
pyoanda Documentation

Release

Author

August 25, 2015

| | | |
|----------|-------------------------------------|-----------|
| 1 | The MIT License (MIT) | 3 |
| 2 | Installing pyoanda | 5 |
| 2.1 | Pypi | 5 |
| 2.2 | Manual | 5 |
| 3 | Usage | 7 |
| 4 | pyoanda package | 9 |
| 4.1 | Submodules | 9 |
| 4.2 | pyoanda.client module | 9 |
| 4.3 | pyoanda.exceptions module | 12 |
| 4.4 | pyoanda.order module | 12 |
| 4.5 | Module contents | 12 |
| 5 | Indices and tables | 13 |
| | Python Module Index | 15 |

Oanda's API python wrapper. Robust and Fast API wrapper for your Forex bot.

Python library that wraps [oanda](#) API. Built on top of requests, it's easy to use and makes sense.

Pyoanda is released under the MIT license. The source code is on [GitHub](#) and [issues are also tracked on GitHub](#). Works well with python 2.7, 3, 3.1, 3.2, 3.3, 3.4 and pypy.

Contents:

The MIT License (MIT)

Copyright (c) 2015 Tolo Palmer

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Installing pyoanda

Pyoanda is on the Python Package Index (PyPI), so it can be installed standard Python tools like `pip` or `easy_install`, and as well you can install from sources

2.1 Pypi

For an easy and always standard setup:

```
$ pip install pyoanda
```

2.2 Manual

For a custom or developer installation:

```
$ git clone git@github.com:toloco/pyoanda.git
$ cd pyoanda
$ python setup.py install
# Make sure it works
$ python setup.py test
```

Usage

```
from pyoanda import Client, PRACTICE

c = Client(
    environment=PRACTICE,
    account_id="Your Oanda account ID",
    access_token="Your Oanda access token"
)

c.get_instrument_history(
    instrument="EUR_GBP",
    candle_format="midpoint",
    granularity="S30"
)
```

Note: that if you are intending to use the sandbox environment, you should first use the API to create an account then use the account_id to run the example above.

```
from pyoanda import Client, SANDBOX

c = Client(environment=SANDBOX)

# Create an account
user = c.create_account()

# Retrieve the username and accountId values for future use
print "username: %s\naccount_id: %d" % (user['username'], user['accountId'])
```

pyoanda package

4.1 Submodules

4.2 pyoanda.client module

class `pyoanda.client.Client` (*environment*, *account_id=None*, *access_token=None*,
json_options=None)

Bases: `object`

API_VERSION = 'v1'

close_order (*order_id*)
See more: <http://developer.oanda.com/rest-live/orders/#closeOrder>

close_position (*instrument*)
Close an existing position

instrument [string] The instrument to close the position for.

See more: <http://developer.oanda.com/rest-live/positions/#closeExistingPosition>

close_trade (*trade_id*)
Close an open trade.

trade_id [int] The id of the trade to close.

See more: <http://developer.oanda.com/rest-live/trades/#closeOpenTrade>

create_account (*currency=None*)
Create a new account.

This call is only available on the sandbox system. Please create accounts on fxtrade.oanda.com on our production system.

See more: <http://developer.oanda.com/rest-sandbox/accounts/#-a-name-createtestaccount-a-create-a-test-account>

create_order (*order*)
See more: <http://developer.oanda.com/rest-live/orders/#createNewOrder>

get_accounts (*username=None*)
Get a list of accounts owned by the user.

username [string] The name of the user. Note: This is only required on the sandbox, on production systems your access token will identify you.

See more: <http://developer.oanda.com/rest-sandbox/accounts/#-a-name-getaccountsforuser-a-get-accounts-for-a-user>

get_credentials()

See more: <http://developer.oanda.com/rest-live/accounts/>

get_instrument_history (*instrument*, *candle_format='bidask'*, *granularity='S5'*,
count=500, *daily_alignment=None*, *alignment_timezone=None*,
weekly_alignment='Monday', *start=None*, *end=None*)

See more: <http://developer.oanda.com/rest-live/rates/#retrieveInstrumentHistory>

get_instruments()

See more: <http://developer.oanda.com/rest-live/rates/#getInstrumentList>

get_order (*order_id*)

See more: <http://developer.oanda.com/rest-live/orders/#getInformationForAnOrder>

get_orders (*instrument=None*, *count=50*)

See more: <http://developer.oanda.com/rest-live/orders/#getOrdersForAnAccount>

get_position (*instrument*)

Get the position for an instrument.

instrument [string] The instrument to get the open position for.

See more: <http://developer.oanda.com/rest-live/positions/#getPositionForInstrument>

get_positions()

Get a list of all open positions.

See more: <http://developer.oanda.com/rest-live/positions/#getListAllOpenPositions>

get_prices (*instruments*, *stream=True*)

See more: <http://developer.oanda.com/rest-live/rates/#getCurrentPrices>

get_trade (*trade_id*)

Get information on a specific trade.

trade_id [int] The id of the trade to get information on.

See more: <http://developer.oanda.com/rest-live/trades/#getInformationSpecificTrade>

get_trades (*max_id=None*, *count=None*, *instrument=None*, *ids=None*)

Get a list of open trades

max_id [int] The server will return trades with id less than or equal to this, in descending order (for pagination)

count [int] Maximum number of open trades to return. Default: 50 Max value: 500

instrument [str] Retrieve open trades for a specific instrument only Default: all

ids [list] A list of trades to retrieve. Maximum number of ids: 50. No other parameter may be specified with the ids parameter.

See more: <http://developer.oanda.com/rest-live/trades/#getListOpenTrades>

get_transaction (*transaction_id*)

Get information on a specific transaction.

transaction_id [int] The id of the transaction to get information on.

See more: <http://developer.oanda.com/rest-live/transaction-history/#getInformationForTransaction>
<http://developer.oanda.com/rest-live/transaction-history/#transactionTypes>

get_transaction_history (*max_wait=5.0*)

Download full account history.

Uses `request_transaction_history` to get the transaction history URL, then polls the given URL until it's ready (or the `max_wait` time is reached) and provides the decoded response.

max_wait [float] The total maximum time to spend waiting for the file to be ready; if this is exceeded a failed response will be returned. This is not guaranteed to be strictly followed, as one last attempt will be made to check the file before giving up.

See more: <http://developer.oanda.com/rest-live/transaction-history/#getFullAccountHistory>
<http://developer.oanda.com/rest-live/transaction-history/#transactionTypes>

get_transactions (*max_id=None, count=None, instrument='all', ids=None*)

Get a list of transactions.

max_id [int] The server will return transactions with id less than or equal to this, in descending order (for pagination).

count [int] Maximum number of open transactions to return. Default: 50. Max value: 500.

instrument [str] Retrieve open transactions for a specific instrument only. Default: all.

ids [list] A list of transactions to retrieve. Maximum number of ids: 50. No other parameter may be specified with the `ids` parameter.

See more: <http://developer.oanda.com/rest-live/transaction-history/#getTransactionHistory>
<http://developer.oanda.com/rest-live/transaction-history/#transactionTypes>

request_transaction_history ()

Request full account history.

Submit a request for a full transaction history. A successfully accepted submission results in a response containing a URL in the Location header to a file that will be available once the request is served. Response for the URL will be HTTP 404 until the file is ready. Once served the URL will be valid for a certain amount of time.

See more: <http://developer.oanda.com/rest-live/transaction-history/#getFullAccountHistory>
<http://developer.oanda.com/rest-live/transaction-history/#transactionTypes>

update_order (*order_id, order*)

See more: <http://developer.oanda.com/rest-live/orders/#modifyExistingOrder>

update_trade (*trade_id, stop_loss=None, take_profit=None, trailing_stop=None*)

Modify an existing trade.

Note: Only the specified parameters will be modified. All other parameters will remain unchanged. To remove an optional parameter, set its value to 0.

trade_id [int] The id of the trade to modify.

stop_loss [number] Stop Loss value.

take_profit [number] Take Profit value.

trailing_stop [number] Trailing Stop distance in pips, up to one decimal place

See more: <http://developer.oanda.com/rest-live/trades/#modifyExistingTrade>

4.3 pyoanda.exceptions module

exception `pyoanda.exceptions.BadCredentials`

Bases: `Exception`

exception `pyoanda.exceptions.BadRequest`

Bases: `Exception`

4.4 pyoanda.order module

class `pyoanda.order.Order` (***kwargs*)

Bases: `object`

check ()

Logic extracted from: <http://developer.oanda.com/rest-live/orders/#createNewOrder>

4.5 Module contents

Indices and tables

- `genindex`
- `modindex`
- `search`

p

`pyoanda`, [12](#)
`pyoanda.client`, [9](#)
`pyoanda.exceptions`, [12](#)
`pyoanda.order`, [12](#)

A

API_VERSION (pyoanda.client.Client attribute), 9

B

BadCredentials, 12

BadRequest, 12

C

check() (pyoanda.order.Order method), 12

Client (class in pyoanda.client), 9

close_order() (pyoanda.client.Client method), 9

close_position() (pyoanda.client.Client method), 9

close_trade() (pyoanda.client.Client method), 9

create_account() (pyoanda.client.Client method), 9

create_order() (pyoanda.client.Client method), 9

G

get_accounts() (pyoanda.client.Client method), 9

get_credentials() (pyoanda.client.Client method), 10

get_instrument_history() (pyoanda.client.Client method),
10

get_instruments() (pyoanda.client.Client method), 10

get_order() (pyoanda.client.Client method), 10

get_orders() (pyoanda.client.Client method), 10

get_position() (pyoanda.client.Client method), 10

get_positions() (pyoanda.client.Client method), 10

get_prices() (pyoanda.client.Client method), 10

get_trade() (pyoanda.client.Client method), 10

get_trades() (pyoanda.client.Client method), 10

get_transaction() (pyoanda.client.Client method), 10

get_transaction_history() (pyoanda.client.Client method),
10

get_transactions() (pyoanda.client.Client method), 11

O

Order (class in pyoanda.order), 12

P

pyoanda (module), 12

pyoanda.client (module), 9

pyoanda.exceptions (module), 12

pyoanda.order (module), 12

R

request_transaction_history() (pyoanda.client.Client
method), 11

U

update_order() (pyoanda.client.Client method), 11

update_trade() (pyoanda.client.Client method), 11