Package ‘yahoofinancer’

November 14, 2024

Type Package

Title Fetch Data from Yahoo Finance API

Version 0.4.0

Description Obtain historical and near real time data related to stocks, index and currencies from the Yahoo Finance API. This package is community maintained and is not officially supported by 'Yahoo'. The accuracy of data is only as correct as provided on <https://finance.yahoo.com/>.

Depends R(>= 3.4)

Imports curl, httr, jsonlite, lubridate, magrittr, purrr, R6, stringr

Suggests covr, httptest, testthat (>= 3.0.0)

Config/testthat/edition 3

License MIT + file LICENSE

Encoding UTF-8

URL https://yahoofinancer.rsquaredacademy.com/,
https://github.com/rsquaredacademy/yahoofinancer

BugReports https://github.com/rsquaredacademy/yahoofinancer/issues

RoxygenNote 7.3.2

NeedsCompilation no

Author Aravind Hebbali [aut, cre]

Maintainer Aravind Hebbali <hebbali.aravind@gmail.com>

Repository CRAN

Date/Publication 2024-11-14 16:20:02 UTC

Contents

`currency_converter` *Currency converter*

Description

Retrieve current conversion rate between two currencies as well as historical rates.

Usage

```
currency_converter(
  from = "EUR",
  to = "USD",
  start = NULL,
  end = NULL,
  period = "ytd",
  interval = "1d"
)
```

Arguments

`from`	Currency to convert from.
`to`	Currency to convert to.
`start`	Specific starting date. `String` or `date` object in `yyyy-mm-dd` format.
`end`	Specific ending date. `String` or `date` object in `yyyy-mm-dd` format.
`period`	Length of time. Defaults to `'ytd'` Valid values are: • `'1d'` • `'5d'` • `'1mo'` • `'3mo'` • `'6mo'` • `'1y'` • `'2y'` • `'5y'` • `'10y'` • `'ytd'` • `'max'`
`interval`	Time between data points. Defaults to `'1d'` Valid values are: • `'1h'`

- '1d'
- '5d'
- '1wk'
- '1mo'
- '3mo'

Value

A `data.frame`.

Examples

```
currency_converter('GBP', 'USD', '2022-07-01', '2022-07-10')
currency_converter('GBP', 'USD', period = '1mo', interval = '1d')
```

`get_currencies` *Currencies*

Description

List of currencies Yahoo Finance supports.

Usage

```
get_currencies()
```

Value

Symbol, short and long name of the currencies.

Examples

```
get_currencies()
```

`get_market_summary` *Market Summary*

Description

Summary info of relevant exchanges for specific country.

Usage

```
get_market_summary(country = "US")
```

Arguments

`country`	Name of the country.

Value

A `data.frame`.

Examples

```
get_market_summary(country = 'US')
```

`get_trending` *Trending securities*

Description

List of trending securities for specific country.

Usage

```
get_trending(country = "US", count = 10)
```

Arguments

`country`	Name of the country.
`count`	Number of securities.

Value

Securities trending in the country.

Examples

```
get_trending()
```

`Index-class` *R6 Class Representing a Ticker*

Description

Base class for getting all data related to indices from Yahoo Finance API.

Format

An R6 class object

Public fields

`index` Index for which data is retrieved

Methods

Public methods:

- `Index$new()`
- `Index$set_index()`
- `Index$get_history()`
- `Index$clone()`

Method `new()`**:** Create a new Index object

Usage:

```
Index$new(index = NA)
```

Arguments:

`index` Index

Returns: A new 'Index' object

Examples:

```
nifty_50 <- Index$new('^NSEI')
```

Method `set_index()`**:** Set a new index.

Usage:

```
Index$set_index(index)
```

Arguments:

`index` New index

Examples:

```
indice <- Index$new('^NSEI')
indice$set_index('^NDX')
```

Method `get_history()`**:** Retrieves historical data

Usage:

```
Index$get_history(period = "ytd", interval = "1d", start = NULL, end = NULL)
```

Arguments:

`period` Length of time. Defaults to `'ytd'`. Valid values are:

- `'1d'`
- `'5d'`
- `'1mo'`
- `'3mo'`
- `'6mo'`
- `'1y'`
- `'2y'`
- `'5y'`
- `'10y'`
- `'ytd'`
- `'max'`

`interval` Time between data points. Defaults to `'1d'`. Valid values are:

- `'1m'`
- `'2m'`
- `'5m'`
- `'15m'`
- `'30m'`
- `'60m'`
- `'90m'`
- `'1h'`
- `'1d'`
- `'5d'`
- `'1wk'`
- `'1mo'`
- `'3mo'`

`start` Specific starting date. `String` or `date` object in `yyyy-mm-dd` format.

`end` Specific ending date. `String` or `date` object in `yyyy-mm-dd` format.

Returns: A `data.frame`.

Examples:

```
\donttest{
nifty <- Index$new('^NSEI')
nifty$get_history(start = '2022-07-01', interval = '1d')
nifty$get_history(start = '2022-07-01', end = '2022-07-14', interval = '1d')
nifty$get_history(period = '1mo', interval = '1d')
}
```

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
Index$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Examples

```
## ------------------------------------------------
## Method `Index$new`
## ------------------------------------------------

nifty_50 <- Index$new('^NSEI')

## ------------------------------------------------
## Method `Index$set_index`
## ------------------------------------------------

indice <- Index$new('^NSEI')
indice$set_index('^NDX')

## ------------------------------------------------
## Method `Index$get_history`
## ------------------------------------------------


nifty <- Index$new('^NSEI')
nifty$get_history(start = '2022-07-01', interval = '1d')
nifty$get_history(start = '2022-07-01', end = '2022-07-14', interval = '1d')
nifty$get_history(period = '1mo', interval = '1d')
```

`Ticker-class` *R6 Class Representing a Ticker*

Description

Base class for getting all data related to ticker from Yahoo Finance API.

Format

An R6 class object

Public fields

`symbol` Symbol for which data is retrieved.

Active bindings

`valuation_measures` Retrieves valuation measures for most recent four quarters

`recommendations` Recommended symbols

`technical_insights` Technical indicators for given symbol

`currency` Currency

`exchange_name` Exchange name

`full_exchange_name` Full exchange name

`first_trade_date` First trade date

`regular_market_time` Regular market time

`timezone` Time zone

`exchange_timezone_name` Exchange timezone name

`regular_market_price` Regular market price

`fifty_two_week_high` Fifty two week high

`fifty_two_week_low` Fifty two week low

`regular_market_day_high` Regular market day high

`regular_market_day_low` Regular market day low

`regular_market_volume` Regular market volume

`previous_close` Previous close

Methods

Public methods:

- `Ticker$new()`
- `Ticker$set_symbol()`
- `Ticker$get_history()`
- `Ticker$clone()`

Method `new()`**:** Create a new Ticker object.

Usage:

```
Ticker$new(symbol = NA)
```

Arguments:

`symbol` Symbol.

Returns: A new ‘Ticker‘ object

Examples:

```
aapl <- Ticker$new('aapl')
```

Method `set_symbol()`**:** Set a new symbol.

Usage:

```
Ticker$set_symbol(symbol)
```

Arguments:

`symbol` New symbol

Examples:

```
aapl <- Ticker$new('aapl')
aapl$set_symbol('msft')
```

Method `get_history()`**:** Retrieves historical pricing data.

Usage:

```
Ticker$get_history(period = "ytd", interval = "1d", start = NULL, end = NULL)
```

Arguments:

`period` Length of time. Defaults to `'ytd'`. Valid values are:

- `'1d'`
- `'5d'`
- `'1mo'`
- `'3mo'`
- `'6mo'`
- `'1y'`
- `'2y'`
- `'5y'`
- `'10y'`
- `'ytd'`
- `'max'`

`interval` Time between data points. Defaults to `'1d'`. Valid values are:

- `'1m'`
- `'2m'`
- `'5m'`
- `'15m'`
- `'30m'`
- `'60m'`
- `'90m'`
- `'1h'`
- `'1d'`
- `'5d'`
- `'1wk'`
- `'1mo'`
- `'3mo'`

`start` Specific starting date. `String` or `date` object in `yyyy-mm-dd` format.

`end` Specific ending date. `String` or `date` object in `yyyy-mm-dd` format.

Returns: A `data.frame`.

Examples:

```
\donttest{
aapl <- Ticker$new('aapl')
aapl$get_history(start = '2022-07-01', interval = '1d')
aapl$get_history(start = '2022-07-01', end = '2022-07-14', interval = '1d')
aapl$get_history(period = '1mo', interval = '1d')
}
```

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
Ticker$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Examples

```
## ------------------------------------------------
## Method `Ticker$new`
## ------------------------------------------------

aapl <- Ticker$new('aapl')

## ------------------------------------------------
## Method `Ticker$set_symbol`
## ------------------------------------------------

aapl <- Ticker$new('aapl')
aapl$set_symbol('msft')

## ------------------------------------------------
## Method `Ticker$get_history`
## ------------------------------------------------


aapl <- Ticker$new('aapl')
aapl$get_history(start = '2022-07-01', interval = '1d')
aapl$get_history(start = '2022-07-01', end = '2022-07-14', interval = '1d')
aapl$get_history(period = '1mo', interval = '1d')
```

`validate` *Symbol validation*

Description

Validate symbols before retrieving data.

Usage

```
validate(symbol = NULL)
```

Arguments

symbol	Ticker, index or fund name.

Examples

```
validate("aapl")
validate("aapls")
```

Index