Package 'rgdax'

October 14, 2022

Type Package

Title Wrapper for 'Coinbase Pro (erstwhile GDAX)' Cryptocurrency Exchange

Version 1.2.1

Maintainer Dheeraj Agarwal <dheeeraj.agarwal@gmail.com>

Description Allow access to both public and private end points to Coinbase Pro (erstwhile GDAX) cryptocurrency exchange. For authenticated flow, users must have valid api, secret and passphrase to be able to connect.

URL https://github.com/DheerajAgarwal/rgdax/

BugReports https://github.com/DheerajAgarwal/rgdax/issues

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.1.1

Depends R(>= 4.0), digest, jsonlite, RCurl, httr, plyr

Suggests testthat

NeedsCompilation no

Author Dheeraj Agarwal [aut, cre]

Repository CRAN

Date/Publication 2021-08-03 04:20:02 UTC

R topics documented:

`account` *Get Account Details For An Account*

Description

This is an auth based function. User must have valid api keys generated by GDAX which must be passed as mandatory arguments. The function takes an account id as an additional input and returns the account details for that account. The account details currently include information about the currency (fiat or crypto) and the details on the balance (total, available and help for other transactions).

Usage

```
account(acct_id, api.key, secret, passphrase)
```

Arguments

`acct_id`	Mandatory character value. This is case senstivite. Must be one of the `id` generated from `accounts`
`api.key`	Mandatory character value. This is the API key as generated by GDAX. Typically a 32 character value.
`secret`	Mandatory character value. This is the API secret as generated by GDAX. Typically a 88 character value.
`passphrase`	Mandatory character value. This is the passphrase as generated by GDAX. Typically a 11 character value.

Value

Dataframe with a single row, provides the currency, the current balance, available, holds and profile_id of the user.

Examples

```
## Not run:
account(acct_id = "9kb58e27-276s-4g99-r2v7-128723948k1c",
       api.key = your_key, secret = your_api_secret, passphrase = your_api_pass)

## End(Not run)
```

`accounts` *Get Account Details For All Accounts*

Description

This is an auth based function. User must have valid api keys generated by GDAX which must be passed as mandatory arguments. The function takes no additional arguments and returns the account details for all accounts linked to that authenticated user. The account details currently include information about the currency (fiat or crypto) and the details on the balance (total, available and helpful information for other transactions). All accounts are returned even ones with zero balance.

Usage

```
accounts(api.key, secret, passphrase)
```

Arguments

`api.key`	Mandatory character value. This is the API key as generated by GDAX. Typically a 32 character value.
`secret`	Mandatory character value. This is the API secret as generated by GDAX. Typically a 88 character value.
`passphrase`	Mandatory character value. This is the passphrase as generated by GDAX. Typically a 11 character value.

Value

Dataframe with a single row for each account_id, the currency, the current balance, available, holds and profile_id of the user.

Examples

```
## Not run:
accounts(api.key = your_key, secret = your_api_secret, passphrase = your_api_pass)

## End(Not run)
```

`account_hist` *Get Account History For An Account Using Currency*

Description

This is an auth based function. User must have valid api keys generated by GDAX which must be passed as mandatory arguments. The function takes a currency as an additional input and returns the ledger for that currency. Since currency and account id have one to one mapping, the currency is being used a proxy.

Usage

```
account_hist(currency = "BTC", api.key, secret, passphrase)
```

Arguments

`currency`	Optional character value. The default is `"BTC"`. This is case insensitive and must be a valid currency as provided by `accounts` or `account`.
`api.key`	Mandatory character value. This is the API key as generated by GDAX. Typically a 32 character value.
`secret`	Mandatory character value. This is the API secret as generated by GDAX. Typically a 88 character value.
`passphrase`	Mandatory character value. This is the passphrase as generated by GDAX. Typically a 11 character value.

Value

Dataframe with account activity for that currency. It indiactes the type of activity, when the activity occured and other associated details.

Examples

```
## Not run:
account_hist(api.key = your_key, secret = your_api_secret, passphrase = your_api_pass)
account_hist("BTC", api.key = your_key, secret = your_api_secret, passphrase = your_api_pass)
account_hist("ETH", api.key = your_key, secret = your_api_secret, passphrase = your_api_pass)

## End(Not run)
```

add_order *Create a New Order For A Currency*

Description

This is an auth based function. User must have valid api keys generated by GDAX which must be passed as mandatory arguments. The users can place different types of orders like "limit", "stop" or "market". Orders will be placed succesfully only if there is sufficient funds. Each order will result in a hold and the details of the hold can be tracked using holds. Margin Orders are currently not supported.

Usage

```
add_order(
  api.key,
  secret,
  passphrase,
  product_id = "BTC-USD",
  type = "limit",
  stop = NULL,
  stop_price = NULL,
  side = "b",
  price = NULL,
  size
)
```

Arguments

api.key	Mandatory character value. This is the API key as generated by GDAX. Typically a 32 character value.
secret	Mandatory character value. This is the API secret as generated by GDAX. Typically a 88 character value.
passphrase	Mandatory character value. This is the passphrase as generated by GDAX. Typically a 11 character value.
product_id	Optional character value for the currency pair. The default is "BTC-USD". This param is case insensitive and must be one of the valid currency-pair. The list of valid currency-pairs can be fetched using public_info.
type	Optional character value for the order type. The default is "limit". This param is case insensitive and must be one of the valid order types. The current valid order types is only "limit".
stop	Optional parameter. This parameter is required if a stop order needs to be placed. Possible values apart from default NULL are "loss" or "entry". If a non-default value is provided, stop_price argument must be defined.

`stop_price`	Optional parameter.The value is needed only if `stop` is defined. Sets the trigger price for stop order. If `stop ="loss"`, the triger price is when market price is at or below the trigger. If `stop = "entry"`, the trigger price is when market price is at or above the trigger.
`side`	Optional character value for the order side The default is `"b"` which stands for buy. This param is case insensitive and must be one of either `"b"` (buy) or `"s"` (sell).
`price`	Conditional mandatory numeric value. It can either be an integer or float. Float values of greater than 2 decimals will be rounded to 2 decimals using the generic `round` function from R. The value is mandatory for `type = "limit"`.
`size`	Mandatory numeric value. It can either be an integer or float. Float values will **NOT** be rounded. The user must ensure that the fractional unit of a currency pair is valid for acceptance by GDAX. This information can also be determined by `public_info` which provides the minimun and maximum order sizes for each currency pairs.

Value

Dataframe with status of the order, posted details and created time stamp etc.

Examples

```
## Not run:
add_order("BTC-USD", api.key = your_key, secret = your_api_secret, passphrase = your_api_pass,
     type="limit", side = "s", price = 1000.25, size = 1)

## End(Not run)
```

auth *Parse Authenticated Calls To COINBASE PRO (erstwhile GDAX) API*

Description

This is an internal function that will be used for all private connections to the user account. This function determines the kind of API call (GET / POST / DELETE).

Usage

```
auth(method, req.url = "/accounts/", api.key, secret, passphrase, order = NULL)
```

Arguments

`method`	Mandatory character value. Value must be upper case.
`req.url`	THE URL component for the API. Default to "/accounts".
`api.key`	Mandatory character value. This is the API key as generated by GDAX. Typically a 32 character value.

`secret`	Mandatory character value. This is the API secret as generated by GDAX. Typically a 88 character value.
`passphrase`	Mandatory character value. This is the pass-phrase as generated by GDAX. Typically a 11 character value.
`order`	Optional named list. If method is `POST` then the field is required for post to work, else the api end point will return an error.

Value

A named list of server response.

`cancel_order` *Cancel Pending Orders*

Description

This is an auth based function. User must have valid api keys generated by GDAX which must be passed as mandatory arguments. The users can cancel all pending orders except stop orders (GDAX does not treat STOP order as an open order unless the stop price is kicked in). User can now pass an optional order id to cancel a specific order including stop orders. Open orders can now be determined with `open_orders`. This function is a common call to delete one individual order or deleting all open orders.

Usage

```
cancel_order(order_id = "all", api.key, secret, passphrase)
```

Arguments

`order_id`	Optional character value. This is the order id as generated by GDAX. Default value is "all" which will cancel all open orders.
`api.key`	Mandatory character value. This is the API key as generated by GDAX. Typically a 32 character value.
`secret`	Mandatory character value. This is the API secret as generated by GDAX. Typically a 88 character value.
`passphrase`	Mandatory character value. This is the passphrase as generated by GDAX. Typically a 11 character value.

Value

A Dataframe of order-ids of all orders that were cancelled.

Examples

```
## Not run:
cancel_order(api.key = your_key,
             secret = your_api_secret,
             passphrase = your_api_pass)

cancel_order(order_id = "a0a00000-0000-000a-a000-a0a0aa00000a",
             api.key = your_key,
             secret = your_api_secret,
             passphrase = your_api_pass)

## End(Not run)
```

---

fills — *Get List of Most Recent Fills*

Description

This is an auth based function. User must have valid api keys generated by GDAX which must be passed as mandatory arguments. The functions takes product_id as an optional param and returns a list of all previously filled orders.

Usage

```
fills(api.key, secret, passphrase, product_id = NULL)
```

Arguments

`api.key`	Mandatory character value. This is the API key as generated by GDAX. Typically a 32 character value.
`secret`	Mandatory character value. This is the API secret as generated by GDAX. Typically a 88 character value.
`passphrase`	Mandatory character value. This is the passphrase as generated by GDAX. Typically a 11 character value.
`product_id`	Optional character value for the currency pair. The default is `NULL` which is equivalent of 100 most recent fills. This param when provided is case insensitive and must be one of the valid currency-pair. The list of valid currency-pairs can be fetched using `public_info`.

Value

Dataframe with fills for all products or for the provided products. The volume is quoted in USD.

Examples

```
## Not run:
fills(api.key = your_key,
secret = your_api_secret,
passphrase = your_api_pass,
product_id = "BTC-USD")

## End(Not run)
```

---

holds — *Get Hold Details for An Account Using Currency*

Description

This is an auth based function. User must have valid api keys generated by GDAX which must be passed as mandatory arguments. The function takes a currency as an additional input and returns the hold information. Since currency and account id have one to one mapping, the currency is being used a proxy. The basic hold amounts can also be fetched using `account` or `accounts`, however this function provides additional details and ties to active open orders. Please note that each trade buy order, results in a hold on the currency like `USD` and each sell order will result in a hold on the cryptoasset like `BTC`.

Usage

```
holds(currency = "BTC", api.key, secret, passphrase)
```

Arguments

`currency`	Optional character value. The default is `"BTC"`. This is case insensitive and must be a valid currency as provided by `accounts` or `account`.
`api.key`	Mandatory character value. This is the API key as generated by GDAX. Typically a 32 character value.
`secret`	Mandatory character value. This is the API secret as generated by GDAX. Typically a 88 character value.
`passphrase`	Mandatory character value. This is the passphrase as generated by GDAX. Typically a 11 character value.

Value

Dataframe with timestamp of all the holds for the currency, the amount & type of hold and a reference id which created the hold.

Examples

```
## Not run:
holds(api.key = your_key, secret = your_api_secret, passphrase = your_api_pass)
holds("ETH", api.key = your_key, secret = your_api_secret, passphrase = your_api_pass)
holds("USD", api.key = your_key, secret = your_api_secret, passphrase = your_api_pass)

## End(Not run)
```

`list_orders` *Get List of All Orders for the User*

Description

This is an auth based function. User must have valid api keys generated by GDAX which must be passed as mandatory arguments. The function will return all open orders by default for Bitcoin This is an extension of `open_orders`.

Usage

```
list_orders(
  api.key,
  secret,
  passphrase,
  product_id = "BTC-USD",
  status = "open"
)
```

Arguments

`api.key`	Mandatory character value. This is the API key as generated by GDAX. Typically a 32 character value.
`secret`	Mandatory character value. This is the API secret as generated by GDAX. Typically a 88 character value.
`passphrase`	Mandatory character value. This is the passphrase as generated by GDAX. Typically a 11 character value.
`product_id`	Optional character value for the currency pair. The default is `"BTC-USD"`. This param is case insensitive and must be one of the valid currency-pair. The list of valid currency-pairs can be fetched using `public_info`.
`status`	Optional character value. Limit list of orders to either of ‘open‘, ‘pending‘, or ‘active‘ statuses. Passing ‘all‘ returns orders of all statuses.

Value

Dataframe with all orders for a given status for that currency.

Examples

```
## Not run:
list_orders(api.key = your_key,
secret = your_api_secret,
passphrase = your_api_pass)

list_orders(api.key=your_api_key,secret=your_secret,
passphrase=your_passphrase,
product_id="BTC-EUR",status="active")

## End(Not run)
```

open_orders *Get List of All Orders the User*

Description

This is an auth based function. User must have valid api keys generated by GDAX which must be passed as mandatory arguments. The function will return all open orders for all currencies. This is an extendion of `holds` & `fills`.

Usage

```
open_orders(api.key, secret, passphrase)
```

Arguments

`api.key`	Mandatory character value. This is the API key as generated by GDAX. Typically a 32 character value.
`secret`	Mandatory character value. This is the API secret as generated by GDAX. Typically a 88 character value.
`passphrase`	Mandatory character value. This is the passphrase as generated by GDAX. Typically a 11 character value.

Value

Dataframe with all orders and their status for that currency.

Examples

```
## Not run:
order_info(api.key = your_key, secret = your_api_secret, passphrase = your_api_pass)

## End(Not run)
```

parse_response — *Utility Function To Parse Message From GDAX API For Public Functions*

Description

An internal function to parse the api response for various public functions.

Usage

```
parse_response(path, query = NA)
```

Arguments

path	Mandatory character parameter. This is an extension of the api end point and passed as an argument from the function calling `parse_response`.
query	Optional named list parameter. This would consist of any named params to be passed to the api end point.

Value

A named list or a dataframe object of the response.

public_candles — *Get bids and asks for provided currency-pair (products) by GDAX.*

Description

This function is a public function and will fetch historic rates for provided currency pair (product_id). Rates are returned in grouped buckets based on requested granularity. A maximum of 300 records are returned. If the date range and granularity are not within the resultant volume set, it will be rejected by Coinbase Pro.

Usage

```
public_candles(
  product_id = "BTC-USD",
  start = NULL,
  end = NULL,
  granularity = NULL
)
```

Arguments

`product_id`	Optional character parameter. This is a case insensitive value of the product id for which the order book is desired. Default to `'LTC-USD'`. For all valid product ids, refer to `public_info`.
`start`	Optional string parameter. Start time in ISO 8601. Format YYYY-MM-DD
`end`	Optional string parameter. End time in ISO 8601. Format YYYY-MM-DD
`granularity`	Optional parameter. Desired timeslice in seconds. The granularity field must be one of the following values: `60, 300, 900, 3600, 21600, 86400`. Otherwise, the request will be rejected. These values correspond to timeslices representing one minute, five minutes, fifteen minutes, one hour, six hours, and one day, respectively.

Value

Dataframe with a time of the candle, low, high, open , close and volume for that candle.

Examples

```
## Not run:
public_candles()
public_candles("ETH-EUR")
ublic_candles("ETH-EUR",start="2012-01-18", end="2012-01-25",granularity=3600)

## End(Not run)
```

`public_daystats` *24 Hour Stats For A Given Product*

Description

This function is a public function and will fetch get 24 hr stats for the provided currency pair (product_id). Volume is in base currency units. open, high, low are in quote currency units.

Usage

```
public_daystats(product_id = "BTC-USD")
```

Arguments

`product_id`	Optional character parameter. This is a case insensitive value of the product id for which the order book is desired. Default to `'BTC-USD'`. For all valid product ids, refer to `public_info`.

Value

Dataframe with a single row of product's last 24 hour stats.

Examples

```
## Not run:
public_daystats()

## End(Not run)
```

public_info   *Get all supported products (currency - pairs) by GDAX.*

Description

This function is a public function and will fetch all supported currency pairs by default. The function can also fetch all the supported currencies based on the `source` argument.

Usage

```
public_info(product = TRUE)
```

Arguments

`product`	Optional Boolean Parameter. Default to `TRUE`. `NA` behavior is similar to default. `FALSE` is equivalent of currency public end point.

Value

Dataframe with ALL supported currency pairs. A dataframe of base currencies is returned when the flag is et as `FALSE`.

Examples

```
## Not run:
public_info()
public_info(product = FALSE)

## End(Not run)
```

`public_orderbook` *Get bids and asks for provided currency-pair (products) by GDAX.*

Description

This function is a public function and will fetch all bids/asks for provided currency pair (product_id). User should change the level to get the best bid/ask or to fetch all bids/asks.

Usage

```
public_orderbook(product_id = "BTC-USD", level = 1)
```

Arguments

`product_id`	Optional character parameter. This is a case insensitive value of the product id for which the order book is desired. Default to `'BTC-USD'`. For all valid product ids, refer to `public_info`.
`level`	Optional parameter. Integer value of level denoting level of detail. Valid user values of 1, 2, 3. Default to 1.

Value

A named list with bids and asks for the provided currency pair.

Examples

```
## Not run:
public_orderbook()
public_orderbook(product_id = "BCH-USD", level = 3)
public_orderbook(product_id = "BTC-EUR", level = 2)

## End(Not run)
```

`public_ticker` *Get Latest Buy & Sell Trade.*

Description

Snapshot information about the last trade (tick), best bid/ask and 24h volume.

Usage

```
public_ticker(product_id = "BTC-USD")
```

Arguments

`product_id`	Optional character parameter. This is a case insensitive value of the product id for which the order book is desired. Default to `'BTC-USD'`. For all valid product ids, refer to `public_info`.

Value

A dataframe of most recent trade and 24 hr volume.

Examples

```
## Not run:
public_ticker()
public_ticker("BTC-EUR")

## End(Not run)
```

`public_time` *Get GDAX API Server Time*

Description

Gets the server time from GDAX for reference purposes. This function does not take any arguments.

Usage

```
public_time()
```

Value

Dataframe with time in ISO and the epoch field represents decimal seconds since Unix Epoch.

Examples

```
## Not run:
public_time()

## End(Not run)
```

public_trades *Get Latest Buy & Sell Trades.*

Description

This is a public function and will fetch the 100 latest trades for the provided currency pair (product_id).

Usage

```
public_trades(product_id = "BTC-USD")
```

Arguments

`product_id`	Optional character parameter. This is a case insensitive value of the product id for which the order book is desired. Default to `'LTC-USD'`. For all valid product ids, refer to `public_info`.

Value

A dataframe of most recent trades indicating if it was buy / sell and what the trade size was.

Examples

```
## Not run:
public_trades()
public_trades("BTC-EUR")

## End(Not run)
```

pymt_methods *View All Linked payment Methods.*

Description

This is an auth based function. User must have valid api keys generated by GDAX which must be passed as mandatory arguments. These keys must have the appropriate access. Note that for this function to work, the API key must have transfer access.

Usage

```
pymt_methods(api.key, secret, passphrase)
```

Arguments

`api.key`	Mandatory character value. This is the API key as generated by GDAX. Typically a 32 character value.
`secret`	Mandatory character value. This is the API secret as generated by GDAX. Typically a 88 character value.
`passphrase`	Mandatory character value. This is the passphrase as generated by GDAX. Typically a 11 character value.

Value

A named list.

Examples

```
## Not run:
pmt_methods(api.key = your_key, secret = your_api_secret, passphrase = your_api_pass)

## End(Not run)
```

Index