Not run)
```

pvt_quidel *Quidel COVID-19 and influenza testing data*

Description

API docs: https://cmu-delphi.github.io/delphi-epidata/api/quidel.html

Data provided by Quidel Corp., which contains flu lab test results.

Usage

```
pvt_quidel(auth, locations, epiweeks, fetch_args = fetch_args_list())
```

Arguments

auth	string. Restricted access key (not the same as API key).
locations	character. Locations to fetch.
epiweeks	timeset. Epiweeks to fetch.
fetch_args	fetch_args. Additional arguments to pass to fetch().

Value

tibble::tibble

Examples

```
## Not run:
pvt_quidel(
  auth = Sys.getenv("SECRET_API_AUTH_QUIDEL"),
  epiweeks = epirange(201201, 202001),
  locations = "hhs1"
)

## End(Not run)
```

---

pvt_sensors *Digital surveillance sensors*

Description

API docs: https://cmu-delphi.github.io/delphi-epidata/api/sensors.html

This is the documentation of the API for accessing the Digital Surveillance Sensors (sensors) endpoint of the Delphi's epidemiological Note: this repository was built to support modeling and forecasting efforts surrounding seasonal influenza (and dengue). In the current COVID-19 pandemic, syndromic surveillance data, like ILI data (influenza-like illness) through FluView, will likely prove very useful. However, we urge caution to users examining the digital surveillance sensors, like ILI Nearby, Google Flu Trends, etc., during the COVID-19 pandemic, because these were designed to track ILI as driven by seasonal influenza, and were NOT designed to track ILI during the COVID-19 pandemic.

Usage

```
pvt_sensors(auth, names, locations, epiweeks, fetch_args = fetch_args_list())
```

Arguments

auth	string. Restricted access key (not the same as API key).
names	character. Sensor names to fetch.
locations	character. Locations to fetch.
epiweeks	timeset. Epiweeks to fetch.
fetch_args	fetch_args. Additional arguments to pass to fetch().

Value

tibble::tibble

Examples

```
## Not run:
pvt_sensors(
  auth = Sys.getenv("SECRET_API_AUTH_SENSORS"),
  names = "sar3",
  locations = "nat",
  epiweeks = epirange(201501, 202001)
)

## End(Not run)
```

---

pvt_twitter *HealthTweets data*

Description

API docs: https://cmu-delphi.github.io/delphi-epidata/api/twitter.html

This is the API documentation for accessing the Twitter Stream (twitter) endpoint of Delphi's epidemiological data. Sourced from Healthtweets

Usage

```
pvt_twitter(
  auth,
  locations,
  ...,
  dates = NULL,
  epiweeks = NULL,
  fetch_args = fetch_args_list()
)
```

Arguments

auth	string. Restricted access key (not the same as API key).
locations	character. Locations to fetch.
...	not used for values, forces later arguments to bind by name
dates	timeset. Dates to fetch. Mutually exclusive with epiweeks.
epiweeks	timeset. Epiweeks to fetch. Mutually exclusive with dates.
fetch_args	fetch_args. Additional arguments to pass to fetch().

Value

tibble::tibble

Examples

```
## Not run:
pvt_twitter(
  auth = Sys.getenv("SECRET_API_AUTH_TWITTER"),
  locations = "CA",
  epiweeks = epirange(201501, 202001)
)

## End(Not run)
```

`set_cache` *Create or renew a cache for this session*

Description

By default, epidatr re-requests data from the API on every call of `fetch`. In case you find yourself repeatedly calling the same data, you can enable the cache using either this function for a given session, or environmental variables for a persistent cache. The typical recommended workflow for using the cache is to set the environmental variables `EPIDATR_USE_CACHE=TRUE` and `EPIDATR_CACHE_DIRECTORY="/your/directory/here"`in your `.Renviron`, for example by calling `usethis::edit_r_environ()`. See the parameters below for some more configurables if you're so inclined.

`set_cache` (re)defines the cache to use in a particular R session. This does not clear existing data at any previous location, but instead creates a handle to the new cache using cachem that seamlessly handles caching for you. Say your cache is normally stored in some default directory, but for the current session you want to save your results in `~/my/temporary/savedirectory`, then you would call `set_cache(dir = "~/my/temporary/savedirectory")`. Or if you know the data from 2 days ago is wrong, you could call `set_cache(days = 1)` to clear older data whenever the cache is referenced. In both cases, these changes would only last for a single session (though the deleted data would be gone permanently!).

An important feature of the caching in this package is that only calls which specify either `issues` before a certain date, or `as_of` before a certain date will actually cache. For example the call

```
covidcast(
  source = "jhu-csse",
  signals = "confirmed_7dav_incidence_prop",
  geo_type = "state",
  time_type = "day",
  geo_values = "ca,fl",
  time_values = epirange(20200601, 20230801)
)
```

won't cache, since it is possible for the cache to be invalidated by new releases with no warning. On the other hand, the call

```
covidcast(
  source = "jhu-csse",
  signals = "confirmed_7dav_incidence_prop",
  geo_type = "state",
  time_type = "day",
  geo_values = "ca,fl",
  time_values = epirange(20200601, 20230801),
  as_of = "2023-08-01"
)
```

will cache, since normal new versions of data can't invalidate it (since they would be `as_of` a later date). It is still possible that Delphi may patch such data, but the frequency is on the order of months rather than days. We are working on creating a public channel to communicate such updates. While specifying `issues` will usually cache, a call with `issues="*"` won't cache, since its subject to cache invalidation by normal versioning.

On the backend, the cache uses cachem, with filenames generated using an md5 encoding of the call url. Each file corresponds to a unique epidata-API call.

Usage

```
set_cache(
  cache_dir = NULL,
  days = NULL,
  max_size = NULL,
  logfile = NULL,
  confirm = TRUE
)
```

Arguments

`cache_dir`	the directory in which the cache is stored. By default, this is `tools::R_user_dir()` if on R 4.0+, but must be specified for earlier versions of R. The path can be either relative or absolute. The environmental variable is `EPIDATR_CACHE_DIR`.
`days`	the maximum length of time in days to keep any particular cached call. By default this is `1`. The environmental variable is `EPIDATR_CACHE_MAX_AGE_DAYS`.
`max_size`	the size of the entire cache, in MB, at which to start pruning entries. By default this is `1024`, or 1GB. The environmental variable is `EPIDATR_CACHE_MAX_SIZE_MB`.
`logfile`	where cachem's log of transactions is stored, relative to the cache directory. By default, it is `"logfile.txt"`. The environmental variable is `EPIDATR_CACHE_LOGFILE`.
`confirm`	whether to confirm directory creation. default is `TRUE`; should only be set in non-interactive scripts

Value

`NULL` no return value, all effects are stored in the package environment

See Also

clear_cache to delete the old cache while making a new one, disable_cache to disable without deleting, and cache_info

Examples

```
set_cache(
  cache_dir = tempdir(),
  days = 14,
  max_size = 512,
  logfile = "logs.txt"
)
```

---

timeset *Timeset*

Description

Many API calls accept timesets to specify the time ranges of data being requested. Timesets can be specified with `epirange()`, as `Date` objects, or with wildcards.

Timesets are not special R types; the term simply describes any value that would be accepted by epidatr to specify the time value of an epidata query. The allowed values are:

- Dates: `Date` instances, integer-like values, or integer-like strings that take the form YYYYMMDD.
- Epiweeks: Integer-like values or integer-like strings that take the form YYYYWW.
- EpiRanges: A range returned by `epirange()`, or a list of multiple ranges.
- Wildcard: The string `"*"`, which request all available time values.

Please refer to the specific endpoint documentation for guidance on using dates vs weeks. Most endpoints support only one or the other. Some (less commonly used) endpoints may not accept the `"*"` wildcard, but this can be simulated with a large `epirange()`.

Index