Package 'openmeteo'

March 10, 2023

Title Retrieve Weather Data from the Open-Meteo API

Version 0.1.1

Description A client for the Open-Meteo API that retrieves Open-Meteo weather data in a tidy format. No API key is required. The API specification is located at <https://open-meteo.com/en/docs>.

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.2.3

Imports httr, tibble, tidyr, tibblify, dplyr, yaml, lutz, testthat (>= 3.0.0)

Suggests httptest

Config/testthat/edition 3

NeedsCompilation no

Author Tom Pisel [aut, cre, cph]

Maintainer Tom Pisel <mail@tompisel.com>

Repository CRAN

Date/Publication 2023-03-10 07:50:06 UTC

R topics documented:

geocode *Retrieve data from the Open-Meteo geocoding API*

Description

Call the Open-Meteo Geocoding API to retrieve co-ordinates and other information for a given place name. The closest n matching records can be requested.

Geocoding API documentation is available at https://open-meteo.com/en/docs/geocoding-api.

Usage

```
geocode(location_name, n_results = 1, language = "en", silent = TRUE)
```

Arguments

location_name	Required. The location name to search for via fuzzy matching.
n_results	The number of matching locations provided in response, sorted by relevance (default 1, up to 100).
language	Desired response language for place names (lower-cased two-letter string, i.e. "en" for *London*,"fr" for *Londres*).
silent	If FALSE, the top match will be printed to the console, to aid in confirming the match is correct when used within other functions.

Value

Details for each matching location (latitude, longitude, elevation, population, timezone, and administrative areas)

Examples

```
# obtain co-ordinates of Sydney
gc <- geocode("Sydney")
sydney_coords <- c(gc$latitude, gc$longitude)
sydney_coords

# elevation of Kathmandu
geocode("kathmandu")$elevation

# 10 places named 'Paris'
geocode("paris", 10)
```

openmeteo *openmeteo: retrieve weather data from the Open-Meteo API*

Description

openmeteo provides functions for accessing the Open-Meteo weather API, enabling the desired weather data or forecasts to be retrieved in a tidy data format. An API key is *not* required to access the Open-Meteo API.

Open-Meteo provides several API endpoints. This package currently enables access to the *Weather Forecast API*, the *Historical Weather API*, and the *Geocoding API* through the following functions:

- weather_forecast() - retrieve weather forecasts for a location
- weather_history() - retrieve historical weather observations for a location
- weather_now() - simple function to return current weather for a location
- geocode() - return the co-ordinates and other data for a location name
- weather_variables() - retrieve a shortlist of valid forecast or historical weather variables provided

Please review the API documentation at https://open-meteo.com/ for details regarding the data available, its types, units, and other caveats and considerations.

weather_forecast *Retrieve weather forecasts from the Open-Meteo API*

Description

weather_forecast() calls the Open-Meteo Weather Forecast API to obtain meteorological forecasts for a given location.

Usage

```
weather_forecast(
  location,
  start = NULL,
  end = NULL,
  hourly = NULL,
  daily = NULL,
  response_units = NULL,
  model = NULL,
  timezone = "auto"
)
```

Arguments

`location`	Required. The location for which data will be retrieved. Supplied as either a `c(latitude,longitude)` WGS84 coordinate pair or a place name string (with co-ordinates obtained via `geocode()`).
`start, end`	Start and end dates in ISO 8601 (e.g. "2020-12-31"). If no dates are supplied, data for the next 7 days will be provided by default.
`hourly, daily`	At least one required. A weather variable accepted by the API, or list thereof. See details below.
`response_units`	Supply to convert temperature, windspeed, or precipitation units. This defaults to: `list(temperature_unit = "celsius", windspeed_unit = "kmh", precipitation_unit = "mm"`
`model`	Supply to specify a model for forecasted values (defaults to autoselection of best model).
`timezone`	specify timezone for time data as a string, i.e. "australia/sydney" (defaults to "auto", the timezone local to the specified `location`).

Details

You will need to specify at least one weather variable, such as temperature, that you want forecasted data for. These variables are sampled or aggregated at *hourly* or *daily* intervals, and can be supplied as a list to request multiple variables over the same time period.

Example *Hourly* forecast variables include:

Variable	Description
`temperature_2m`	Air temperature at 2 meters above ground
`precipitation`	Sum of rain, showers, and snow over the preceding hour
`windspeed_10m`	Wind speed at 10 meters above ground
`cloudcover`	Total cloud cover as an area fraction
`pressure_msl`	Atmospheric air pressure at mean sea level

Example *Daily* forecast variables include:

Variable	Description
`temperature_2m_max`	Maximum daily air temperature at 2 meters above ground
`precipitation_sum`	Sum of rain, showers, and snow over the preceding day
`windspeed_10m_max`	Maximum daily wind speed at 10 meters above ground

Full documentation for the forecast API is available at: `https://open-meteo.com/en/docs`

You can also call `weather_variables()` to retrieve an (incomplete) shortlist of valid hourly and daily weather variables.

Value

Requested weather forecast data for the given location and time, as a tidy tibble.

Examples

```
# obtain temperature forecasts for the South Pole's next 7 days
weather_forecast(c(-90, 0), hourly = "temperature_2m")

# obtain temperature and precipitation forecasts for NYC in Imperial units
weather_forecast("nyc",
  hourly = c("temperature_2m", "precipitation"),
  response_units = list(
    temperature_unit = "fahrenheit",
    precipitation_unit = "inch"
  )
)

# will it rain tomorrow in Jakarta?
tomorrow <- Sys.Date() + 1
weather_forecast("jakarta", tomorrow, tomorrow, daily = "precipitation_sum")
```

`weather_history` *Retrieve historical weather data from the Open-Meteo API*

Description

`weather_history()` calls the Open-Meteo Historical Weather API to obtain weather data for a given location and historical time period.

Usage

```
weather_history(
  location,
  start,
  end,
  hourly = NULL,
  daily = NULL,
  response_units = NULL,
  model = NULL,
  timezone = "auto"
)
```

Arguments

`location`	Required. The location for which data will be retrieved. Supplied as either a `c(latitude,longitude)` WGS84 coordinate pair or a place name string (with co-ordinates obtained via `geocode()`).
`start, end`	Required. Start and end dates in ISO 8601 (e.g. "2020-12-31").
`hourly, daily`	At least one required. A weather variable accepted by the API, or list thereof. See details below.

`response_units`	Supply to convert temperature, windspeed, or precipitation units. This defaults to: `list(temperature_unit = "celsius", windspeed_unit = "kmh", precipitation_unit = "mm"`
`model`	Supply to specify a model for re-analysis.
`timezone`	specify timezone for time data as a string, i.e. "australia/sydney" (defaults to "auto", the timezone local to the specified `location`).

Details

You will need to specify at least one weather variable, such as temperature, that you want historical data for. These variables have been sampled or aggregated at *hourly* or *daily* intervals, and can be supplied as a list to request multiple variables over the same time period.

Example *Hourly* historical weather variables include:

Variable	Description
`temperature_2m`	Air temperature at 2 meters above ground
`precipitation`	Sum of rain, showers, and snow over the preceding hour
`windspeed_10m`	Wind speed at 10 meters above ground
`cloudcover`	Total cloud cover as an area fraction
`pressure_msl`	Atmospheric air pressure at mean sea level

Example *Daily* historical weather variables include:

Variable	Description
`temperature_2m_max`	Maximum daily air temperature at 2 meters above ground
`precipitation_sum`	Sum of rain, showers, and snow over the preceding day
`windspeed_10m_max`	Maximum daily wind speed at 10 meters above ground

Full documentation for the historical weather API is available at: https://open-meteo.com/en/docs/historical-weather-api

You can also call `weather_variables()` to retrieve an (incomplete) shortlist of valid hourly and daily weather variables.

Value

Specified weather forecast data for the given location and time

Examples

```
# obtain cloud cover history for London over 2020
weather_history("London",
  start = "2020-01-01",
  end = "2021-12-31",
  hourly = "cloudcover"
)
```

`weather_now` *Retrieve Current Weather from the Open-Meteo API*

Description

`weather_now()` calls the Open-Meteo weather API for the most recently recorded weather conditions a given location. Location is provided either as string or `c(latitude,longitude)`.

Usage

```
weather_now(location, response_units = NULL, timezone = "auto")
```

Arguments

`location`	Required. The location for which data will be retrieved. Supplied as either a `c(latitude,longitude)` WGS84 coordinate pair or a place name string (with co-ordinates obtained via `geocode()`).
`response_units`	Supply to convert temperature, windspeed, or precipitation units. This defaults to: `list(temperature_unit = "celsius", windspeed_unit = "kmh", precipitation_unit = "mm"`
`timezone`	specify timezone for time data as a string, i.e. "australia/sydney" (defaults to "auto", the timezone local to the specified `location`).

Value

Current weather conditions: temperature, windspeed, wind direction and weathercode.

Examples

```
# current weather in Montreal
weather_now("Montreal")

# current weather at the North Pole in Imperial units
weather_now(c(90, 0),
  response_units = list(
    temperature_unit = "fahrenheit",
    windspeed_unit = "mph"
  )
)
```

`weather_variables` *Retrieve valid hourly and daily weather variables*

Description

`weather_variables()` retrieves an incomplete list of *hourly* and *daily* variables accepted by `weather_forecast()` and `weather_history()`, such as temperature or precipitation.

Refer to the following documentation for the forecast and history API endpoints for detailed descriptions, units, and caveats:

Forecast API https://open-meteo.com/en/docs

Historical API https://open-meteo.com/en/docs/historical-weather-api

Usage

```
weather_variables()
```

Value

A list of valid hourly and daily weather variables

Examples

```
weather_variables()
```

Index