Package 'filters.trade'

September 17, 2026

Title A Filter System for Selecting Trading Instruments

Version 0.0.1

Description Enables filtering datasets of tradable instruments by prior specified identifiers which correspond to saved filter expressions. A filter is a named expression bound to a target dataset, stored once in a package level registry, and later applied to select trading codes such as tickers or symbols out of a universe, price or signal dataset. The design follows the `filters` package, replacing the clinical study dataset convention with a trading instrument convention.

Depends R (>= 4.1.0)

Imports yaml

Suggests roxygen2 (>= 7.0.0), testthat (>= 3.0.0)

Encoding UTF-8

Config/testthat/edition 3

License Apache License (>= 2.0)

Config/roxygen2/version 8.1.0

LazyData true

NeedsCompilation no

Author Joe Zhu [aut, cre]

Maintainer Joe Zhu <sha.joe.zhu@gmail.com>

Repository CRAN

Date/Publication 2026-09-17 13:20:20 UTC

Contents

`filters.trade-package` *filters.trade Package*

Description

A filter system for selecting trading instruments. `add_filter()` and `apply_filter()` are the two entry points of the package.

Author(s)

Maintainer: Joe Zhu <sha.joe.zhu@gmail.com>

Authors:

- Joe Zhu <sha.joe.zhu@gmail.com>

`add_filter` *Add a New Filter Definition*

Description

Add a new filter definition or overwrite an existing one. A filter binds an unevaluated `condition` to a `target` dataset so that it can be applied later by `apply_filter()`. Unless `character_only = TRUE`, `condition` is captured with `substitute()` and therefore written without quotes.

Usage

```
add_filter(
  id,
  title,
  target,
  condition,
  character_only = FALSE,
  overwrite = FALSE
)
```

Arguments

id	character The id of this filter. Only upper case letters and numbers are allowed, e.g. "LARGECAP".
title	character The title of the filter
target	character The target dataset of the filter, e.g. "TICKERS" or "PRICES"
condition	The filter condition
character_only	logical. Is condition a string? Defaults to FALSE.
overwrite	logical Should existing filters be overwritten? Defaults to FALSE.

Value

A list of title, target and condition, invisibly

Examples

```
add_filter(
  id = "LARGECAP",
  title = "Large Cap",
  target = "TICKERS",
  condition = MARKET_CAP >= 1e10
)

add_filter(
  id = "TECH",
  title = "Technology Sector",
  target = "TICKERS",
  condition = "SECTOR == 'Technology'",
  character_only = TRUE
)
```

---

apply_filter *Apply a Filter to a Dataset or List of Datasets*

Description

Apply a Filter to a Dataset or List of Datasets

Usage

```
apply_filter(data, ...)

## Default S3 method:
apply_filter(data, ...)

## S3 method for class 'data.frame'
```

```
apply_filter(data, id, target = deparse(substitute(data)), verbose = TRUE, ...)

## S3 method for class 'list'
apply_filter(data, id, verbose = TRUE, ...)
```

Arguments

data	data.frame or named list of data.frames
...	Not used.
id	character. The ID of one or more filters defined with add_filter(), combined with underscores. An empty string applies no filter.
target	character. The name of the dataset, e.g. "TICKERS" or "PRICES"
verbose	logical. Should informative messages be printed? Defaults to TRUE.

Value

A new data.frame or list of data.frames filtered based upon the condition defined for id. When the master trading code dataset is among the filter targets, every other dataset is restricted to the surviving trading codes.

Examples

```
tickers <- data.frame(
  SYMBOL = c("AAA", "BBB", "CCC"),
  SECTOR = c("Technology", "Energy", "Technology"),
  stringsAsFactors = FALSE
)
prices <- data.frame(
  SYMBOL = c("AAA", "BBB", "CCC"),
  CLOSE = c(10, 20, 30),
  stringsAsFactors = FALSE
)

add_filter(
  id = "TECH",
  title = "Technology Sector",
  target = "TICKERS",
  condition = SECTOR == "Technology",
  overwrite = TRUE
)

apply_filter(tickers, "TECH", target = "TICKERS")
apply_filter(list(tickers = tickers, prices = prices), "TECH")
```

get_filter *Get a Filter Definition*

Description

Get a Filter Definition

Usage

```
get_filter(id)
```

Arguments

id	character. The filter ID

Value

A list with elements title, target and condition

Examples

```
add_filter(
  id = "LARGECAP",
  title = "Large Cap",
  target = "TICKERS",
  condition = MARKET_CAP >= 1e10,
  overwrite = TRUE
)
get_filter("LARGECAP")

## Filter `FOO` does not exist
try(get_filter("FOO"))
```

get_filters *Get Multiple Filter Definitions*

Description

Filter IDs are combined into a single string separated by underscores, so "US_LARGECAP" requests the two filters US and LARGECAP.

Usage

```
get_filters(ids)
```

Arguments

ids	character. The filter IDs as a single string separated by underscores

Value

A named list of filter definitions

Examples

```
add_filter(
  id = "US",
  title = "US Listed",
  target = "TICKERS",
  condition = EXCHANGE %in% c("NYSE", "NASDAQ"),
  overwrite = TRUE
)
get_filters("US_LARGECAP")
```

---

list_all_filters *List All Filters*

Description

List all filter definitions currently held in the registry.

Usage

```
list_all_filters()
```

Value

A data.frame with columns id, title, target and condition

Examples

```
list_all_filters()
```

load_filters *Load Filter Definitions*

Description

Load filter definitions from a yaml file. Each top level key is a filter id and maps to the fields `title`, `target` and `condition`.

Usage

```
load_filters(yaml_file, overwrite = FALSE)
```

Arguments

`yaml_file`	`character`. A path to a yaml file containing filter definitions
`overwrite`	`logical`. Should existing filter definitions be overwritten? Defaults to `FALSE`.

Value

On success, `load_filters()` returns `TRUE` invisibly

Examples

```
filter_definitions <- system.file("filters_eg.yaml", package = "filters.trade")
if (interactive()) file.edit(filter_definitions)
load_filters(filter_definitions, overwrite = TRUE)
```

nz_data *New Zealand Exchange (NZX) trading instruments*

Description

A master list of trading instruments listed on New Zealand's Exchange (NZX), in the shape `filters.trade` expects of the master trading-code dataset. The join key is the `SYMBOL` column; every other column is an attribute that a filter condition can reference unquoted.

Usage

```
nz_data
```

Format

A data.frame with 168 rows and 7 columns:

SYMBOL Trading code, e.g. `"ACE.NZ"`. The code column.

NAME Company or instrument name.

EXCHANGE Exchange code; always `"NZX"`.

SECTOR Sector classification.

MARKET_CAP Market capitalisation in NZD.

DIV_YIELD Dividend yield in percent.

ADV_20D Average daily volume over 20 days, in shares.

Source

`SYMBOL` and `NAME` are from `~/homepage-stock/data/nz_list.csv`. `EXCHANGE`, `SECTOR`, `MARKET_CAP`, `DIV_YIELD` and `ADV_20D` are synthetic placeholders generated in `data-raw/nz_data.R` for example use.

Index