Package 'riingo'

October 14, 2022

Type Package

Title An R Interface to the 'Tiingo' Stock Price API

Version 0.3.1

Maintainer Davis Vaughan <davis@rstudio.com>

Description Functionality to download stock prices, cryptocurrency data, and more from the 'Tiingo' API <https://api.tiingo.com/>.

License MIT + file LICENSE

URL https://github.com/business-science/riingo

BugReports https://github.com/business-science/riingo/issues

Depends R (>= 2.10)

Imports crayon (>= 1.3.4), glue (>= 1.3.1), httr (>= 1.3.1), jsonlite (>= 1.5), purrr (>= 0.2.4), rlang (>= 0.2.0), tibble (>= 1.4.2), vctrs (>= 0.3.0)

Suggests covr, testthat, withr

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

NeedsCompilation no

Author Davis Vaughan [aut, cre],
Matt Dancho [aut]

Repository CRAN

Date/Publication 2020-09-12 05:50:11 UTC

R topics documented:

`convert_to_local_time` *Convert the `POSIXct` columns of a data frame to the local time zone*

Description

Tiingo returns data with a UTC time zone. Often you will want to view this in your own time zone. This function converts each `POSIXct` column of the returned `tibble` to the local (or specified) time zone.

Usage

```
convert_to_local_time(.data, tz = "")
```

Arguments

`.data`	A tibble with `POSIXct` columns
`tz`	The time zone to convert to. The default is the local time zone.

is_supported_ticker *Ticker information*

Description

is_supported_ticker() can tell you if a given ticker is supported on Tiingo. supported_tickers() returns a tibble listing every available ticker.

Usage

```
is_supported_ticker(ticker, type = "tiingo")

supported_tickers(type = "tiingo")
```

Arguments

ticker	The single ticker to check for on Tiingo.
type	One of: "tiingo", "iex", or "crypto".

Examples

```

## Not run:

# VOO is supported on both Tiingo and IEX
is_supported_ticker("VOO")
is_supported_ticker("VOO", type = "iex")

# PRHSX is a mutual fund that is supported by Tiingo but not IEX
is_supported_ticker("PRHSX")
is_supported_ticker("PRHSX", type = "iex")

# BTCUSD is available
is_supported_ticker("btcusd", type = "crypto")


## End(Not run)
```

riingo_browse_usage *Browse various pages of the Tiingo site*

Description

Note that you **must** be signed into the site on the opened browser for most of these functions to work properly, otherwise you will redirected to the sign in page.

Usage

```
riingo_browse_usage()

riingo_browse_token()

riingo_browse_documentation()

riingo_browse_signup()
```

`riingo_crypto_latest` *The latest day's worth of intraday data for a given cryptocurrency*

Description

This returns only the most recent day of intraday data for the supplied ticker(s).

Usage

```
riingo_crypto_latest(
  ticker,
  resample_frequency = "1min",
  base_currency = NULL,
  exchanges = NULL,
  convert_currency = NULL,
  raw = FALSE
)
```

Arguments

`ticker` One or more cryptocurrency tickers. Specified as `"btcusd"` for bitcoin quoted in USD. A character vector.

`resample_frequency`

For Tiingo data, a character specified as one of: `"daily"`, `"weekly"`, `"monthly"` or `"annually"`.

For IEX data, a character specified at the `"min"` or `"hour"` frequencies in the form: `"1min"`, `"5min"`, or `"2hour"`.

For Crypto data, a character specified at the `"min"`, `"hour"` or `"day"` frequencies similar to IEX.

`base_currency` *Instead* of `ticker` you may pass a base currency. This selects all currencies with that base currency. For example if ‘base_currency="btc"“ tickers *btcusd*, *btcjpy*, *btceur*, etc.. will all be returned.

`exchanges` If you would like to limit the query to a subset of exchanges, pass a comma-separated list of exchanges to select. Example) `"POLONIEX, GDAX"`

`convert_currency`
: This parameter will convert the return data into another fx rate. For example if querying `BTCUSD` and convert_currency is `'cure'`, the bitcoin prices will be converted to CureCoin prices. Setting this to a value will add `fxOpen`, `fxHigh`, `fxLow`, `fxClose`, `fxVolumeNotional`, and `fxRate` accordingly. `fxRate` is the rate used to perform the currency calculation. If `exchanges` is specified, the conversion rate will be calculated using the exchanges passed.

`raw`
: If `TRUE`, the raw underlying data from multiple exchanges will be returned, rather than the clean prices. This is the data that calculates the aggregated prices and quotes.

Examples

```
## Not run:

# The latest available day of intraday data for QQQ
riingo_crypto_latest("btcusd")



## End(Not run)
```

`riingo_crypto_meta` *Get meta data about a cryptocurrency on Tiingo*

Description

Relevant returned meta data include: ticker, name, description, quote currency, and base currency.

Usage

```
riingo_crypto_meta(ticker)
```

Arguments

`ticker`
: One or more cryptocurrency tickers. Specified as `"btcusd"` for bitcoin quoted in USD. A character vector.

Examples

```
## Not run:

# Bitcoin meta
riingo_crypto_meta("btcusd")

# A trick to return ALL crypto meta data
# For some reason Descriptions are not returned here
riingo_crypto_meta("")


## End(Not run)
```

riingo_crypto_prices *Get cryptocurrency prices aggregated through Tiingo*

Description

Get cryptocurrency prices aggregated through Tiingo

Usage

```
riingo_crypto_prices(
  ticker,
  start_date = NULL,
  end_date = NULL,
  resample_frequency = "1day",
  base_currency = NULL,
  exchanges = NULL,
  convert_currency = NULL,
  raw = FALSE
)
```

Arguments

`ticker` One or more cryptocurrency tickers. Specified as `"btcusd"` for bitcoin quoted in USD. A character vector.

`start_date` The first date to download data for. A character in the form YYYY-MM-DD, or a `Date` variable. The default is to download 1 year's worth of data.

`end_date` The last date to download data for. A character in the form YYYY-MM-DD, or a `Date` variable.

`resample_frequency`

For Tiingo data, a character specified as one of: `"daily"`, `"weekly"`, `"monthly"` or `"annually"`.

For IEX data, a character specified at the `"min"` or `"hour"` frequencies in the form: `"1min"`, `"5min"`, or `"2hour"`.

For Crypto data, a character specified at the `"min"`, `"hour"` or `"day"` frequencies similar to IEX.

`base_currency` *Instead* of `ticker` you may pass a base currency. This selects all currencies with that base currency. For example if ‘base_currency="btc"“ tickers *btcusd*, *btcjpy*, *btceur*, etc.. will all be returned.

`exchanges` If you would like to limit the query to a subset of exchanges, pass a comma-separated list of exchanges to select. Example) `"POLONIEX, GDAX"`

`convert_currency`
This parameter will convert the return data into another fx rate. For example if querying `BTCUSD` and convert_currency is `'cure'`, the bitcoin prices will be converted to CureCoin prices. Setting this to a value will add `fxOpen`, `fxHigh`, `fxLow`, `fxClose`, `fxVolumeNotional`, and `fxRate` accordingly. `fxRate` is the rate used to perform the currency calculation. If `exchanges` is specified, the conversion rate will be calculated using the exchanges passed.

`raw` If `TRUE`, the raw underlying data from multiple exchanges will be returned, rather than the clean prices. This is the data that calculates the aggregated prices and quotes.

Examples

```
## Not run:

# Bitcoin prices
riingo_crypto_prices("btcusd")

# Bitcoin in USD and EUR
riingo_crypto_prices(c("btcusd", "btceur"), start_date = "2018-01-01", resample_frequency = "5min")

# Bitcoin raw data
riingo_crypto_prices("btcusd", raw = TRUE)

# Only use the POLONIEX exchange
riingo_crypto_prices("btcusd", raw = TRUE, exchanges = "POLONIEX")

# All btc___ crypotcurrency pairs
riingo_crypto_prices(base_currency = "btc")


## End(Not run)
```

`riingo_crypto_quote` *Quote and Top of Book data for a given cryptocurrency*

Description

Tiingo provides TOP (top of book) bid and ask quotes for cryptocurrencies. Note that this cannot be historically queried.

Usage

```
riingo_crypto_quote(
  ticker,
  exchanges = NULL,
  convert_currency = NULL,
  raw = FALSE
)
```

Arguments

`ticker` One or more cryptocurrency tickers. Specified as `"btcusd"` for bitcoin quoted in USD. A character vector.

`exchanges` If you would like to limit the query to a subset of exchanges, pass a comma-separated list of exchanges to select. Example) `"POLONIEX, GDAX"`

`convert_currency`
This parameter will convert the return data into another fx rate. For example if querying `BTCUSD` and convert_currency is `'cure'`, the bitcoin prices will be converted to CureCoin prices. Setting this to a value will add `fxOpen`, `fxHigh`, `fxLow`, `fxClose`, `fxVolumeNotional`, and `fxRate` accordingly. `fxRate` is the rate used to perform the currency calculation. If `exchanges` is specified, the conversion rate will be calculated using the exchanges passed.

`raw` If `TRUE`, the raw underlying data from multiple exchanges will be returned, rather than the clean prices. This is the data that calculates the aggregated prices and quotes.

Details

At the end of the day, the, `askPrice`, `bidSize`, `bidPrice`, `askSize`, and `lastSize` fields may be `NA`. This is normal.

Examples

```

## Not run:

riingo_crypto_quote("btcusd")

# The raw data can provide more insight into each individual exchange
riingo_crypto_quote("btcusd", raw = TRUE)


## End(Not run)
```

riingo_fundamentals_definitions

Fundamentals - Definitions

Description

This function can be used to check which fields are available from the fundamentals endpoint. As Tiingo adds more indicators, the output of this function may change to reflect the addition of new indicators.

Usage

```
riingo_fundamentals_definitions()
```

Value

A data frame with columns describing the various data codes that will be returned by `riingo_fundamentals_statements()`.

See Also

Other fundamentals: `riingo_fundamentals_meta()`, `riingo_fundamentals_metrics()`, `riingo_fundamentals_statem`

Examples

```
## Not run:
riingo_fundamentals_definitions()

## End(Not run)
```

riingo_fundamentals_meta

Fundamentals - Meta

Description

This function can be used to get fundamentals meta data for individual tickers, which includes information about when that ticker was last updated with new fundamentals data, along with things such as the sector and industry that that company belongs to.

Usage

```
riingo_fundamentals_meta(ticker)
```

Arguments

`ticker`	One or more tickers to download meta information for.

Value

A data frame with one row per ticker containing fundamentals meta data.

See Also

Other fundamentals: riingo_fundamentals_definitions(), riingo_fundamentals_metrics(), riingo_fundamentals_statements()

Examples

```
## Not run:
riingo_fundamentals_meta(c("AAPL", "MSFT"))

## End(Not run)
```

riingo_fundamentals_metrics

Fundamentals - Metrics

Description

This function collects daily financial metrics for the specificed tickers. These might include market capitalization, P/E ratios, and more.

Usage

```
riingo_fundamentals_metrics(ticker, start_date = NULL, end_date = NULL)
```

Arguments

ticker	One or more tickers to download financial metrics for.
start_date	The first date to download data for. A character in the form YYYY-MM-DD, or a Date variable. The default is to download 1 year's worth of data.
end_date	The last date to download data for. A character in the form YYYY-MM-DD, or a Date variable.

Value

A data frame containing the financial metrics for the requested tickers.

See Also

Other fundamentals: riingo_fundamentals_definitions(), riingo_fundamentals_meta(), riingo_fundamentals_statements()

Examples

```
## Not run:
riingo_fundamentals_metrics(c("AAPL", "MSFT"), start_date = "2020-01-01")

## End(Not run)
```

riingo_fundamentals_statements

Fundamentals - Statements

Description

This function collects financial statement information for individual tickers. Cash flow, income statement, and balance sheet information are returned at the quarterly level, with an additional annual report attached if available.

In the returned data frame, `quarter == 0` represents an annual report for the corresponding `year`.

If `as_reported = FALSE`, an `overview` list column is also returned that contains a combination of metrics from various statements.

The returned data frame is in a very compact form containing *list columns*. Each list column is made up of more data frames, where each data frame represents that particular financial statement for that quarter. The easiest way to get at the underlying data is to unnest the list columns individually using `tidyr::unnest()`.

Usage

```
riingo_fundamentals_statements(
  ticker,
  start_date = NULL,
  end_date = NULL,
  as_reported = FALSE
)
```

Arguments

`ticker`	One or more tickers to download financial statements for.
`start_date`	The first date to download data for. A character in the form YYYY-MM-DD, or a `Date` variable. The default is to download 1 year's worth of data.
`end_date`	The last date to download data for. A character in the form YYYY-MM-DD, or a `Date` variable.
`as_reported`	A single logical. When `FALSE`, the most recent data will be returned, including any revisions for the reporting period. The dates will correspond to the fiscal end of the quarter or year (note that this can vary from company to company). When `TRUE`, the endpoint will return the data as it was reported on the release date. Similarly, the date will correspond to the date the filings were posted on the SEC website.

Value

A data frame containing the financial statement information for the requested tickers.

See Also

Other fundamentals: riingo_fundamentals_definitions(), riingo_fundamentals_meta(), riingo_fundamentals_metrics()

Examples

```
## Not run:
riingo_fundamentals_statements(c("AAPL", "MSFT"))

riingo_fundamentals_statements(c("AAPL", "MSFT"), as_reported = TRUE)

## End(Not run)
```

riingo_fx_prices *Forex - Prices*

Description

This function collects forex prices for specified tickers. It can return daily, hourly, and minutely data, however, the amount of returned data becomes more limited with a finer resolution.

Usage

```
riingo_fx_prices(
  ticker,
  start_date = NULL,
  end_date = NULL,
  resample_frequency = "1day"
)
```

Arguments

ticker
: One or more fx tickers to download financial metrics for, such as "audusd" or "eurusd".

start_date
: The first date to download data for. A character in the form YYYY-MM-DD, or a Date variable. The default is to download 1 year's worth of data.

end_date
: The last date to download data for. A character in the form YYYY-MM-DD, or a Date variable.

resample_frequency
: A single character specified at the "day", "min" or "hour" frequencies in the form: "1day", "1min", "5min", or "2hour".

Value

A data frame containing the fx prices for the requested tickers.

Examples

```
## Not run:
start <- Sys.Date() - 10
riingo_fx_prices(c("audusd", "eurusd"), start_date = start)

## End(Not run)
```

`riingo_fx_quote` *Quote and Top of Book data for a given forex ticker*

Description

This function queries the top of book data for the provided forex tickers.

Usage

```
riingo_fx_quote(ticker)
```

Arguments

`ticker` One or more tickers to download data for from Tiingo. Can be a stock, mutual fund, or ETF. A character vector.

Value

A data frame containing 1 row per valid ticker with the top of book data.

Examples

```
## Not run:
riingo_fx_quote(c("audusd", "usdjpy"))

## End(Not run)
```

riingo_iex_latest *The latest day's worth of intraday data for a given ticker*

Description

This returns only the most recent day of intraday data for the supplied ticker(s).

Usage

```
riingo_iex_latest(ticker, resample_frequency = "1min")
```

Arguments

`ticker` One or more tickers to download data for from Tiingo. Can be a stock, mutual fund, or ETF. A character vector.

`resample_frequency`

For Tiingo data, a character specified as one of: `"daily"`, `"weekly"`, `"monthly"` or `"annually"`.

For IEX data, a character specified at the `"min"` or `"hour"` frequencies in the form: `"1min"`, `"5min"`, or `"2hour"`.

For Crypto data, a character specified at the `"min"`, `"hour"` or `"day"` frequencies similar to IEX.

Examples

```
## Not run:

# The latest available day of intraday data for QQQ
riingo_iex_latest("QQQ")

riingo_iex_latest("QQQ", "1hour")


## End(Not run)
```

riingo_iex_prices *Get stock or ETF prices from IEX through Tiingo*

Description

The Tiingo API provides a way to access data from IEX, The Investors Exchange. This data is supplied at a much lower (intraday!) frequency than the data from Tiingo's native API.

Usage

```
riingo_iex_prices(
  ticker,
  start_date = NULL,
  end_date = NULL,
  resample_frequency = "5min",
  after_hours = FALSE,
  force_fill = FALSE
)
```

Arguments

`ticker` One or more tickers to download data for from Tiingo. Can be a stock, mutual fund, or ETF. A character vector.

`start_date` The first date to download data for. A character in the form YYYY-MM-DD, or a `Date` variable. The default is to download 1 year's worth of data.

`end_date` The last date to download data for. A character in the form YYYY-MM-DD, or a `Date` variable.

`resample_frequency`

For Tiingo data, a character specified as one of: `"daily"`, `"weekly"`, `"monthly"` or `"annually"`.

For IEX data, a character specified at the `"min"` or `"hour"` frequencies in the form: `"1min"`, `"5min"`, or `"2hour"`.

For Crypto data, a character specified at the `"min"`, `"hour"` or `"day"` frequencies similar to IEX.

`after_hours` A single logical. Should pre and post market data be returned if available?

`force_fill` A single logical. Some tickers do not have a trade/quote update for a given time period. If `force_fill` is set to `TRUE`, then the previous OHLC will be used to fill the current OHLC.

Details

This feed returns the most recent 2000 ticks of data at the specified frequency. For example, `"5min"` would return the 2000 most recent data points spaced 5 minutes apart. You can subset the returned range with `start_date` and `end_date`, but **you cannot request data older than today's date minus 2000 data points.**

Because the default attempts to pull 1 year's worth of data, at a 5 minute frequency, all available data will be pulled so there is no need to use `start_date` and `end_date`. Only use them if you set the frequency to hourly.

Examples

```
## Not run:

# Pulling all available minute level data for Apple
riingo_iex_prices("AAPL", resample_frequency = "1min")
```

```
# This would result in an error, as you are pulling outside the available range
# riingo_iex_prices("AAPL", "1990-01-01", "2000-01-01", resample_frequency = "5min")


## End(Not run)
```

---

riingo_iex_quote *Quote and Top of Book data for a given ticker*

Description

Tiingo is plugged into the IEX feed, and they provide last sale data along with TOP (top of book) bid and ask quotes. Note that this cannot be historically queried.

Usage

```
riingo_iex_quote(ticker)
```

Arguments

ticker	One or more tickers to download data for from Tiingo. Can be a stock, mutual fund, or ETF. A character vector.

Details

At the end of the day, the mid, askPrice, bidSize, bidPrice, askSize, and lastSize fields will be NA. This is normal.

Examples

```
## Not run:

riingo_iex_quote("QQQ")


## End(Not run)
```

`riingo_latest` *The latest day's worth of data for a given ticker*

Description

This returns only the most recent day of daily data for the supplied ticker(s).

Usage

```
riingo_latest(ticker)
```

Arguments

`ticker`	One or more tickers to download data for from Tiingo. Can be a stock, mutual fund, or ETF. A character vector.

Examples

```
## Not run:

# The latest available day of daily data for QQQ
riingo_latest("QQQ")


## End(Not run)
```

`riingo_meta` *Get meta data about a ticker available on Tiingo*

Description

Retrieve start and end dates for available ticker data, along with the name, exchange, and description of the ticker.

Usage

```
riingo_meta(ticker)
```

Arguments

`ticker`	One or more tickers to download data for from Tiingo. Can be a stock, mutual fund, or ETF. A character vector.

Examples

```
## Not run:

riingo_meta("AAPL")

riingo_meta("QQQ")


## End(Not run)
```

riingo_news *Get news articles cultivated by Tiingo*

Description

This function retrieves news articles filtered by tickers, tags, or sources. It returns them as a data frame with one row per article. The original URL to the article, its description, and a number of other features are returned.

Usage

```
riingo_news(
  ticker = NULL,
  start_date = NULL,
  end_date = NULL,
  tags = NULL,
  source = NULL,
  limit = 100,
  offset = 0
)
```

Arguments

ticker	One or more tickers to download data for from Tiingo. Can be a stock, mutual fund, or ETF. A character vector.
start_date	The first date to download data for. A character in the form YYYY-MM-DD, or a Date variable. The default is to download 1 year's worth of data.
end_date	The last date to download data for. A character in the form YYYY-MM-DD, or a Date variable.
tags	A character vector of one word tags to filter with, such as "Election" or "Australia".
source	A character vector of URLs corresponding to news sources to collect articles from (such as "bloomberg.com" or "seekingalpha.com").
limit	The maximum number of articles to be retrieved. The default is 100, the maximum is 1000.

`offset` A single integer representing the "pagination". This is generally used in combination with `limit` to retrieve more articles. For example, if `limit = 100`, you can request the first 100 articles. If `offset` is then set to `100`, you can request again to get the next 100 articles.

Details

Returns a data frame of news article descriptions, urls, sources, and more.

Examples

```
## Not run:
riingo_news(ticker = "QQQ")

# Filter by either source URL or tag
riingo_news(ticker = "QQQ", source = "bloomberg.com")
riingo_news(ticker = "QQQ", tags = "Earnings")

# A ticker is not required
riingo_news(tags = "Earnings")

## End(Not run)
```

riingo_prices *Get stock or ETF prices from the Tiingo API*

Description

The Tiingo API provides a large feed of historical data at the daily (and weekly, monthly, or annual) level.

Usage

```
riingo_prices(
  ticker,
  start_date = NULL,
  end_date = NULL,
  resample_frequency = "daily"
)
```

Arguments

`ticker` One or more tickers to download data for from Tiingo. Can be a stock, mutual fund, or ETF. A character vector.

`start_date` The first date to download data for. A character in the form YYYY-MM-DD, or a `Date` variable. The default is to download 1 year's worth of data.

`end_date` The last date to download data for. A character in the form YYYY-MM-DD, or a `Date` variable.

`resample_frequency`
: For Tiingo data, a character specified as one of: `"daily"`, `"weekly"`, `"monthly"` or `"annually"`.

  For IEX data, a character specified at the `"min"` or `"hour"` frequencies in the form: `"1min"`, `"5min"`, or `"2hour"`.

  For Crypto data, a character specified at the `"min"`, `"hour"` or `"day"` frequencies similar to IEX.

Details

Multiple downloads are done *sequentially*, meaning that downloading 5 tickers costs 5 requests against your usage limits. Sadly Tiingo does not support batch downloads at the moment.

Tiingo supplied documentation regarding the resample frequency:

- daily: Values returned as daily periods, with a holiday calendar
- weekly: Values returned as weekly data, with days ending on Friday
- monthly: Values returned as monthly data, with days ending on the last standard business day (Mon-Fri) of each month
- annually: Values returned as annual data, with days ending on the last standard business day (Mon-Fri) of each year
- Note, that if you choose a value in-between the resample period for weekly, monthly, and daily, the start date rolls back to consider the entire period. For example, if you choose to resample weekly, but your "start_date" parameter is set to Wednesday of that week, the start_date will be adjusted to Monday, so the entire week is captured. Another example is if you send a start_date mid-month, we roll back the start_date to the beginning of the month.
- Similarly, if you provide an end_date, and it's midway through the period, we roll-forward the date to capture the whole period. In the above example, if the end date is set to a wednesday with a weekly resample, the end date is rolled forward to the Friday of that week.

Examples

```
## Not run:

# Downloading 1 year's worth of prices for AAPL
riingo_prices("AAPL")

# Downloading a range of data, using 2 tickers
riingo_prices(c("AAPL", "MSFT"), "1999-01-01", "2005-01-01")

# Monthly data
riingo_prices(c("AAPL", "MSFT"), "1999-01-01", "2005-01-01", "monthly")


## End(Not run)
```

riingo_set_token *Set and get your Tiingo API token*

Description

There are two methods for setting your token, an environment variable in your `.Renviron` file, or setting an option. If both are set, the environment variable will always be used. It is encouraged to use the environment variable approach, as this will persist between R sessions. See details for how to get started.

Usage

```
riingo_set_token(token, inform = TRUE)

riingo_get_token()
```

Arguments

`token`	Tiingo API token. A character.
`inform`	A single logical. Should a message be displayed encouraging you to instead use an environment variable?

Details

To use the Tiingo API, you must create an account and set an API token. It is completely free to get started and use their free source of data.

To sign up, use `riingo_browse_signup()` and click Sign-up.

To find your API token, use `riingo_browse_token()`. Note that you must be signed in on the opened browser.

With your API token in hand, you can do one of two things:

- Set the API token using the `RIINGO_TOKEN` environment variable in an `.Renviron` file. This is recommended. The easiest way to access this file is with `usethis::edit_r_environ()`. Add a line containing `RIINGO_TOKEN = <your-token-here>`. Do not put the token in quotes, and make sure to restart R once you have set it. After that, you shouldn't have to worry about it again.
- Set the API token with `riingo_set_token()`. This is only valid for the current R session and must be done each time you open R.

Index