Package 'geckor'

July 13, 2021

Title R Client for the 'CoinGecko' API

Version 0.1.1

Description Collect the current and historical cryptocurrency market data using the public 'CoinGecko' API (<https://www.coingecko.com/en/api>).

License MIT + file LICENSE

URL https://github.com/next-game-solutions/geckor

BugReports https://github.com/next-game-solutions/geckor/issues

Depends R (>= 4.0)

Encoding UTF-8

RoxygenNote 7.1.1

Suggests covr, markdown, readr, rmarkdown, spelling, testthat (>= 3.0.0)

Config/testthat/edition 3

Language en-US

Imports dplyr, jsonlite, httr, knitr, lifecycle, magrittr, rlang, tibble, tidyr, tidyselect

VignetteBuilder knitr

RdMacros lifecycle

NeedsCompilation no

Author Sergey Mastitsky [aut, cre], Next Game Solutions [cph]

Maintainer Sergey Mastitsky <sergey@nextgamesolutions.com>

Repository CRAN

Date/Publication 2021-07-13 14:40:05 UTC

R topics documented:

coin_history . . . 2
coin_history_ohlc . . . 3
coin_history_range . . . 5
coin_history_snapshot . . . 6
coin_tickers . . . 7
current_market . . . 9
current_price . . . 11
exchange_rate . . . 12
ping . . . 13
supported_coins . . . 14
supported_currencies . . . 15
supported_exchanges . . . 15
trending_coins . . . 16

Index **18**

`coin_history` *Get historical market data*

Description

Retrieves market data for a coin for the last *n* days

Usage

```
coin_history(
  coin_id,
  vs_currency = "usd",
  days,
  interval = NULL,
  max_attempts = 3
)
```

Arguments

`coin_id`	(character): ID of the coin of interest. An up-to-date list of supported coins and their IDs can be retrieved with the `supported_coins()` function.
`vs_currency`	(character): name of the reference currency to express the price in. An up-to-date list of supported reference currencies (both fiat and cryptocurrencies) can be obtained with the `supported_currencies()` function. If an unsupported `vs_currency` is requested, the call will fail with the respective error message.
`days`	(numeric or `"max"`): number of days to look back. If `days = "max"`, the entire available history for `coin_id` will be retrieved. Depending on the value of `days`, the time interval used to present the data will differ - see "Details".
`interval`	(character or `NULL`): time interval used to present the data. The only currently supported value is `daily`. Defaults to `NULL`.

`max_attempts` (double, positive): specifies the maximum number of attempts to call the CoinGecko API (e.g., if the first call fails for some reason). Additional attempts are implemented with an exponential backoff. Defaults to 3.

Details

If `days = 1` and `interval = NULL`, the data will be returned for every few minutes (typically 3-8 minutes). If `days` is between 2 and 90 (inclusive) and `interval = NULL`, an (approximately) hourly time step will be used. Daily data are used for `days` above 90. If `interval = "daily"`, daily data will be used irrespective of the value of `days`.

This function is based on the public CoinGecko API, which has a limit of 50 calls per minute. Please keep this limit in mind when developing your applications.

Value

A tibble with the following columns:

- `timestamp` (POSIXct);
- `coin_id` (character): same as the argument `coin_id`;
- `vs_currency` (character): same as the argument `vs_currency`;
- `price` (double): coin price, as of `timestamp`;
- `total_volume` (double): a 24 hours rolling-window trading volume, as of `timestamp`;
- `market_cap` (double): market capitalisation, as of `timestamp`.

Examples

```
r <- coin_history(coin_id = "bitcoin", vs_currency = "usd", days = 30)
print(r)
```

`coin_history_ohlc` *Historical OHLC price data*

Description

Retrieves open-high-low-close price data for the last *n* days

Usage

```
coin_history_ohlc(coin_id, vs_currency = "usd", days, max_attempts = 3)
```

Arguments

coin_id	(character): ID of the coin of interest. An up-to-date list of supported coins and their IDs can be retrieved with the supported_coins() function.
vs_currency	(character): name of the reference currency to express the price in. An up-to-date list of supported reference currencies (both fiat and cryptocurrencies) can be obtained with the supported_currencies() function. If an unsupported vs_currency is requested, the call will fail with the respective error message.
days	(numeric or "max"): number of days to look back. The only acceptable values are 1, 7, 14, 30, 90, 180, 365 and "max". Attempts to assign any other values will fail with the corresponding error message. If days = "max", the entire available history will be retrieved. Depending on the value of days, the time interval used to present the data will differ - see "Details".
max_attempts	(double, positive): specifies the maximum number of attempts to call the CoinGecko API (e.g., if the first call fails for some reason). Additional attempts are implemented with an exponential backoff. Defaults to 3.

Details

Granularity of the retrieved data (i.e. candle's body) depends on the value of days as follows:

- 1 day: 30 minutes;
- 7 - 30 days: 4 hours;
- above 30: 4 days.

This function is based on the public CoinGecko API, which has a limit of 50 calls per minute. Please keep this limit in mind when developing your applications.

Value

A tibble with the following columns:

- timestamp (POSIXct);
- coin_id (character): same as the argument coin_id;
- vs_currency (character): same as the argument vs_currency;
- price_open (double): coin price in the beginning of a time interval;
- price_high (double): highest coin price observed within a time interval;
- price_low (double): lowest coin price observed within a time interval;
- price_close (double): coin price in the end of a time interval.

Examples

```
r <- coin_history_ohlc(
  coin_id = "cardano",
  vs_currency = "usd",
  days = 7
)
print(r)
```

`coin_history_range` *Get historical market data*

Description

Retrieves coin-specific market data for a range of historical dates

Usage

```
coin_history_range(coin_id, vs_currency = "usd", from, to, max_attempts = 3)
```

Arguments

`coin_id`	(character): ID of the coin of interest. An up-to-date list of supported coins and their IDs can be retrieved with the `supported_coins()` function.
`vs_currency`	(character): name of the reference currency to express the price in. An up-to-date list of supported reference currencies (both fiat and cryptocurrencies) can be obtained with the `supported_currencies()` function. If an unsupported `vs_currency` is requested, the call will fail with the respective error message.
`from`	(POSIXct): timestamp of the beginning of the period of interest (YYYY-MM-DD HH:MM:SS).
`to`	(POSIXct): timestamp of the end of the period of interest (YYYY-MM-DD HH:MM:SS).
`max_attempts`	(double, positive): specifies the maximum number of attempts to call the CoinGecko API (e.g., if the first call fails for some reason). Additional attempts are implemented with an exponential backoff. Defaults to 3.

Details

This function returns hourly data for periods of up to 90 days and daily data for periods that are longer than 90 days.

This function is based on the public CoinGecko API, which has a limit of 50 calls per minute. Please keep this limit in mind when developing your applications.

Value

A tibble with the following columns:

- `timestamp` (POSIXct);
- `vs_currency` (character): same as the argument `vs_currency`;
- `price` (double): coin price as of `timestamp`;
- `total_trading_vol` (double): a 24 hours rolling-window trading volume, as of `timestampt`;
- `market_cap` (double): market capitalisation, as of `timestamp`.

Examples

```
r <- coin_history_range(
  coin_id = "cardano",
  vs_currency = "usd",
  from = as.POSIXct("2021-01-01 13:00:00", tz = "UTC"),
  to = as.POSIXct("2021-01-01 18:00:00", tz = "UTC")
)
print(r)
```

---

coin_history_snapshot *History snapshot for a coin*

Description

Coin-specific market data for a given historical date

Usage

```
coin_history_snapshot(coin_id, date, vs_currencies = "usd", max_attempts = 3)
```

Arguments

coin_id	(character): ID of the coin of interest. An up-to-date list of supported coins and their IDs can be retrieved with the supported_coins() function.
date	(Date): date of interest. If no data exist for the requested date, nothing (NULL) will be returned.
vs_currencies	(character): a vector with names of the reference currencies to express the price in, e.g. c("usd","eur","btc"). An up-to-date list of supported vs_currencies (both fiat and cryptocurrencies) can be obtained with the supported_currencies() function. If vs_currencies contains at least one unsupported currency, the call will fail with the respective error message.
max_attempts	(double, positive): specifies the maximum number of attempts to call the CoinGecko API (e.g., if the first call fails for some reason). Additional attempts are implemented with an exponential backoff. Defaults to 3.

Details

This function is based on the public CoinGecko API, which has a limit of 50 calls per minute. Please keep this limit in mind when developing your applications.

Value

If the requested data exist, this function will return a tibble with with as many rows as the length of `vs_currencies` and the following columns:

- `coin_id` (character): same as the argument `coin_id`;
- `symbol` (character): symbol of the coin;
- `name` (character): common name of the coin;
- `date` (Date): same as the argument `date`;
- `vs_currency` (character): reference currency, in which the `price` is expressed (ordered alphabetically);
- `price` (double): price of the coin;
- `market_cap` (double): market capitalisation of the coin;
- `total_volume` (double): total trading volume recorded on that `date`.

Examples

```
r <- coin_history_snapshot(
  coin_id = "cardano",
  date = as.Date("2021-05-01"),
  vs_currencies = c("usd", "eth")
)
print(r)
```

---

coin_tickers *Exchange tickers*

Description

Retrieves the most recent exchange tickers for a coin

Usage

```
coin_tickers(coin_id, exchange_id, max_attempts = 3)
```

Arguments

`coin_id`	(character): ID of the coin of interest. An up-to-date list of supported coins and their IDs can be retrieved with the `supported_coins()` function.
`exchange_id`	(character): ID of the exchange to retrieve the data from. An up-to-date list of supported exchanges and their IDs can be obtained with the `supported_exchanges()` function.
`max_attempts`	(double, positive): specifies the maximum number of attempts to call the CoinGecko API (e.g., if the first call fails for some reason). Additional attempts are implemented with an exponential backoff. Defaults to 3.

Details

This function is based on the public CoinGecko API, which has a limit of 50 calls per minute. Please keep this limit in mind when developing your applications.

Value

A tibble with the following columns:

- `exchange_id` (character): same as the argument `exchange_id`;
- `exchange_name` (character): common name of the exchange;
- `coin_id` (character): same as the argument `coin_id`;
- `name` (character): common name of the coin;
- `base` (character): symbol of the base currency in a trading pair;
- `target` (character): symbol of the target currency in a trading pair;
- `trust_score` (character): trust score of the trading pair (`"green"`, `"yellow"`, `"red"`; see this and this article on the CoinGecko website for details);
- `last_price` (double): last price reported by this exchange for this trading pair;
- `last_fetch_at` (POSIXct, UTC time zone): timestamp of when `last_price` was recorded;
- `last_traded_at` (POSIXct, UTC time zone): timestamp of the most recent trade;
- `bid_ask_spread_percentage` (double): percentage difference between the ask price (lowest price a seller is willing to sell for) and the bid price (highest price a buyer is willing to buy for; see Investopedia for details);
- `trading_volume_24h` (double): trading volume (in `target` currency) recorded in the last 24 h (as of `last_traded_at`);
- `last_price_btc` (double): last price reported by this exchange for this coin, expressed in Bitcoin;
- `last_price_eth` (double): last price reported by this exchange for this coin, expressed in Ethereum;
- `last_price_usd` (double): last price reported by this exchange for this coin, expressed in US dollar;
- `trading_volume_24h_btc` (double): 24 hours trading volume expressed in Bitcoin (as of `last_traded_at`);
- `trading_volume_24h_eth` (double): 24 hours trading volume expressed in Ethereum;
- `trading_volume_24h_usd` (double): 24 hours trading volume expressed in US dollars;
- `cost_to_move_up_2percent_usd` and `cost_to_move_down_2percent_usd` (double): 2% market depth (see this article on the CoinGecko website for details);
- `is_anomaly` (logical): an indicator of whether the ticker's price is an outlier (see "Outlier detection" in Methodology on the GoinGecko website);
- `is_stale` (logical): an indicator of whether the ticker's price has not been updated for a while (see "Outlier detection" in Methodology on the GoinGecko website);
- `trade_url` (character): URL to this trading pair's page.

Examples

```
r <- coin_tickers(coin_id = "cardano", exchange_id = "binance")
print(r)
```

---

current_market *Current market data*

Description

Retrieves current market data for a set of coins

Usage

```
current_market(coin_ids, vs_currency = "usd", max_attempts = 3)
```

Arguments

coin_ids	(character): a vector with IDs of the coins of interest. An up-to-date list of supported coins and their IDs can be obtained with the supported_coins() function.
vs_currency	(character): name of the reference currency to express the price in. An up-to-date list of supported reference currencies (both fiat and cryptocurrencies) can be obtained with the supported_currencies() function. If an unsupported vs_currency is requested, the call will fail with the respective error message.
max_attempts	(double, positive): specifies the maximum number of attempts to call the CoinGecko API (e.g., if the first call fails for some reason). Additional attempts are implemented with an exponential backoff. Defaults to 3.

Details

If no data can be retrieved (e.g., because of a misspecified query parameter), nothing (NULL) will be returned.

This function is based on the public CoinGecko API, which has a limit of 50 calls per minute. Please keep this limit in mind when developing your applications.

Value

If the requested data exist, this function will return a tibble with as many rows as the length of coin_ids and the following columns:

- coin_id (character): coin IDs, ordered by market_cap (see below);
- symbol (character): symbol of the coin;
- name (character): common name of the coin;
- last_updated_at (POSIXct, UTC time zone): timestamp of the last update;

- current_price (double): current price (as of last_updated_at) expressed in vs_currency;
- market_cap (double): current market capitalisation;
- market_cap_rank (integer): current rank of the coin in terms of its market capitalisation;
- fully_diluted_valuation (double): fully diluted valuation of the coin's project;
- total_volume (double): total trading volume in the last 24 hours;
- high_24h (double): max price recorded in the last 24 hours;
- low_24h (double): min price recorded in the last 24 hours;
- price_change_24h (double): price change as compared to 24 hours ago;
- price_change_percentage_24h (double): percentage change of the price as compared to 24 hours ago;
- circulating_supply (double): coin supply currently in circulation;
- total_supply (double): total supply that can potentially be circulated;
- max_supply (double): max possible supply;
- ath (double): all-time high price;
- ath_change_percentage (double): percentage change of the all-time high price compared to the current price;
- ath_date (POSIXct, UTC time zone): timestamp of when the all-time high price was recorded;
- atl (double): all-time low price;
- atl_change_percentage (double): percentage change of the all-time low price compared to the current price;
- atl_date (POSIXct, UTC timezone): timestamp of when the all-time low price was recorded;
- price_change_percentage_<time>_in_currency: columns containing the percentage change in price as compared to various points in the past (in particular, 1 hour, 24 hours, 7 days, 14 days, 30 days, 200 days, and 1 year).

Examples

```
r <- current_market(
  coin_ids = c("bitcoin", "ethereum", "cardano"),
  vs_currency = "usd"
)
print(r)
```

current_price *Current coin prices*

Description

Retrieves current prices of supported coins in any supported reference currencies

Usage

```
current_price(
  coin_ids,
  vs_currencies = c("usd"),
  include_market_cap = TRUE,
  include_24h_vol = TRUE,
  include_24h_change = TRUE,
  max_attempts = 3
)
```

Arguments

coin_ids	(character): a vector with IDs of the coins of interest. An up-to-date list of supported coins and their IDs can be obtained with the supported_coins() function.
vs_currencies	(character): a vector with names of the reference currencies to express the price in, e.g. c("usd","eur","btc"). An up-to-date list of supported vs_currencies (both fiat and cryptocurrencies) can be obtained with the supported_currencies() function. If vs_currencies contains at least one unsupported currency, the call will fail with the respective error message.
include_market_cap	(boolean, defaults to TRUE): whether to return the market capitalisation information.
include_24h_vol	(boolean, defaults to TRUE): whether to return the trading volume for the last 24 hours.
include_24h_change	(boolean, defaults to TRUE): whether to return the price percentage change compared to 24 hours ago.
max_attempts	(double, positive): specifies the maximum number of attempts to call the CoinGecko API (e.g., if the first call fails for some reason). Additional attempts are implemented with an exponential backoff. Defaults to 3.

Details

This function is based on the public CoinGecko API, which has a limit of 50 calls per minute. Please keep this limit in mind when developing your applications.

If no data can be retrieved (e.g. because of a misspecified query parameter), nothing (NULL) will be returned.

Value

A tibble, which by the default will contain the following columns (use arguments `include_market_cap`, `include_24h_vol` and `include_24h_change` to control the inclusion of the corresponding columns):

- `coin_id` (character): coin IDs, ordered alphabetically;
- `vs_currency` (character): reference currency, in which the price of `coin_id` is expressed;
- `last_updated_at` (POSIXct, UTC time zone): timestamp of the last price update;
- `market_cap` (double): current market capitalisation;
- `vol_24h` (double): trading volume in the last 24 hours;
- `price_percent_change_24h` (double): percentage change of the price as compared to 24 hours ago.

Examples

```
r <- current_price(
  coin_ids = c("aave", "tron", "bitcoin"),
  vs_currencies = c("usd", "eur", "gbp")
)
print(r)
```

---

exchange_rate *Exchange rate*

Description

Retrieves the current exchange rate for a crypto- of fiat currency in Bitcoin

Usage

```
exchange_rate(currency = NULL, max_attempts = 3)
```

Arguments

`currency`	(character or `NULL`): a vector with abbreviated names of the currencies of interest. An up-to-date list of supported currencies (both fiat and cryptocurrencies) can be retrieved with the `supported_currencies()` function. If an unsupported `currency` is requested, the call will fail with the respective error message. If `currency = NULL` (default), the function will return exchange rates for all supported currencies.
`max_attempts`	(double, positive): specifies the maximum number of attempts to call the CoinGecko API (e.g., if the first call fails for some reason). Additional attempts are implemented with an exponential backoff. Defaults to 3.

Details

This function is based on the public CoinGecko API, which has a limit of 50 calls per minute. Please keep this limit in mind when developing your applications.

Value

A tibble with the following columns:

- `timestamp` (POSIXct): date and time of the API request;
- `currency` (character): abbreviated name of the currency;
- `name` (character): common name of the currency;
- `price_in_btc` (double): price in Bitcoin;
- `type` (character): type of the currency (`"fiat"` or `"crypto"`).

Examples

```
# get exchange rates for all supported currencies
r1 <- exchange_rate()
print(r1)

# get exchange rates for a set of currencies:
r2 <- exchange_rate(currency = c("usd", "eur", "gbp"))
print(r2)
```

`ping` *API health check*

Description

Pings the CoinGecko API to check if the service is available

Usage

```
ping()
```

Details

This function has no arguments.

Value

Returns `TRUE` if the service is available and `FALSE` otherwise.

`supported_coins` *CoinGecko coins*

Description

Retrieves a list of coins currently supported by the CoinGecko API

Usage

```
supported_coins(max_attempts = 3)
```

Arguments

`max_attempts`	(double, positive): specifies the maximum number of attempts to call the CoinGecko API (e.g., if the first call fails for some reason). Additional attempts are implemented with an exponential backoff. Defaults to 3.

Details

This function is based on the public CoinGecko API, which has a limit of 50 calls per minute. Please keep this limit in mind when developing your applications.

Value

A tibble with three columns:

- `coin_id` (character): coin IDs, ordered alphabetically;
- `symbol` (character): coin symbols;
- `name` (character): common names of the coins.

Examples

```
r <- supported_coins()
head(r, 10)
```

supported_currencies *CoinGecko currencies*

Description

Retrieves a list of base currencies currently supported by the CoinGecko API

Usage

```
supported_currencies(max_attempts = 3)
```

Arguments

max_attempts	(double, positive): specifies the maximum number of attempts to call the CoinGecko API (e.g., if the first call fails for some reason). Additional attempts are implemented with an exponential backoff. Defaults to 3.

Details

This function is based on the public CoinGecko API, which has a limit of 50 calls per minute. Please keep this limit in mind when developing your applications.

Value

Character vector with abbreviated names of the currencies.

Examples

```
r <- supported_currencies()
print(r)
```

supported_exchanges *CoinGecko exchanges*

Description

Retrieves a list of exchanges supported by the CoinGecko API

Usage

```
supported_exchanges(max_attempts = 3)
```

Arguments

`max_attempts`	(double, positive): specifies the maximum number of attempts to call the CoinGecko API (e.g., if the first call fails for some reason). Additional attempts are implemented with an exponential backoff. Defaults to 3.

Details

This function is based on the public CoinGecko API, which has a limit of 50 calls per minute. Please keep this limit in mind when developing your applications.

Value

A tibble with the following columns:

- `exchange_id` (character): exchange ID;
- `name` (character): common name of the exchange;
- `year_established` (integer): year when the exchange was established;
- `country` (character): country where the exchange is registered and / or has its headquarters;
- `url` (character): web address of the exchange;
- `trust_score` (integer): an indicator of how much an exchange can be trusted (ranges from 1 to 10; see Methodology on the CoinGecko website);
- `trading_volume_24h_btc` (double): trading volume in the last 24 hours, expressed in Bitcoin.

Examples

```
r <- supported_exchanges()
print(r)
```

`trending_coins` *Trending coins*

Description

Retrieves a list of top-7 coins on CoinGecko according to their search popularity

Usage

```
trending_coins(max_attempts = 3)
```

Arguments

`max_attempts`	(double, positive): specifies the maximum number of attempts to call the CoinGecko API (e.g., if the first call fails for some reason). Additional attempts are implemented with an exponential backoff. Defaults to 3.

Details

Popularity of a coin is determined from search patterns at the CoinGecko website over the last 24 hours.

This function is based on the public CoinGecko API, which has a limit of 50 calls per minute. Please keep this limit in mind when developing your applications.

Value

A tibble with the following columns:

- `timestamp` (POSIXct): date and time of the API request;
- `popularity_rank_24h` (integer): popularity rank in the last 24 hours;
- `coin_id` (character): coin ID;
- `name` (character): common name of the coin;
- `symbol` (character): symbol of the coin;
- `market_cap_rank` (integer): market capitalisation rank;
- `price_btc` (double): price expressed in Bitcoin.

Examples

```
r <- trending_coins()
print(r)
```

Index

coin_history, 2
coin_history_ohlc, 3
coin_history_range, 5
coin_history_snapshot, 6
coin_tickers, 7
current_market, 9
current_price, 11

exchange_rate, 12

ping, 13

supported_coins, 14
supported_coins(), *2*, *4–7*, *9*, *11*
supported_currencies, 15
supported_currencies(), *2*, *4–6*, *9*, *11, 12*
supported_exchanges, 15
supported_exchanges(), *7*

trending_coins, 16