Package: etfdata (via r-universe)

May 23, 2026

Title European UCITS ETF Data Collection

Version 0.0.0.9000

Description Tools to fetch, clean, and store historical and metadata for European bond and equity index ETFs (UCITS) listed on the London Stock Exchange.

License MIT + file LICENSE

URL https://johngavin.github.io/etf-data

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Config/testthat/edition 3

Imports dplyr, httr2, janitor, jsonlite, logger, lubridate, magrittr, purrr, quantmod, readr, rlang, rvest, stringr, tibble, utf8

Suggests ggplot2, knitr, rmarkdown, targets, quarto, visNetwork, DiagrammeR, shinylive, DT, htmltools, tidyquant, scales

VignetteBuilder knitr, quarto

Config/pak/sysreqs libicu-dev libxml2-dev libssl-dev libx11-dev

Repository https://johngavin.r-universe.dev

Date/Publication 2026-02-22 11:19:36 UTC

RemoteUrl https://github.com/JohnGavin/etf-data

RemoteRef HEAD

RemoteSha 1ac4f23df24eff47ab6e8d675d135bb130cf67ef

RemoteSubdir etfdata

Contents

fetch_etf_holdings 2
fetch_etf_metadata 2

fetch_justetf_metadata 3
fetch_justetf_screener 4
fetch_price_history 5
get_etf_universe 5
parse_aum 6

Index **7**

`fetch_etf_holdings` *Fetch ETF Holdings from Yahoo Finance*

Description

Retrieves historical fund holdings (composition) for a given ETF ticker using tidyquant. This includes top holdings, sector weights, and asset allocation data.

Usage

```
fetch_etf_holdings(symbol)
```

Arguments

`symbol`	Character string. The ticker symbol of the ETF (e.g., "SPY").

Value

A tibble containing holding details. Returns NULL or empty tibble on failure.

Examples

```
## Not run:
  fetch_etf_holdings("SPY")

## End(Not run)
```

`fetch_etf_metadata` *Fetch ETF Metadata from Yahoo Finance API*

Description

Fetches metadata (AUM, TER, etc.) for a given ETF ticker from the Yahoo Finance API.

Usage

```
fetch_etf_metadata(symbol)
```

Arguments

symbol	Character string. The Yahoo Finance ticker symbol of the ETF (e.g., "VUSA.L").

Value

A tibble with metadata columns. Returns an empty tibble with the symbol if fetching fails.

Examples

```
## Not run:
  fetch_etf_metadata("VUSA.L")

## End(Not run)
```

---

fetch_justetf_metadata

Fetch ETF Metadata from JustETF

Description

Scrapes metadata (AUM, TER, etc.) for a given ETF ISIN from JustETF.

Usage

```
fetch_justetf_metadata(isin)
```

Arguments

isin	Character string. The ISIN of the ETF (e.g., "IE00B3XXRP09").

Value

A tibble with metadata columns.

Examples

```
## Not run:
  fetch_etf_metadata("IE00B3XXRP09")

## End(Not run)
```

`fetch_justetf_screener` *Fetch ETF Data from JustETF*

Description

Retrieves a list of ETFs from JustETF, applying optional filters for AUM and TER. The function first tries the legacy JSON API, then falls back to the JustETF search table endpoint if needed.

Usage

```
fetch_justetf_screener(
  min_aum_gbp = 0,
  max_ter = Inf,
  page_size = 500,
  max_pages = Inf,
  source = c("auto", "api", "wicket"),
  quiet = FALSE
)
```

Arguments

`min_aum_gbp`	Numeric. Minimum Assets Under Management in GBP (millions). Default is 0.
`max_ter`	Numeric. Maximum Total Expense Ratio (percentage). Default is Inf.
`page_size`	Integer. Number of rows per page when paging the table endpoint.
`max_pages`	Integer. Maximum number of pages to fetch when paging. Default Inf.
`source`	Character. Data source preference: "auto", "api", or "wicket".
`quiet`	Logical. If TRUE, suppresses warnings when endpoints fail.

Value

A tibble containing the filtered ETF data.

Examples

```
## Not run:
  etfs <- fetch_justetf_screener()
  print(etfs)

## End(Not run)
```

fetch_price_history *Fetch Price History from Yahoo Finance*

Description

Retrieves daily OHLCV data for a given ticker using quantmod/Yahoo.

Usage

```
fetch_price_history(ticker, start_date = "2020-01-01", end_date = Sys.Date())
```

Arguments

ticker	Character string. The ticker symbol (e.g., "VUSA.L").
start_date	Date or character. Start date (default "2020-01-01").
end_date	Date or character. End date (default Sys.Date()).

Value

A tibble with Date, Open, High, Low, Close, Volume, Adjusted columns.

Examples

```
## Not run:
  fetch_price_history("VUSA.L")

## End(Not run)
```

get_etf_universe *Get ETF Universe*

Description

Retrieves the list of ETFs to track. Prefers the curated universe file when available, then falls back to the cached snapshot or seed list.

Usage

```
get_etf_universe(n = 20, live = FALSE, file = NULL)
```

Arguments

n	Integer. Maximum number of ETFs to return. Default is 20. Set to Inf for all.
live	Logical. If TRUE, attempts to fetch the latest universe from an external source.
file	Character. Path to a local CSV file to read the universe from. Overrides `live` and default cache.

Value

A tibble with columns: ticker, name, isin, currency

Examples

```
## Not run:
  get_etf_universe() # Returns top 20
  get_etf_universe(n = 5)
  get_etf_universe(n = Inf) # Returns all
  get_etf_universe(file = "my_etfs.csv")

## End(Not run)
```

---

parse_aum *Parse AUM Text*

Description

Converts an AUM (Assets Under Management) string (e.g., "GBP 100,642 m") into structured components.

Usage

```
parse_aum(aum_text)
```

Arguments

aum_text	Character vector. The AUM string(s) to parse.

Value

A tibble with columns: currency (factor), aum_amount (numeric), aum_units (factor), aum_units_amount (numeric).

Examples

```
parse_aum("GBP 100,642 m")
```

Index

fetch_etf_holdings, 2
fetch_etf_metadata, 2
fetch_justetf_metadata, 3
fetch_justetf_screener, 4
fetch_price_history, 5

get_etf_universe, 5

parse_aum, 6