Package ‘igfetchr’

November 21, 2025

Type Package

Title Access the 'IG Trading REST API'

Version 0.1.0

Maintainer Saw Simeon <saw.s@ku.th>

Description
Provides functions to fetch market data, search historical prices, execute trades, and get account details from the 'IG Trading REST API' <https://labs.ig.com>. Returns tidy tibbles for easy analysis. Trading contracts for difference (CFDs), options and spread bets carries a high risk of losing money. This package is not financial or trading advice.

License GPL-3

Encoding UTF-8

Collate 'igfetchr.R' 'igfetchr-package.R'

Imports httr, jsonlite, tibble, purrr

Suggests testthat (>= 3.0.0), knitr, rmarkdown, lubridate, httptest

VignetteBuilder knitr

URL https://github.com/sawsimeon/igfetchr,
https://sawsimeon.github.io/igfetchr/

BugReports https://github.com/sawsimeon/igfetchr/issues

RoxygenNote 7.3.3

NeedsCompilation no

Author Saw Simeon [aut, cre]

Repository CRAN

Date/Publication 2025-11-21 15:00:08 UTC

Contents

`igfetchr` *igfetchr: Lightweight IG Trading REST API wrapper*

Description

Helper functions to authenticate and fetch market/account data from the IG Trading REST API. All exported functions return tibbles where appropriate. This package is not financial advice. Trading CFDs and spread bets carries high risk of losing money. Key functions: ‘ig_auth‘ to authenticate, ‘ig_execute_trade‘ to place trades.

Author(s)

Maintainer: Saw Simeon <`saw.s@ku.th`>

See Also

Useful links:

- `https://github.com/sawsimeon/igfetchr`
- `https://sawsimeon.github.io/igfetchr/`
- Report bugs at `https://github.com/sawsimeon/igfetchr/issues`

`ig_auth` *Authenticate with the IG API*

Description

Authenticates with the IG API to obtain session tokens (CST and X-SECURITY-TOKEN) for subsequent API requests. Supports environment variables for credentials and optional account type and number.

Usage

```
ig_auth(
  username = Sys.getenv("IG_SERVICE_USERNAME"),
  password = Sys.getenv("IG_SERVICE_PASSWORD"),
  api_key = Sys.getenv("IG_SERVICE_API_KEY"),
  acc_type = Sys.getenv("IG_SERVICE_ACC_TYPE", "DEMO"),
  acc_number = Sys.getenv("IG_SERVICE_ACC_NUMBER")
)
```

Arguments

`username`	Character. IG account username. Defaults to ‘IG_SERVICE_USERNAME‘.
`password`	Character. IG account password. Defaults to ‘IG_SERVICE_PASSWORD‘.
`api_key`	Character. IG API key. Defaults to ‘IG_SERVICE_API_KEY‘.
`acc_type`	Character. Account type, either "DEMO" or "LIVE". Defaults to "DEMO".
`acc_number`	Character. Optional account number. Defaults to ‘IG_SERVICE_ACC_NUMBER‘ or NULL.

Value

A list containing client session token (cst), security token (X-SECURITY-TOKEN), base URL, API key, account type, and account number (or NULL if not provided).

Examples

```
## Not run:
# Using environment variables
Sys.setenv(IG_SERVICE_USERNAME = "your_username")
Sys.setenv(IG_SERVICE_PASSWORD = "your_password")
Sys.setenv(IG_SERVICE_API_KEY = "your_api_key")
Sys.setenv(IG_SERVICE_ACC_NUMBER = "ABC123")
Sys.setenv(IG_SERVICE_ACC_TYPE = "DEMO")
auth <- ig_auth()

# Using explicit arguments
auth <- ig_auth(
  username = "your_username",
  password = "your_password",
  api_key = "your_api_key",
  acc_type = "DEMO",
  acc_number = "ABC123"
)

## End(Not run)
```

ig_close_session *Close session*

Description

Closes the authenticated IG API session.

Alias for ‘ig_close_session()‘.

Usage

```
ig_close_session(auth, mock_response = NULL)

ig_logout(auth, mock_response = NULL)
```

Arguments

auth	List. Authentication details from ‘ig_auth()‘, including ‘cst‘, ‘security‘, ‘base_url‘, ‘api_key‘, and ‘acc_number‘.
mock_response	Logical. Optional mock response for testing (returns ‘TRUE‘ if provided). Defaults to ‘NULL‘.

Value

Logical ‘TRUE‘ if the session was closed successfully.

Examples

```
## Not run:
# Authenticate and close session
auth <- ig_auth(
  username = "your_username",
  password = "your_password",
  api_key = "your_api_key",
  acc_type = "DEMO",
  acc_number = "ABC123"
)
ig_close_session(auth)

# Using mock response for testing
ig_close_session(auth, mock_response = TRUE)

## End(Not run)
```

`ig_execute_trade` *Execute a trade (place OTC position)*

Description

Places a market trade using the IG API. If stops/limits fail, falls back to placing the trade without them and adding via PUT.

Usage

```
ig_execute_trade(
  epic,
  direction,
  size,
  auth,
  currency_code = NULL,
  expiry = NULL,
  guaranteed_stop = FALSE,
  level = NULL,
  time_in_force = "FILL_OR_KILL",
  order_type = "MARKET",
  limit_distance = NULL,
  limit_level = NULL,
  stop_distance = NULL,
  stop_level = NULL,
  deal_reference = NULL,
  force_open = NULL,
  mock_response = NULL
)
```

Arguments

`epic`	Character. Market epic (e.g., "CS.D.USDCHF.MINI.IP").
`direction`	Character. "BUY" or "SELL".
`size`	Numeric. Trade size (units).
`auth`	List. Authentication details from `ig_auth`.
`currency_code`	Character. Currency code (e.g., "CHF"). Defaults to NULL.
`expiry`	Character. Expiry date (e.g., "-"). Defaults to NULL.
`guaranteed_stop`	Logical. Use guaranteed stop. Defaults to FALSE.
`level`	Numeric. Price level for LIMIT orders. Defaults to NULL.
`time_in_force`	Character. "EXECUTE_AND_ELIMINATE" or "FILL_OR_KILL". Defaults to "FILL_OR_KILL".
`order_type`	Character. "MARKET" or "LIMIT". Defaults to "MARKET".

`limit_distance`	Numeric. Limit distance in points. Defaults to NULL.
`limit_level`	Numeric. Limit price. Defaults to NULL.
`stop_distance`	Numeric. Stop distance in points. Defaults to NULL.
`stop_level`	Numeric. Stop price. Defaults to NULL.
`deal_reference`	Character. Custom deal reference. Defaults to NULL.
`force_open`	Logical. Force new position. Defaults to TRUE if stops/limits specified.
`mock_response`	List or data frame. Mock response for testing.

Value

A tibble with trade confirmation details including deal ID and deal reference.

Examples

```
## Not run:
auth <- ig_auth(
username = "your_username",
password = "your_password",
api_key = "your_api_key",
acc_type = "DEMO",
acc_number = "ABC123")
res <- ig_execute_trade(
  epic = "CS.D.USDCHF.MINI.IP",
  direction = "BUY",
  size = 1.0,
  auth = auth,
  currency_code = "CHF",
  order_type = "MARKET",
  time_in_force = "FILL_OR_KILL",
  limit_distance = 2000,
  stop_distance = 2000,
  guaranteed_stop = FALSE,
  force_open = TRUE
)
print(res)

## End(Not run)
```

`ig_get_accounts` *Retrieve IG account details*

Description

Fetches details of IG accounts associated with the authenticated session. Returns a tibble containing account information such as account ID, name, balance, and status.

Usage

```
ig_get_accounts(auth, mock_response = NULL)
```

Arguments

`auth`	List. Authentication details from ‘ig_auth()‘, including ‘cst‘, ‘security‘, ‘base_url‘, ‘api_key‘, and ‘acc_number‘.
`mock_response`	List or data frame. Optional mock response for testing, bypassing the API call.

Value

A tibble with account information including account ID, name, balance and currency.

Examples

```
## Not run:
# Authenticate and get accounts
auth <- ig_auth(
  username = "your_username",
  password = "your_password",
  api_key = "your_api_key",
  acc_type = "DEMO",
  acc_number = "ABC123"
)
accounts <- ig_get_accounts(auth)
print(accounts)

# Using mock response for testing
mock_response <- data.frame(
  accountId = "ABC123",
  accountName = "Demo Account",
  balance.balance = 10000,
  currency = "SEK"
)
accounts <- ig_get_accounts(auth, mock_response = mock_response)

## End(Not run)
```

`ig_get_historical` *Get historical prices for a market*

Description

Fetches historical prices for a market epic between specified dates at a given resolution from the IG API. Uses the /prices/{epic}/{resolution}/{startDate}/{endDate} endpoint (version 2) with a fallback to version 3.

Usage

```
ig_get_historical(
  epic,
  from,
  to,
  resolution = "D",
  page_size = 20,
  auth,
  mock_response = NULL,
  wait = 1
)
```

Arguments

`epic`	Character. Market epic (e.g., "CS.D.USDCHF.MINI.IP").
`from`	Character or Date. Start date (e.g., "2025-09-01" or "2025-09-01 00:00:00"). Required.
`to`	Character or Date. End date (e.g., "2025-09-28" or "2025-09-28 23:59:59"). Required.
`resolution`	Character. Resolution code (e.g., "D", "1MIN", "HOUR"). Defaults to "D".
`page_size`	Integer. Number of data points per page (v3 only). Defaults to 20.
`auth`	List. Authentication details from ‘ig_auth()‘, including ‘cst‘, ‘security‘, ‘base_url‘, ‘api_key‘, and ‘acc_number‘.
`mock_response`	List or data frame. Optional mock response for testing, bypassing the API call.
`wait`	Numeric. Seconds to wait between paginated API calls (v3 only). Defaults to 1.

Value

A tibble with historical OHLC data including snapshot time, bid, as, open, close, high, low prices and last traded volume.

Examples

```
## Not run:
# Authenticate and get historical prices
auth <- ig_auth(
  username = "your_username",
  password = "your_password",
  api_key = "your_api_key",
  acc_type = "DEMO",
  acc_number = "ABC123"
)
hist <- ig_get_historical(
  "CS.D.USDCHF.MINI.IP",
  from = "2025-09-01",
  to = "2025-09-28",
  resolution = "D",
  page_size = 20,
  auth
)
print(hist)

# Using time
hist <- ig_get_historical(
  "CS.D.USDCHF.MINI.IP",
  from = "2025-09-01 00:00:00",
  to = "2025-09-28 23:59:59",
  resolution = "D",
  page_size = 20,
  auth
)

# Using mock response
mock_response <- list(
  prices = data.frame(
    snapshotTime = "2025/09/01 00:00:00",
    openPrice.bid = 0.970,
    openPrice.ask = 0.971,
    openPrice.lastTraded = NA,
    highPrice.bid = 0.975,
    highPrice.ask = 0.976,
    highPrice.lastTraded = NA,
    lowPrice.bid = 0.965,
    lowPrice.ask = 0.966,
    lowPrice.lastTraded = NA,
    closePrice.bid = 0.971,
    closePrice.ask = 0.972,
    closePrice.lastTraded = NA,
    lastTradedVolume = 50000
  ),
  metadata = list(
    allowance = list(remainingAllowance = 10000),
    pageData = list(pageNumber = 1, totalPages = 1)
  )
)
hist <- ig_get_historical(
  "CS.D.USDCHF.MINI.IP",
  from = "2025-09-01",
  to = "2025-09-28",
  resolution = "D",
  page_size = 20,
  auth,
  mock_response = mock_response
)

## End(Not run)
```

`ig_get_markets_by_epic` *Get market details for one or more epics*

Description

Fetches detailed market information for the specified market epic(s) from the IG API using the '/markets' endpoint (version 2). Returns instrument details, dealing rules, and snapshot data.

Usage

```
ig_get_markets_by_epic(
  epics,
  auth,
  detailed = TRUE,
  mock_response = NULL,
  verbose = FALSE
)
```

Arguments

`epics`	Character vector. One or more market epics (e.g., "CS.D.USDCHF.MINI.IP" or c("CS.D.USDCHF.MINI.IP", "CS.D.EURUSD.MINI.IP")).
`auth`	List. Authentication details from `ig_auth`.
`detailed`	Logical. Whether to return detailed info (instrument and dealing rules) or snapshot data only. Defaults to TRUE.
`mock_response`	List or data frame. Optional mock response for testing, bypassing the API call.
`verbose`	Logical. Whether to print the raw API response for debugging. Defaults to FALSE.

Value

A list with market details including instrument (epic, currencies, marginDepositBands), dealingRules (minStepDistance, minDealSize) and snapshot data (marketStatus, bid, offer, high, low).

Examples

```
## Not run:
# Authenticate with IG API
auth <- ig_auth(
  username = Sys.getenv("IG_SERVICE_USERNAME"),
  password = Sys.getenv("IG_SERVICE_PASSWORD"),
  api_key = Sys.getenv("IG_SERVICE_API_KEY"),
  acc_type = Sys.getenv("IG_SERVICE_ACC_TYPE"),
  acc_number = Sys.getenv("IG_SERVICE_ACC_NUMBER")
)

# Example 1: Fetch details for a single epic
```

```
markets <- ig_get_markets_by_epic("CS.D.USDCHF.MINI.IP", auth)
print(markets)
# Expected output: A tibble with 1 row and 3 columns (instrument, dealingRules, snapshot)

# Example 2: Fetch details for multiple epics
markets <- ig_get_markets_by_epic(c("CS.D.USDCHF.MINI.IP", "CS.D.EURUSD.MINI.IP"), auth)
print(markets)
# Expected output: A tibble with 2 rows and 3 columns

# Example 3: Fetch snapshot data only (no instrument or dealingRules)
markets <- ig_get_markets_by_epic("CS.D.USDCHF.MINI.IP", auth, detailed = FALSE)
print(markets)
# Expected output: A tibble with 1 row and 1 column (snapshot)

# Example 4: Fetch with verbose output for debugging
markets <- ig_get_markets_by_epic("CS.D.USDCHF.MINI.IP", auth, verbose = TRUE)
print(markets)
# Expected output: Prints raw JSON response, followed by a tibble with 1 row and 3 columns

# Example 5: Use a mock response for testing
mock_response <- list(
  marketDetails = list(
    list(
      instrument = list(
        epic = "CS.D.USDCHF.MINI.IP",
        name = "USD/CHF Mini",
        currencies = list(list(code = "CHF",
        symbol = "SF",
        baseExchangeRate = 0.08467604,
        isDefault = FALSE)),
        marginDepositBands = list(
          list(min = 0, max = 124, margin = 3.33, currency = "CHF"),
          list(min = 124, max = 310, margin = 3.33, currency = "CHF")
        ),
        marginFactor = 3.33,
        marginFactorUnit = "PERCENTAGE"
      ),
      dealingRules = list(
        minStepDistance = list(unit = "POINTS", value = 1.0),
        minDealSize = list(unit = "POINTS", value = 0.1)
      ),
      snapshot = list(
        marketStatus = "TRADEABLE",
        bid = 0.79715,
        offer = 0.79739,
        high = 0.79888,
        low = 0.79512,
        updateTime = "2025/09/29 18:40:51",
        binaryOdds = NA,
        decimalPlacesFactor = 5,
        scalingFactor = 10000
      )
    )
  )
)
markets <- ig_get_markets_by_epic("CS.D.USDCHF.MINI.IP", auth = NULL, mock_response = mock_response)
print(markets)

## End(Not run)
```

`ig_get_options` *Get options/derivatives positions*

Description

Retrieves positions filtered for options/derivatives from the IG API. Returns a tibble with position details.

Usage

```
ig_get_options(auth, mock_response = NULL)
```

Arguments

`auth`	List. Authentication details from ‘ig_auth()‘, including ‘cst‘, ‘security‘, ‘base_url‘, ‘api_key‘, and ‘acc_number‘.
`mock_response`	List or data frame. Optional mock response for testing, bypassing the API call.

Value

A tibble with options/derivative position details including deal ID, size, direction and instrument type.

Examples

```
## Not run:
# Authenticate and get options positions
auth <- ig_auth(
  username = "your_username",
  password = "your_password",
  api_key = "your_api_key",
  acc_type = "DEMO",
  acc_number = "ABC123"
)
options <- ig_get_options(auth)
print(options)

# Using mock response for testing
mock_response <- data.frame(
  dealId = "DIXXXX",
  size = 1.5,
  direction = "BUY",
  instrumentType = "OPTION"
)
options <- ig_get_options(auth, mock_response = mock_response)

## End(Not run)
```

`ig_get_price` *Get current price for a market*

Description

Fetches current price(s) for the given market epic from the IG API using the '/markets/{epic}' endpoint.

Usage

```
ig_get_price(epic, auth, mock_response = NULL)
```

Arguments

`epic`	Character. Market epic (e.g., "CS.D.USDCHF.CFD.IP").
`auth`	List. Authentication details from 'ig_auth()', including 'cst', 'security', 'base_url', 'api_key', and 'acc_number'.
`mock_response`	List or data frame. Optional mock response for testing, bypassing the API call.

Value

A tibble with price information including makret status, bid, offer, high, low, and update time.

Examples

```
## Not run:
# Authenticate and get price
auth <- ig_auth(
  username = "your_username",
  password = "your_password",
  api_key = "your_api_key",
  acc_type = "DEMO",
  acc_number = "ABC123"
)
price <- ig_get_price("CS.D.USDCHF.CFD.IP", auth)
print(price)

# Using mock response for testing
mock_response <- list(
  snapshot = data.frame(
    marketStatus = "TRADEABLE",
    bid = 0.798,
    offer = 0.798,
    high = 0.801,
    low = 0.797,
    updateTime = "2025/09/26 21:58:57",
    binaryOdds = NA
  )
)
price <- ig_get_price("CS.D.USDCHF.CFD.IP", auth, mock_response = mock_response)

## End(Not run)
```

`ig_search_markets` *Search markets*

Description

Search markets by text query. Returns a tibble of matching markets from the IG API.

Usage

```
ig_search_markets(query, auth, mock_response = NULL)
```

Arguments

`query`	Character. Search string for markets (e.g., "USD/CHF").
`auth`	List. Authentication details from ‘ig_auth()‘, including ‘cst‘, ‘security‘, ‘base_url‘, ‘api_key‘, and ‘acc_number‘.
`mock_response`	List or data frame. Optional mock response for testing, bypassing the API call.

Value

A tibble with market information including epic, instrument name and market status.

Examples

```
## Not run:
# Authenticate and search markets
auth <- ig_auth(
  username = "your_username",
  password = "your_password",
  api_key = "your_api_key",
  acc_type = "DEMO",
  acc_number = "ABC123"
)
markets <- ig_search_markets("USD/CHF", auth)
print(markets)
```

```
# Using mock response
mock_response <- data.frame(
  epic = "CS.D.USDCHF.MINI.IP",
  instrumentName = "USD/CHF Mini",
  marketStatus = "OPEN"
)
markets <- ig_search_markets("USD/CHF", auth, mock_response = mock_response)

## End(Not run)
```

Index