Package ‘RClimacell’

October 12, 2022

Title R Wrapper for the 'Climacell' API
Version 0.1.4
Description 'Climacell' is a weather platform that provides hyper-local forecasts and weather data. This package enables the user to query the core layers of the timeline interface of the 'Climacell' v4 API <https://www.climacell.co/weather-api/>. This package requires a valid API key. See vignettes for instructions on use.
License MIT + file LICENSE
URL https://nikdata.github.io/RClimacell/
BugReports https://github.com/nikdata/RClimacell/issues
Encoding UTF-8
LazyData true
RoxygenNote 7.1.1
Imports dplyr (>= 1.0.0), magrittr (>= 2.0.1), stringr (>= 1.4.0), tibble (>= 3.0.6), httr (>= 1.4.2), lubridate (>= 1.7.9.2), rlang (>= 0.4.10), tidyr (>= 1.1.2), assertthat (>= 0.2.1), tidyselect
Suggests knitr, rmarkdown
VignetteBuilder knitr
Depends R (>= 2.10)
NeedsCompilation no
Author Nikhil Agarwal [aut, cre] (<https://orcid.org/0000-0003-1470-7472>)
Maintainer Nikhil Agarwal <gitnik@niks.me>
Repository CRAN
Date/Publication 2021-03-23 19:00:03 UTC

R topics documented:

climacell_celestial .......................................................... 2
climacell_core ............................................................. 3
climacell_celestial

Sunrise, Sunset, and Moon Phase Readings from Climacell

Description
This function will make a call to the Climacell API and retrieve sunrise, sunset times and moon phase variables.

Usage
climacell_celestial(
  api_key,
  lat,
  long,
  timestep = "1d",
  start_time = NULL,
  end_time = NULL
)

Arguments
api_key character string representing the private API key. Provided by user or loaded automatically from environment variable (environment variable must be called “CLIMACELL_API”).
lat a numeric value (or a string that can be coerced to numeric) representing the latitude of the location.
long a numeric value (or a string that can be coerced to numeric) representing the longitude of the location.
timestep a ‘step’ value for the time. For the climacell_celestial function, the only acceptable value (per the limitations of the Climacell API) is ’1d’.
start_time the start time of the query. This input must be a character string that can be parsed into a data/time or a date/time value. If the input does not contain a timezone, the value will be assumed to be in UTC. It is recommended that the lubridate::now() function or Sys.time() be used to define the start_time. For this function, the start_time cannot be less than 6 hours from the current time.
climacell_core

end_time

the end time of the query. This input must be a character string that can be parsed into a data/time or a date/time value. If the input does not contain a timezone, the value will be assumed to be in UTC. For this function, the end_time cannot be greater than 15 days from the current date/time.

Details

climacell_celestial returns a tibble that consists of sunrise/sunset times along with the moon phase (code & description).

Value

a tibble

Examples

## Not run:
climacell_celestial(
  api_key = Sys.getenv('CLIMACELL_API'),
  lat = 0,
  long = 0,
  timestep = '1d',
  start_time = lubridate::now(),
  end_time = lubridate::now() + lubridate::days(5))

## End(Not run)

climacell_core

Climacell Core Layer Data

Description

climacell_core returns a tibble that contains all of the Core Layer data from the Climacell version 4 API using the Timelines interface. The intent of this function is to retrieve all of the Core Layer data in a single API call. This is especially handy when using the free API as it limits the usage of the API based on hourly rate and daily usage.

Usage

climacell_core(
    api_key, lat, long,
    timestep, start_time = NULL,
    end_time = NULL
)
Leveraging climacell_precip

**Arguments**

- **api_key**: character string representing the private API key. Provided by user or loaded automatically from environment variable (environment variable must be called "CLIMACELL_API").
- **lat**: a numeric value (or a string that can be coerced to numeric) representing the latitude of the location.
- **long**: a numeric value (or a string that can be coerced to numeric) representing the longitude of the location.
- **timestep**: a 'step' value for the time. Choose one of the following valid values: c(’1d’, ’1h’, ’30m’, ’15m’, ’5m’, ’1m’, ’current’).
- **start_time**: the start time of the query. This input must be a character string that can be parsed into a data/time or a date/time value. If the input does not contain a timezone, the value will be assumed to be in UTC. It is recommended that the lubridate::now() function or Sys.time() be used to define the start_time. For this function, the start_time cannot be less than 6 hours from the current time.
- **end_time**: the end time of the query. This input must be a character string that can be parsed into a data/time or a date/time value. If the input does not contain a timezone, the value will be assumed to be in UTC. OPTIONAL if timestep is 'current' or if the user desires to get the maximum results possible (depends on the timestep chosen).

**Value**

a tibble

**Examples**

```r
## Not run:
climacell_core(
  api_key = Sys.getenv('CLIMACELL_API'),
  lat = 0,
  long = 0,
  timestep = '1d',
  start_time = lubridate::now(),
  end_time = lubridate::now + lubridate::days(5))
## End(Not run)
```

---

**climacell_precip**

Precipitation Readings from Climacell

**Description**

This function will make a call to the Climacell API and retrieve precipitation related (including cloud cover & pressure) values.
climacell_precip

Usage

climacell_precip(
  api_key,
  lat,
  long,
  timestep,
  start_time = NULL,
  end_time = NULL
)

Arguments

api_key character string representing the private API key. Provided by user or loaded automatically from environment variable (environment variable must be called "CLIMACELL_API").

lat a numeric value (or a string that can be coerced to numeric) representing the latitude of the location.

long a numeric value (or a string that can be coerced to numeric) representing the longitude of the location.

timestep a 'step' value for the time. Choose one of the following valid values: c('1d', '1h', '30m', '15m', '5m', '1m', 'current').

start_time the start time of the query. This input must be a character string that can be parsed into a date/time or a date/time value. If the input does not contain a timezone, the value will be assumed to be in UTC. It is recommended that the lubridate::now() function or Sys.time() be used to define the start_time. For this function, the start_time cannot be less than 6 hours from the current time.

end_time the end time of the query. This input must be a character string that can be parsed into a date/time or a date/time value. If the input does not contain a timezone, the value will be assumed to be in UTC. OPTIONAL if timestep is 'current' or if the user desires to get the maximum results possible (depends on the timestep chosen).

Details

climacell_precip returns a tibble that consists of precipitation related variables (returned values are in metric units) using the Climacell API. These variables consist of precipitation intensity, precipitation probability, precipitation description, visibility, surface & sea level pressure, cloud cover & ceiling, and a weather description.

Value
	a tibble

Examples

## Not run:
climacell_temperature

Temperature Readings from Climacell

Description
This function will make a call to the Climacell API and retrieve temperature related variables.

Usage
climacell_temperature(
  api_key,
  lat,
  long,
  timestep,
  start_time = NULL,
  end_time = NULL
)

Arguments

- **api_key**: character string representing the private API key. Provided by user or loaded automatically from environment variable (environment variable must be called "CLIMACELL_API").
- **lat**: a numeric value (or a string that can be coerced to numeric) representing the latitude of the location.
- **long**: a numeric value (or a string that can be coerced to numeric) representing the longitude of the location.
- **timestep**: a 'step' value for the time. Choose one of the following valid values: c('1d', '1h', '30m', '15m', '5m', '1m', 'current').
- **start_time**: the start time of the query. This input must be a character string that can be parsed into a data/time or a date/time value. If the input does not contain a timezone, the value will be assumed to be in UTC. It is recommended that the lubridate::now() function or Sys.time() be used to define the start_time. For this function, the start_time cannot be less than 6 hours from the current time.
climacell_wind

end_time

the end time of the query. This input must be a character string that can be parsed into a data/time or a date/time value. If the input does not contain a timezone, the value will be assumed to be in UTC. OPTIONAL if timestep is ‘current’ or if the user desires to get the maximum results possible (depends on the timestep chosen).

Details

climacell_temperature returns a tibble that consists of temperature related variables (returned values are in metric units) using the Climacell API. These variables consist of temperature, a "feels like" temperature, dewpoint, and humidity.

Value

a tibble

Examples

## Not run:
climacell_temperature(
    api_key = Sys.getenv('CLIMACELL_API'),
    lat = 0,
    long = 0,
    timestep = 'current')

## End(Not run)

-------------------

climacell_wind      Wind Readings from Climacell

Description

This function will make a call to the Climacell API and retrieve wind related variables.

Usage

climacell_wind(
    api_key,
    lat,
    long,
    timestep,
    start_time = NULL,
    end_time = NULL
)
Arguments

api_key  character string representing the private API key. Provided by user or loaded automatically from environment variable (environment variable must be called "CLIMACELL_API").

lat  a numeric value (or a string that can be coerced to numeric) representing the latitude of the location.

long  a numeric value (or a string that can be coerced to numeric) representing the longitude of the location.

timestep  a 'step' value for the time. Choose one of the following valid values: c('1d', '1h', '30m', '15m', '5m', '1m', 'current').

start_time  the start time of the query. This input must be a character string that can be parsed into a data/time or a date/time value. If the input does not contain a timezone, the value will be assumed to be in UTC. It is recommended that the lubridate::now() function or Sys.time() be used to define the start_time. For this function, the start_time cannot be less than 6 hours from the current time.

end_time  the end time of the query. This input must be a character string that can be parsed into a data/time or a date/time value. If the input does not contain a timezone, the value will be assumed to be in UTC. OPTIONAL if timestep is 'current' or if the user desires to get the maximum results possible (depends on the timestep chosen).

Details

climacell_wind returns a tibble that consists of wind related variables (returned values are in metric units) using the Climacell API. These variables consist of wind speed, wind gust, and wind direction.

Value

a tibble

Examples

## Not run:
climacell_wind(
    api_key = Sys.getenv('CLIMACELL_API'),
    lat = 0,
    long = 0,
    timestep = 'current')

## End(Not run)
dict_moonphase

**Moonphase Dictionary**

**Description**

this is a helper function that returns the moon phase tibble containing the moon phase codes (which are returned by Climacell API) and their appropriate description.

**Usage**

```
dict_moonphase()
```

**Value**

a tibble

---

dict_preciptype

**Precipitation Type Dictionary**

**Description**

this is a helper function that returns the precipitation type tibble containing the precipitation type codes (which are returned by Climacell API) and their appropriate description.

**Usage**

```
dict_preciptype()
```

**Value**

a tibble

---

dict_weathercode

**Weather Dictionary**

**Description**

this is a helper function that returns the weather code tibble containing the weather codes (which are returned by Climacell API) and their appropriate description.

**Usage**

```
dict_weathercode()
```

**Value**

a tibble
Index

climacell_celestial, 2
climacell_core, 3
climacell_precip, 4
climacell_temperature, 6
climacell_wind, 7
dict_moonphase, 9
dict_preciptype, 9
dict_weathercode, 9