Package 'okxr'

April 28, 2026

Title R Interface to the 'OKX' REST API

Version 0.2.4

Author Oliver Zhou [aut, cre]

Maintainer Oliver Zhou <oliver.yxzhou@gmail.com>

Description Provides lightweight R wrappers for the 'OKX' REST API, covering endpoints for market data, trading, account management, asset balances, and copy trading. The upstream API reference is available at <https://www.okx.com/docs-v5/en/>.

License MIT + file LICENSE

Encoding UTF-8

Language en-US

Imports rlang, httr, jsonlite, digest, base64enc, data.table

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

Collate okxr-package.R def_constants.R utils_request_helpers.R utils_auth.R utils_parser.R utils_get_generator.R utils_post_generator.R utils_labels.R wrappers_get_market.R wrappers_get_asset.R wrappers_get_account.R wrappers_get_trade.R wrappers_get_copy_trade.R wrappers_post_configure.R wrappers_post_trade.R

URL https://github.com/OliverLDS/okxr

BugReports https://github.com/OliverLDS/okxr/issues

RoxygenNote 7.2.3

NeedsCompilation no

Repository CRAN

Date/Publication 2026-04-28 19:10:02 UTC

Contents

`okxr-package` *okxr: R Interface to the OKX REST API*

Description

'okxr' provides lightweight wrappers for selected OKX REST API endpoints, including market data, account information, asset metadata, order-book trade queries, trading actions, and copy-trading endpoints.

Details

Public market and public reference endpoints can be called without credentials. Private account, asset, trade, and copy-trading endpoints require an OKX API credential list with 'api_key', 'secret_key', and 'passphrase' entries.

If 'config$demo' is 'TRUE', signed requests include OKX's simulated trading header. Request timeout defaults to 10 seconds and can be set globally with 'set_okxr_options(timeout = 15)' or per request with 'config$timeout'.

By default, wrappers return parsed 'data.table' objects. Use 'set_okxr_options(raw_data = TRUE)' to return raw API 'data' payloads instead.

Network failures, request timeouts, OKX error responses, or empty API 'data' payloads may return 'NULL' with a warning. Live API examples are intentionally non-running because they require credentials, network access, and may have account-specific side effects.

Author(s)

Maintainer: Oliver Zhou <oliver.yxzhou@gmail.com>

See Also

[set_okxr_options()], [get_market_candles()], [post_trade_order()]

`.build_request` *Build a full OKX request object*

Description

Assemble URL, headers, and body for an OKX API call.

Usage

```
.build_request(
  httr_method,
  base_url,
  api_path,
  query_string,
  config = NULL,
  body_json = "",
  auth = TRUE
)
```

Arguments

httr_method	HTTP method ("GET" or "POST").
base_url	Base URL of the OKX API.
api_path	API path (e.g., "/api/v5/account/balance").
query_string	Query string starting with "?" or empty.
config	List with API credentials.
body_json	Optional JSON string for POST body.
auth	Logical. Whether to sign the request with OKX credentials.

Value

A list with elements: ‘method‘, ‘url‘, ‘full_path‘, ‘headers‘, and ‘body_json‘.

.execute_get_action *Execute a GET request to OKX*

Description

Perform a signed GET call to the OKX API.

Usage

```
.execute_get_action(api_path, query_string, config = NULL, auth = TRUE)
```

Arguments

api_path	API path (e.g., "/api/v5/account/balance").
query_string	Query string starting with "?" or empty.
config	List with API credentials.
auth	Logical. Whether to sign the request with OKX credentials.

Value

An ‘httr‘ response object, or ‘NULL‘ if the request fails.

.execute_post_action *Execute a POST request to OKX*

Description

Perform a signed POST call to the OKX API.

Usage

```
.execute_post_action(api_path, body_list, config)
```

Arguments

api_path	API path (e.g., "/api/v5/trade/order").
body_list	List to be converted to JSON for the request body.
config	List with API credentials.

Value

An ‘httr‘ response object, or ‘NULL‘ if the request fails.

.get_headers *Create OKX API request headers*

Description

Generate the required signed headers for an OKX REST request.

Usage

```
.get_headers(config, httr_method, httr_path, body_json = "")
```

Arguments

config	List with ‘api_key‘, ‘secret_key‘, and ‘passphrase‘.
httr_method	HTTP method as a string (e.g., "GET" or "POST").
httr_path	Path portion of the API endpoint.
body_json	Optional JSON string of the request body.

Value

A ‘httr::add_headers‘ object containing the signed headers.

`get_account_balance` *Get account balance*

Description

Retrieve account-level margin and equity information for your OKX account.

Usage

```
get_account_balance(config, tz = .okx_default_tz)
```

Arguments

`config`	List. API credentials/config, typically containing ‘api_key‘, ‘secret_key‘, and ‘passphrase‘. May also include ‘base_url‘.
`tz`	Character. Time zone for parsing timestamps. Default ‘"Asia/Hong_Kong"‘.

Details

This wraps ‘/api/v5/account/balance‘. Returns one row per account-level equity snapshot. Timestamps are parsed into ‘POSIXct‘ in the given ‘tz‘.

Value

A ‘data.frame‘ with account balance and margin metrics (e.g., ‘totalEq‘, ‘isoEq‘, ‘adjEq‘, ‘availEq‘, ‘ordFroz‘, ‘imr‘, ‘mmr‘, ‘upl‘, ‘mgnRatio‘, ...). Timestamp columns (‘uTime‘) are ‘POSIXct‘.

Note

Since okxr 0.1.1

See Also

[get_account_positions()], [get_account_leverage_info()]

Other okxr-account: `get_account_config()`, `get_account_leverage_info()`, `get_account_positions_history()`, `get_account_positions()`

Examples

```
## Not run:
cfg <- list(api_key = "xxx", secret_key = "xxx", passphrase = "xxx")
bal <- get_account_balance(config = cfg)
head(bal)

## End(Not run)
```

`get_account_bills` *Get account bills*

Description

Retrieve account bill details from the last 7 days.

Usage

```
get_account_bills(
  inst_type = NULL,
  ccy = NULL,
  mgn_mode = NULL,
  ct_type = NULL,
  type = NULL,
  sub_type = NULL,
  after = NULL,
  before = NULL,
  limit = NULL,
  config,
  tz = .okx_default_tz
)
```

Arguments

`inst_type`	Character or ‘NULL‘. Instrument type filter.
`ccy`	Character or ‘NULL‘. Currency filter.
`mgn_mode`	Character or ‘NULL‘. Margin mode filter.
`ct_type`	Character or ‘NULL‘. Contract type filter.
`type`	Character or ‘NULL‘. Bill type filter.
`sub_type`	Character or ‘NULL‘. Bill subtype filter.
`after`	Character or ‘NULL‘. Pagination cursor for earlier records.
`before`	Character or ‘NULL‘. Pagination cursor for newer records.
`limit`	Integer or ‘NULL‘. Number of rows to request.
`config`	List. API credentials/config.
`tz`	Character. Time zone for parsing timestamps. Default ‘"Asia/Hong_Kong"‘.

Value

A ‘data.frame‘ with account bill rows.

get_account_bills_archive *Get archived account bills*

Description

Retrieve archived account bill details.

Usage

```
get_account_bills_archive(
  inst_type = NULL,
  ccy = NULL,
  mgn_mode = NULL,
  ct_type = NULL,
  type = NULL,
  sub_type = NULL,
  after = NULL,
  before = NULL,
  limit = NULL,
  config,
  tz = .okx_default_tz
)
```

Arguments

inst_type	Character or 'NULL'. Instrument type filter.
ccy	Character or 'NULL'. Currency filter.
mgn_mode	Character or 'NULL'. Margin mode filter.
ct_type	Character or 'NULL'. Contract type filter.
type	Character or 'NULL'. Bill type filter.
sub_type	Character or 'NULL'. Bill subtype filter.
after	Character or 'NULL'. Pagination cursor for earlier records.
before	Character or 'NULL'. Pagination cursor for newer records.
limit	Integer or 'NULL'. Number of rows to request.
config	List. API credentials/config.
tz	Character. Time zone for parsing timestamps. Default '"Asia/Hong_Kong"'.

Value

A 'data.frame' with archived account bill rows.

get_account_config *Get account configuration*

Description

Retrieve account-level configuration information.

Usage

```
get_account_config(config, tz = .okx_default_tz)
```

Arguments

config	List. API credentials/config, typically containing 'api_key', 'secret_key', and 'passphrase'.
tz	Character. Time zone for parsing timestamps. Default '"Asia/Hong_Kong"'.

Details

Wraps '/api/v5/account/config'. Includes account ID, mode, and position mode flags. Returns one row.

Value

A 'data.frame' with columns like 'uid', 'mainUid', 'acctLv', 'posMode', 'autoLoan', etc.

Note

Since okxr 0.1.2

See Also

[get_account_balance()], [get_account_leverage_info()]

Other okxr-account: `get_account_balance()`, `get_account_leverage_info()`, `get_account_positions_history()`, `get_account_positions()`

Examples

```
## Not run:
cfg <- list(api_key = "xxx", secret_key = "xxx", passphrase = "xxx")
cfg_info <- get_account_config(config = cfg)
cfg_info

## End(Not run)
```

`get_account_leverage_info` *Get account leverage settings*

Description

Retrieve leverage configuration for a given instrument and margin mode.

Usage

```
get_account_leverage_info(inst_id, mgn_mode, config, tz = .okx_default_tz)
```

Arguments

`inst_id`	Character. Instrument ID, e.g. '"BTC-USDT"'.
`mgn_mode`	Character. Margin mode. One of '"cross"' or '"isolated"'.
`config`	List. API credentials/config, typically containing 'api_key', 'secret_key', and 'passphrase'.
`tz`	Character. Time zone for parsing timestamps. Default '"Asia/Hong_Kong"'.

Details

Wraps '/api/v5/account/leverage-info'. Requires both 'inst_id' and 'mgn_mode'. Returns current leverage values (numeric).

Value

A 'data.frame' with columns 'instId', 'mgnMode', 'posSide', and 'lever'.

Note

Since okxr 0.1.1

See Also

[get_account_balance()], [get_account_positions()]

Other okxr-account: `get_account_balance()`, `get_account_config()`, `get_account_positions_history()`, `get_account_positions()`

Examples

```
## Not run:
cfg <- list(api_key = "xxx", secret_key = "xxx", passphrase = "xxx")
get_account_leverage_info(
  inst_id = "BTC-USDT",
  mgn_mode = "cross",
  config = cfg
```

```
)

## End(Not run)
```

`get_account_positions` *Get account open positions*

Description

Retrieve all currently open positions under the account.

Usage

```
get_account_positions(config, tz = .okx_default_tz)
```

Arguments

`config`	List. API credentials/config, typically containing ‘api_key‘, ‘secret_key‘, and ‘passphrase‘.
`tz`	Character. Time zone for parsing timestamps. Default ‘"Asia/Hong_Kong"‘.

Details

Wraps ‘/api/v5/account/positions‘. Returns one row per open position.

Value

A ‘data.frame‘ with columns such as ‘instId‘, ‘posId‘, ‘posSide‘, ‘pos‘, ‘lever‘, ‘avgPx‘, ‘markPx‘, ‘upl‘, ‘realizedPnl‘, etc. Timestamps (‘cTime‘, ‘uTime‘) are ‘POSIXct‘.

Note

Since okxr 0.1.1

See Also

[get_account_balance()], [get_account_positions_history()]

Other okxr-account: `get_account_balance()`, `get_account_config()`, `get_account_leverage_info()`, `get_account_positions_history()`

Examples

```
## Not run:
cfg <- list(api_key = "xxx", secret_key = "xxx", passphrase = "xxx")
pos <- get_account_positions(config = cfg)
pos

## End(Not run)
```

`get_account_positions_history` *Get account position history*

Description

Retrieve historical records of closed or adjusted positions.

Usage

```
get_account_positions_history(config, tz = .okx_default_tz)
```

Arguments

`config`	List. API credentials/config, typically containing ‘api_key‘, ‘secret_key‘, and ‘passphrase‘.
`tz`	Character. Time zone for parsing timestamps. Default ‘"Asia/Hong_Kong"‘.

Details

Wraps ‘/api/v5/account/positions-history‘. Includes closed positions and their realized PnL. Returns one row per historical record.

Value

A ‘data.frame‘ with columns such as ‘instId‘, ‘posId‘, ‘posSide‘, ‘pos‘, ‘lever‘, ‘realizedPnl‘, ‘fee‘, plus timestamp fields (‘cTime‘, ‘uTime‘).

Note

Since okxr 0.1.1

See Also

[get_account_positions()]

Other okxr-account: `get_account_balance()`, `get_account_config()`, `get_account_leverage_info()`, `get_account_positions()`

Examples

```
## Not run:
cfg <- list(api_key = "xxx", secret_key = "xxx", passphrase = "xxx")
hist <- get_account_positions_history(config = cfg)
tail(hist)

## End(Not run)
```

get_asset_balances *Get asset balances*

Description

Retrieves the available, total, and frozen balance for each asset in the account.

Usage

```
get_asset_balances(config, tz = .okx_default_tz)
```

Arguments

config	API credentials as a list with api_key, secret_key, and passphrase.
tz	Timezone string (default: "Asia/Hong_Kong").

Value

A data.frame with balances per currency.

get_asset_currencies *Get funding currencies*

Description

Retrieve currencies available to the current account.

Usage

```
get_asset_currencies(ccy = NULL, config, tz = .okx_default_tz)
```

Arguments

ccy	Character or 'NULL'. Single currency or comma-separated currencies.
config	API credentials as a list with api_key, secret_key, and passphrase.
tz	Timezone string (default: "Asia/Hong_Kong").

Value

A data.frame with currency and chain metadata.

get_asset_deposit_address

Get deposit address

Description

Retrieve deposit addresses for a currency.

Usage

```
get_asset_deposit_address(ccy, config, tz = .okx_default_tz)
```

Arguments

ccy	Character. Currency, e.g. "BTC" or "USDT".
config	API credentials as a list with api_key, secret_key, and passphrase.
tz	Timezone string (default: "Asia/Hong_Kong").

Value

A data.frame with deposit address rows.

get_asset_deposit_history

Get asset deposit history

Description

Retrieves a record of all asset deposits made to your account.

Usage

```
get_asset_deposit_history(config, tz = .okx_default_tz)
```

Arguments

config	API credentials as a list with api_key, secret_key, and passphrase.
tz	Timezone string (default: "Asia/Hong_Kong").

Value

A data.frame with deposit timestamps, amounts, and currencies.

get_asset_withdrawal_history

Get asset withdrawal history

Description

Retrieves a record of all asset withdrawals from your account.

Usage

```
get_asset_withdrawal_history(config, tz = .okx_default_tz)
```

Arguments

config	API credentials as a list with api_key, secret_key, and passphrase.
tz	Timezone string (default: "Asia/Hong_Kong").

Value

A data.frame with withdrawal timestamps, amounts, and currencies.

get_copy_trade_current_subpos

Get current copy trading subpositions

Description

Retrieve your currently active subpositions under copy trading.

Usage

```
get_copy_trade_current_subpos(config, tz = .okx_default_tz)
```

Arguments

config	List. API credentials/config, typically containing ‘api_key‘, ‘secret_key‘, and ‘passphrase‘.
tz	Character. Time zone for parsing timestamps. Default ‘"Asia/Hong_Kong"‘.

Details

Wraps ‘/api/v5/copytrading/current-subpositions‘. Returns one row per subposition, associated with the relevant lead trader.

Value

A ‘data.frame‘ with fields like ‘instId‘ and ‘uniqueCode‘.

Note

Since okxr 0.1.2

See Also

[get_copy_trade_historical_subpos()]

Other okxr-copytrading: `get_copy_trade_historical_subpos()`, `get_copy_trade_my_leaders()`, `get_copy_trade_settings()`

Examples

```
## Not run:
cfg <- list(api_key = "xxx", secret_key = "xxx", passphrase = "xxx")
get_copy_trade_current_subpos(config = cfg)

## End(Not run)
```

`get_copy_trade_historical_subpos`

Get historical copy trading subpositions

Description

Retrieve your historical copy trading subpositions.

Usage

```
get_copy_trade_historical_subpos(config, tz = .okx_default_tz)
```

Arguments

`config`	List. API credentials/config, typically containing ‘api_key‘, ‘secret_key‘, and ‘passphrase‘.
`tz`	Character. Time zone for parsing timestamps. Default ‘"Asia/Hong_Kong"‘.

Details

Wraps ‘/api/v5/copytrading/subpositions-history‘. Returns one row per closed or historical subposition.

Value

A ‘data.frame‘ with fields like ‘instId‘ and ‘uniqueCode‘.

Note

Since okxr 0.1.2

See Also

[get_copy_trade_current_subpos()]

Other okxr-copytrading: `get_copy_trade_current_subpos()`, `get_copy_trade_my_leaders()`, `get_copy_trade_settings()`

Examples

```
## Not run:
cfg <- list(api_key = "xxx", secret_key = "xxx", passphrase = "xxx")
hist <- get_copy_trade_historical_subpos(config = cfg)
head(hist)

## End(Not run)
```

---

get_copy_trade_my_leaders *Get my lead traders*

Description

Retrieve the list of lead traders you are currently copying.

Usage

```
get_copy_trade_my_leaders(
  inst_type = NULL,
  config,
  tz = .okx_default_tz,
  instType = inst_type
)
```

Arguments

`inst_type`	Character or ‘NULL‘. Filter by instrument type (e.g., ‘"SWAP"‘, ‘"MARGIN"‘, ‘"SPOT"‘). If ‘NULL‘, returns all.
`config`	List. API credentials/config, typically containing ‘api_key‘, ‘secret_key‘, and ‘passphrase‘.
`tz`	Character. Time zone for parsing timestamps. Default ‘"Asia/Hong_Kong"‘.
`instType`	Deprecated alias for ‘inst_type‘.

Details

Wraps '/api/v5/copytrading/current-lead-traders'. Returns one row per lead trader followed by your account.

Value

A 'data.frame' with fields such as 'nickName' and 'uniqueCode'.

Note

Since okxr 0.1.2

See Also

[get_copy_trade_settings()], [get_copy_trade_current_subpos()]

Other okxr-copytrading: `get_copy_trade_current_subpos()`, `get_copy_trade_historical_subpos()`, `get_copy_trade_settings()`

Examples

```
## Not run:
cfg <- list(api_key = "xxx", secret_key = "xxx", passphrase = "xxx")
get_copy_trade_my_leaders(inst_type = "SWAP", config = cfg)

## End(Not run)
```

---

get_copy_trade_settings

Get copy trading settings

Description

Retrieve your account's copy trading configuration.

Usage

```
get_copy_trade_settings(
  unique_code,
  config,
  tz = .okx_default_tz,
  uniqueCode = unique_code
)
```

Arguments

unique_code	Character. Lead trader unique code.
config	List. API credentials/config, typically containing ‘api_key‘, ‘secret_key‘, and ‘passphrase‘.
tz	Character. Time zone for parsing timestamps. Default ‘"Asia/Hong_Kong"‘.
uniqueCode	Deprecated alias for ‘unique_code‘.

Details

Wraps ‘/api/v5/copytrading/copy-settings‘. Returns one row with the current copy mode and copy state for the given ‘unique_code‘.

Value

A ‘data.frame‘ with fields like ‘copyMode‘ and ‘copyState‘.

Note

Since okxr 0.1.2

See Also

[get_copy_trade_my_leaders()], [get_copy_trade_current_subpos()]

Other okxr-copytrading: `get_copy_trade_current_subpos()`, `get_copy_trade_historical_subpos()`, `get_copy_trade_my_leaders()`

Examples

```
## Not run:
cfg <- list(api_key = "xxx", secret_key = "xxx", passphrase = "xxx")
get_copy_trade_settings(unique_code = "1129E65755274C36", config = cfg)

## End(Not run)
```

---

get_market_books *Get order book*

Description

Retrieve the current order book for an instrument.

Usage

```
get_market_books(inst_id, sz = NULL, config = NULL, tz = .okx_default_tz)
```

Arguments

inst_id	Character. Instrument ID, e.g. '"BTC-USDT"'.
sz	Integer or 'NULL'. Order book depth. If 'NULL', OKX uses its endpoint default.
config	Optional list. Public endpoint request options, such as 'timeout'; credentials are not required.
tz	Character. Time zone for parsing timestamps. Default '"Asia/Hong_Kong"'.

Value

A 'data.frame' with JSON-encoded 'asks' and 'bids' columns plus 'ts'.

---

get_market_candles *Get recent market candles*

Description

Retrieve the latest candlestick data for a given instrument and bar size.

Usage

```
get_market_candles(
  inst_id,
  bar,
  limit = 100L,
  config = NULL,
  tz = .okx_default_tz,
  standardize_names = TRUE
)
```

Arguments

inst_id	Character. Instrument ID, e.g. '"BTC-USDT"', '"ETH-USDT-SWAP"'.
bar	Character. Candlestick granularity, e.g. '"1m"', '"5m"', '"1H"', '"1D"'.
limit	Integer. Number of bars to retrieve. Default '100L'.
config	Optional list. Public endpoint request options, such as 'timeout'; credentials are not required.
tz	Character. Time zone for parsing timestamps. Default '"Asia/Hong_Kong"'.
standardize_names	Logical. If 'TRUE' (default), renames columns to 'timestamp', 'open', 'high', 'low', 'close', 'volume', 'volQuote'.

Details

Wraps '/api/v5/market/candles'. Returns up to 'limit' bars, sorted by timestamp. Candlestick fields can be standardized to common OHLCV names via 'standardize_names = TRUE'.

Value

A ‘data.frame‘ with columns including ‘timestamp‘, ‘open‘, ‘high‘, ‘low‘, ‘close‘, ‘volume‘, and ‘volQuote‘. Timestamps are ‘POSIXct‘ in ‘tz‘.

Note

Since okxr 0.1.1

See Also

[get_market_history_candles()], [get_public_mark_price()]

Other okxr-market: `get_market_history_candles()`, `get_public_instruments()`, `get_public_mark_price()`

Examples

```
## Not run:
cfg <- list(api_key = "xxx", secret_key = "xxx", passphrase = "xxx")
get_market_candles("BTC-USDT", bar = "5m", limit = 50, config = cfg)

## End(Not run)
```

---

get_market_history_candles

Get historical market candles

Description

Retrieve candlestick data before a specific datetime.

Usage

```
get_market_history_candles(
  inst_id,
  bar,
  before = NULL,
  limit = 100L,
  config = NULL,
  tz = .okx_default_tz,
  standardize_names = TRUE
)
```

Arguments

inst_id	Character. Instrument ID, e.g. ‘"BTC-USDT"‘.
bar	Character. Candlestick granularity, e.g. ‘"1m"‘, ‘"5m"‘, ‘"1H"‘.
before	Character or ‘NULL‘. Timestamp string in format ‘"%Y-%m-%d %H:%M:%S"‘. If ‘NULL‘ (default), fetches most recent history.
limit	Integer. Number of bars to retrieve. Default ‘100L‘.
config	Optional list. Public endpoint request options, such as ‘timeout‘; credentials are not required.
tz	Character. Time zone for parsing timestamps. Default ‘"Asia/Hong_Kong"‘.
standardize_names	Logical. If ‘TRUE‘ (default), renames columns to ‘timestamp‘, ‘open‘, ‘high‘, ‘low‘, ‘close‘, ‘volume‘, ‘volQuote‘.

Details

Wraps ‘/api/v5/market/history-candles‘. If ‘before‘ is supplied, it is converted to milliseconds since epoch (in ‘tz‘) and sent as ‘after=...‘ (per OKX semantics: *return data before this time*).

Value

A ‘data.frame‘ of candlestick bars with standardized column names if ‘standardize_names = TRUE‘. Timestamps are ‘POSIXct‘ in ‘tz‘.

Note

Since okxr 0.1.1

See Also

[get_market_candles()]

Other okxr-market: get_market_candles(), get_public_instruments(), get_public_mark_price()

Examples

```
## Not run:
cfg <- list(api_key = "xxx", secret_key = "xxx", passphrase = "xxx")
get_market_history_candles(
  "ETH-USDT-SWAP", bar = "1H",
  before = "2025-08-20 00:00:00", config = cfg
)

## End(Not run)
```

get_market_history_trades *Get historical public trades*

Description

Retrieve public trade history for an instrument.

Usage

```
get_market_history_trades(
  inst_id,
  type = NULL,
  after = NULL,
  before = NULL,
  limit = NULL,
  config = NULL,
  tz = .okx_default_tz
)
```

Arguments

inst_id	Character. Instrument ID, e.g. '"BTC-USDT"'.
type	Character or 'NULL'. Pagination type, using OKX values.
after	Character or 'NULL'. Pagination cursor for earlier records.
before	Character or 'NULL'. Pagination cursor for newer records.
limit	Integer or 'NULL'. Number of rows to request.
config	Optional list. Public endpoint request options, such as 'timeout'; credentials are not required.
tz	Character. Time zone for parsing timestamps. Default '"Asia/Hong_Kong"'.

Value

A 'data.frame' with historical public trades.

get_market_ticker *Get market ticker*

Description

Retrieve the latest ticker snapshot for a specific instrument.

Usage

```
get_market_ticker(inst_id, config = NULL, tz = .okx_default_tz)
```

Arguments

inst_id	Character. Instrument ID, e.g. ‘"BTC-USDT"‘ or ‘"ETH-USDT-SWAP"‘.
config	Optional list. Public endpoint request options, such as ‘timeout‘; credentials are not required.
tz	Character. Time zone for parsing timestamps. Default ‘"Asia/Hong_Kong"‘.

Value

A ‘data.frame‘ with the latest ticker fields returned by OKX.

---

get_market_tickers *Get market tickers*

Description

Retrieve ticker snapshots for all instruments under an instrument type.

Usage

```
get_market_tickers(
  inst_type,
  uly = NULL,
  inst_family = NULL,
  config = NULL,
  tz = .okx_default_tz
)
```

Arguments

inst_type	Character. Instrument type, e.g. ‘"SPOT"‘, ‘"SWAP"‘, ‘"FUTURES"‘, or ‘"OPTION"‘.
uly	Character or ‘NULL‘. Underlying. Optional filter for derivatives.
inst_family	Character or ‘NULL‘. Instrument family. Optional filter for derivatives and options.
config	Optional list. Public endpoint request options, such as ‘timeout‘; credentials are not required.
tz	Character. Time zone for parsing timestamps. Default ‘"Asia/Hong_Kong"‘.

Value

A ‘data.frame‘ with one row per ticker.

get_market_trades *Get recent public trades*

Description

Retrieve recent public trades for an instrument.

Usage

```
get_market_trades(inst_id, limit = NULL, config = NULL, tz = .okx_default_tz)
```

Arguments

inst_id	Character. Instrument ID, e.g. '"BTC-USDT"'.
limit	Integer or 'NULL'. Number of rows to request.
config	Optional list. Public endpoint request options, such as 'timeout'; credentials are not required.
tz	Character. Time zone for parsing timestamps. Default '"Asia/Hong_Kong"'.

Value

A 'data.frame' with recent public trades.

get_public_funding_rate *Get current funding rate*

Description

Retrieve the current funding rate for a perpetual swap instrument.

Usage

```
get_public_funding_rate(inst_id, config = NULL, tz = .okx_default_tz)
```

Arguments

inst_id	Character. Instrument ID, e.g. '"BTC-USDT-SWAP"'.
config	Optional list. Public endpoint request options, such as 'timeout'; credentials are not required.
tz	Character. Time zone for parsing timestamps. Default '"Asia/Hong_Kong"'.

Value

A 'data.frame' containing the current funding rate fields returned by OKX.

get_public_funding_rate_history

Get funding rate history

Description

Retrieve historical funding rate entries for a perpetual swap instrument.

Usage

```
get_public_funding_rate_history(
  inst_id,
  before = NULL,
  after = NULL,
  limit = 400,
  config = NULL,
  tz = .okx_default_tz
)
```

Arguments

inst_id	Character. Instrument ID, e.g. '"BTC-USDT-SWAP"'.
before	Optional cursor for records earlier than the supplied value.
after	Optional cursor for records later than the supplied value.
limit	Integer. Number of records to request. Default '400'.
config	Optional list. Public endpoint request options, such as 'timeout'; credentials are not required.
tz	Character. Time zone for parsing timestamps. Default '"Asia/Hong_Kong"'.

Value

A 'data.frame' containing funding rate history rows returned by OKX.

get_public_instruments

Get instrument metadata

Description

Retrieve metadata for instruments of a given type.

Usage

```
get_public_instruments(
  inst_id = NULL,
  inst_type = "SWAP",
  config = NULL,
  tz = .okx_default_tz
)
```

Arguments

inst_id	Character or 'NULL'. Specific instrument ID to query. Use 'NULL' to fetch all instruments of 'inst_type'.
inst_type	Character. Instrument type. One of '"SPOT"', '"MARGIN"', '"SWAP"' (default), '"FUTURES"', '"OPTION"'.
config	Optional list. Public endpoint request options, such as 'timeout'; credentials are not required.
tz	Character. Time zone for parsing timestamps. Default '"Asia/Hong_Kong"'.

Details

Wraps '/api/v5/public/instruments'. Returns one row per instrument, including contract specifications, tick size, lot size, expiry, and state.

Value

A 'data.frame' with instrument metadata (e.g., 'instType', 'instId', 'uly', 'baseCcy', 'quoteCcy', 'settleCcy', 'ctVal', 'ctMult', 'tickSz', 'lotSz', 'minSz', 'expTime', 'lever', 'state', ...).

Note

Since okxr 0.1.2

See Also

[get_public_mark_price()]

Other okxr-market: get_market_candles(), get_market_history_candles(), get_public_mark_price()

Examples

```
## Not run:
cfg <- list(api_key = "xxx", secret_key = "xxx", passphrase = "xxx")
# Get metadata for all SWAP instruments
df <- get_public_instruments(inst_type = "SWAP", config = cfg)

# Get metadata for one instrument
get_public_instruments("ETH-USDT-SWAP", inst_type = "SWAP", config = cfg)

## End(Not run)
```

`get_public_mark_price` *Get current mark price*

Description

Retrieve the current mark price for a given instrument.

Usage

```
get_public_mark_price(
  inst_id,
  inst_type = "SWAP",
  config = NULL,
  tz = .okx_default_tz
)
```

Arguments

`inst_id`	Character. Instrument ID, e.g. '"BTC-USDT"', '"ETH-USDT-SWAP"'.
`inst_type`	Character. Instrument type. One of '"SPOT"', '"MARGIN"', '"SWAP"' (default), '"FUTURES"', '"OPTION"'.
`config`	Optional list. Public endpoint request options, such as 'timeout'; credentials are not required.
`tz`	Character. Time zone for parsing timestamps. Default '"Asia/Hong_Kong"'.

Details

Wraps '/api/v5/public/mark-price'. Useful for margin calculations and PnL estimation. Returns a single row with the latest mark price and timestamp.

Value

A 'data.frame' with columns 'timestamp', 'instId', 'markPx'.

Note

Since okxr 0.1.1

See Also

[get_public_instruments()]

Other okxr-market: `get_market_candles()`, `get_market_history_candles()`, `get_public_instruments()`

Examples

```
## Not run:
cfg <- list(api_key = "xxx", secret_key = "xxx", passphrase = "xxx")
get_public_mark_price("BTC-USDT", inst_type = "SWAP", config = cfg)

## End(Not run)
```

`get_public_open_interest` *Get open interest*

Description

Retrieve current open interest for an instrument.

Usage

```
get_public_open_interest(
  inst_id,
  inst_type,
  config = NULL,
  tz = .okx_default_tz
)
```

Arguments

`inst_id`	Character. Instrument ID, e.g. ‘"BTC-USDT-SWAP"‘.
`inst_type`	Character. Instrument type such as ‘"SWAP"‘ or ‘"FUTURES"‘.
`config`	Optional list. Public endpoint request options, such as ‘timeout‘; credentials are not required.
`tz`	Character. Time zone for parsing timestamps. Default ‘"Asia/Hong_Kong"‘.

Value

A ‘data.frame‘ containing open interest fields returned by OKX.

get_public_price_limit *Get price limit*

Description

Retrieve buy and sell price limits for an instrument.

Usage

```
get_public_price_limit(inst_id, config = NULL, tz = .okx_default_tz)
```

Arguments

inst_id	Character. Instrument ID, e.g. ‘"BTC-USDT-SWAP"‘.
config	Optional list. Public endpoint request options, such as ‘timeout‘; credentials are not required.
tz	Character. Time zone for parsing timestamps. Default ‘"Asia/Hong_Kong"‘.

Value

A ‘data.frame‘ with price limit fields.

get_public_time *Get OKX system time*

Description

Retrieve OKX system time.

Usage

```
get_public_time(config = NULL, tz = .okx_default_tz)
```

Arguments

config	Optional list. Public endpoint request options, such as ‘timeout‘; credentials are not required.
tz	Character. Time zone for parsing timestamps. Default ‘"Asia/Hong_Kong"‘.

Value

A one-row ‘data.frame‘ with system time.

`get_trade_fills` *Get trade fills*

Description

Retrieve recently filled transaction details from the last 3 days.

Usage

```
get_trade_fills(
  inst_type = NULL,
  inst_family = NULL,
  inst_id = NULL,
  ord_id = NULL,
  sub_type = NULL,
  after = NULL,
  before = NULL,
  limit = NULL,
  config,
  tz = .okx_default_tz
)
```

Arguments

`inst_type`	Character or ‘NULL‘. Instrument type filter.
`inst_family`	Character or ‘NULL‘. Instrument family filter.
`inst_id`	Character or ‘NULL‘. Instrument ID filter.
`ord_id`	Character or ‘NULL‘. Order ID filter.
`sub_type`	Character or ‘NULL‘. Transaction subtype filter.
`after`	Character or ‘NULL‘. Pagination cursor for earlier records.
`before`	Character or ‘NULL‘. Pagination cursor for newer records.
`limit`	Integer or ‘NULL‘. Number of rows to request.
`config`	List. API credentials/config.
`tz`	Character. Time zone for parsing timestamps. Default ‘"Asia/Hong_Kong"‘.

Value

A ‘data.frame‘ with fill rows.

`get_trade_fills_history` *Get trade fills history*

Description

Retrieve historical filled transaction details from the last 3 months.

Usage

```
get_trade_fills_history(
  inst_type,
  inst_family = NULL,
  inst_id = NULL,
  ord_id = NULL,
  sub_type = NULL,
  after = NULL,
  before = NULL,
  limit = NULL,
  config,
  tz = .okx_default_tz
)
```

Arguments

`inst_type`	Character. Instrument type, e.g. ‘"SPOT"‘ or ‘"SWAP"‘.
`inst_family`	Character or ‘NULL‘. Instrument family filter.
`inst_id`	Character or ‘NULL‘. Instrument ID filter.
`ord_id`	Character or ‘NULL‘. Order ID filter.
`sub_type`	Character or ‘NULL‘. Transaction subtype filter.
`after`	Character or ‘NULL‘. Pagination cursor for earlier records.
`before`	Character or ‘NULL‘. Pagination cursor for newer records.
`limit`	Integer or ‘NULL‘. Number of rows to request.
`config`	List. API credentials/config.
`tz`	Character. Time zone for parsing timestamps. Default ‘"Asia/Hong_Kong"‘.

Value

A ‘data.frame‘ with historical fill rows.

`get_trade_order` *Get trade order details*

Description

Retrieve detailed information about a specific OKX order by either the exchange-assigned ‘ord_id‘ or your client-assigned ‘cl_ord_id‘.

Usage

```
get_trade_order(
  inst_id,
  ord_id = NULL,
  cl_ord_id = NULL,
  config,
  tz = .okx_default_tz
)
```

Arguments

`inst_id`	Character. Instrument ID, e.g. ‘"ETH-USDT-SWAP"‘ (perps), ‘"BTC-USDT"‘ (spot), or ‘"BTC-USD-240927"‘ (dated futures).
`ord_id`	Character, optional. The OKX order ID. Provide this **or** ‘cl_ord_id‘.
`cl_ord_id`	Character, optional. Your client order ID. Provide this **or** ‘ord_id‘.
`config`	List. API credentials/config, typically containing ‘api_key‘, ‘secret_key‘, and ‘passphrase‘. May also include ‘base_url‘.
`tz`	Character. Time zone for parsing timestamps. Default ‘"Asia/Hong_Kong"‘.

Details

You must provide exactly one identifier: ‘ord_id‘ **or** ‘cl_ord_id‘. Timestamps in the response are converted to ‘POSIXct‘ in the supplied ‘tz‘.

Value

A ‘data.frame‘ (one row) with order details following OKX schema (e.g., ‘ordId‘, ‘clOrdId‘, ‘instId‘, ‘ordType‘, ‘px‘, ‘sz‘, ‘side‘, ‘posSide‘, ‘tdMode‘, ‘accFillSz‘, ‘fillPx‘, ‘fillSz‘, ‘fillTime‘, ‘avgPx‘, ‘state‘, ‘lever‘, etc.). Timestamp columns are ‘POSIXct‘ in ‘tz‘.

Common errors

- ‘Either ’ord_id’ or ’cl_ord_id’ must be provided.‘ (client-side) - HTTP 401 Unauthorized (missing/invalid credentials) - OKX ‘code‘ like ‘51000‘ invalid sign or ‘51603‘ order not found

Note

Since okxr 0.1.1

See Also

[get_trade_orders_pending()], [get_trade_orders_history_7d()]

Other okxr-trade: `get_trade_orders_history_7d()`, `get_trade_orders_pending()`

Examples

```
## Not run:
cfg <- list(api_key = "xxx", secret_key = "xxx", passphrase = "xxx")
get_trade_order(
  inst_id = "ETH-USDT-SWAP",
  ord_id  = "1234567890",
  config  = cfg
)

## End(Not run)
```

`get_trade_orders_history_7d`

Get trade orders history (last 7 days)

Description

Retrieve recent order history (up to ~7 days) for the specified instrument type. This wraps '/api/v5/trade/orders-history'. For older data, OKX provides an archive endpoint ('orders-history-archive') which is not covered here.

Usage

```
get_trade_orders_history_7d(inst_type = "SWAP", config, tz = .okx_default_tz)
```

Arguments

`inst_type`	Character. Instrument type. One of '"SPOT"', '"MARGIN"', '"SWAP"', '"FUTURES"', '"OPTION"'. Default '"SWAP"'.
`config`	List. API credentials/config, typically containing 'api_key', 'secret_key', and 'passphrase'. May also include 'base_url'.
`tz`	Character. Time zone for parsing timestamps (e.g. '"Asia/Hong_Kong"').

Value

A 'data.frame' with one row per historical order and columns following the OKX schema (same layout as pending orders, plus final states). Timestamp columns are 'POSIXct' in 'tz'.

Common errors

- HTTP 401 Unauthorized (missing/invalid credentials) - HTTP 400 Bad Request (invalid 'inst_type')

Note

Since okxr 0.1.2

See Also

[get_trade_order()], [get_trade_orders_pending()]

Other okxr-trade: `get_trade_orders_pending()`, `get_trade_order()`

Examples

```
## Not run:
cfg <- list(api_key = "xxx", secret_key = "xxx", passphrase = "xxx")
hist <- get_trade_orders_history_7d(inst_type = "SWAP", config = cfg, tz = "Asia/Hong_Kong")
tail(hist)

## End(Not run)
```

---

get_trade_orders_pending

Get all pending trade orders

Description

Retrieve all currently open (unfilled) orders for your OKX account.

Usage

```
get_trade_orders_pending(config, tz = .okx_default_tz)
```

Arguments

`config`	List. API credentials/config, typically containing ‘api_key‘, ‘secret_key‘, and ‘passphrase‘. May also include ‘base_url‘.
`tz`	Character. Time zone for parsing timestamps (e.g. ‘"Asia/Hong_Kong"‘).

Details

Returns one row per open order. Timestamps are parsed to ‘POSIXct‘ using ‘tz‘.

Value

A ‘data.frame‘ with one row per pending order and columns following the OKX schema (e.g., ‘cTime‘, ‘ordId‘, ‘clOrdId‘, ‘tag‘, ‘instId‘, ‘ordType‘, ‘px‘, ‘sz‘, ‘side‘, ‘posSide‘, ‘tdMode‘, ‘accFillSz‘, ‘fillPx‘, ‘fillSz‘, ‘fillTime‘, ‘avgPx‘, ‘state‘, ‘lever‘, . . .). Timestamp columns are ‘POSIXct‘.

Common errors

- HTTP 401 Unauthorized (missing/invalid credentials) - Rate limiting: HTTP 429 / OKX throttle codes

Note

Since okxr 0.1.1

See Also

[get_trade_order()], [get_trade_orders_history_7d()]

Other okxr-trade: `get_trade_orders_history_7d()`, `get_trade_order()`

Examples

```
## Not run:
cfg <- list(api_key = "xxx", secret_key = "xxx", passphrase = "xxx")
df <- get_trade_orders_pending(config = cfg, tz = "Asia/Hong_Kong")
head(df)

## End(Not run)
```

---

get_var_label *Retrieve variable labels from OKX data frames*

Description

Returns human-readable labels attached to a data frame produced by OKX API parsers. You can retrieve all labels or a label for a specific variable by name or index.

Usage

```
get_var_label(df, var = NULL, default = NA_character_)
```

Arguments

`df`	A `data.frame` with `"var_labels"` attribute (as returned by OKX API parsers).
`var`	Optional variable name (character) or index (numeric). If `NULL`, all labels are returned.
`default`	Value to return if the variable has no label (default: `NA_character_`).

Value

A character label if `var` is provided, or a named character vector of all labels if `var = NULL`.

Examples

```
df <- data.frame(ordId = "123", px = 10)
attr(df, "var_labels") <- c(ordId = "Order ID", px = "Price")

get_var_label(df, "ordId")
get_var_label(df, 2)
get_var_label(df)
```

`post_account_set_leverage` *Set Account Leverage*

Description

Sets the leverage level for a specific trading instrument and margin mode.

Usage

```
post_account_set_leverage(
  inst_id,
  lever,
  mgn_mode,
  pos_side = NULL,
  tz = .okx_default_tz,
  config
)
```

Arguments

`inst_id`	Instrument ID (e.g., `"BTC-USDT"`).
`lever`	Leverage level to apply (as a string or numeric, e.g., `"10"`).
`mgn_mode`	Margin mode: `"cross"` or `"isolated"`.
`pos_side`	Optional. Position side: `"long"` or `"short"`. Required for isolated mode.
`tz`	Timezone used for any timestamp parsing (default: `"Asia/Hong_Kong"`).
`config`	A list containing API credentials: `api_key`, `secret_key`, and `passphrase`.

Value

A `data.frame` with leverage update confirmation (including instrument ID and leverage settings).

post_trade_cancel_order *Cancel a Trade Order*

Description

Submits a cancellation request for a previously placed trade order.

Usage

```
post_trade_cancel_order(inst_id, ord_id, config, tz = .okx_default_tz)
```

Arguments

inst_id	Instrument ID (e.g., "BTC-USDT").
ord_id	Order ID to cancel. Alternatively, you can modify the function to accept clOrdId.
config	A list with API credentials: api_key, secret_key, passphrase.
tz	Timezone for parsing any timestamps (default: "Asia/Hong_Kong").

Value

A data.frame containing cancellation result and timestamp.

post_trade_close_position *Close a Position*

Description

Submits a request to close a position for a given instrument and position side.

Usage

```
post_trade_close_position(
  inst_id,
  mgn_mode,
  pos_side,
  tz = .okx_default_tz,
  config
)
```

Arguments

inst_id	Instrument ID (e.g., "BTC-USDT").
mgn_mode	Margin mode: "cross" or "isolated".
pos_side	Position side to close: "long" or "short".
tz	Timezone for parsing any timestamps (default: "Asia/Hong_Kong").
config	A list with API credentials: api_key, secret_key, passphrase.

Value

A data.frame with close position confirmation details.

---

post_trade_order *Place a Trade Order*

Description

Submits a trade order to the OKX exchange.

Usage

```
post_trade_order(
  inst_id,
  td_mode,
  side,
  ord_type,
  sz,
  pos_side = NULL,
  px = NULL,
  reduce_only = NULL,
  tgt_ccy = NULL,
  cl_ord_id = NULL,
  tag = NULL,
  config,
  tz = .okx_default_tz
)
```

Arguments

inst_id	Instrument ID (e.g., "BTC-USDT").
td_mode	Trade mode: "cross" or "isolated".
side	Order side: "buy" or "sell".
ord_type	Order type: "limit", "market", etc.
sz	Size of the order (quantity to buy/sell).
pos_side	Optional. Position side: "long" or "short".

px	Optional. Price (required for limit orders).
reduce_only	Optional. Logical flag to indicate a reduce-only order.
tgt_ccy	Optional. Quote currency (e.g., "base", "quote").
cl_ord_id	Optional. Custom client order ID (auto-generated if NULL).
tag	Optional. Tag used for identifying the strategy or bot.
config	A list with API credentials: api_key, secret_key, passphrase.
tz	Timezone for parsing any timestamps (default: "Asia/Hong_Kong").

Value

A data.frame containing fields like order ID, client order ID, and timestamp.

---

set_okxr_options *Set or get okxr options*

Description

Convenience wrapper to set global options for okxr, such as whether to return raw data instead of parsed data.

Usage

```
set_okxr_options(raw_data = NULL, timeout = NULL)
```

Arguments

raw_data	Logical. Whether functions should return raw data (default = FALSE). If 'NULL', the current value is left unchanged.
timeout	Numeric. HTTP request timeout in seconds. If 'NULL', the current value is left unchanged.

Value

An invisible named list with the current package options: 'raw_data' (logical) and 'timeout' (numeric seconds). This return value can be used to inspect the effective option state after updating it.

Examples

```
old <- getOption("okxr.raw_data")
old_timeout <- getOption("okxr.timeout")
set_okxr_options(raw_data = TRUE)
set_okxr_options(timeout = 5)
options(okxr.raw_data = old, okxr.timeout = old_timeout)

set_okxr_options()  # check current values
```

Index