Package ‘yahoofinancer’

October 14, 2022

Type  Package
Title  Fetch Data from Yahoo Finance API
Version 0.1.0
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 httr, jsonlite, lubridate, magrittr, purrr, R6, stringr
Suggests covr, dplyr, 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  7.1.1
NeedsCompilation no
Author  Aravind Hebbali [aut, cre]
Maintainer Aravind Hebbali <hebbali.aravind@gmail.com>
Repository CRAN
Date/Publication 2022-09-13 10:10:02 UTC

R topics documented:

  currency_converter .................................................. 2
  currency_summary .................................................... 3
  get_currencies ....................................................... 4
  get_market_summary .................................................. 4
currency_converter

get_trending .......................................................... 5
Index-class ............................................................... 5
Ticker-class ................................................................. 8
validate ................................................................. 13

Index 15

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:
currency_summary

- '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")

currency_summary('GBP', 'USD')

currency_summary('GBP', 'USD', from = "USD", to = "INR")

Argument:
from: Currency to convert from.
to: Currency to convert to.

Value
A list.

Examples

currency_summary('GBP', 'USD')
get_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  

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

Description

List of trending securities for specific country.

Usage

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

Arguments

<table>
<thead>
<tr>
<th>name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>country</td>
<td>Name of the country.</td>
</tr>
<tr>
<td>count</td>
<td>Number of securities.</td>
</tr>
</tbody>
</table>

Value

Securities trending in the country.

Examples

get_trending()

Index-class

\textit{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

Active bindings

summary_detail Contains information available via the Summary tab in Yahoo Finance
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:

```r
\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

```r
## Method `Index$new`

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

## Method `Index$set_index`

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

asset_profile Information related to the company’s location, operations, and officers.
calendar_events Earnings and Revenue expectations for upcoming earnings date.
company_officers Retrieves top executives for given symbol and their total pay package.
earnings_history Data related to historical earnings (actual vs. estimate)
earnings Historical earnings data.
earnings_trend Historical trend data for earnings and revenue estimations
esg_scores Data related to environmental, social, and governance metrics
financial_data Financial key performance indicators
fund_bond_holdings Retrieves aggregated maturity and duration information for a given symbol
fund_bond_ratings Retrieves aggregated maturity and duration information
fund_equity_holdings Fund equity holdings
fund_holding_info Contains information for a funds top holdings, bond ratings, bond holdings,
equity holdings, sector weightings, and category breakdown
fund_ownership Top 10 owners of a given symbol
fund_performance Historical return data for a given symbol and its specific category

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

fund_profile Summary level information for a given symbol
fund_section_weightings Retrieves aggregated sector weightings for a given symbol
fund_top_holdings Retrieves Top 10 holdings for a given symbol
fund_holdings Holding info for the given fund
grading_history Data related to upgrades / downgrades by companies
index_trend Trend data related to given symbol’s index, specifically PE and PEG ratios
inside_holders Data related to stock holdings of a given symbol(s) insiders
insider_transactions Transactions by insiders for a given symbol(s)
institution_ownership Top 10 owners of a given symbol
key_stats KPIs for given symbol
major_holders Data showing breakdown of owners of given symbol(s), insiders, institutions, etc.
page_views Short, Mid, and Long-term trend data regarding a symbol’s page views
price Detailed pricing data for given symbol, exchange, quote type, currency, market cap, pre / post market data, etc.
quote_type Stock exchange specific data for given symbol
recommendation_trend Data related to historical recommendations (buy, hold, sell) for a given symbol
security_filings Historical SEC filings
share_purchase_activity High-level buy / sell data
summary_detail Contains information available via the Summary tab in Yahoo Finance
summary_profile Return business summary of given symbol
valuation_measures Retrieves valuation measures for most recent four quarters
option_chain Option chain data for all expiration dates for a given symbol
optionExpiration_dates Option expiration dates
option_strikes Option strikes
quote Get real-time quote information for given symbol
recommendations Recommended symbols
technical_insights Technical indicators for given symbol

Methods

Public methods:

• Ticker$new()
• Ticker$set_symbol()
• Ticker$get_balance_sheet()
• Ticker$get_cash_flow()
• Ticker$get_income_statement()
• Ticker$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:*

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

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

*Usage:*

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

*Arguments:*

- `symbol`: New symbol

*Examples:*

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

**Method** `get_balance_sheet()`: Retrieves balance sheet data for most recent four quarters or most recent four years.

*Usage:*

```
Ticker$get_balance_sheet(
  frequency = c("annual", "quarter"),
  clean_names = TRUE
)
```

*Arguments:*

- `frequency`: Annual or quarter.
- `clean_names`: Logical; if TRUE, converts column names to snake case.

*Returns:* A tibble.

*Examples:*

```r
\donttest{
aapl <- Ticker$new('aapl')
aapl$get_balance_sheet('annual')
aapl$get_balance_sheet('quarter')
}
```

**Method** `get_cash_flow()`: Retrieves cash flow data for most recent four quarters or most recent four years.

*Usage:*

```
Ticker$get_cash_flow(frequency = c("annual", "quarter"), clean_names = TRUE)
```

*Arguments:*

- `frequency`: Annual or quarter.
clean_names Logical; if TRUE, converts column names to snake case.

Returns: A tibble.

Examples:
\donttest{
aapl <- Ticker$new('aapl')
aapl$get_cash_flow('annual')
aapl$get_cash_flow('quarter')
}

Method get_income_statement(): Retrieves income statement data for most recent four quarters or most recent four years.

Usage:
Ticker$get_income_statement(
  frequency = c("annual", "quarter"),
  clean_names = TRUE
)

Arguments:
frequency Annual or quarter.
clean_names Logical; if TRUE, converts column names to snake case.

Returns: A tibble.

Examples:
\donttest{
aapl <- Ticker$new('aapl')
aapl$get_income_statement('annual')
aapl$get_income_statement('quarter')
}

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:
```r
\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:
deepl Whether to make a deep clone.

Examples
```r
## Method `Ticker$new`
## ------------------------------------------------
aapl <- Ticker$new('aapl')
## ------------------------------------------------
## Method `Ticker$set_symbol`
## ------------------------------------------------
aapl <- Ticker$new('aapl')
aapl$set_symbol('msft')
```
## Method `Ticker$get_balance_sheet`

aapl <- Ticker$new('aapl')
aapl$get_balance_sheet('annual')
aapl$get_balance_sheet('quarter')

## Method `Ticker$get_cash_flow`

aapl <- Ticker$new('aapl')
aapl$get_cash_flow('annual')
aapl$get_cash_flow('quarter')

## Method `Ticker$get_income_statement`

aapl <- Ticker$new('aapl')
aapl$get_income_statement('annual')
aapl$get_income_statement('quarter')

## 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')

---

**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

currency_converter, 2
currency_summary, 3

get_currencies, 4
get_market_summary, 4
get_trending, 5

Index (Index-class), 5
Index-class, 5

Ticker (Ticker-class), 8
Ticker-class, 8

validate, 13