Package 'yahoofinancer'

October 16, 2026

Type Package

Title Fetch Data from Yahoo Finance API

Version 0.6.1

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 (>= 4.1)

Imports curl, dplyr, httr2 (>= 1.0.0), jsonlite, lubridate, magrittr, purrr, R6, tibble

Suggests covr, ggplot2, httptest2, knitr, rmarkdown, scales, shiny, testthat (>= 3.0.0), tidyr, zoo

VignetteBuilder knitr

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 7.3.3

NeedsCompilation no

Author Aravind Hebbali [aut, cre]

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

Repository CRAN

Date/Publication 2026-10-16 08:50:02 UTC

Contents

currency_converter 2
get_currencies 3
get_market_summary 4
get_trending 5
Indice-class 6
Ticker-class 7
validate 10
yf_download_prices 11
yf_get_financials 12
yf_get_index_quotes 13
yf_get_market_stats 13

Index **15**

`currency_converter` *Currency Converter*

Description

Retrieve current and historical exchange rates between two currencies via the Yahoo Finance chart API.

Usage

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

Arguments

`from`	ISO 4217 three-letter currency code to convert from (e.g., `"EUR"`, `"GBP"`). Defaults to `"EUR"`.
`to`	ISO 4217 three-letter currency code to convert to (e.g., `"USD"`, `"JPY"`). Defaults to `"USD"`.
`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: • `'1d'` • `'5d'`

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

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

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

Value

A tidy tibble with columns: date (POSIXct), high, low, open, close, volume (all numeric), and conditionally adj_close (numeric, present for daily and longer intervals). Rows with NA volume are excluded. Returns an empty tibble on network failure.

See Also

Other currency: get_currencies()

Examples

```
## Not run:
currency_converter("GBP", "USD", "2022-07-01", "2022-07-10")
currency_converter("GBP", "USD", period = "1mo", interval = "1d")

## End(Not run)
```

---

get_currencies *Get Supported Currencies*

Description

Retrieves the list of currencies supported by Yahoo Finance.

Usage

```
get_currencies()
```

Value

A `data.frame` with 4 columns: `short_name` (character), `long_name` (character), `symbol` (character), `local_long_name` (character). Returns `invisible(NULL)` on network failure.

See Also

Other currency: `currency_converter()`

Examples

```
## Not run:
get_currencies()

## End(Not run)
```

`get_market_summary` *Get Real-Time Market Summary*

Description

Retrieve live market overview snapshots across global benchmark indices, commodities, currencies, and futures.

Usage

```
get_market_summary(as_tibble = TRUE)
```

Arguments

`as_tibble`	Logical; if `TRUE` (default), returns a tidy tibble. If `FALSE`, returns the raw nested list from the API.

Value

If `as_tibble = TRUE`, a `tibble` with 9 columns: `symbol`, `short_name`, `regular_market_price`, `regular_market_change`, `regular_market_change_percent`, `regular_market_previous_close`, `market_state`, `exchange`, `market_time` (POSIXct, UTC). If `as_tibble = FALSE`, a nested list. Returns an empty tibble (or `invisible(NULL)`) on failure.

See Also

Other market data: `get_trending()`, `yf_get_market_stats()`

Examples

```
## Not run:
# Tidy tibble output (default)
market_df <- get_market_summary()

# Raw list output
market_list <- get_market_summary(as_tibble = FALSE)

## End(Not run)
```

`get_trending` *Get Trending Securities*

Description

Retrieves the list of currently trending securities for a specific region from Yahoo Finance.

Usage

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

Arguments

`country`	ISO 3166-1 alpha-2 region code (e.g., `"US"`, `"GB"`, `"IN"`). Defaults to `"US"`.
`count`	Maximum number of trending securities to return. Defaults to `10`.

Value

A character vector of trending ticker symbols. Returns `invisible(NULL)` on network failure or if no data is available (with a message).

See Also

Other market data: `get_market_summary()`, `yf_get_market_stats()`

Examples

```
## Not run:
get_trending()
get_trending(country = "GB", count = 5)

## End(Not run)
```

Indice-class *R6 Class Representing an Index*

Description

Class for retrieving data related to market indices from the Yahoo Finance API. Inherits `get_history()` from `YahooFinanceBase-class`.

Format

An R6 class object

Super class

`yahoofinancer::YahooFinanceBase` -> `Index`

Active bindings

`index` Deprecated. Returns `self$symbol`.

Methods

Public methods:

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

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

Usage:

```
Index$new(symbol = NA, index = NA, validate = TRUE)
```

Arguments:

`symbol` Symbol (e.g., `"^NSEI"`).

`index` Deprecated. Use `symbol` instead.

`validate` Logical; if TRUE, validate symbol against Yahoo Finance. Defaults to TRUE.

Returns: A new `Index` object.

Examples:

```
\dontrun{
nifty_50 <- Index$new('^NSEI')
}
```

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

Usage:

```
Index$set_index(symbol = NA, index = NA, validate = TRUE)
```

Arguments:

`symbol` New symbol.

`index` Deprecated. Use `symbol` instead.

`validate` Logical; if TRUE, validate symbol against Yahoo Finance. Defaults to TRUE.

Examples:

```
\dontrun{
indice <- Index$new('^NSEI')
indice$set_index('^NDX')
}
```

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.

See Also

Other historical data: `Ticker-class`, `Tickers`, `yf_download_prices()`, `yf_get_index_quotes()`

Examples

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

## Not run:
nifty_50 <- Index$new('^NSEI')

## End(Not run)

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

## Not run:
indice <- Index$new('^NSEI')
indice$set_index('^NDX')

## End(Not run)
```

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

Description

Class for retrieving all data related to a single ticker from the Yahoo Finance API. Inherits `get_history()` from `YahooFinanceBase-class`.

Format

An R6 class object

Super class

yahoofinancer::YahooFinanceBase -> Ticker

Active bindings

valuation_measures Retrieves valuation measures

recommendations Related symbols recommended by Yahoo Finance and their scores.

technical_insights Technical insights and indicators snapshot.

currency Currency code for the symbol.

exchange_name Short name of the stock exchange.

full_exchange_name Full name of the stock exchange.

first_trade_date Timestamp of the first recorded trade.

regular_market_time Timestamp of the last market trade.

timezone Timezone code of the exchange.

exchange_timezone_name Full timezone name of the exchange.

regular_market_price Current regular market price.

fifty_two_week_high 52-week high price.

fifty_two_week_low 52-week low price.

regular_market_day_high Highest price during the current trading session.

regular_market_day_low Lowest price during the current trading session.

regular_market_volume Current trading volume.

previous_close Closing price of the previous trading day.

Methods

Public methods:

- Ticker$new()
- Ticker$set_symbol()
- Ticker$get_income_statement()
- Ticker$get_balance_sheet()
- Ticker$get_cash_flow()
- Ticker$clone()

Method new(): Create a new Ticker object.

Usage:

```
Ticker$new(symbol = NA, validate = TRUE)
```

Arguments:

symbol Symbol (e.g., "AAPL").

validate Logical; if TRUE, validate symbol against Yahoo Finance. Defaults to TRUE.

Method `set_symbol()`**:** Set a new symbol and clear cached metadata.

Usage:

```
Ticker$set_symbol(symbol, validate = TRUE)
```

Arguments:

symbol New symbol (e.g., "AAPL").

validate Logical; if TRUE, validate symbol against Yahoo Finance. Defaults to TRUE.

Method `get_income_statement()`**:** Retrieve income statement data.

Usage:

```
Ticker$get_income_statement(frequency = c("annual", "quarterly"))
```

Arguments:

frequency One of "annual" or "quarterly". Defaults to "annual".

Returns: A tibble with columns date, period_type, and income statement line items, or invisible(NULL) on failure.

Method `get_balance_sheet()`**:** Retrieve balance sheet data.

Usage:

```
Ticker$get_balance_sheet(frequency = c("annual", "quarterly"))
```

Arguments:

frequency One of "annual" or "quarterly". Defaults to "annual".

Returns: A tibble with columns date, period_type, and balance sheet line items, or invisible(NULL) on failure.

Method `get_cash_flow()`**:** Retrieve cash flow statement data.

Usage:

```
Ticker$get_cash_flow(frequency = c("annual", "quarterly"))
```

Arguments:

frequency One of "annual" or "quarterly". Defaults to "annual".

Returns: A tibble with columns date, period_type, and cash flow statement line items, or invisible(NULL) on failure.

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.

See Also

Other historical data: Indice-class, Tickers, yf_download_prices(), yf_get_index_quotes()

validate *Symbol validation*

Description

Validate symbols before retrieving data. This function checks whether the provided ticker strings are valid according to the Yahoo Finance API.

Usage

```
validate(symbol = NULL, index = NA, return_logical = FALSE)
```

Arguments

symbol	A character vector of one or more ticker strings, index or fund names (e.g., c("AAPL", "MSFT")).
index	Deprecated. Use symbol instead.
return_logical	A logical value indicating whether to return a named logical vector. If FALSE (default), returns a character vector containing only the valid symbols. If TRUE, returns a named logical vector indicating validation status (TRUE/FALSE) for each symbol.

Details

Symbol validation

Value

If return_logical = FALSE (the default), a character vector of valid symbols. If return_logical = TRUE, a named logical vector indicating validation status for each symbol.

Examples

```
## Not run:
validate("AAPL")
validate(c("AAPL", "NOTASTOCK", "MSFT"))
validate(c("AAPL", "NOTASTOCK", "MSFT"), return_logical = TRUE)

## End(Not run)
```

`yf_download_prices` *Download Historical Prices*

Description

Fetches historical OHLCV price data for one or more tickers using the Yahoo Finance chart API. Results are combined into a single long-format tibble suitable for `dplyr` pipelines.

Intraday lookback limits (imposed by Yahoo Finance):

- `"1m"`: max 7 days
- `"5m"`, `"15m"`, `"30m"`: max 60 days
- `"1h"`: max 730 days

Usage

```
yf_download_prices(
  tickers,
  start = NULL,
  end = NULL,
  interval = "1d",
  period = NULL
)
```

Arguments

`tickers`	Character vector of one or more stock symbols (e.g., `c("AAPL", "MSFT")`).
`start`	Date or character string in `"YYYY-MM-DD"` format representing the start date. When provided, `period` is ignored.
`end`	Date or character string in `"YYYY-MM-DD"` format representing the end date. Defaults to today if `start` is set but `end` is `NULL`.
`interval`	Time between data points. Valid values: `"1m"`, `"2m"`, `"5m"`, `"15m"`, `"30m"`, `"60m"`, `"90m"`, `"1h"`, `"1d"`, `"5d"`, `"1wk"`, `"1mo"`, `"3mo"`. Defaults to `"1d"`.
`period`	Relative time period. Valid values: `"1d"`, `"5d"`, `"1mo"`, `"3mo"`, `"6mo"`, `"1y"`, `"2y"`, `"5y"`, `"10y"`, `"ytd"`, `"max"`. Defaults to `"1y"` when both `start` and `period` are `NULL`.

Value

A `tibble` with 8 columns: `symbol`, `date`, `open`, `high`, `low`, `close`, `adj_close`, `volume`. Returns an empty tibble if all tickers fail.

See Also

Other historical data: `Indice-class`, `Ticker-class`, `Tickers`, `yf_get_index_quotes()`

Examples

```
## Not run:
# Single ticker with date range
yf_download_prices("AAPL", start = "2023-01-01", end = "2023-01-10")

# Multiple tickers with relative period
yf_download_prices(c("AAPL", "MSFT"), period = "6mo", interval = "1mo")

## End(Not run)
```

---

yf_get_financials *Get Financial Statements*

Description

Retrieves quarterly or annual financial statements (income statement, balance sheet, or cash flow) for one or more tickers.

Usage

```
yf_get_financials(
  tickers,
  statement_type = c("income", "balance-sheet", "cash-flow"),
  frequency = c("annual", "quarterly")
)
```

Arguments

`tickers`	Character vector of stock symbols.
`statement_type`	The type of financial statement to retrieve. One of `"income"`, `"balance-sheet"`, or `"cash-flow"`.
`frequency`	One of `"annual"` or `"quarterly"`. Defaults to `"annual"`.

Value

A `tibble` with `symbol` as the first column. Remaining columns vary by `statement_type` and reflect the line items returned by the Yahoo Finance API. Returns an empty tibble if all tickers fail.

Examples

```
## Not run:
yf_get_financials("AAPL", statement_type = "income")
yf_get_financials(c("AAPL", "MSFT"), statement_type = "balance-sheet", frequency = "quarterly")

## End(Not run)
```

yf_get_index_quotes *Get Index Quotes*

Description

Retrieves the most recent 1-day price snapshot for a given market index.

Usage

```
yf_get_index_quotes(index_symbol)
```

Arguments

index_symbol	Character string of the index symbol (e.g., "^GSPC" for S&P 500, "^NSEI" for Nifty 50).

Value

A tibble with the same 8-column schema as yf_download_prices: symbol, date, open, high, low, close, adj_close, volume. Returns an empty tibble on failure.

See Also

Other historical data: Indice-class, Ticker-class, Tickers, yf_download_prices()

Examples

```
## Not run:
yf_get_index_quotes("^GSPC")

## End(Not run)
```

yf_get_market_stats *Get Market Statistics*

Description

Retrieves key real-time valuation and summary fields for one or more tickers.

Usage

```
yf_get_market_stats(tickers)
```

Arguments

tickers	Character vector of stock symbols.

Value

A tibble with 7 columns: symbol (character), regular_market_price (numeric), fifty_two_week_high (numeric), fifty_two_week_low (numeric), regular_market_volume (numeric), previous_close (numeric), currency (character). Returns an empty tibble if all tickers fail.

See Also

Other market data: get_market_summary(), get_trending()

Examples

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

## End(Not run)
```

Index

∗ **currency**
    currency_converter, 2
    get_currencies, 3
∗ **fundamental data**
    yf_get_financials, 12
∗ **historical data**
    Indice-class, 6
    Ticker-class, 7
    yf_download_prices, 11
    yf_get_index_quotes, 13
∗ **market data**
    get_market_summary, 4
    get_trending, 5
    yf_get_market_stats, 13
∗ **validation**
    validate, 10

currency_converter, 2, *4*

get_currencies, *3*, 3
get_market_summary, 4, *5*, *14*
get_trending, *4*, 5, *14*

Index *(*Indice-class*)*, 6
Index-class *(*Indice-class*)*, 6
Indice-class, 6

tibble, *3*, *4*, *9*, *11–14*
Ticker *(*Ticker-class*)*, 7
Ticker-class, 7
Tickers, *7*, *9*, *11*, *13*

validate, 10

yahoofinancer::YahooFinanceBase, *6*, *8*
yf_download_prices, *7*, *9*, 11, *13*
yf_get_financials, 12
yf_get_index_quotes, *7*, *9*, *11*, 13
yf_get_market_stats, *4*, *5*, 13