OANDA REST-V20 API Documentation

Release 0.6.1

Feite Brekeveld

oandapyV20 REST-V20 API wrapper

1	Intro	duction	3
	1.1	Install	3
	1.2	Download from Github	3
2	Inter	face OANDA's REST-V20	5
	2.1	The client	5
	2.2	Exceptions	8
	2.3	Logging	8
3	oand	apyV20.endpoints	11
	3.1	oandapyV20.endpoints.accounts	11
	3.2	oandapyV20.endpoints.forexlabs	20
	3.3	oandapyV20.endpoints.instruments	34
	3.4	oandapyV20.endpoints.orders	42
	3.5	oandapyV20.endpoints.positions	50
	3.6		56
	3.7		62
	3.8	oandapyV20.endpoints.transactions	70
4	oand	apyV20.definitions	7 9
	4.1	oandapyV20.definitions.accounts	79
	4.2	oandapyV20.definitions.instruments	80
	4.3	oandapyV20.definitions.orders	82
	4.4	oandapyV20.definitions.pricing	86
	4.5	oandapyV20.definitions.trades	87
	4.6		89
5 oan	oand	apyV20.types	01
	5.1	AccountID	01
	5.2	AccountUnits	
	5.3	ClientComment	
	5.4	ClientID	
	5.5	ClientTag	
	5.6	DateTime	
	5.7	OrderID	
	5.8	OrderIdentifier	
	<i>7</i> 0	OrderSpecifier	
	5.9	Orderspecifier	U4

	5.10	PriceValue	104					
	5.11	TradeID	104					
	5.12	Units	104					
6	oand	apyV20.contrib	107					
	6.1	apyV20.contrib Factories	107					
	6.2	Generic						
	6.3	Order Classes	109					
	6.4	support classes						
7	Exar	mples	125					
	7.1	Example for trades-endpoints	125					
8 Indices and tables		ces and tables	127					
Рy	Python Module Index							

Contents:

CHAPTER 1

Introduction

The oandapyV20 package offers an API to the OANDA V20 REST service. To use the REST-API-service you will need a *token* and an *account*. This applies for both *live* and *practice* accounts. For details check oanda.com.

1.1 Install

Install the pypi package with pip:

```
$ pip install oandapyV20
```

Or alternatively install the latest development version from github:

```
$ pip install git+https://github.com/hootnot/oanda-api-v20.git
```

You may consider using *virtualenv* to create isolated Python environments. Python 3.4 has *pyvenv* providing the same kind of functionality.

1.2 Download from Github

If you want to run the tests, download the source from github:

```
$ git clone https://github.com/hootnot/oanda-api-v20.git
$ cd oanda-api-v20
$ python setup.py test
$ python setup.py install
```

Interface OANDA's REST-V20

2.1 The client

The oandapyV20 package contains a client class, oandapyV20.API, to communicate with the REST-V20 interface. It processes requests that can be created from the endpoint classes. For it's communication it relies on: requests (requests).

The client keeps no state of a requests. The response of a request is assigned to the request instance. The response is also returned as a return value by the client.

```
class oandapy V20.API (access_token, environment='practice', headers=None, request_params=None)

Bases: object
```

API - class to handle APIRequests objects to access API endpoints.

Examples

```
# get a list of trades
from oandapyV20 import API
import oandapyV20.endpoints.trades as trades

api = API(access_token="xxx")
accountID = "101-305-3091856-001"

r = trades.TradesList(accountID)
# show the endpoint as it is constructed for this call
print("REQUEST:{}".format(r))
rv = api.request(r)
print("RESPONSE:\n{}".format(json.dumps(rv, indent=2)))
```

```
REQUEST: v3/accounts/101-305-3091856-001/trades
RESPONSE:
"trades": [
   {
      "financing": "0.0000",
      "openTime": "2016-07-21T15:47:05.170212014Z",
      "price": "10133.9",
      "unrealizedPL": "8.0000",
      "realizedPL": "0.0000",
      "instrument": "DE30_EUR",
      "state": "OPEN",
      "initialUnits": "-10",
      "currentUnits": "-10",
      "id": "1032"
    },
      "financing": "0.0000",
      "openTime": "2016-07-21T15:47:04.963590941Z",
      "price": "10134.4",
      "unrealizedPL": "13.0000",
      "realizedPL": "0.0000",
      "instrument": "DE30_EUR",
      "state": "OPEN",
      "initialUnits": "-10",
      "currentUnits": "-10",
      "id": "1030"
   }
  ],
  "lastTransactionID": "1040"
```

```
# reduce a trade by it's id
from oandapyV20 import API
import oandapyV20.endpoints.trades as trades

api = API(access_token="...")
accountID = "101-305-3091856-001"
tradeID = "1030"
cfg = {"units": 5}
r = trades.TradeClose(accountID, tradeID=tradeID, data=cfg)
# show the endpoint as it is constructed for this call
print("REQUEST:{}".format(r))
rv = api.request(r)
print("RESPONSE\n{}".format(json.dumps(rv, indent=2)))
```

```
REQUEST: v3/accounts/101-305-3091856-001/trades/1030/close
RESPONSE: {
    "orderFillTransaction": {
        "orderID": "1041",
        "financing": "-0.1519",
        "instrument": "DE30_EUR",
        "userID": 1435156,
        "price": "10131.6",
        "tradeReduced": {
        "units": "5",
```

```
"financing": "-0.1519",
    "realizedPL": "14.0000",
    "tradeID": "1030"
  },
  "batchID": "1041",
  "accountBalance": "44876.2548",
  "reason": "MARKET_ORDER_TRADE_CLOSE",
  "time": "2016-07-21T17:32:51.361464739Z",
  "units": "5",
  "type": "ORDER_FILL",
  "id": "1042",
  "pl": "14.0000",
  "accountID": "101-305-3091856-001"
"orderCreateTransaction": {
  "timeInForce": "FOK",
  "positionFill": "REDUCE_ONLY",
  "userID": 1435156,
  "batchID": "1041",
  "instrument": "DE30_EUR",
  "reason": "TRADE_CLOSE",
  "tradeClose": {
   "units": "5",
    "tradeID": "1030"
  "time": "2016-07-21T17:32:51.361464739Z",
  "units": "5",
  "type": "MARKET_ORDER",
  "id": "1041",
  "accountID": "101-305-3091856-001"
},
"relatedTransactionIDs": [
 "1041",
 "1042"
1.
"lastTransactionID": "1042"
```

__init__ (access_token, environment='practice', headers=None, request_params=None)
Instantiate an instance of OandaPy's API wrapper.

Parameters

- access_token (string) Provide a valid access token.
- **environment** (*string*) Provide the environment for OANDA's REST api. Valid values: 'practice' or 'live'. Default: 'practice'.
- headers (dict (optional)) Provide request headers to be set for a request.

Note: There is no need to set the 'Content-Type: application/json' for the endpoints that require this header. The API-request classes covering those endpoints will take care of the header.

request_params [(optional)] parameters to be passed to the request. This can be used to apply for instance a timeout value:

request_params={"timeout": 0.1}

2.1. The client 7

See specs of the requests module for full details of possible parameters.

Warning: parameters belonging to a request need to be set on the requestinstance and are NOT passed via the client.

```
request (endpoint)
```

Perform a request for the APIRequest instance 'endpoint'.

Parameters endpoint (APIRequest) – The endpoint parameter contains an instance of an APIRequest containing the endpoint, method and optionally other parameters or body data.

Raises V20Error in case of HTTP response code >= 400

request_params

request_params property.

2.2 Exceptions

```
class oandapyV20.V20Error(code, msg)
    Bases: exceptions.Exception
```

Generic error class.

In case of HTTP response codes >= 400 this class can be used to raise an exception representing that error.

```
___init___(code, msg)
Instantiate a V20Error.
```

Parameters

- code (int) the HTTP-code of the response
- msg(str) the message returned with the response

2.3 Logging

The oandapy V20 package has *logging* integrated. Logging can be simply applied by enabling a *logger*. The example below will log INFO-level logging to the file v20.log. For details check the logger module in the standard Python documentation.

```
# code snippet
from oandapyV20 import API
import oandapyV20.endpoints.orders as orders
from oandapyV20.exceptions import V20Error
from exampleauth import exampleAuth
import logging

logging.basicConfig(
    filename="v20.log",
    level=logging.INFO,
    format='%(asctime)s [%(levelname)s] %(name)s : %(message)s',
)

accountID, token = exampleAuth()
...
```

Resulting loglines:

2.3. Logging 9

CHAPTER 3

oandapyV20.endpoints

3.1 oandapyV20.endpoints.accounts

3.1.1 AccountChanges

```
 \textbf{class} \  \, \texttt{oandapyV20.endpoints.accounts.AccountChanges} \, (\textit{accountID}, \textit{params=None}) \\ \text{Bases:} \, \texttt{oandapyV20.endpoints.accounts.Accounts} \, . \\
```

AccountChanges.

Endpoint used to poll an Account for its current state and changes since a specified TransactionID.

```
ENDPOINT = 'v3/accounts/{accountID}/changes'
EXPECTED_STATUS = 200

METHOD = 'GET'
__init__(accountID, params=None)
    Instantiate an AccountChanges request.
```

Parameters

- account ID (string (required)) id of the account to perform the request on.
- params (dict (optional)) query params to send, check developer.oanda.com for details.

Query Params example:

```
{
  "sinceTransactionID": 2308
}
```

```
>>> import oandapyV20
>>> import oandapyV20.endpoints.accounts as accounts
>>> client = oandapyV20.API(access_token=...)
```

```
>>> params = ...
>>> r = accounts.AccountChanges(accountID=..., params=params)
>>> client.request(r)
>>> print r.response
```

```
"state": {
 "trades": [],
 "marginCloseoutNAV": "33848.2663",
  "marginUsed": "0.0000",
  "marginAvailable": "33848.2663",
  "marginCallPercent": "0.00000",
 "NAV": "33848.2663",
 "marginCloseoutMarginUsed": "0.0000",
 "orders": [],
 "withdrawalLimit": "33848.2663",
 "marginCloseoutPercent": "0.00000",
 "positions": [],
 "unrealizedPL": "0.0000",
  "marginCallMarginUsed": "0.0000",
  "marginCloseoutUnrealizedPL": "0.0000",
  "positionValue": "0.0000"
},
"changes": {
 "tradesReduced": [],
 "tradesOpened": [],
 "ordersFilled": [],
 "tradesClosed": [],
  "transactions": [
      "price": "1.20000",
      "stopLossOnFill": {
       "timeInForce": "GTC",
        "price": "1.22000"
      "timeInForce": "GTC",
      "reason": "CLIENT_ORDER",
      "id": "2309",
      "batchID": "2309",
      "triggerCondition": "TRIGGER_DEFAULT",
      "positionFill": "DEFAULT",
      "userID": 1435156,
      "instrument": "EUR_USD",
      "time": "2016-10-25T21:07:21.065554321Z",
      "units": "-100",
      "type": "LIMIT_ORDER",
      "accountID": "101-004-1435156-001"
    }
  ],
  "ordersCreated": [
    {
      "partialFill": "DEFAULT_FILL",
      "price": "1.20000",
      "stopLossOnFill": {
       "timeInForce": "GTC",
        "price": "1.22000"
```

```
},
    "timeInForce": "GTC",
    "createTime": "2016-10-25T21:07:21.065554321Z",
    "triggerCondition": "TRIGGER_DEFAULT",
    "positionFill": "POSITION_DEFAULT",
    "id": "2309",
    "instrument": "EUR_USD",
    "state": "PENDING",
    "units": "-100",
    "type": "LIMIT"
    }
],
    "positions": [],
    "ordersTriggered": [],
    "ordersCancelled": []
},
    "lastTransactionID": "2309"
}
```

3.1.2 AccountConfiguration

```
class oandapyV20.endpoints.accounts.AccountConfiguration(accountID, data)
    Bases: oandapyV20.endpoints.accounts.
```

Set the client-configurable portions of an Account.

antiate an riceounteeningaration req

Parameters

- accountID(string (required)) id of the account to perform the request on.
- data (dict (required)) json body to send

body example:

```
{
    "marginRate": "0.05"
}
```

```
>>> import oandapyV20
>>> import oandapyV20.endpoints.accounts as accounts
>>> client = oandapyV20.API(access_token=...)
>>> r = accounts.AccountConfiguration(accountID, data=data)
>>> client.request(r)
>>> print r.response
```

```
{
  "lastTransactionID": "830",
  "clientConfigureTransaction": {
```

```
"userID": 1435156,

"marginRate": "0.05",

"batchID": "830",

"time": "2016-07-12T19:48:11.657494168Z",

"type": "CLIENT_CONFIGURE",

"id": "830",

"accountID": "101-004-1435156-001"

}
```

3.1.3 AccountDetails

 $\verb|class| oandapy V20.endpoints.accounts. Account Details (|account ID)|$

Bases: oandapyV20.endpoints.accounts.Accounts

AccountDetails.

Get the full details for a single Account that a client has access to. Full pending Order, open Trade and open Position representations are provided.

```
ENDPOINT = 'v3/accounts/{accountID}'
EXPECTED_STATUS = 200
METHOD = 'GET'
__init__(accountID)
    Instantiate an AccountDetails request.
```

Parameters accountID (string (required)) - id of the account to perform the request

```
>>> import oandapyV20
>>> import oandapyV20.endpoints.accounts as accounts
>>> client = oandapyV20.API(access_token=...)
>>> r = accounts.AccountDetails(accountID)
>>> client.request(r)
>>> print r.response
```

```
"account": {
  "positions": [
      "short": {
        "units": "0",
        "resettablePL": "0.0000",
        "unrealizedPL": "0.0000",
        "pl": "0.0000"
      },
      "unrealizedPL": "0.0000",
      "long": {
        "units": "0",
        "resettablePL": "-3.8046",
        "unrealizedPL": "0.0000",
        "pl": "-3.8046"
      "instrument": "EUR_USD",
      "resettablePL": "-3.8046",
```

```
"pl": "-3.8046"
  },
  {
    "short": {
      "unrealizedPL": "682.0000",
      "units": "-20",
      "resettablePL": "-1744.8000",
      "tradeIDs": [
        "821",
       "823"
      ],
      "averagePrice": "9984.7",
      "pl": "-1744.8000"
    "unrealizedPL": "682.0000",
    "long": {
      "units": "0",
      "resettablePL": "447.6000",
      "unrealizedPL": "0.0000",
      "pl": "447.6000"
    },
    "instrument": "DE30_EUR",
    "resettablePL": "-1297.2000",
    "pl": "-1297.2000"
  }
"unrealizedPL": "682.0000",
"marginCloseoutNAV": "49393.6580",
"marginUsed": "9948.9000",
"currency": "EUR",
"resettablePL": "-1301.0046",
"NAV": "49377.6580",
"marginCloseoutMarginUsed": "9949.8000",
"id": "101-004-1435156-001",
"marginCloseoutPositionValue": "198996.0000",
"openTradeCount": 2,
"orders": [
    "partialFill": "DEFAULT_FILL",
    "price": "0.87000",
    "stopLossOnFill": {
      "timeInForce": "GTC",
      "price": "0.88000"
    },
    "timeInForce": "GTC",
    "clientExtensions": {
      "comment": "myComment",
      "id": "myID"
    },
    "id": "204",
    "triggerCondition": "TRIGGER_DEFAULT",
    "replacesOrderID": "200",
    "positionFill": "POSITION_DEFAULT",
    "createTime": "2016-07-08T07:18:47.623211321Z",
    "instrument": "EUR_GBP",
    "state": "PENDING",
    "units": "-50000",
    "type": "LIMIT"
```

```
],
  "openPositionCount": 1,
  "marginCloseoutPercent": "0.10072",
 "marginCallMarginUsed": "9949.8000",
 "hedgingEnabled": false,
 "positionValue": "198978.0000",
 "pl": "-1301.0046",
 "lastTransactionID": "833",
  "marginAvailable": "39428.7580",
  "marginRate": "0.05",
  "marginCallPercent": "0.20144",
  "pendingOrderCount": 1,
  "withdrawalLimit": "39428.7580",
  "trades": [
   {
      "instrument": "DE30_EUR",
      "financing": "0.0000",
      "openTime": "2016-07-12T09:32:18.062823776Z",
      "initialUnits": "-10",
      "currentUnits": "-10",
      "price": "9984.7",
      "unrealizedPL": "341.0000",
      "realizedPL": "0.0000",
      "state": "OPEN",
      "id": "821"
   },
      "instrument": "DE30_EUR",
      "financing": "0.0000",
      "openTime": "2016-07-12T09:32:18.206929733Z",
      "initialUnits": "-10",
      "currentUnits": "-10",
      "price": "9984.7",
      "unrealizedPL": "341.0000",
      "realizedPL": "0.0000",
      "state": "OPEN",
      "id": "823"
 ],
  "alias": "hootnotv20",
  "createdByUserID": 1435156,
  "marginCloseoutUnrealizedPL": "698.0000",
  "createdTime": "2016-06-24T21:03:50.914647476Z",
  "balance": "48695.6580"
},
"lastTransactionID": "833"
```

3.1.4 AccountInstruments

class oandapyV20.endpoints.accounts.AccountInstruments(accountID, params=None)
 Bases: oandapyV20.endpoints.accounts.Accounts

AccountInstruments.

Get the list of tradable instruments for the given Account. The list of tradeable instruments is dependent on the

regulatory division that the Account is located in, thus should be the same for all Accounts owned by a single user.

```
ENDPOINT = 'v3/accounts/{accountID}/instruments'
EXPECTED_STATUS = 200

METHOD = 'GET'
__init__(accountID, params=None)
    Instantiate an AccountInstruments request.
```

Parameters

- account ID (string (required)) id of the account to perform the request on.
- params (dict (optional)) query params to send, check developer.oanda.com for details.

Query Params example:

```
{
   "instruments": "EU50_EUR, EUR_USD, US30_USD, FR40_EUR, EUR_CHF, DE30_EUR"
}
```

```
>>> import oandapyV20
>>> import oandapyV20.endpoints.accounts as accounts
>>> client = oandapyV20.API(access_token=...)
>>> params = ...
>>> r = accounts.AccountInstruments(accountID=..., params=params)
>>> client.request(r)
>>> print r.response
```

```
"instruments": [
   "marginRate": "0.05",
    "minimumTrailingStopDistance": "5.0",
    "maximumPositionSize": "0",
    "minimumTradeSize": "1",
    "displayName": "Europe 50",
    "name": "EU50_EUR",
   "displayPrecision": 1,
   "maximumTrailingStopDistance": "10000.0",
   "maximumOrderUnits": "3000",
   "tradeUnitsPrecision": 0,
    "pipLocation": 0,
    "type": "CFD"
  },
    "marginRate": "0.05",
    "minimumTrailingStopDistance": "0.00050",
    "maximumPositionSize": "0",
    "minimumTradeSize": "1",
    "displayName": "EUR/USD",
    "name": "EUR_USD",
    "displayPrecision": 5,
    "maximumTrailingStopDistance": "1.00000",
    "maximumOrderUnits": "100000000",
```

```
"tradeUnitsPrecision": 0,
  "pipLocation": -4,
  "type": "CURRENCY"
},
 "marginRate": "0.05",
 "minimumTrailingStopDistance": "5.0",
 "maximumPositionSize": "0",
 "minimumTradeSize": "1",
  "displayName": "US Wall St 30",
  "name": "US30_USD",
  "displayPrecision": 1,
  "maximumTrailingStopDistance": "10000.0",
  "maximumOrderUnits": "1000",
  "tradeUnitsPrecision": 0,
  "pipLocation": 0,
  "type": "CFD"
},
 "marginRate": "0.05",
 "minimumTrailingStopDistance": "5.0",
 "maximumPositionSize": "0",
  "minimumTradeSize": "1",
  "displayName": "France 40",
  "name": "FR40_EUR",
  "displayPrecision": 1,
  "maximumTrailingStopDistance": "10000.0",
  "maximumOrderUnits": "2000",
  "tradeUnitsPrecision": 0,
 "pipLocation": 0,
 "type": "CFD"
},
 "marginRate": "0.05",
 "minimumTrailingStopDistance": "0.00050",
  "maximumPositionSize": "0",
  "minimumTradeSize": "1",
  "displayName": "EUR/CHF",
  "name": "EUR_CHF",
  "displayPrecision": 5,
  "maximumTrailingStopDistance": "1.00000",
  "maximumOrderUnits": "100000000",
  "tradeUnitsPrecision": 0,
  "pipLocation": -4,
  "type": "CURRENCY"
},
  "marginRate": "0.05",
  "minimumTrailingStopDistance": "5.0",
  "maximumPositionSize": "0",
  "minimumTradeSize": "1",
  "displayName": "Germany 30",
  "name": "DE30_EUR",
  "displayPrecision": 1,
  "maximumTrailingStopDistance": "10000.0",
  "maximumOrderUnits": "2500",
  "tradeUnitsPrecision": 0,
  "pipLocation": 0,
```

```
"type": "CFD"
}
],
"lastTransactionID": "2124"
}
```

3.1.5 AccountList

```
class oandapyV20.endpoints.accounts.AccountList
    Bases: oandapyV20.endpoints.accounts.Accounts

Get a list of all Accounts authorized for the provided token.

ENDPOINT = 'v3/accounts'

EXPECTED_STATUS = 200

METHOD = 'GET'

__init__()

Instantiate an AccountList request.

>>> import oandapyV20
>>> import oandapyV20.endpoints.accounts as accounts
>>> client = oandapyV20.API(access_token=...)
>>> r = accounts.AccountList()
>>> client.request(r)
>>> print r.response
```

3.1.6 AccountSummary

Parameters accountID (string (required)) - id of the account to perform the request on.

```
>>> import oandapyV20
>>> import oandapyV20.endpoints.accounts as accounts
>>> client = oandapyV20.API(access_token=...)
>>> r = accounts.AccountSummary(accountID)
>>> client.request(r)
>>> print r.response
```

```
"account": {
  "marginCloseoutNAV": "35454.4740",
  "marginUsed": "10581.5000",
  "currency": "EUR",
  "resettablePL": "-13840.3525",
  "NAV": "35454.4740",
  "marginCloseoutMarginUsed": "10581.5000",
  "marginCloseoutPositionValue": "211630.0000",
  "openTradeCount": 2,
  "id": "101-004-1435156-001",
  "hedgingEnabled": false,
  "marginCloseoutPercent": "0.14923",
  "marginCallMarginUsed": "10581.5000",
  "openPositionCount": 1,
  "positionValue": "211630.0000",
  "pl": "-13840.3525",
  "lastTransactionID": "2123",
  "marginAvailable": "24872.9740",
  "marginRate": "0.05",
  "marginCallPercent": "0.29845",
  "pendingOrderCount": 0,
  "withdrawalLimit": "24872.9740",
  "unrealizedPL": "0.0000",
  "alias": "hootnotv20",
  "createdByUserID": 1435156,
  "marginCloseoutUnrealizedPL": "0.0000",
  "createdTime": "2016-06-24T21:03:50.914647476Z",
  "balance": "35454.4740"
},
"lastTransactionID": "2123"
```

3.2 oandapyV20.endpoints.forexlabs

3.2.1 Autochartist

```
class oandapyV20.endpoints.forexlabs.Autochartist(params=None)
    Bases: oandapyV20.endpoints.forexlabs.ForexLabs
    Autochartist.
    Get the 'autochartist data'.
    ENDPOINT = 'labs/v1/signal/autochartist'
    EXPECTED STATUS = 200
```

METHOD = 'GET'

__init__(params=None)

Instantiate an Autochartist request.

Parameters params (dict (optional)) - query params to send, check developer.oanda.com for details.

```
>>> r = labs.Autochartist(params=params)
>>> client.request(r)
>>> print(r.response)
```

```
"signals": [
 {
    "data": {
      "points": {
        "support": {
         "y1": 0.72456,
          "y0": 0.725455,
          "x0": 1520420400,
          "x1": 1520503200
        },
        "resistance": {
          "y1": 0.729755,
          "y0": 0.731095,
          "x0": 1520323200,
          "x1": 1520463600
        }
      },
      "patternendtime": 1520589600,
      "prediction": {
        "pricelow": 0.7316,
        "timefrom": 1520589600,
        "pricehigh": 0.7349,
        "timeto": 1520773200
      }
    "meta": {
      "direction": 1,
      "completed": 1,
      "probability": 72.36,
      "scores": {
        "clarity": 7,
        "breakout": 10,
        "quality": 8,
        "initialtrend": 10,
        "uniformity": 6
```

```
"pattern": "Channel Down",
      "historical stats": {
        "hourofday": {
          "total": 1909,
          "percent": 71.08,
          "correct": 1357
        },
        "pattern": {
          "total": 3361,
          "percent": 73.61,
          "correct": 2474
        "symbol": {
          "total": 429,
          "percent": 65.5,
          "correct": 281
        }
      },
      "interval": 60,
      "trendtype": "Continuation",
      "length": 73
    "type": "chartpattern",
    "id": 458552738,
    "instrument": "NZD_USD"
],
"provider": "autochartist"
```

3.2.2 Calendar

```
class oandapyV20.endpoints.forexlabs.Calendar(params)
   Bases: oandapyV20.endpoints.forexlabs.ForexLabs

Calendar.

Get calendar information.

ENDPOINT = 'labs/v1/calendar'

EXPECTED_STATUS = 200

METHOD = 'GET'

__init__(params)
   Instantiate a Calendar request.

Parameters params (dict (required)) - query params to send, check developer.oanda.com for details.
```

```
>>> import oandapyV20
>>> import oandapyV20.endpoints.forexlabs as labs
>>> accountID = ...
>>> client = oandapyV20.API(access_token=...)
>>> params =
{
    "instrument": "EUR_USD",
```

```
"period": 86400
}
```

```
>>> r = labs.Calendar(params=params)
>>> client.request(r)
>>> print(r.response)
```

Output:

```
[
    "impact": 3,
    "currency": "USD",
    "actual": "60.8",
    "market": "58.7",
    "title": "ISM Manufacturing",
    "timestamp": 1519916400,
    "region": "americas",
    "previous": "59.1",
    "unit": "index",
    "forecast": "59.5"
}
```

3.2.3 CommitmentsOfTraders

class oandapyV20.endpoints.forexlabs.CommitmentsOfTraders (params)

Bases: oandapyV20.endpoints.forexlabs.ForexLabs

CommitmentsOfTraders.

Get the 'commitments of traders' information for an instrument.

```
ENDPOINT = 'labs/v1/commitments_of_traders'
EXPECTED_STATUS = 200
METHOD = 'GET'
__init__(params)
```

Instantiate a CommitmentsOfTraders request.

Parameters params (dict (required)) – query params to send, check developer.oanda.com for details.

```
>>> r = labs.CommitmentOfTraders(params=params)
>>> client.request(r)
>>> print(r.response)
```

Output:

```
"EUR_USD": [
  {
    "oi": "603460",
    "price": "1.2315925",
    "ncs": "109280",
    "ncl": "258022",
    "date": 1517288400,
    "unit": "Contracts Of EUR 125,000"
  },
    "oi": "596937",
    "price": "1.2364",
    "ncs": "110546",
    "ncl": "251369",
    "date": 1517893200,
    "unit": "Contracts Of EUR 125,000"
  },
    "oi": "564233",
    "price": "1.2330275",
   "ncs": "103496",
    "ncl": "230785",
    "date": 1518498000,
    "unit": "Contracts Of EUR 125,000"
  },
    "oi": "567534",
    "price": "1.2346025",
    "ncs": "103147",
    "ncl": "229273",
    "date": 1519102800,
    "unit": "Contracts Of EUR 125,000"
    "oi": "567463",
    "price": "1.23557",
    "ncs": "100310",
    "ncl": "238287",
    "date": 1519707600,
    "unit": "Contracts Of EUR 125,000"
]
```

3.2.4 HistoricalPositionRatios

```
class oandapyV20.endpoints.forexlabs.HistoricalPositionRatios(params)
    Bases: oandapyV20.endpoints.forexlabs.ForexLabs
```

HistoricalPositionRatios.

Get the historical position ratios for an instrument.

```
ENDPOINT = 'labs/v1/historical_position_ratios'
```

Parameters params (dict (required)) – query params to send, check developer.oanda.com for details.

```
>>> r = labs.HistoricalPositionRatios(params=params)
>>> client.request(r)
>>> print(r.response)
```

```
"data": {
  "EUR_USD": {
    "data": [
      [
        1519924801,
        44.22,
        1.2209
      ],
        1519926001,
        44.33,
        1.221
      ],
        1519927200,
        44.16,
        1.2212
      ],
        1519928400,
        44.2,
        1.2209
      ],
        1519929601,
        43.88,
        1.2201
        1519930800,
        44.15,
        1.2197
```

```
],
[
1519932000,
 44.51,
 1.2204
],
 1519933200,
 44.55,
 1.2233
],
 1519934401,
 44.55,
 1.2254
],
 1519935601,
 44.08,
 1.226
],
 1519936801,
 43.67,
 1.2264
  1519938001,
 43.55,
  1.2263
],
 1519939201,
 43.25,
 1.2261
],
 1519940401,
 43.28,
 1.2263
 1519941601,
 43.39,
 1.2267
],
  1519942801,
 43.69,
 1.227
],
  1519944001,
  43.57,
  1.2269
],
  1519945201,
```

```
43.68,
  1.2272
],
 1519946400,
 43.51,
 1.2268
],
 1519947601,
 43.53,
 1.2267
 1519948801,
 43.71,
 1.2271
],
 1519950001,
43.66,
 1.2265
],
 1519951201,
 43.78,
 1.2269
],
1519952401,
 43.86,
 1.2273
],
 1519953600,
 43.85,
 1.2273
],
 1519954800,
 43.81,
 1.2271
],
 1519956001,
 44,
 1.2275
],
 1519957200,
 43.89,
 1.2274
 1519958401,
 43.95,
 1.2273
],
```

```
1519959601,
 43.93,
 1.2273
],
 1519960800,
 43.86,
 1.2276
],
 1519962000,
 44.02,
 1.2278
],
 1519963200,
 44.18,
 1.228
],
 1519964401,
 44.52,
 1.2283
],
 1519965600,
 44.19,
 1.2281
],
 1519966801,
 44.14,
 1.2278
],
[
 1519968000,
 43.93,
 1.2276
 1519969201,
 43.82,
 1.2277
],
 1519970401,
 43.77,
 1.2279
],
 1519971601,
 43.02,
 1.2269
],
 1519972801,
 42.99,
```

```
1.2265
],
[
 1519974001,
 42.73,
 1.2263
],
  1519975201,
 42.22,
 1.2262
],
 1519976400,
 42.13,
 1.2255
],
 1519977601,
 42.02,
 1.2263
],
  1519978801,
 42.15,
 1.2261
 1519980000,
 42.5,
 1.2273
],
 1519981201,
 42.2,
 1.2274
],
 1519982400,
 42.06,
 1.2271
],
  1519983600,
 42.38,
 1.2279
],
 1519984800,
 42.29,
  1.2276
  1519986000,
 42.16,
  1.2281
],
[
```

```
1519987201,
  43.46,
  1.2291
],
[
 1519988401,
 43.51,
 1.2291
],
[
 1519989601,
 43.4,
 1.2317
 1519990800,
 43.46,
 1.2317
],
 1519992001,
 43.07,
 1.2304
],
 1519993201,
 43.56,
  1.2316
],
 1519994401,
 43.75,
 1.2319
],
 1519995601,
 43.15,
 1.2308
],
 1519996801,
 42.94,
 1.2309
],
  1519998001,
 42.99,
 1.2315
],
  1519999201,
 42.33,
  1.2309
],
  1520000400,
 41.93,
  1.2299
```

```
],
      [
        1520001601,
       42.31,
        1.2303
      ],
        1520002801,
       42.5,
        1.2313
      ],
        1520004000,
        42.8,
        1.2326
        1520005201,
       42.67,
       1.2317
      ],
        1520006401,
        42.29,
       1.2309
        1520007600,
        42.33,
        1.2309
      ],
        1520008800,
       42.63,
        1.2321
      ],
        1520010001,
        42.11,
        1.2314
    ],
    "label": "EUR/USD"
}
```

3.2.5 OrderbookData

```
class oandapyV20.endpoints.forexlabs.OrderbookData(params)
    Bases: oandapyV20.endpoints.forexlabs.ForexLabs
    OrderbookData.
    Get the 'orderbook data' for an instrument.
    ENDPOINT = 'labs/v1/orderbook_data'
```

```
EXPECTED_STATUS = 200

METHOD = 'GET'

__init__ (params)

Instantiate an OrderbookData request.
```

Parameters params (dict (required)) – query params to send, check developer.oanda.com for details.

```
>>> r = labs.CommitmentOfTraders(params=params)
>>> client.request(r)
>>> print(r.response)
```

```
"1520066400": {
  "rate": 1.2318,
  "price_points": {
    "1.23": {
      "ps": 1.2155,
      "ol": 0.3871,
      "os": 0.2615,
      "pl": 0.5633
    "1.223": {
      "ps": 1.1266,
      "ol": 0.5021,
      "os": 0.2197,
      "pl": 0.3854
    "1.288": {
      "ps": 0,
      "ol": 0.0105,
      "os": 0.0105,
      "pl": 0
    "1.22": {
      "ps": 0.9191,
      "ol": 0.6486,
      "os": 0.136,
      "pl": 0.2965
    "1.2245": {
      "ps": 0.5336,
      "ol": 0.5021,
      "os": 0.3975,
      "pl": 0.4447
```

```
"1.1825": {
      "ps": 0.1779,
      "ol": 0.1465,
      "os": 0.0628,
      "pl": 0
    "1.2085": {
      "ps": 0.1482,
      "ol": 0.2092,
      "os": 0.2197,
      "pl": 0.1482
    "1.26": {
      "ps": 0,
      "ol": 0.2197,
      "os": 0.68,
      "pl": 0
    "1.25": {
      "ps": 0.0593,
      "ol": 0.272,
      "os": 1.0566,
      "pl": 0.1186
    "1.24": {
      "ps": 0.1186,
      "ol": 0.4289,
      "os": 0.8264,
      "pl": 0.4447
   }
}
```

3.2.6 Spreads

```
class oandapyV20.endpoints.forexlabs.Spreads(params)
   Bases: oandapyV20.endpoints.forexlabs.ForexLabs
   Spreads.
   Get the spread information for an instrument.
   ENDPOINT = 'labs/v1/spreads'
   EXPECTED_STATUS = 200
   METHOD = 'GET'
   __init__(params)
   Instantiate a Spreads request.

   Parameters params (dict (required)) - query params to send, check developer.oanda.com for details.

>>> import oandapyV20
>>> import oandapyV20.endpoints.forexlabs as labs
```

>>> accountID = ...

```
>>> r = labs.Spreads(params=params)
>>> client.request(r)
>>> print(r.response)
```

Output:

3.3 oandapyV20.endpoints.instruments

3.3.1 InstrumentsCandles

Parameters

• instrument (string (required)) - the instrument to fetch candle data for

• params (dict) - optional request query parameters, check developer.oanda.com for details

Params example:

```
{
    "count": 5,
    "granularity": "M5"
}
```

Candle data example:

```
>>> import oandapyV20
>>> import oandapyV20.endpoints.instruments as instruments
>>> client = oandapyV20.API(access_token=...)
>>> params = ...
>>> r = instruments.InstrumentsCandles(instrument="DE30_EUR",
>>> params=params)
>>> client.request(r)
>>> print r.response
```

```
"candles": [
  {
    "volume": 132,
    "mid": {
     "h": "10508.0",
     "c": "10506.0",
     "1": "10503.8",
      "o": "10503.8"
   },
    "complete": true,
    "time": "2016-10-17T19:35:00.00000000Z"
  },
    "volume": 162,
    "mid": {
     "h": "10507.0",
     "c": "10504.9",
     "1": "10502.0",
     "o": "10506.0"
   },
    "complete": true,
    "time": "2016-10-17T19:40:00.00000000Z"
  },
    "volume": 196,
    "mid": {
     "h": "10509.8",
      "c": "10505.0",
      "1": "10502.6",
      "o": "10504.9"
    },
    "complete": true,
    "time": "2016-10-17T19:45:00.00000000Z"
  },
  {
```

```
"volume": 153,
    "mid": {
      "h": "10510.1",
      "c": "10509.0",
      "1": "10504.2",
      "o": "10505.0"
    "complete": true,
    "time": "2016-10-17T19:50:00.00000000Z"
  },
    "volume": 172,
    "mid": {
      "h": "10509.8",
      "c": "10507.8",
      "1": "10503.2",
      "o": "10509.0"
    "complete": true,
    "time": "2016-10-17T19:55:00.00000000Z"
  }
],
"granularity": "M5",
"instrument": "DE30/EUR"
```

3.3.2 InstrumentsOrderBook

Parameters

- instrument (string (required)) the instrument to fetch candle data for
- params (dict) optional request query parameters, check developer.oanda.com for details

Params example:

```
{}
```

OrderBook data example:

```
>>> import oandapyV20
>>> import oandapyV20.endpoints.instruments as instruments
>>> client = oandapyV20.API(access_token=...)
```

```
>>> params = ...
>>> r = instruments.InstrumentsOrderBook(instrument="EUR_USD",
>>> params=params)
>>> client.request(r)
>>> print r.response
```

```
"orderBook": {
  "instrument": "EUR_USD",
  "buckets": [
    {
      "price": "1.12850",
      "shortCountPercent": "0.2352",
      "longCountPercent": "0.2666"
    },
      "price": "1.12900",
      "shortCountPercent": "0.2195",
      "longCountPercent": "0.3293"
    },
      "price": "1.12950",
      "shortCountPercent": "0.3136",
      "longCountPercent": "0.2901"
    },
      "price": "1.13000",
      "shortCountPercent": "0.3842",
      "longCountPercent": "0.4156"
    },
    {
      "price": "1.13050",
      "shortCountPercent": "0.1960",
      "longCountPercent": "0.3685"
    },
      "price": "1.13100",
      "shortCountPercent": "0.2431",
      "longCountPercent": "0.2901"
    },
      "price": "1.13150",
      "shortCountPercent": "0.2509",
      "longCountPercent": "0.3136"
    },
      "price": "1.13200",
      "shortCountPercent": "0.2587",
      "longCountPercent": "0.3450"
    },
    {
      "price": "1.13250",
      "shortCountPercent": "0.3842",
      "longCountPercent": "0.2666"
    },
    {
```

```
"price": "1.13300",
  "shortCountPercent": "0.3371",
  "longCountPercent": "0.3371"
},
  "price": "1.13350",
  "shortCountPercent": "0.3528",
  "longCountPercent": "0.2744"
},
{
  "price": "1.13400",
  "shortCountPercent": "0.3842",
  "longCountPercent": "0.3136"
},
{
  "price": "1.13450",
  "shortCountPercent": "0.2039",
  "longCountPercent": "0.2744"
},
  "price": "1.13500",
  "shortCountPercent": "0.1882",
  "longCountPercent": "0.3371"
},
{
  "price": "1.13550",
  "shortCountPercent": "0.0235",
  "longCountPercent": "0.0392"
},
  "price": "1.13600",
  "shortCountPercent": "0.0549",
  "longCountPercent": "0.0314"
},
{
  "price": "1.13650",
  "shortCountPercent": "0.1333",
  "longCountPercent": "0.0314"
},
  "price": "1.13700",
  "shortCountPercent": "0.1176",
  "longCountPercent": "0.1019"
},
  "price": "1.13750",
  "shortCountPercent": "0.1568",
  "longCountPercent": "0.0784"
},
  "price": "1.13800",
  "shortCountPercent": "0.1176",
  "longCountPercent": "0.0862"
},
  "price": "1.13850",
  "shortCountPercent": "0.2117",
  "longCountPercent": "0.1960"
```

38

```
},
  {
    "price": "1.13900",
    "shortCountPercent": "0.4548",
    "longCountPercent": "0.2587"
  },
    "price": "1.13950",
    "shortCountPercent": "0.2979",
    "longCountPercent": "0.3215"
  },
    "price": "1.14000",
    "shortCountPercent": "0.7449",
    "longCountPercent": "0.2901"
  },
  {
    "price": "1.14050",
    "shortCountPercent": "0.2117",
    "longCountPercent": "0.1176"
  },
    "price": "1.14100",
    "shortCountPercent": "0.1960",
    "longCountPercent": "0.1333"
  },
    "price": "1.14150",
    "shortCountPercent": "0.1882",
    "longCountPercent": "0.1176"
  }
],
"time": "2017-06-28T10:00:00Z",
"price": "1.13609",
"bucketWidth": "0.00050"
```

3.3.3 InstrumentsPositionBook

• instrument (string (required)) - the instrument to fetch candle data for

• params (dict) - optional request query parameters, check developer.oanda.com for details

Params example:

```
()
```

PositionBook data example:

```
"positionBook": {
 "instrument": "EUR_USD",
  "buckets": [
   {
      "price": "1.12800",
     "shortCountPercent": "0.2670",
      "longCountPercent": "0.2627"
   },
    {
     "price": "1.12850",
     "shortCountPercent": "0.2034",
      "longCountPercent": "0.2712"
   },
    {
      "price": "1.12900",
      "shortCountPercent": "0.2034",
      "longCountPercent": "0.2161"
   },
      "price": "1.12950",
      "shortCountPercent": "0.2670",
      "longCountPercent": "0.2839"
   },
    {
      "price": "1.13000",
      "shortCountPercent": "0.2755",
      "longCountPercent": "0.3221"
   },
      "price": "1.13050",
      "shortCountPercent": "0.1949",
      "longCountPercent": "0.2839"
   },
     "price": "1.13100",
      "shortCountPercent": "0.2288",
      "longCountPercent": "0.2712"
   },
```

```
"price": "1.13150",
  "shortCountPercent": "0.2416",
  "longCountPercent": "0.2712"
},
  "price": "1.13200",
  "shortCountPercent": "0.2204",
  "longCountPercent": "0.3178"
},
{
  "price": "1.13250",
  "shortCountPercent": "0.2543",
  "longCountPercent": "0.2458"
},
{
  "price": "1.13300",
  "shortCountPercent": "0.2839",
  "longCountPercent": "0.2585"
},
  "price": "1.13350",
  "shortCountPercent": "0.3602",
  "longCountPercent": "0.3094"
},
  "price": "1.13400",
  "shortCountPercent": "0.2882",
  "longCountPercent": "0.3560"
},
  "price": "1.13450",
  "shortCountPercent": "0.2500",
  "longCountPercent": "0.3009"
},
{
  "price": "1.13500",
  "shortCountPercent": "0.1738",
  "longCountPercent": "0.3475"
},
  "price": "1.13550",
  "shortCountPercent": "0.2119",
  "longCountPercent": "0.2797"
},
  "price": "1.13600",
  "shortCountPercent": "0.1483",
  "longCountPercent": "0.3094"
},
  "price": "1.13650",
  "shortCountPercent": "0.1483",
  "longCountPercent": "0.1314"
},
  "price": "1.13700",
  "shortCountPercent": "0.1568",
```

```
"longCountPercent": "0.2034"
  },
  {
    "price": "1.13750",
    "shortCountPercent": "0.1398",
    "longCountPercent": "0.1271"
  },
  {
    "price": "1.13800",
    "shortCountPercent": "0.1314",
    "longCountPercent": "0.2034"
  },
    "price": "1.13850",
    "shortCountPercent": "0.1483",
    "longCountPercent": "0.1695"
  },
    "price": "1.13900",
    "shortCountPercent": "0.2924",
    "longCountPercent": "0.1653"
  },
    "price": "1.13950",
    "shortCountPercent": "0.1526",
    "longCountPercent": "0.1865"
  },
    "price": "1.14000",
    "shortCountPercent": "0.4365",
    "longCountPercent": "0.2034"
  },
    "price": "1.14050",
    "shortCountPercent": "0.1398",
    "longCountPercent": "0.1144"
  }
],
"time": "2017-06-28T10:00:00Z",
"price": "1.13609",
"bucketWidth": "0.00050"
```

3.4 oandapyV20.endpoints.orders

3.4.1 OrderCancel

```
class oandapyV20.endpoints.orders.OrderCancel (accountID, orderID)
    Bases: oandapyV20.endpoints.orders.Orders
    Cancel a pending Order in an Account.
    ENDPOINT = 'v3/accounts/{accountID}/orders/{orderID}/cancel'
    EXPECTED_STATUS = 200
```

Parameters

- account ID (string (required)) id of the account to perform the request on.
- orderID (string (required)) id of the account to perform the request on.

Example:

```
>>> import oandapyV20
>>> import oandapyV20.endpoints.orders as orders
>>> client = oandapyV20.API(access_token=...)
>>> r = orders.OrderCancel(accountID= ..., orderID=...)
>>> client.request(r)
>>> print r.response
```

Output:

```
{
   "orderCancelTransaction": {
      "orderID": "2307",
      "clientOrderID": "myID",
      "reason": "CLIENT_REQUEST",
      "batchID": "2308",
      "time": "2016-10-25T20:53:03.789670387Z",
      "type": "ORDER_CANCEL",
      "userID": 1435156,
      "id": "2308",
      "accountID": "101-004-1435156-001"
    },
     "lastTransactionID": "2308",
      "relatedTransactionIDs": [
      "2308"
    ]
}
```

3.4.2 OrderClientExtensions

class oandapyV20.endpoints.orders.OrderClientExtensions (accountID, orderID, data)
 Bases: oandapyV20.endpoints.orders.Orders

Update the Client Extensions for an Order in an Account.

Warning: Do not set, modify or delete clientExtensions if your account is associated with MT4.

```
ENDPOINT = 'v3/accounts/{accountID}/orders/{orderID}/clientExtensions'
EXPECTED_STATUS = 200
HEADERS = {'Content-Type': 'application/json'}
METHOD = 'PUT'
__init__(accountID, orderID, data)
    Instantiate an OrderCreate request.
```

Parameters

- accountID (string (required)) id of the account to perform the request on.
- orderID (string (required)) id of the order to perform the request on.
- data (JSON (required)) json orderbody to send

Orderbody example:

```
{
  "clientExtensions": {
    "comment": "myComment",
    "id": "myID"
    }
}
```

```
>>> import oandapyV20
>>> import oandapyV20.endpoints.orders as orders
>>> client = oandapyV20.API(access_token=...)
>>> r = orders.OrderClientExtensions(accountID, orderID, data=data)
>>> client.request(r)
>>> print r.response
```

3.4.3 OrderCreate

```
class oandapyV20.endpoints.orders.OrderCreate(accountID, data)
   Bases: oandapyV20.endpoints.orders.Orders

   Create an Order for an Account.

ENDPOINT = 'v3/accounts/{accountID}/orders'

EXPECTED_STATUS = 201

HEADERS = {'Content-Type': 'application/json'}

METHOD = 'POST'
```

__init__ (accountID, data)

Instantiate an OrderCreate request.

Parameters

- accountID (string (required)) id of the account to perform the request on.
- data (JSON (required)) json orderbody to send

Orderbody example:

```
{
   "order": {
      "price": "1.2",
      "stopLossOnFill": {
        "timeInForce": "GTC",
        "price": "1.22"
      },
      "timeInForce": "GTC",
      "instrument": "EUR_USD",
      "units": "-100",
      "type": "LIMIT",
      "positionFill": "DEFAULT"
    }
}
```

```
>>> import oandapyV20
>>> import oandapyV20.endpoints.orders as orders
>>> client = oandapyV20.API(access_token=...)
>>> r = orders.OrderCreate(accountID, data=data)
>>> client.request(r)
>>> print r.response
```

```
"orderCreateTransaction": {
 "price": "1.20000",
  "stopLossOnFill": {
   "timeInForce": "GTC",
   "price": "1.22000"
  },
  "timeInForce": "GTC",
  "reason": "CLIENT_ORDER",
  "id": "2304",
  "batchID": "2304",
  "triggerCondition": "TRIGGER_DEFAULT",
  "positionFill": "DEFAULT",
  "userID": 1435156,
  "instrument": "EUR_USD",
  "time": "2016-10-24T21:48:18.593753865Z",
  "units": "-100",
  "type": "LIMIT_ORDER",
  "accountID": "101-004-1435156-001"
},
"lastTransactionID": "2304",
"relatedTransactionIDs": [
  "2304"
```

3.4.4 OrderDetails

```
class oandapyV20.endpoints.orders.OrderDetails(accountID, orderID)
   Bases: oandapyV20.endpoints.orders.Orders
   Get details for a single Order in an Account.

ENDPOINT = 'v3/accounts/{accountID}/orders/{orderID}'

EXPECTED_STATUS = 200

METHOD = 'GET'

__init__(accountID, orderID)
   Instantiate an OrderDetails request.
```

Parameters

- accountID (string (required)) id of the account to perform the request on.
- **orderID** (*string* (*required*)) id of the order to perform the request on.

```
>>> import oandapyV20
>>> import oandapyV20.endpoints.orders as orders
>>> client = oandapyV20.API(access_token=...)
>>> r = orders.OrderDetails(accountID=..., orderID=...)
>>> client.request(r)
>>> print r.response
```

Output:

```
"order": {
 "partialFill": "DEFAULT_FILL",
 "price": "1.20000",
 "stopLossOnFill": {
   "timeInForce": "GTC",
   "price": "1.22000"
 },
 "timeInForce": "GTC",
 "createTime": "2016-10-25T21:07:21.065554321Z",
 "triggerCondition": "TRIGGER_DEFAULT",
 "positionFill": "POSITION_DEFAULT",
 "id": "2309",
 "instrument": "EUR_USD",
 "state": "PENDING",
  "units": "-100",
  "type": "LIMIT"
},
"lastTransactionID": "2309"
```

3.4.5 OrderList

```
class oandapyV20.endpoints.orders.OrderList(accountID, params=None)
    Bases: oandapyV20.endpoints.orders.Orders

Create an Order for an Account.

ENDPOINT = 'v3/accounts/{accountID}/orders'
```

Parameters

- account ID (string (required)) id of the account to perform the request on.
- params (dict) optional request query parameters, check developer.oanda.com for details

Example:

```
>>> import oandapyV20
>>> import oandapyV20.endpoints.orders as orders
>>> client = oandapyV20.API(access_token=...)
>>> r = orders.OrderList(accountID)
>>> client.request(r)
>>> print r.response
```

Output:

```
"orders": [
  {
    "partialFill": "DEFAULT_FILL",
    "price": "1.20000",
    "stopLossOnFill": {
     "timeInForce": "GTC",
      "price": "1.22000"
    },
    "timeInForce": "GTC",
    "createTime": "2016-10-05T10:25:47.627003645Z",
    "triggerCondition": "TRIGGER_DEFAULT",
    "positionFill": "POSITION_DEFAULT",
    "id": "2125",
    "instrument": "EUR_USD",
    "state": "PENDING",
    "units": "-100",
    "type": "LIMIT"
],
"lastTransactionID": "2129"
```

3.4.6 OrderReplace

```
class oandapyV20.endpoints.orders.OrderReplace (accountID, orderID, data)
    Bases: oandapyV20.endpoints.orders.Orders
    OrderReplace.
    Replace an Order in an Account by simultaneously cancelling it and createing a replacement Order.
    ENDPOINT = 'v3/accounts/{accountID}/orders/{orderID}'
    EXPECTED_STATUS = 201
```

Parameters

- account ID (string (required)) id of the account to perform the request on.
- orderID (string (required)) id of the order to perform the request on.
- data (JSON (required)) json orderbody to send

Orderbody example:

```
{
  "order": {
    "units": "-500000",
    "instrument": "EUR_USD",
    "price": "1.25000",
    "type": "LIMIT"
  }
}
```

```
>>> import oandapyV20
>>> import oandapyV20.endpoints.orders as orders
>>> client = oandapyV20.API(access_token=...)
>>> data =

{
    "order": {
        "units": "-500000",
        "instrument": "EUR_USD",
        "price": "1.25000",
        "type": "LIMIT"
        }
    }
}
```

```
>>> r = orders.OrderReplace(accountID=..., orderID=..., data=data)
>>> client.request(r)
>>> print r.response
```

```
"orderCreateTransaction": {
    "price": "1.25000",
    "timeInForce": "GTC",
    "reason": "REPLACEMENT",
    "clientExtensions": {
        "comment": "myComment",
        "id": "myID"
    },
    "id": "2307",
    "batchID": "2306",
    "triggerCondition": "TRIGGER_DEFAULT",
    "replacesOrderID": "2304",
    "positionFill": "DEFAULT",
    "userID": 1435156,
```

```
"instrument": "EUR_USD",
  "time": "2016-10-25T19:45:38.558056359Z",
 "units": "-500000",
 "type": "LIMIT_ORDER",
 "accountID": "101-004-1435156-001"
"orderCancelTransaction": {
 "orderID": "2304",
 "clientOrderID": "myID",
 "reason": "CLIENT_REQUEST_REPLACED",
 "batchID": "2306",
 "time": "2016-10-25T19:45:38.558056359Z",
 "type": "ORDER_CANCEL",
 "replacedByOrderID": "2307",
  "userID": 1435156,
  "id": "2306",
  "accountID": "101-004-1435156-001"
"lastTransactionID": "2307",
"relatedTransactionIDs": [
 "2306",
  "2307"
]
```

3.4.7 OrdersPending

```
class oandapyV20.endpoints.orders.OrdersPending(accountID)
    Bases: oandapyV20.endpoints.orders.Orders

List all pending Orders in an Account.

ENDPOINT = 'v3/accounts/{accountID}/pendingOrders'

EXPECTED_STATUS = 200

METHOD = 'GET'

__init__(accountID)

Instantiate an OrdersPending request.
```

Parameters accountID(string (required))—id of the account to perform the request on.

Example:

```
>>> import oandapyV20
>>> import oandapyV20.endpoints.orders as orders
>>> client = oandapyV20.API(access_token=...)
>>> r = orders.OrdersPending(accountID)
>>> client.request(r)
>>> print r.response
```

```
{
    "orders": [
    {
```

```
"partialFill": "DEFAULT_FILL",
    "price": "1.20000",
    "stopLossOnFill": {
      "timeInForce": "GTC",
      "price": "1.22000"
    "timeInForce": "GTC",
    "clientExtensions": {
      "comment": "myComment",
      "id": "myID"
    "id": "2304",
    "triggerCondition": "TRIGGER_DEFAULT",
    "positionFill": "POSITION_DEFAULT",
    "createTime": "2016-10-24T21:48:18.593753865Z",
    "instrument": "EUR_USD",
    "state": "PENDING",
    "units": "-100",
    "type": "LIMIT"
],
"lastTransactionID": "2305"
```

3.5 oandapyV20.endpoints.positions

3.5.1 OpenPositions

OpenPositions.

List all open Positions for an Account. An open Position is a Position in an Account that currently has a Trade opened for it.

```
ENDPOINT = 'v3/accounts/{accountID}/openPositions'
EXPECTED_STATUS = 200

METHOD = 'GET'
__init__(accountID)
    Instantiate an OpenPositions request.
```

Parameters accountID (string (required)) - id of the account to perform the request on.

```
>>> import oandapyV20
>>> import oandapyV20.endpoints.positions as positions
>>> accountID = ...
>>> client = oandapyV20.API(access_token=...)
>>> r = positions.OpenPositions(accountID=accountID)
>>> client.request(r)
>>> print r.response
```

```
"positions": [
  {
    "short": {
      "units": "0",
      "resettablePL": "-14164.3000",
      "unrealizedPL": "0.0000",
      "pl": "-14164.3000"
    },
    "unrealizedPL": "-284.0000",
    "long": {
      "unrealizedPL": "-284.0000",
      "tradeIDs": [
        "2315"
      ],
      "resettablePL": "404.5000",
      "units": "10",
      "averagePrice": "10678.3",
      "pl": "404.5000"
    "instrument": "DE30_EUR",
    "resettablePL": "-13759.8000",
    "pl": "-13759.8000"
  },
    "short": {
      "unrealizedPL": "-0.0738",
      "tradeIDs": [
       "2323"
      ],
      "resettablePL": "0.0000",
      "units": "-100",
      "averagePrice": "1.09843",
      "pl": "0.0000"
    "unrealizedPL": "-0.0738",
    "long": {
      "units": "0",
      "resettablePL": "-44.6272",
      "unrealizedPL": "0.0000",
      "pl": "-44.6272"
    },
    "instrument": "EUR_USD",
    "resettablePL": "-44.6272",
    "pl": "-44.6272"
],
"lastTransactionID": "2327"
```

3.5.2 PositionClose

class oandapyV20.endpoints.positions.**PositionClose** (*accountID*, *instrument*, *data*)

Bases: oandapyV20.endpoints.positions.Positions

Closeout the open Position regarding instrument in an Account.

```
ENDPOINT = 'v3/accounts/{accountID}/positions/{instrument}/close'
EXPECTED_STATUS = 200
HEADERS = {'Content-Type': 'application/json'}
METHOD = 'PUT'
__init__(accountID, instrument, data)
    Instantiate a PositionClose request.
```

Parameters

- accountID (string (required)) id of the account to perform the request on.
- instrument (string (required)) instrument to close partially or fully.
- data (dict (required)) closeout specification data to send, check developer.oanda.com for details.

Data body example:

```
{
  "longUnits": "ALL"
}
```

```
"longOrderCreateTransaction": {
    "longPositionCloseout": {
        "units": "ALL",
        "instrument": "EUR_USD"
    },
    "batchID": "6390",
    "reason": "POSITION_CLOSEOUT",
    "id": "6390",
    "timeInForce": "FOK",
    "positionFill": "REDUCE_ONLY",
    "userID": "<USERID>",
    "instrument": "EUR_USD",
    "time": "2016-06-22T18:41:35.034041665Z",
    "units": "-251",
    "type": "MARKET_ORDER",
    "accountID": "<ACCOUNT>"
```

```
"relatedTransactionIDs": [
  "6390",
  "6391"
],
"lastTransactionID": "6391",
"longOrderFillTransaction": {
  "price": "1.13018",
  "batchID": "6390",
  "accountBalance": "43650.69807",
  "reason": "MARKET_ORDER_POSITION_CLOSEOUT",
  "tradesClosed": [
    {
      "units": "-1",
      "financing": "0.00000",
      "realizedPL": "-0.00013",
      "tradeID": "6383"
    },
    {
      "units": "-250",
      "financing": "0.00000",
      "realizedPL": "-0.03357",
      "tradeID": "6385"
    }
  ],
  "id": "6391",
  "orderID": "6390",
  "financing": "0.00000",
  "userID": "<USERID>",
  "instrument": "EUR_USD",
  "time": "2016-06-22T18:41:35.034041665Z",
  "units": "-251",
  "type": "ORDER_FILL",
  "pl": "-0.03370",
  "accountID": "<ACCOUNT>"
```

3.5.3 PositionDetails

```
class oandapyV20.endpoints.positions.PositionDetails (accountID, instrument)
    Bases: oandapyV20.endpoints.positions.Positions

PositionDetails.

Get the details of a single instrument's position in an Account. The position may be open or not.

ENDPOINT = 'v3/accounts/{accountID}/positions/{instrument}'

EXPECTED_STATUS = 200

METHOD = 'GET'

__init__ (accountID, instrument)
    Instantiate a PositionDetails request.

Parameters
```

• account ID (string (required)) – id of the account to perform the request on.

• instrument (string (required)) - id of the instrument to get the position details for.

```
>>> import oandapyV20
>>> import oandapyV20.endpoints.positions as positions
>>> accountID = ...
>>> instrument = ...
>>> client = oandapyV20.API(access_token=...)
>>> r = positions.PositionDetails(accountID=accountID, instrument)
>>> client.request(r)
>>> print r.response
```

Output:

```
"position": {
 "short": {
   "unrealizedPL": "-0.0738",
   "tradeIDs": [
     "2323"
   ],
    "resettablePL": "0.0000",
   "units": "-100",
    "averagePrice": "1.09843",
    "pl": "0.0000"
  "unrealizedPL": "-0.0738",
  "long": {
    "units": "0",
    "resettablePL": "-44.6272",
    "unrealizedPL": "0.0000",
    "pl": "-44.6272"
 },
  "instrument": "EUR_USD",
  "resettablePL": "-44.6272",
  "pl": "-44.6272"
},
"lastTransactionID": "2327"
```

3.5.4 PositionList

```
class oandapyV20.endpoints.positions.PositionList(accountID)
    Bases: oandapyV20.endpoints.positions.Positions
```

PositionList.

List all Positions for an Account. The Positions returned are for every instrument that has had a position during the lifetime of the Account.

```
ENDPOINT = 'v3/accounts/{accountID}/positions'
EXPECTED_STATUS = 200
METHOD = 'GET'
__init__(accountID)
    Instantiate a PositionList request.
```

Parameters accountID(string (required))-id of the account to perform the request on.

```
>>> import oandapyV20
>>> import oandapyV20.endpoints.positions as positions
>>> accountID = ...
>>> client = oandapyV20.API(access_token=...)
>>> r = positions.PositionList(accountID=accountID)
>>> client.request(r)
>>> print r.response
```

```
"positions": [
 {
    "short": {
     "units": "0",
     "resettablePL": "-272.6805",
     "unrealizedPL": "0.0000",
     "pl": "-272.6805"
   "unrealizedPL": "0.0000",
   "long": {
     "units": "0",
     "resettablePL": "0.0000",
     "unrealizedPL": "0.0000",
     "pl": "0.0000"
   },
   "instrument": "EUR_GBP",
    "resettablePL": "-272.6805",
    "pl": "-272.6805"
 },
    "short": {
     "unrealizedPL": "870.0000",
     "tradeIDs": [
       "2121",
       "2123"
      "resettablePL": "-13959.3000",
      "units": "-20",
      "averagePrice": "10581.5",
     "pl": "-13959.3000"
   },
    "unrealizedPL": "870.0000",
    "long": {
     "units": "0",
     "resettablePL": "404.5000",
     "unrealizedPL": "0.0000",
     "pl": "404.5000"
   },
   "instrument": "DE30_EUR",
    "resettablePL": "-13554.8000",
    "pl": "-13554.8000"
  },
    "short": {
      "units": "0",
```

```
"resettablePL": "0.0000",
    "unrealizedPL": "0.0000",
    "pl": "0.0000"
},
    "unrealizedPL": "0.0000",
    "long": {
        "units": "0",
        "resettablePL": "-12.8720",
        "unrealizedPL": "0.0000",
        "pl": "-12.8720"
        },
        "instrument": "EUR_USD",
        "resettablePL": "-12.8720",
        "pl": "-12.8720"
        }
],
        "lastTransactionID": "2124"
}
```

3.6 oandapyV20.endpoints.pricing

3.6.1 PricingInfo

```
class oandapyV20.endpoints.pricing.PricingInfo(accountID, params=None)
    Bases: oandapyV20.endpoints.pricing.Pricing
Pricing.
Get pricing information for a specified list of Instruments within an account.

ENDPOINT = 'v3/accounts/{accountID}/pricing'

EXPECTED_STATUS = 200

METHOD = 'GET'

__init__(accountID, params=None)
    Instantiate a PricingStream APIRequest instance.
```

Parameters

- accountID (string (required)) the accountID of the account.
- params (dict (required)) parameters for the request, check developer.oanda.com for details.

Example

```
>>> import oandapyV20
>>> from oandapyV20 import API
>>> import oandapyV20.endpoints.pricing as pricing
>>> accountID = "..."
>>> api = API (access_token="...")
>>> params =
{
```

```
"instruments": "EUR_USD,EUR_JPY"
}
```

```
>>> r = pricing.PricingInfo(accountID=accountID, params=params)
>>> rv = api.request(r)
>>> print r.response
```

```
"prices": [
 {
   "status": "tradeable",
    "instrument": "EUR_USD",
    "quoteHomeConversionFactors": {
     "negativeUnits": "0.89160730",
     "positiveUnits": "0.89150397"
   },
    "asks": [
        "price": "1.12170",
        "liquidity": 10000000
      },
        "price": "1.12172",
        "liquidity": 10000000
   ],
    "time": "2016-10-05T05:28:16.729643492Z",
    "closeoutAsk": "1.12174",
    "bids": [
      {
        "price": "1.12157",
        "liquidity": 10000000
      },
        "price": "1.12155",
        "liquidity": 10000000
   ],
    "closeoutBid": "1.12153",
    "unitsAvailable": {
      "default": {
       "short": "506246",
        "long": "506128"
      "reduceOnly": {
        "short": "0",
        "long": "0"
      "openOnly": {
        "short": "506246",
        "long": "506128"
      "reduceFirst": {
        "short": "506246",
        "long": "506128"
```

```
},
 "status": "tradeable",
 "instrument": "EUR_JPY",
 "quoteHomeConversionFactors": {
   "negativeUnits": "0.00867085",
    "positiveUnits": "0.00866957"
 },
  "asks": [
   {
      "price": "115.346",
      "liquidity": 1000000
      "price": "115.347",
      "liquidity": 2000000
    },
      "price": "115.348",
      "liquidity": 5000000
   },
      "price": "115.350",
      "liquidity": 10000000
    }
 ],
  "time": "2016-10-05T05:28:15.621238671Z",
  "closeoutAsk": "115.350",
  "bids": [
      "price": "115.329",
      "liquidity": 1000000
    },
      "price": "115.328",
      "liquidity": 2000000
   },
      "price": "115.327",
      "liquidity": 5000000
   },
      "price": "115.325",
      "liquidity": 10000000
    }
 ],
  "closeoutBid": "115.325",
  "unitsAvailable": {
    "default": {
     "short": "506262",
      "long": "506112"
    },
    "reduceOnly": {
      "short": "0",
      "long": "0"
    },
    "openOnly": {
```

```
"short": "506262",
    "long": "506112"
    },
    "reduceFirst": {
        "short": "506262",
        "long": "506112"
    }
    }
    }
}
```

3.6.2 PricingStream

```
class oandapyV20.endpoints.pricing.PricingStream(accountID, params=None)
    Bases: oandapyV20.endpoints.pricing.Pricing
    PricingStream.

Get realtime pricing information for a specified list of Instruments.

ENDPOINT = 'v3/accounts/{accountID}/pricing/stream'
```

```
EXPECTED_STATUS = 200

METHOD = 'GET'

STREAM = True
```

__init__ (accountID, params=None)

Instantiate a PricingStream APIRequest instance.

Parameters

- accountID (string (required)) the accountID of the account.
- params (dict (required)) parameters for the request, check developer.oanda.com for details.

Example

```
>>> r = pricing.PricingStream(accountID=accountID, params=params)
>>> rv = api.request(r)
>>> maxrecs = 100
>>> for ticks in r:
>>> print json.dumps(R, indent=4),","
>>> if maxrecs == 0:
>>> r.terminate("maxrecs records received")
```

```
"status": "tradeable",
 "instrument": "EUR_JPY",
  "asks": [
      "price": "114.312",
      "liquidity": 1000000
    },
      "price": "114.313",
      "liquidity": 2000000
    },
      "price": "114.314",
      "liquidity": 5000000
    },
      "price": "114.316",
      "liquidity": 10000000
    }
  "time": "2016-10-27T08:38:43.094548890Z",
  "closeoutAsk": "114.316",
  "type": "PRICE",
  "closeoutBid": "114.291",
  "bids": [
     "price": "114.295",
     "liquidity": 1000000
   },
      "price": "114.294",
      "liquidity": 2000000
      "price": "114.293",
      "liquidity": 5000000
    },
      "price": "114.291",
     "liquidity": 10000000
 ]
},
{
 "type": "HEARTBEAT",
  "time": "2016-10-27T08:38:44.327443673Z"
},
{
 "status": "tradeable",
 "instrument": "EUR_USD",
  "asks": [
      "price": "1.09188",
      "liquidity": 10000000
```

```
},
    {
     "price": "1.09190",
      "liquidity": 10000000
   }
 ],
  "time": "2016-10-27T08:38:45.664613867Z",
  "closeoutAsk": "1.09192",
  "type": "PRICE",
  "closeoutBid": "1.09173",
  "bids": [
      "price": "1.09177",
      "liquidity": 10000000
    },
      "price": "1.09175",
      "liquidity": 10000000
 ]
},
{
  "status": "tradeable",
 "instrument": "EUR_JPY",
  "asks": [
      "price": "114.315",
      "liquidity": 1000000
   },
     "price": "114.316",
     "liquidity": 2000000
   },
      "price": "114.317",
     "liquidity": 5000000
    },
      "price": "114.319",
      "liquidity": 10000000
 ],
  "time": "2016-10-27T08:38:45.681572782Z",
  "closeoutAsk": "114.319",
  "type": "PRICE",
  "closeoutBid": "114.294",
  "bids": [
      "price": "114.298",
      "liquidity": 1000000
    },
      "price": "114.297",
      "liquidity": 2000000
    },
      "price": "114.296",
      "liquidity": 5000000
```

```
},
    {
        "price": "114.294",
        "liquidity": 10000000
      }
      ]
}
```

terminate (message=")

terminate the stream.

Calling this method will stop the generator yielding tickrecords. A message can be passed optionally.

3.7 oandapyV20.endpoints.trades

3.7.1 OpenTrades

```
class oandapyV20.endpoints.trades.OpenTrades(accountID)
   Bases: oandapyV20.endpoints.trades.Trades

Get the list of open Trades for an Account.

ENDPOINT = 'v3/accounts/{accountID}/openTrades'

EXPECTED_STATUS = 200

METHOD = 'GET'

__init__(accountID)

Instantiate an OpenTrades request.
```

Parameters accountID(string (required))—id of the account to perform the request on.

```
>>> import oandapyV20
>>> import oandapyV20.endpoints.trades as trades
>>> client = oandapyV20.API(access_token=...)
>>> r = trades.OpenTrades(accountID=...)
>>> client.request(r)
>>> print r.response
```

```
l,
  "lastTransactionID": "2317"
}
```

3.7.2 TradeCRCDO

```
class oandapyV20.endpoints.trades.TradeCRCDO (accountID, tradeID, data)
    Bases: oandapyV20.endpoints.trades.Trades

Trade Create Replace Cancel Dependent Orders.

ENDPOINT = 'v3/accounts/{accountID}/trades/{tradeID}/orders'

EXPECTED_STATUS = 200

HEADERS = {'Content-Type': 'application/json'}

METHOD = 'PUT'

__init__ (accountID, tradeID, data)
    Instantiate a TradeClientExtensions request.
```

Parameters

- account ID (string (required)) id of the account to perform the request on.
- tradeID (string (required)) id of the trade to update client extensions for.
- data(dict (required)) clientextension data to send, check developer.oanda.com for details.

Data body example:

```
{
  "takeProfit": {
    "timeInForce": "GTC",
    "price": "1.05"
},
  "stopLoss": {
    "timeInForce": "GTC",
    "price": "1.10"
}
}
```

```
>>> import oandapyV20
>>> import oandapyV20.endpoints.trades as trades
>>> accountID = ...
>>> tradeID = ...
>>> client = oandapyV20.API(access_token=...)
>>> data =

{
    "takeProfit": {
        "timeInForce": "GTC",
        "price": "1.05"
      },
      "stopLoss": {
        "timeInForce": "GTC",
        "price": "1.10"
      }
}
```

```
>>> r = trades.TradeCRCDO(accountID=accountID,
>>> tradeID=tradeID,
>>> data=data)
>>> client.request(r)
>>> print r.response
```

```
"lastTransactionID": "2327",
"stopLossOrderCancelTransaction": {
 "orderID": "2324",
  "batchID": "2325",
  "reason": "CLIENT_REQUEST_REPLACED",
  "time": "2016-10-28T21:00:19.978476830Z",
  "type": "ORDER_CANCEL",
  "replacedByOrderID": "2327",
  "userID": 1435156,
  "id": "2326",
  "accountID": "101-004-1435156-001"
"stopLossOrderTransaction": {
  "tradeID": "2323",
  "price": "1.10000",
  "timeInForce": "GTC",
  "reason": "REPLACEMENT",
  "id": "2327",
  "batchID": "2325",
  "triggerCondition": "TRIGGER_DEFAULT",
  "replacesOrderID": "2324",
  "userID": 1435156,
  "time": "2016-10-28T21:00:19.978476830Z",
  "cancellingTransactionID": "2326",
  "type": "STOP_LOSS_ORDER",
  "accountID": "101-004-1435156-001"
"relatedTransactionIDs": [
  "2325",
  "2326",
  "2327"
],
"takeProfitOrderTransaction": {
 "tradeID": "2323",
  "price": "1.05000",
 "timeInForce": "GTC",
 "reason": "CLIENT_ORDER",
  "id": "2325",
  "batchID": "2325",
  "triggerCondition": "TRIGGER_DEFAULT",
  "userID": 1435156,
  "time": "2016-10-28T21:00:19.978476830Z",
  "type": "TAKE_PROFIT_ORDER",
  "accountID": "101-004-1435156-001"
```

3.7.3 TradeClientExtensions

TradeClientExtensions.

Update the Client Extensions for a Trade. Do not add, update or delete the Client Extensions if your account is associated with MT4.

Parameters

- accountID (string (required)) id of the account to perform the request on.
- **tradeID** (*string* (*required*)) id of the trade to update client extensions for.
- data (dict (required)) clientextension data to send, check developer.oanda.com for details.

Data body example:

```
{
  "clientExtensions": {
    "comment": "myComment",
    "id": "myID2315"
  }
}
```

```
>>> import oandapyV20
>>> import oandapyV20.endpoints.trades as trades
>>> accountID = ...
>>> tradeID = ...
>>> client = oandapyV20.API(access_token=...)
>>> data =

{
    "clientExtensions": {
        "comment": "myComment",
        "id": "myID2315"
      }
}
```

```
>>> r = trades.TradeClientExtensions(accountID=accountID,
>>> tradeID=tradeID,
>>> data=data)
>>> client.request(r)
>>> print r.response
```

```
"tradeClientExtensionsModifyTransaction": {
 "batchID": "2319",
 "tradeID": "2315",
 "time": "2016-10-28T20:32:39.356516787Z",
 "tradeClientExtensionsModify": {
   "comment": "myComment",
   "id": "myID2315"
 },
  "type": "TRADE_CLIENT_EXTENSIONS_MODIFY",
  "userID": 1435156,
 "id": "2319",
  "accountID": "101-004-1435156-001"
"lastTransactionID": "2319",
"relatedTransactionIDs": [
  "2319"
1
```

3.7.4 TradeClose

class oandapyV20.endpoints.trades.TradeClose(accountID, tradeID, data=None)

Parameters

- accountID (string (required)) id of the account to perform the request on.
- tradeID (string (required)) id of the trade to close.
- data (dict (optional)) data to send, use this to close a trade partially. Check developer.oanda.com for details.

Data body example:

```
{
    "units": 100
}
```

```
>>> import oandapyV20
>>> import oandapyV20.endpoints.trades as trades
>>> client = oandapyV20.API(access_token=...)
>>> data =
{
```

```
"units": 100
}
```

```
>>> r = trades.TradeClose(accountID=..., data=data)
>>> client.request(r)
>>> print r.response
```

```
"orderFillTransaction": {
 "price": "1.09289",
  "batchID": "2316",
  "accountBalance": "33848.1208",
  "reason": "MARKET_ORDER_TRADE_CLOSE",
  "tradesClosed": [
   {
      "units": "-100",
      "financing": "0.0000",
      "realizedPL": "-0.1455",
      "tradeID": "2313"
   }
  ],
  "id": "2317",
  "orderID": "2316",
  "financing": "0.0000",
  "userID": 1435156,
  "instrument": "EUR_USD",
  "time": "2016-10-28T15:11:58.023004583Z",
  "units": "-100",
  "type": "ORDER_FILL",
  "pl": "-0.1455",
  "accountID": "101-004-1435156-001"
},
"orderCreateTransaction": {
  "timeInForce": "FOK",
  "reason": "TRADE_CLOSE",
  "tradeClose": {
   "units": "100",
   "tradeID": "2313"
  },
  "id": "2316",
  "batchID": "2316",
  "positionFill": "REDUCE_ONLY",
  "userID": 1435156,
  "instrument": "EUR_USD",
  "time": "2016-10-28T15:11:58.023004583Z",
  "units": "-100",
  "type": "MARKET_ORDER",
  "accountID": "101-004-1435156-001"
"lastTransactionID": "2317",
"relatedTransactionIDs": [
  "2316",
  "2317"
]
```

3.7.5 TradeDetails

```
class oandapyV20.endpoints.trades.TradeDetails(accountID, tradeID)
   Bases: oandapyV20.endpoints.trades.Trades
   Get the details of a specific Trade in an Account.
   ENDPOINT = 'v3/accounts/{accountID}/trades/{tradeID}'
   EXPECTED_STATUS = 200
   METHOD = 'GET'
   __init__(accountID, tradeID)
        Instantiate a TradeDetails request.
```

Parameters

- accountID (string (required)) id of the account to perform the request on.
- **tradeID** (*string* (*required*)) id of the trade.

```
>>> import oandapyV20
>>> import oandapyV20.endpoints.trades as trades
>>> client = oandapyV20.API(access_token=...)
>>> r = accounts.TradeDetails(accountID=..., tradeID=...)
>>> client.request(r)
>>> print r.response
```

Output:

```
{
  "lastTransactionID": "2317",
  "trade": {
      "instrument": "DE30_EUR",
      "financing": "0.0000",
      "openTime": "2016-10-28T14:28:05.231759081Z",
      "initialUnits": "10",
      "currentUnits": "10",
      "price": "10678.3",
      "unrealizedPL": "226.0000",
      "realizedPL": "0.0000",
      "state": "OPEN",
      "id": "2315"
    }
}
```

3.7.6 TradesList

```
class oandapyV20.endpoints.trades.TradesList(accountID, params=None)
    Bases: oandapyV20.endpoints.trades.Trades

Get a list of trades for an Account.

ENDPOINT = 'v3/accounts/{accountID}/trades'

EXPECTED_STATUS = 200

METHOD = 'GET'
```

__init__ (accountID, params=None)
Instantiate a TradesList request.

Parameters

- accountID (string (required)) id of the account to perform the request on.
- params (dict (optional)) query params to send, check developer.oanda.com for details.

Query Params example:

```
{
  "instrument": "DE30_EUR,EUR_USD"
}
```

```
>>> r = trades.TradesList(accountID=..., params=params)
>>> client.request(r)
>>> print r.response
```

```
"trades": [
   "instrument": "DE30_EUR",
    "financing": "0.0000",
    "openTime": "2016-10-28T14:28:05.231759081Z",
    "initialUnits": "10",
    "currentUnits": "10",
    "price": "10678.3",
    "unrealizedPL": "25.0000",
    "realizedPL": "0.0000",
    "state": "OPEN",
    "id": "2315"
  },
    "instrument": "EUR_USD",
    "financing": "0.0000",
    "openTime": "2016-10-28T14:27:19.011002322Z",
    "initialUnits": "100",
    "currentUnits": "100",
    "price": "1.09448",
    "unrealizedPL": "-0.0933",
    "realizedPL": "0.0000",
    "state": "OPEN",
    "id": "2313"
],
"lastTransactionID": "2315"
```

3.8 oandapyV20.endpoints.transactions

3.8.1 TransactionDetails

Parameters

- accountID (string (required)) id of the account to perform the request on.
- transactionID (string (required)) id of the transaction

```
>>> import oandapyV20
>>> import oandapyV20.endpoints.transactions as trans
>>> client = oandapyV20.API(access_token=...)
>>> r = trans.TransactionDetails(accountID=..., transactionID=...)
>>> client.request(r)
>>> print r.response
```

```
"transaction": {
 "price": "1.20000",
  "stopLossOnFill": {
   "timeInForce": "GTC",
   "price": "1.22000"
  "timeInForce": "GTC",
  "reason": "CLIENT_ORDER",
  "id": "2304",
  "batchID": "2304",
  "triggerCondition": "TRIGGER_DEFAULT",
  "positionFill": "DEFAULT",
  "userID": 1435156,
  "instrument": "EUR_USD",
  "time": "2016-10-24T21:48:18.593753865Z",
  "units": "-100",
  "type": "LIMIT_ORDER",
  "accountID": "101-004-1435156-001"
},
"lastTransactionID": "2311"
```

3.8.2 TransactionIDRange

Parameters

- accountID (string (required)) id of the account to perform the request on.
- params (dict (required)) query params to send, check developer.oanda.com for details.

Query Params example:

```
{
  "to": 2306,
  "from": 2304
}
```

```
>>> import oandapyV20
>>> import oandapyV20.endpoints.transactions as trans
>>> client = oandapyV20.API(access_token=...)
>>> params =
{
    "to": 2306,
    "from": 2304
}
```

```
>>> r = trans.TransactionIDRange(accountID=..., params=params)
>>> client.request(r)
>>> print r.response
```

```
"triggerCondition": "TRIGGER_DEFAULT",
    "positionFill": "DEFAULT",
    "userID": 1435156,
    "instrument": "EUR_USD",
    "time": "2016-10-24T21:48:18.593753865Z",
    "units": "-100",
    "type": "LIMIT_ORDER",
    "accountID": "101-004-1435156-001"
  },
    "orderID": "2304",
    "batchID": "2305",
    "clientExtensionsModify": {
      "comment": "myComment",
      "id": "myID"
    "time": "2016-10-25T15:56:43.075594239Z",
    "type": "ORDER_CLIENT_EXTENSIONS_MODIFY",
    "userID": 1435156,
    "id": "2305",
    "accountID": "101-004-1435156-001"
  },
    "orderID": "2304",
    "clientOrderID": "myID",
    "reason": "CLIENT_REQUEST_REPLACED",
    "batchID": "2306",
    "time": "2016-10-25T19:45:38.558056359Z",
    "type": "ORDER_CANCEL",
    "replacedByOrderID": "2307",
    "userID": 1435156,
    "id": "2306",
    "accountID": "101-004-1435156-001"
1
```

3.8.3 TransactionList

• accountID (string (required)) - id of the account to perform the request on.

• params (dict (optional)) - query params to send, check developer.oanda.com for details.

Query Params example:

```
{
    "pageSize": 200
}
```

```
>>> import oandapyV20
>>> import oandapyV20.endpoints.transactions as trans
>>> client = oandapyV20.API(access_token=...)
>>> r = trans.TransactionList(accountID) # params optional
>>> client.request(r)
>>> print r.response
```

```
"count": 2124,
 "from": "2016-06-24T21:03:50.914647476Z",
 "lastTransactionID": "2124",
 "pageSize": 100,
 "to": "2016-10-05T06:54:14.025946546Z",
 "pages": [
   "https://api-fxpractice.oanda.com/v3/accounts/101-004-1435156-001/
→transactions/idrange?from=1&to=100",
   "https://api-fxpractice.oanda.com/v3/accounts/101-004-1435156-001/
→transactions/idrange?from=101&to=200",
   "https://api-fxpractice.oanda.com/v3/accounts/101-004-1435156-001/
→transactions/idrange?from=201&to=300",
   "https://api-fxpractice.oanda.com/v3/accounts/101-004-1435156-001/
→transactions/idrange?from=301&to=400",
   "https://api-fxpractice.oanda.com/v3/accounts/101-004-1435156-001/
→transactions/idrange?from=401&to=500",
   "https://api-fxpractice.oanda.com/v3/accounts/101-004-1435156-001/
→transactions/idrange?from=501&to=600",
   "https://api-fxpractice.oanda.com/v3/accounts/101-004-1435156-001/
→transactions/idrange?from=601&to=700",
   "https://api-fxpractice.oanda.com/v3/accounts/101-004-1435156-001/
→transactions/idrange?from=701&to=800",
   "https://api-fxpractice.oanda.com/v3/accounts/101-004-1435156-001/
→transactions/idrange?from=801&to=900",
   "https://api-fxpractice.oanda.com/v3/accounts/101-004-1435156-001/
→transactions/idrange?from=901&to=1000",
   "https://api-fxpractice.oanda.com/v3/accounts/101-004-1435156-001/
→transactions/idrange?from=1001&to=1100",
   "https://api-fxpractice.oanda.com/v3/accounts/101-004-1435156-001/
→transactions/idrange?from=1101&to=1200",
   "https://api-fxpractice.oanda.com/v3/accounts/101-004-1435156-001/
→transactions/idrange?from=1201&to=1300",
   "https://api-fxpractice.oanda.com/v3/accounts/101-004-1435156-001/
→transactions/idrange?from=1301&to=1400",
   "https://api-fxpractice.oanda.com/v3/accounts/101-004-1435156-001/
→transactions/idrange?from=1401&to=1500",
   "https://api-fxpractice.oanda.com/v3/accounts/101-004-1435156-001/
→transactions/idrange?from=1501&to=1600",
   "https://api-fxpractice.oanda.com/v3/accounts/101-004-1435156-001/

→transactions/idrange?from=1601&to=1700",
```

3.8.4 TransactionsSinceID

Bases: oandapyV20.endpoints.transactions.Transactions

TransactionsSinceID.

Get a range of Transactions for an Account starting at (but not including) a provided Transaction ID.

```
ENDPOINT = 'v3/accounts/{accountID}/transactions/sinceid'
```

```
EXPECTED_STATUS = 200
METHOD = 'GET'
```

__init__ (accountID, params=None)

Instantiate an TransactionsSince request.

Parameters

- accountID (string (required)) id of the account to perform the request on.
- params (dict (required)) query params to send, check developer.oanda.com for details.

Query Params example:

```
{
    "id": 2306
}
```

```
>>> r = trans.TransactionsSinceID(accountID=..., params=params)
>>> client.request(r)
>>> print r.response
```

```
"lastTransactionID": "2311",
"transactions": [
   "price": "1.25000",
   "timeInForce": "GTC",
   "reason": "REPLACEMENT",
   "clientExtensions": {
     "comment": "myComment",
     "id": "myID"
   "id": "2307",
    "batchID": "2306",
    "triggerCondition": "TRIGGER_DEFAULT",
    "replacesOrderID": "2304",
    "positionFill": "DEFAULT",
    "userID": 1435156,
    "instrument": "EUR_USD",
   "time": "2016-10-25T19:45:38.558056359Z",
   "units": "-500000",
   "type": "LIMIT_ORDER",
   "accountID": "101-004-1435156-001"
 },
   "orderID": "2307",
   "clientOrderID": "myID",
    "reason": "CLIENT_REQUEST",
    "batchID": "2308",
    "time": "2016-10-25T20:53:03.789670387Z",
    "type": "ORDER_CANCEL",
   "userID": 1435156,
   "id": "2308",
   "accountID": "101-004-1435156-001"
  },
   "price": "1.20000",
    "stopLossOnFill": {
     "timeInForce": "GTC",
      "price": "1.22000"
   },
    "timeInForce": "GTC",
    "reason": "CLIENT_ORDER",
    "id": "2309",
   "batchID": "2309",
   "triggerCondition": "TRIGGER_DEFAULT",
   "positionFill": "DEFAULT",
   "userID": 1435156,
   "instrument": "EUR_USD",
   "time": "2016-10-25T21:07:21.065554321Z",
   "units": "-100",
   "type": "LIMIT_ORDER",
   "accountID": "101-004-1435156-001"
  },
   "userID": 1435156,
    "marginRate": "0.01",
    "batchID": "2310",
    "time": "2016-10-26T13:28:00.507651360Z",
```

```
"type": "CLIENT_CONFIGURE",
   "id": "2310",
   "accountID": "101-004-1435156-001"
},
{
    "userID": 1435156,
    "marginRate": "0.01",
    "batchID": "2311",
    "time": "2016-10-26T13:28:13.597103123Z",
    "type": "CLIENT_CONFIGURE",
    "id": "2311",
    "accountID": "101-004-1435156-001"
}
```

3.8.5 TransactionsStream

```
class oandapy V20.endpoints.transactions. Transactions Stream (accountID, params=None)

Bases: oandapy V20.endpoints.transactions. Transactions
```

TransactionsStream.

Get a stream of Transactions for an Account starting from when the request is made.

```
ENDPOINT = 'v3/accounts/{accountID}/transactions/stream'
EXPECTED_STATUS = 200

METHOD = 'GET'

STREAM = True
__init__(accountID, params=None)
    Instantiate an TransactionsStream request.
```

Performing this request will result in a generator yielding transactions.

Parameters accountID (string (required))—id of the account to perform the request on.

```
>>> import oandapyV20
>>> import oandapyV20.endpoints.transactions as trans
>>> client = oandapyV20.API(access_token=...)
>>> r = trans.TransactionsStream(accountID=...)
>>> rv = client.request(r)
>>> maxrecs = 5
>>> try:
>>>
        for T in r.response: # or rv ...
>>>
            print json.dumps(R, indent=4), ","
>>>
            maxrecs -= 1
           if maxrecs == 0:
>>>
                r.terminate("Got them all")
>>>
>>> except StreamTerminated as e:
       print("Finished: {msg}".format(msg=e))
>>>
```

```
{
        "type": "HEARTBEAT",
        "lastTransactionID": "2311",
        "time": "2016-10-28T11:56:12.002855862Z"
      },
        "type": "HEARTBEAT",
        "lastTransactionID": "2311",
        "time": "2016-10-28T11:56:17.059535527Z"
      },
      {
        "type": "HEARTBEAT",
        "lastTransactionID": "2311",
        "time": "2016-10-28T11:56:22.142256403Z"
      },
      {
        "type": "HEARTBEAT",
        "lastTransactionID": "2311",
        "time": "2016-10-28T11:56:27.238853774Z"
      },
      {
        "type": "HEARTBEAT",
       "lastTransactionID": "2311",
        "time": "2016-10-28T11:56:32.289316796Z"
      }
Finished: Got them all
```

terminate (message=")

terminate the stream.

Calling this method will stop the generator yielding transaction records. A message can be passed optionally.

CHAPTER 4

oandapyV20.definitions

The oandapyV20.definitions module holds all the definitions as in the definitions section of the REST-V20 specs of OANDA, see developer.oanda.com.

4.1 oandapyV20.definitions.accounts

Account Definitions.

```
\begin{tabular}{ll} \textbf{class} & \texttt{oandapyV20.definitions.accounts.AccountFinancingMode} \\ & \textbf{Bases: object} \end{tabular}
```

Definition representation of AccountFinancingMode

```
DAILY = 'DAILY'

NO_FINANCING = 'NO_FINANCING'

SECOND_BY_SECOND = 'SECOND_BY_SECOND'

__getitem__(definitionID)

return description for definitionID.
```

definitions

readonly property holding definition dict.

```
class oandapyV20.definitions.accounts.PositionAggregationMode
    Bases: object
```

Definition representation of PositionAggregationMode

Definitions used in requests and responses. This class provides the ID and the description of the definitions.

```
>>> import oandapyV20.definitions.accounts as defaccounts
>>> print defaccounts.PositionAggregationMode.NET_SUM
NET_SUM
>>> c = defaccounts.PositionAggregationMode()
>>> print c[c.NET_SUM]
The units for each side (long and short) of the Position are netted together and_
→the resulting value (long or short) is used to compute the Position value or_
⇔margin.
>>> # or
>>> print defaccounts.PositionAggregationMode().definitions[c.NET_SUM]
>>> # all kevs
>>> print defaccounts.PositionAggregationMode().definitions.keys()
>>> . . .
ABSOLUTE SUM = 'ABSOLUTE SUM'
MAXIMAL_SIDE = 'MAXIMAL_SIDE'
NET_SUM = 'NET_SUM'
```

definitions

__getitem__(definitionID)

readonly property holding definition dict.

return description for definitionID.

4.2 oandapyV20.definitions.instruments

Instruments Definitions.

Definition representation of CandlestickGranularity

```
>>> import oandapyV20.definitions.instruments as definstruments
>>> print definstruments.CandlestickGranularity.H4
H4
>>> c = definstruments.CandlestickGranularity()
>>> print c[c.H4]
4 hour candlesticks, day alignment
>>> # or
>>> print definstruments.CandlestickGranularity().definitions[c.H4]
>>> # all keys
>>> print definstruments.CandlestickGranularity().definitions.keys()
>>> ...
```

```
D = 'D'
```

```
H1 = 'H1'
H12 = 'H12'
H2 = 'H2'
H3 = 'H3'
H4 = 'H4'
H6 = 'H6'
H8 = 'H8'
M = 'M'
M1 = 'M1'
M10 = 'M10'
M15 = 'M15'
M2 = 'M2'
M30 = 'M30'
M4 = 'M4'
M5 = 'M5'
S10 = 'S10'
S15 = 'S15'
$30 = '$30'
S5 = 'S5'
W = W'
__getitem__(definitionID)
    return description for definitionID.
```

definitions

readonly property holding definition dict.

```
\textbf{class} \  \, \texttt{oandapyV20.definitions.instruments.WeeklyAlignment} \\ Bases: \  \, \texttt{object}
```

Definition representation of WeeklyAlignment

```
>>> import oandapyV20.definitions.instruments as definstruments
>>> print definstruments.WeeklyAlignment.Monday
Monday
>>> c = definstruments.WeeklyAlignment()
>>> print c[c.Monday]
Monday
>>> # or
>>> print definstruments.WeeklyAlignment().definitions[c.Monday]
>>> # all keys
>>> print definstruments.WeeklyAlignment().definitions.keys()
>>> ...
```

Friday = 'Friday'

```
Monday = 'Monday'
     Saturday = 'Saturday'
     Sunday = 'Sunday'
     Thursday = 'Thursday'
     Tuesday = 'Tuesday'
     Wednesday = 'Wednesday'
     __getitem__(definitionID)
         return description for definitionID.
     definitions
         readonly property holding definition dict.
class oandapyV20.definitions.instruments.PriceComponents
     Bases: object
     Definition representation of PriceComponents
     Definitions used in requests and responses. This class provides the ID and the description of the definitions.
     >>> import oandapyV20.definitions.instruments as definstruments
     >>> print definstruments.PriceComponents.A
     >>> c = definstruments.PriceComponents()
     >>> print c[c.A]
     >>> print definstruments.PriceComponents().definitions[c.A]
     >>> # all keys
     >>> print definstruments.PriceComponents().definitions.keys()
     A = 'A'
     B = 'B'
    M = 'M'
     __getitem__(definitionID)
         return description for definitionID.
```

4.3 oandapyV20.definitions.orders

readonly property holding definition dict.

Order related definitions.

definitions

```
class oandapyV20.definitions.orders.OrderStateFilter
    Bases: object
```

Definition representation of OrderStateFilter

```
>>> import oandapyV20.definitions.orders as deforders
    >>> print deforders.OrderStateFilter.CANCELLED
    CANCELLED
    >>> c = deforders.OrderStateFilter()
    >>> print c[c.CANCELLED]
    The orders that have been cancelled
    >>> # or
    >>> print deforders.OrderStateFilter().definitions[c.CANCELLED]
    >>> # all keys
    >>> print deforders.OrderStateFilter().definitions.keys()
    ALL = 'ALL'
    CANCELLED = 'CANCELLED'
    FILLED = 'FILLED'
    PENDING = 'PENDING'
    TRIGGERED = 'TRIGGERED'
    __getitem__(definitionID)
         return description for definitionID.
    definitions
         readonly property holding definition dict.
class oandapyV20.definitions.orders.OrderType
    Bases: object
    Definition representation of OrderType
    Definitions used in requests and responses. This class provides the ID and the description of the definitions.
    >>> import oandapyV20.definitions.orders as deforders
    >>> print deforders.OrderType.MARKET_IF_TOUCHED
    MARKET_IF_TOUCHED
    >>> c = deforders.OrderType()
    >>> print c[c.MARKET_IF_TOUCHED]
    A Market-if-touched Order
    >>> # or
    >>> print deforders.OrderType().definitions[c.MARKET_IF_TOUCHED]
    >>> # all keys
    >>> print deforders.OrderType().definitions.keys()
    >>> ...
    LIMIT = 'LIMIT'
    MARKET = 'MARKET'
    MARKET_IF_TOUCHED = 'MARKET_IF_TOUCHED'
    STOP = 'STOP'
    STOP LOSS = 'STOP LOSS'
    TAKE_PROFIT = 'TAKE_PROFIT'
    TRAILING_STOP_LOSS = 'TRAILING_STOP_LOSS'
    __getitem__(definitionID)
         return description for definitionID.
```

definitions

readonly property holding definition dict.

```
class oandapyV20.definitions.orders.CancellableOrderType
    Bases: object
```

Definition representation of CancellableOrderType

Definitions used in requests and responses. This class provides the ID and the description of the definitions.

```
>>> import oandapyV20.definitions.orders as deforders
>>> print deforders.CancellableOrderType.MARKET_IF_TOUCHED
MARKET_IF_TOUCHED
>>> c = deforders.CancellableOrderType()
>>> print c[c.MARKET_IF_TOUCHED]
A Market-if-touched Order
>>> # or
>>> print deforders.CancellableOrderType().definitions[c.MARKET_IF_TOUCHED]
>>> # all keys
>>> print deforders.CancellableOrderType().definitions.keys()
>>> ...
```

```
LIMIT = 'LIMIT'

MARKET_IF_TOUCHED = 'MARKET_IF_TOUCHED'

STOP = 'STOP'

STOP_LOSS = 'STOP_LOSS'

TAKE_PROFIT = 'TAKE_PROFIT'

TRAILING_STOP_LOSS = 'TRAILING_STOP_LOSS'

__getitem__ (definitionID)

return description for definitionID.
```

definitions

readonly property holding definition dict.

```
class oandapyV20.definitions.orders.OrderPositionFill
    Bases: object
```

Definition representation of OrderPositionFill

```
>>> import oandapyV20.definitions.orders as deforders
>>> print deforders.OrderPositionFill.REDUCE_ONLY
REDUCE_ONLY
>>> c = deforders.OrderPositionFill()
>>> print c[c.REDUCE_ONLY]
When the Order is filled, only reduce an existing Position.
>>> # or
>>> print deforders.OrderPositionFill().definitions[c.REDUCE_ONLY]
>>> # all keys
>>> print deforders.OrderPositionFill().definitions.keys()
>>> ...
```

```
DEFAULT = 'DEFAULT'
OPEN_ONLY = 'OPEN_ONLY'
REDUCE FIRST = 'REDUCE FIRST'
```

```
REDUCE_ONLY = 'REDUCE_ONLY'
__getitem__(definitionID)
    return description for definitionID.
definitions
```

readonly property holding definition dict.

```
class oandapyV20.definitions.orders.TimeInForce
    Bases: object
```

Definition representation of TimeInForce

Definitions used in requests and responses. This class provides the ID and the description of the definitions.

```
>>> import oandapyV20.definitions.orders as deforders
>>> print deforders.TimeInForce.IOC
IOC
>>> c = deforders.TimeInForce()
>>> print c[c.IOC]
The Order must be "Immediately partially filled Or Killed"
>>> # or
>>> print deforders.TimeInForce().definitions[c.IOC]
>>> # all keys
>>> print deforders.TimeInForce().definitions.keys()
>>> ...
```

```
FOK = 'FOK'
GFD = 'GFD'
GTC = 'GTC'
GTD = 'GTD'
IOC = 'IOC'
__getitem__(definitionID)
    return description for definitionID.
```

definitions

readonly property holding definition dict.

```
class oandapyV20.definitions.orders.OrderState
    Bases: object
```

Definition representation of OrderState

Definitions used in requests and responses. This class provides the ID and the description of the definitions.

```
>>> import oandapyV20.definitions.orders as deforders
>>> print deforders.OrderState.CANCELLED
CANCELLED
>>> c = deforders.OrderState()
>>> print c[c.CANCELLED]
The Order has been cancelled
>>> # or
>>> print deforders.OrderState().definitions[c.CANCELLED]
>>> # all keys
>>> print deforders.OrderState().definitions.keys()
>>> ...
```

CANCELLED = 'CANCELLED'

```
FILLED = 'FILLED'
PENDING = 'PENDING'
TRIGGERED = 'TRIGGERED'
__getitem__(definitionID)
    return description for definitionID.

definitions
    readonly property holding definition dict.

class oandapyV20.definitions.orders.OrderTriggerCondition
    Bases: object
    Definition representation of OrderTriggerCondition
```

Definitions used in requests and responses. This class provides the ID and the description of the definitions.

```
ASK = 'ASK'

BID = 'BID'

DEFAULT = 'DEFAULT'

INVERSE = 'INVERSE'

MID = 'MID'

__getitem__(definitionID)

return description for definitionID.
```

definitions

readonly property holding definition dict.

4.4 oandapyV20.definitions.pricing

Pricing related Definitions.

```
\begin{tabular}{ll} \textbf{class} & \texttt{oandapyV20.definitions.pricing.PriceStatus} \\ \textbf{Bases:} & \texttt{object} \end{tabular}
```

Definition representation of PriceStatus

```
>>> import oandapyV20.definitions.pricing as defpricing
>>> print defpricing.PriceStatus.non_tradeable
non-tradeable
```

```
>>> c = defpricing.PriceStatus()
>>> print c[c.non_tradeable]
The Instrument's price is not tradeable.
>>> # or
>>> print defpricing.PriceStatus().definitions[c.non_tradeable]
>>> # all keys
>>> print defpricing.PriceStatus().definitions.keys()
>>> ...
```

Note: attribute name *non-tradeable* is renamed to *non_tradeable*, value stil is *non-tradeable*. This means that a lookup stil applies.

```
__getitem__ (definitionID)
    return description for definitionID.

definitions
    readonly property holding definition dict.
invalid = 'invalid'
non_tradeable = 'non-tradeable'
tradeable = 'tradeable'
```

4.5 oandapyV20.definitions.trades

Trades definitions.

```
class oandapyV20.definitions.trades.TradePL
    Bases: object
```

Definition representation of TradePL

```
NEGATIVE = 'NEGATIVE'
POSITIVE = 'POSITIVE'
ZERO = 'ZERO'
__getitem__(definitionID)
    return description for definitionID.
```

definitions

readonly property holding definition dict.

```
class oandapyV20.definitions.trades.TradeState
    Bases: object
```

Definition representation of TradeState

Definitions used in requests and responses. This class provides the ID and the description of the definitions.

```
>>> import oandapyV20.definitions.trades as deftrades
>>> print deftrades.TradeState.CLOSE_WHEN_TRADABLE
CLOSE_WHEN_TRADABLE
>>> c = deftrades.TradeState()
>>> print c[c.CLOSE_WHEN_TRADABLE]
The Trade will be closed as soon as the trade's instrument becomes tradeable
>>> # or
>>> print deftrades.TradeState().definitions[c.CLOSE_WHEN_TRADABLE]
>>> # all keys
>>> print deftrades.TradeState().definitions.keys()
>>> ...
CLOSED = 'CLOSED'
CLOSE_WHEN_TRADABLE = 'CLOSE_WHEN_TRADABLE'
```

CLOSE_WHEN_TRADABLE = 'CLOSE_WHEN_T
OPEN = 'OPEN'

__getitem__ (definitionID) return description for definitionID.

definitions

readonly property holding definition dict.

```
class oandapyV20.definitions.trades.TradeStateFilter
    Bases: object
```

Definition representation of TradeStateFilter

```
>>> import oandapyV20.definitions.trades as deftrades
>>> print deftrades.TradeStateFilter.CLOSE_WHEN_TRADEABLE
CLOSE_WHEN_TRADEABLE
>>> c = deftrades.TradeStateFilter()
>>> print c[c.CLOSE_WHEN_TRADEABLE]
The Trades that will be closed as soon as the trades' instrument becomes tradeable
>>> # or
>>> print deftrades.TradeStateFilter().definitions[c.CLOSE_WHEN_TRADEABLE]
>>> # all keys
>>> print deftrades.TradeStateFilter().definitions.keys()
>>> ...
```

```
ALL = 'ALL'

CLOSED = 'CLOSED'

CLOSE_WHEN_TRADEABLE = 'CLOSE_WHEN_TRADEABLE'

OPEN = 'OPEN'

__getitem__ (definitionID)

return description for definitionID.
```

definitions

readonly property holding definition dict.

4.6 oandapyV20.definitions.transactions

Transactions definitions.

```
\textbf{class} \ \ \texttt{oandapyV20.definitions.transactions.} \\ \textbf{MarketOrderMarginCloseoutReason} \\ \textbf{Bases:} \ \texttt{object}
```

Definition representation of MarketOrderMarginCloseoutReason

Definitions used in requests and responses. This class provides the ID and the description of the definitions.

```
>>> import oandapyV20.definitions.transactions as deftransactions
>>> print deftransactions.MarketOrderMarginCloseoutReason.MARGIN_CHECK_VIOLATION
MARGIN_CHECK_VIOLATION
>>> c = deftransactions.MarketOrderMarginCloseoutReason()
>>> print c[c.MARGIN_CHECK_VIOLATION]
Trade closures resulted from violating OANDA's margin policy
>>> # or
>>> print deftransactions.MarketOrderMarginCloseoutReason().definitions[c.MARGIN_
\( \to CHECK_VIOLATION \)]
>>> # all keys
>>> print deftransactions.MarketOrderMarginCloseoutReason().definitions.keys()
>>> ...
```

```
MARGIN_CHECK_VIOLATION = 'MARGIN_CHECK_VIOLATION'
REGULATORY_MARGIN_CALL_VIOLATION = 'REGULATORY_MARGIN_CALL_VIOLATION'
__getitem__ (definitionID)
    return description for definitionID.
```

definitions

readonly property holding definition dict.

```
class oandapyV20.definitions.transactions.StopLossOrderReason
     Bases: object
```

Definition representation of StopLossOrderReason

```
>>> import oandapyV20.definitions.transactions as deftransactions
>>> print deftransactions.StopLossOrderReason.ON_FILL
ON_FILL
>>> c = deftransactions.StopLossOrderReason()
>>> print c[c.ON_FILL]
The Stop Loss Order was initiated automatically when an Order was filled that_
--opened a new Trade requiring a Stop Loss Order.
>>> # or
>>> print deftransactions.StopLossOrderReason().definitions[c.ON_FILL]
>>> # all keys
>>> print deftransactions.StopLossOrderReason().definitions.keys()
>>> ...
```

```
CLIENT_ORDER = 'CLIENT_ORDER'
ON FILL = 'ON FILL'
```

REPLACEMENT = 'REPLACEMENT'

```
__getitem__(definitionID)
        return description for definitionID.
    definitions
        readonly property holding definition dict.
class oandapyV20.definitions.transactions.OrderFillReason
    Bases: object
    Definition representation of OrderFillReason
    Definitions used in requests and responses. This class provides the ID and the description of the definitions.
    >>> import oandapyV20.definitions.transactions as deftransactions
    >>> print deftransactions.OrderFillReason.STOP_ORDER
    STOP ORDER
    >>> c = deftransactions.OrderFillReason()
    >>> print c[c.STOP_ORDER]
    The Order filled was a Stop Order
    >>> print deftransactions.OrderFillReason().definitions[c.STOP_ORDER]
    >>> # all keys
    >>> print deftransactions.OrderFillReason().definitions.keys()
    LIMIT ORDER = 'LIMIT ORDER'
    MARKET IF TOUCHED ORDER = 'MARKET IF TOUCHED ORDER'
    MARKET_ORDER = 'MARKET_ORDER'
    MARKET_ORDER_DELAYED_TRADE_CLOSE = 'MARKET_ORDER_DELAYED_TRADE_CLOSE'
    MARKET ORDER MARGIN CLOSEOUT = 'MARKET ORDER MARGIN CLOSEOUT'
    MARKET ORDER POSITION CLOSEOUT = 'MARKET ORDER POSITION CLOSEOUT'
    MARKET_ORDER_TRADE_CLOSE = 'MARKET_ORDER_TRADE_CLOSE'
    STOP_LOSS_ORDER = 'STOP_LOSS_ORDER'
    STOP_ORDER = 'STOP_ORDER'
    TAKE PROFIT ORDER = 'TAKE PROFIT ORDER'
    TRAILING_STOP_LOSS_ORDER = 'TRAILING_STOP_LOSS_ORDER'
    __getitem__(definitionID)
        return description for definitionID.
         readonly property holding definition dict.
class oandapyV20.definitions.transactions.FundingReason
    Bases: object
    Definition representation of FundingReason
```

```
>>> import oandapyV20.definitions.transactions as deftransactions
>>> print deftransactions.FundingReason.ACCOUNT_TRANSFER
```

Definitions used in requests and responses. This class provides the ID and the description of the definitions.

ACCOUNT_TRANSFER

```
>>> c = deftransactions.FundingReason()
>>> print c[c.ACCOUNT_TRANSFER]
Funds are being transfered between two Accounts.
>>> # or
>>> print deftransactions.FundingReason().definitions[c.ACCOUNT_TRANSFER]
>>> # all keys
>>> print deftransactions.FundingReason().definitions.keys()
>>> ...
```

```
ACCOUNT_TRANSFER = 'ACCOUNT_TRANSFER'

ADJUSTMENT = 'ADJUSTMENT'

CLIENT_FUNDING = 'CLIENT_FUNDING'

DIVISION_MIGRATION = 'DIVISION_MIGRATION'

SITE_MIGRATION = 'SITE_MIGRATION'

__getitem__ (definitionID)

return description for definitionID.
```

definitions

readonly property holding definition dict.

```
\textbf{class} \  \, \text{oandapy V20.} definitions. transactions. \textbf{MarketIfTouchedOrderReason} \\ Bases: \ object
```

Definition representation of MarketIfTouchedOrderReason

Definitions used in requests and responses. This class provides the ID and the description of the definitions.

```
>>> import oandapyV20.definitions.transactions as deftransactions
>>> print deftransactions.MarketIfTouchedOrderReason.CLIENT_ORDER
CLIENT_ORDER
>>> c = deftransactions.MarketIfTouchedOrderReason()
>>> print c[c.CLIENT_ORDER]
The Market-if-touched Order was initiated at the request of a client
>>> # or
>>> print deftransactions.MarketIfTouchedOrderReason().definitions[c.CLIENT_ORDER]
>>> # all keys
>>> print deftransactions.MarketIfTouchedOrderReason().definitions.keys()
>>> ...
```

```
CLIENT_ORDER = 'CLIENT_ORDER'

REPLACEMENT = 'REPLACEMENT'

__getitem__(definitionID)

return description for definitionID.

definitions
```

readonly property holding definition dict.

```
class oandapyV20.definitions.transactions.MarketOrderReason
     Bases: object
```

Definition representation of MarketOrderReason

```
>>> import oandapyV20.definitions.transactions as deftransactions
    >>> print deftransactions.MarketOrderReason.TRADE_CLOSE
    TRADE CLOSE
    >>> c = deftransactions.MarketOrderReason()
    >>> print c[c.TRADE_CLOSE]
    The Market Order was created to close a Trade at the request of a client
    >>> # or
    >>> print deftransactions.MarketOrderReason().definitions[c.TRADE_CLOSE]
    >>> # all keys
    >>> print deftransactions.MarketOrderReason().definitions.keys()
    >>>
    CLIENT_ORDER = 'CLIENT_ORDER'
    DELAYED_TRADE_CLOSE = 'DELAYED_TRADE_CLOSE'
    MARGIN_CLOSEOUT = 'MARGIN_CLOSEOUT'
    POSITION CLOSEOUT = 'POSITION CLOSEOUT'
    TRADE_CLOSE = 'TRADE_CLOSE'
    __getitem__(definitionID)
         return description for definitionID.
    definitions
         readonly property holding definition dict.
class oandapyV20.definitions.transactions.StopOrderReason
    Bases: object
    Definition representation of StopOrderReason
    Definitions used in requests and responses. This class provides the ID and the description of the definitions.
    >>> import oandapyV20.definitions.transactions as deftransactions
    >>> print deftransactions.StopOrderReason.CLIENT_ORDER
    CLIENT_ORDER
    >>> c = deftransactions.StopOrderReason()
    >>> print c[c.CLIENT_ORDER]
    The Stop Order was initiated at the request of a client
    >>> print deftransactions.StopOrderReason().definitions[c.CLIENT_ORDER]
    >>> # all keys
    >>> print deftransactions.StopOrderReason().definitions.keys()
    >>> ...
    CLIENT_ORDER = 'CLIENT_ORDER'
    REPLACEMENT = 'REPLACEMENT'
     __getitem__(definitionID)
         return description for definitionID.
    definitions
         readonly property holding definition dict.
class oandapyV20.definitions.transactions.TransactionType
    Bases: object
    Definition representation of TransactionType
```

```
>>> import oandapyV20.definitions.transactions as deftransactions
>>> print deftransactions.TransactionType.STOP LOSS ORDER
STOP LOSS ORDER
>>> c = deftransactions.TransactionType()
>>> print c[c.STOP_LOSS_ORDER]
Stop Loss Order Transaction
>>> # or
>>> print deftransactions.TransactionType().definitions[c.STOP_LOSS_ORDER]
>>> # all keys
>>> print deftransactions.TransactionType().definitions.keys()
>>>
CLIENT CONFIGURE = 'CLIENT CONFIGURE'
CLIENT_CONFIGURE_REJECT = 'CLIENT_CONFIGURE_REJECT'
CLOSE = 'CLOSE'
CREATE = 'CREATE'
DAILY_FINANCING = 'DAILY_FINANCING'
DELAYED_TRADE_CLOSURE = 'DELAYED_TRADE_CLOSURE'
LIMIT_ORDER = 'LIMIT_ORDER'
LIMIT_ORDER_REJECT = 'LIMIT_ORDER_REJECT'
MARGIN CALL ENTER = 'MARGIN CALL ENTER'
MARGIN_CALL_EXIT = 'MARGIN_CALL_EXIT'
MARGIN CALL EXTEND = 'MARGIN CALL EXTEND'
MARKET_IF_TOUCHED_ORDER = 'MARKET_IF_TOUCHED_ORDER'
MARKET_IF_TOUCHED_ORDER_REJECT = 'MARKET_IF_TOUCHED_ORDER_REJECT'
MARKET ORDER = 'MARKET ORDER'
MARKET ORDER REJECT = 'MARKET ORDER REJECT'
ORDER_CANCEL = 'ORDER_CANCEL'
ORDER_CANCEL_REJECT = 'ORDER_CANCEL_REJECT'
ORDER_CLIENT_EXTENSIONS_MODIFY = 'ORDER_CLIENT_EXTENSIONS_MODIFY'
ORDER CLIENT EXTENSIONS MODIFY REJECT = 'ORDER CLIENT EXTENSIONS MODIFY REJECT'
ORDER FILL = 'ORDER FILL'
REOPEN = 'REOPEN'
RESET_RESETTABLE_PL = 'RESET_RESETTABLE_PL'
STOP LOSS ORDER = 'STOP LOSS ORDER'
STOP LOSS ORDER REJECT = 'STOP LOSS ORDER REJECT'
STOP ORDER = 'STOP ORDER'
STOP_ORDER_REJECT = 'STOP_ORDER_REJECT'
TAKE PROFIT_ORDER = 'TAKE_PROFIT_ORDER'
TAKE PROFIT ORDER REJECT = 'TAKE PROFIT ORDER REJECT'
```

```
TRADE CLIENT EXTENSIONS MODIFY = 'TRADE CLIENT EXTENSIONS MODIFY'
    TRADE CLIENT EXTENSIONS MODIFY REJECT = 'TRADE CLIENT EXTENSIONS MODIFY REJECT'
    TRAILING_STOP_LOSS_ORDER = 'TRAILING_STOP_LOSS_ORDER'
    TRAILING_STOP_LOSS_ORDER_REJECT = 'TRAILING_STOP_LOSS_ORDER_REJECT'
    TRANSFER FUNDS = 'TRANSFER FUNDS'
    TRANSFER FUNDS REJECT = 'TRANSFER FUNDS REJECT'
    __getitem__(definitionID)
         return description for definitionID.
    definitions
         readonly property holding definition dict.
class oandapyV20.definitions.transactions.TakeProfitOrderReason
    Bases: object
    Definition representation of TakeProfitOrderReason
    Definitions used in requests and responses. This class provides the ID and the description of the definitions.
     >>> import oandapyV20.definitions.transactions as deftransactions
    >>> print deftransactions.TakeProfitOrderReason.ON_FILL
    ON_FILL
    >>> c = deftransactions.TakeProfitOrderReason()
    >>> print c[c.ON_FILL]
    The Take Profit Order was initiated automatically when an Order was filled that,
     →opened a new Trade requiring a Take Profit Order.
     >>> # or
     >>> print deftransactions.TakeProfitOrderReason().definitions[c.ON_FILL]
    >>> # all keys
    >>> print deftransactions.TakeProfitOrderReason().definitions.keys()
    CLIENT ORDER = 'CLIENT ORDER'
    ON_FILL = 'ON_FILL'
    REPLACEMENT = 'REPLACEMENT'
     __getitem__(definitionID)
         return description for definitionID.
    definitions
         readonly property holding definition dict.
class oandapyV20.definitions.transactions.TransactionRejectReason
    Bases: object
    Definition representation of TransactionRejectReason
    Definitions used in requests and responses. This class provides the ID and the description of the definitions.
```

>>> import oandapyV20.definitions.transactions as deftransactions
>>> print deftransactions.TransactionRejectReason.STOP_LOSS_ON_FILL_GTD_TIMESTAMP_

->MISSING
STOP_LOSS_ON_FILL_GTD_TIMESTAMP_MISSING
>>> c = deftransactions.TransactionRejectReason()
>>> print c[c.STOP_LOSS_ON_FILL_GTD_TIMESTAMP_MISSING]
The Stop Loss on fill specifies a GTD TimeInForce but does not provide a GTD_

->timestamp

```
>>> # or
>>> print deftransactions.TransactionRejectReason().definitions[c.STOP LOSS ON
→FILL GTD TIMESTAMP MISSING
>>> # all keys
>>> print deftransactions.TransactionRejectReason().definitions.keys()
ACCOUNT CONFIGURATION LOCKED = 'ACCOUNT CONFIGURATION LOCKED'
ACCOUNT DEPOSIT LOCKED = 'ACCOUNT DEPOSIT LOCKED'
ACCOUNT_LOCKED = 'ACCOUNT_LOCKED'
ACCOUNT_NOT_ACTIVE = 'ACCOUNT_NOT_ACTIVE'
ACCOUNT ORDER CANCEL LOCKED = 'ACCOUNT ORDER CANCEL LOCKED'
ACCOUNT_ORDER_CREATION_LOCKED = 'ACCOUNT_ORDER_CREATION_LOCKED'
ACCOUNT WITHDRAWAL LOCKED = 'ACCOUNT WITHDRAWAL LOCKED'
ADMIN_CONFIGURE_DATA_MISSING = 'ADMIN_CONFIGURE_DATA_MISSING'
ALIAS INVALID = 'ALIAS INVALID'
AMOUNT INVALID = 'AMOUNT INVALID'
AMOUNT MISSING = 'AMOUNT MISSING'
CLIENT CONFIGURE DATA MISSING = 'CLIENT CONFIGURE DATA MISSING'
CLIENT EXTENSIONS DATA MISSING = 'CLIENT EXTENSIONS DATA MISSING'
CLIENT_ORDER_COMMENT_INVALID = 'CLIENT_ORDER_COMMENT_INVALID'
CLIENT ORDER ID ALREADY EXISTS = 'CLIENT ORDER ID ALREADY EXISTS'
CLIENT_ORDER_ID_INVALID = 'CLIENT_ORDER_ID_INVALID'
CLIENT_ORDER_TAG_INVALID = 'CLIENT_ORDER_TAG_INVALID'
CLIENT_TRADE_COMMENT_INVALID = 'CLIENT_TRADE_COMMENT_INVALID'
CLIENT TRADE ID ALREADY EXISTS = 'CLIENT TRADE ID ALREADY EXISTS'
CLIENT TRADE ID INVALID = 'CLIENT TRADE ID INVALID'
CLIENT TRADE TAG INVALID = 'CLIENT TRADE TAG INVALID'
CLOSEOUT_POSITION_DOESNT_EXIST = 'CLOSEOUT_POSITION_DOESNT_EXIST'
CLOSEOUT POSITION INCOMPLETE SPECIFICATION = 'CLOSEOUT POSITION INCOMPLETE SPECIFICATI
CLOSEOUT POSITION PARTIAL UNITS MISSING = 'CLOSEOUT POSITION PARTIAL UNITS MISSING'
CLOSEOUT POSITION REJECT = 'CLOSEOUT POSITION REJECT'
CLOSEOUT_POSITION_UNITS_EXCEED_POSITION_SIZE = 'CLOSEOUT_POSITION_UNITS_EXCEED_POSITIO
CLOSE_TRADE_PARTIAL_UNITS_MISSING = 'CLOSE_TRADE_PARTIAL_UNITS_MISSING'
CLOSE_TRADE_TYPE_MISSING = 'CLOSE_TRADE_TYPE_MISSING'
CLOSE_TRADE_UNITS_EXCEED_TRADE_SIZE = 'CLOSE_TRADE_UNITS_EXCEED_TRADE_SIZE'
FUNDING REASON MISSING = 'FUNDING REASON MISSING'
```

INSTRUMENT MISSING = 'INSTRUMENT MISSING'

INSTRUMENT NOT TRADEABLE = 'INSTRUMENT NOT TRADEABLE'

```
INSTRUMENT PRICE UNKNOWN = 'INSTRUMENT PRICE UNKNOWN'
INSTRUMENT UNKNOWN = 'INSTRUMENT UNKNOWN'
INSUFFICIENT FUNDS = 'INSUFFICIENT FUNDS'
INTERNAL SERVER ERROR = 'INTERNAL SERVER ERROR'
INVALID REISSUE IMMEDIATE PARTIAL FILL = 'INVALID REISSUE IMMEDIATE PARTIAL FILL'
MARGIN_RATE_INVALID = 'MARGIN_RATE_INVALID'
MARGIN_RATE_WOULD_TRIGGER_CLOSEOUT = 'MARGIN_RATE_WOULD_TRIGGER_CLOSEOUT'
MARGIN RATE WOULD TRIGGER MARGIN CALL = 'MARGIN RATE WOULD TRIGGER MARGIN CALL'
MARKUP GROUP ID INVALID = 'MARKUP GROUP ID INVALID'
ORDERS_ON_FILL_DUPLICATE_CLIENT_ORDER_IDS = 'ORDERS_ON_FILL_DUPLICATE_CLIENT_ORDER_IDS
ORDER_DOESNT_EXIST = 'ORDER_DOESNT_EXIST'
ORDER FILL POSITION ACTION INVALID = 'ORDER FILL POSITION ACTION INVALID'
ORDER FILL POSITION ACTION MISSING = 'ORDER FILL POSITION ACTION MISSING'
ORDER IDENTIFIER INCONSISTENCY = 'ORDER IDENTIFIER INCONSISTENCY'
ORDER_ID_UNSPECIFIED = 'ORDER_ID_UNSPECIFIED'
ORDER PARTIAL FILL OPTION INVALID = 'ORDER PARTIAL FILL OPTION INVALID'
ORDER PARTIAL FILL OPTION MISSING = 'ORDER PARTIAL FILL OPTION MISSING'
PENDING_ORDERS_ALLOWED_EXCEEDED = 'PENDING_ORDERS_ALLOWED_EXCEEDED'
POSITION AGGREGATION MODE INVALID = 'POSITION AGGREGATION MODE INVALID'
PRICE_BOUND_INVALID = 'PRICE_BOUND_INVALID'
PRICE_BOUND_PRECISION_EXCEEDED = 'PRICE_BOUND_PRECISION_EXCEEDED'
PRICE_DISTANCE_INVALID = 'PRICE_DISTANCE_INVALID'
PRICE DISTANCE MAXIMUM EXCEEDED = 'PRICE DISTANCE MAXIMUM EXCEEDED'
PRICE DISTANCE MINIMUM NOT MET = 'PRICE DISTANCE MINIMUM NOT MET'
PRICE DISTANCE MISSING = 'PRICE DISTANCE MISSING'
PRICE_DISTANCE_PRECISION_EXCEEDED = 'PRICE_DISTANCE_PRECISION_EXCEEDED'
PRICE INVALID = 'PRICE INVALID'
PRICE MISSING = 'PRICE MISSING'
PRICE_PRECISION_EXCEEDED = 'PRICE_PRECISION_EXCEEDED'
REPLACING_ORDER_INVALID = 'REPLACING_ORDER_INVALID'
REPLACING_TRADE_ID_INVALID = 'REPLACING_TRADE_ID_INVALID'
STOP_LOSS_ON_FILL_CLIENT_ORDER_COMMENT_INVALID = 'STOP_LOSS_ON_FILL_CLIENT_ORDER_COMME
STOP_LOSS_ON_FILL_CLIENT_ORDER_ID_INVALID = 'STOP_LOSS_ON_FILL_CLIENT_ORDER_ID_INVALID
STOP LOSS ON FILL CLIENT ORDER TAG INVALID = 'STOP LOSS ON FILL CLIENT ORDER TAG INVAL
STOP_LOSS_ON_FILL_GTD_TIMESTAMP_IN_PAST = 'STOP_LOSS_ON_FILL_GTD_TIMESTAMP_IN_PAST'
```

```
STOP LOSS ON FILL GTD TIMESTAMP MISSING = 'STOP LOSS ON FILL GTD TIMESTAMP MISSING'
STOP LOSS ON FILL PRICE INVALID = 'STOP LOSS ON FILL PRICE INVALID'
STOP_LOSS_ON_FILL_PRICE_MISSING = 'STOP_LOSS_ON_FILL_PRICE_MISSING'
STOP_LOSS_ON_FILL_PRICE_PRECISION_EXCEEDED = 'STOP_LOSS_ON_FILL_PRICE_PRECISION_EXCEED
STOP LOSS ON FILL TIME IN FORCE INVALID = 'STOP LOSS ON FILL TIME IN FORCE INVALID'
STOP LOSS ON FILL TIME IN FORCE MISSING = 'STOP LOSS ON FILL TIME IN FORCE MISSING'
STOP_LOSS_ON_FILL_TRIGGER_CONDITION_INVALID = 'STOP_LOSS_ON_FILL_TRIGGER_CONDITION_INV
STOP_LOSS_ON_FILL_TRIGGER_CONDITION_MISSING = 'STOP_LOSS_ON_FILL_TRIGGER_CONDITION_MIS
STOP_LOSS_ORDER_ALREADY_EXISTS = 'STOP_LOSS_ORDER_ALREADY_EXISTS'
TAKE PROFIT_ON_FILL_CLIENT_ORDER_COMMENT_INVALID = 'TAKE_PROFIT_ON_FILL_CLIENT_ORDER_C
TAKE PROFIT_ON_FILL_CLIENT_ORDER_ID_INVALID = 'TAKE_PROFIT_ON_FILL_CLIENT_ORDER_ID_INV
TAKE PROFIT ON FILL CLIENT ORDER TAG INVALID = 'TAKE PROFIT ON FILL CLIENT ORDER TAG I
TAKE PROFIT ON FILL GTD TIMESTAMP IN PAST = 'TAKE PROFIT ON FILL GTD TIMESTAMP IN PAST
TAKE_PROFIT_ON_FILL_GTD_TIMESTAMP_MISSING = 'TAKE_PROFIT_ON_FILL_GTD_TIMESTAMP_MISSING
TAKE_PROFIT_ON_FILL_PRICE_INVALID = 'TAKE_PROFIT_ON_FILL_PRICE_INVALID'
TAKE PROFIT ON FILL PRICE MISSING = 'TAKE PROFIT ON FILL PRICE MISSING'
TAKE PROFIT ON FILL PRICE PRECISION EXCEEDED = 'TAKE PROFIT ON FILL PRICE PRECISION EX
TAKE_PROFIT_ON_FILL_TIME_IN_FORCE_INVALID = 'TAKE_PROFIT_ON_FILL_TIME_IN_FORCE_INVALID
TAKE_PROFIT_ON_FILL_TIME_IN_FORCE_MISSING = 'TAKE_PROFIT_ON_FILL_TIME_IN_FORCE_MISSING
TAKE PROFIT ON FILL TRIGGER CONDITION INVALID = 'TAKE PROFIT ON FILL TRIGGER CONDITION
TAKE PROFIT ON FILL TRIGGER CONDITION MISSING = 'TAKE PROFIT ON FILL TRIGGER CONDITION
TAKE PROFIT ORDER ALREADY EXISTS = 'TAKE PROFIT ORDER ALREADY EXISTS'
TIME_IN_FORCE_GTD_TIMESTAMP_IN_PAST = 'TIME_IN_FORCE_GTD_TIMESTAMP_IN_PAST'
TIME IN FORCE GTD TIMESTAMP MISSING = 'TIME IN FORCE GTD TIMESTAMP MISSING'
TIME IN FORCE INVALID = 'TIME IN FORCE INVALID'
TIME IN FORCE MISSING = 'TIME IN FORCE MISSING'
TRADE_DOESNT_EXIST = 'TRADE_DOESNT_EXIST'
TRADE IDENTIFIER INCONSISTENCY = 'TRADE IDENTIFIER INCONSISTENCY'
TRADE ID UNSPECIFIED = 'TRADE ID UNSPECIFIED'
TRADE_ON_FILL_CLIENT_EXTENSIONS_NOT_SUPPORTED = 'TRADE_ON_FILL_CLIENT_EXTENSIONS_NOT_S
TRAILING_STOP_LOSS_ON_FILL_CLIENT_ORDER_COMMENT_INVALID = 'TRAILING_STOP_LOSS_ON_FILL_
TRAILING_STOP_LOSS_ON_FILL_CLIENT_ORDER_ID_INVALID = 'TRAILING_STOP_LOSS_ON_FILL_CLIEN
TRAILING_STOP_LOSS_ON_FILL_CLIENT_ORDER_TAG_INVALID = 'TRAILING_STOP_LOSS_ON_FILL_CLIE
TRAILING_STOP_LOSS_ON_FILL_GTD_TIMESTAMP_IN_PAST = 'TRAILING_STOP_LOSS_ON_FILL_GTD_TIM
TRAILING_STOP_LOSS_ON_FILL_GTD_TIMESTAMP_MISSING = 'TRAILING_STOP_LOSS_ON_FILL_GTD_TIM
```

TRAILING_STOP_LOSS_ON_FILL_PRICE_DISTANCE_INVALID = 'TRAILING_STOP_LOSS_ON_FILL_PRICE_

```
TRAILING STOP LOSS ON FILL PRICE DISTANCE MAXIMUM EXCEEDED = 'TRAILING STOP LOSS ON FI
    TRAILING STOP LOSS ON FILL PRICE DISTANCE MINIMUM NOT MET = 'TRAILING STOP LOSS ON FIL
    TRAILING_STOP_LOSS_ON_FILL_PRICE_DISTANCE_MISSING = 'TRAILING_STOP_LOSS_ON_FILL_PRICE_
    TRAILING_STOP_LOSS_ON_FILL_PRICE_DISTANCE_PRECISION_EXCEEDED = 'TRAILING_STOP_LOSS_ON_
    TRAILING STOP LOSS ON FILL TIME IN FORCE INVALID = 'TRAILING STOP LOSS ON FILL TIME IN
    TRAILING STOP LOSS ON FILL TIME IN FORCE MISSING = 'TRAILING STOP LOSS ON FILL TIME IN
    TRAILING_STOP_LOSS_ON_FILL_TRIGGER_CONDITION_INVALID = 'TRAILING_STOP_LOSS_ON_FILL_TRI
    TRAILING_STOP_LOSS_ON_FILL_TRIGGER_CONDITION_MISSING = 'TRAILING_STOP_LOSS_ON_FILL_TRI
    TRAILING_STOP_LOSS_ORDERS_NOT_SUPPORTED = 'TRAILING_STOP_LOSS_ORDERS_NOT_SUPPORTED'
    TRAILING_STOP_LOSS_ORDER_ALREADY_EXISTS = 'TRAILING_STOP_LOSS_ORDER_ALREADY_EXISTS'
    TRIGGER_CONDITION_INVALID = 'TRIGGER_CONDITION_INVALID'
    TRIGGER_CONDITION_MISSING = 'TRIGGER_CONDITION_MISSING'
    UNITS INVALID = 'UNITS INVALID'
    UNITS_LIMIT_EXCEEDED = 'UNITS_LIMIT_EXCEEDED'
    UNITS MIMIMUM NOT MET = 'UNITS MIMIMUM NOT MET'
    UNITS MISSING = 'UNITS MISSING'
    UNITS PRECISION EXCEEDED = 'UNITS PRECISION EXCEEDED'
    __getitem__(definitionID)
        return description for definitionID.
    definitions
        readonly property holding definition dict.
class oandapyV20.definitions.transactions.OrderCancelReason
    Bases: object
```

Definition representation of OrderCancelReason

```
>>> import oandapyV20.definitions.transactions as deftransactions
>>> print deftransactions.OrderCancelReason.LINKED_TRADE_CLOSED
LINKED_TRADE_CLOSED
>>> c = deftransactions.OrderCancelReason()
>>> print c[c.LINKED_TRADE_CLOSED]
The Order is linked to an open Trade that was closed.
>>> # or
>>> print deftransactions.OrderCancelReason().definitions[c.LINKED_TRADE_CLOSED]
>>> # all keys
>>> print deftransactions.OrderCancelReason().definitions.keys()
>>> ...
```

```
ACCOUNT_LOCKED = 'ACCOUNT_LOCKED'

ACCOUNT_NEW_POSITIONS_LOCKED = 'ACCOUNT_NEW_POSITIONS_LOCKED'

ACCOUNT_ORDER_CREATION_LOCKED = 'ACCOUNT_ORDER_CREATION_LOCKED'

ACCOUNT_ORDER_FILL_LOCKED = 'ACCOUNT_ORDER_FILL_LOCKED'
```

```
BOUNDS VIOLATION = 'BOUNDS VIOLATION'
    CLIENT REQUEST = 'CLIENT REQUEST'
    CLIENT_REQUEST_REPLACED = 'CLIENT_REQUEST_REPLACED'
    CLIENT_TRADE_ID_ALREADY_EXISTS = 'CLIENT_TRADE_ID_ALREADY_EXISTS'
    FIFO VIOLATION = 'FIFO VIOLATION'
    INSUFFICIENT LIQUIDITY = 'INSUFFICIENT LIQUIDITY'
    INSUFFICIENT_MARGIN = 'INSUFFICIENT_MARGIN'
    INTERNAL_SERVER_ERROR = 'INTERNAL_SERVER_ERROR'
    LINKED_TRADE_CLOSED = 'LINKED_TRADE_CLOSED'
    LOSING_TAKE_PROFIT = 'LOSING_TAKE_PROFIT'
    MARKET_HALTED = 'MARKET_HALTED'
    MIGRATION = 'MIGRATION'
    OPEN TRADES ALLOWED EXCEEDED = 'OPEN TRADES ALLOWED EXCEEDED'
    PENDING ORDERS ALLOWED EXCEEDED = 'PENDING ORDERS ALLOWED EXCEEDED'
    POSITION_CLOSEOUT_FAILED = 'POSITION_CLOSEOUT_FAILED'
    POSITION SIZE EXCEEDED = 'POSITION SIZE EXCEEDED'
    STOP LOSS ON FILL CLIENT ORDER ID ALREADY EXISTS = 'STOP LOSS ON FILL CLIENT ORDER ID .
    STOP_LOSS_ON_FILL_GTD_TIMESTAMP_IN_PAST = 'STOP_LOSS_ON_FILL_GTD_TIMESTAMP_IN_PAST'
    STOP_LOSS_ON_FILL_LOSS = 'STOP_LOSS_ON_FILL_LOSS'
    TAKE PROFIT ON FILL CLIENT ORDER ID ALREADY EXISTS = 'TAKE PROFIT ON FILL CLIENT ORDER
    TAKE PROFIT ON FILL GTD TIMESTAMP IN PAST = 'TAKE PROFIT ON FILL GTD TIMESTAMP IN PAST
    TAKE_PROFIT_ON_FILL_LOSS = 'TAKE_PROFIT_ON_FILL_LOSS'
    TIME_IN_FORCE_EXPIRED = 'TIME_IN_FORCE_EXPIRED'
    TRAILING STOP LOSS ON FILL CLIENT ORDER ID ALREADY EXISTS = 'TRAILING STOP LOSS ON FIL
    TRAILING STOP LOSS ON FILL GTD TIMESTAMP IN PAST = 'TRAILING STOP LOSS ON FILL GTD TIM
    __getitem__(definitionID)
        return description for definitionID.
    definitions
        readonly property holding definition dict.
\textbf{class} \ \texttt{oandapy} \texttt{V20.} definitions. \textbf{transactions.} \textbf{TrailingStopLossOrderReason}
    Bases: object
    Definition representation of TrailingStopLossOrderReason
```

```
>>> import oandapyV20.definitions.transactions as deftransactions
>>> print deftransactions.TrailingStopLossOrderReason.ON_FILL
ON_FILL
>>> c = deftransactions.TrailingStopLossOrderReason()
>>> print c[c.ON_FILL]
The Trailing Stop Loss Order was initiated automatically when an Order was filled_

== that opened a new Trade requiring a Trailing Stop Loss Order.
```

```
>>> # or
>>> print deftransactions.TrailingStopLossOrderReason().definitions[c.ON_FILL]
>>> # all keys
>>> print deftransactions.TrailingStopLossOrderReason().definitions.keys()
>>> ...

CLIENT_ORDER = 'CLIENT_ORDER'
ON_FILL = 'ON_FILL'
REPLACEMENT = 'REPLACEMENT'
__getitem__(definitionID)
    return description for definitionID.

definitions
```

readonly property holding definition dict.

```
class oandapyV20.definitions.transactions.LimitOrderReason
    Bases: object
```

Definition representation of LimitOrderReason

Definitions used in requests and responses. This class provides the ID and the description of the definitions.

```
>>> import oandapyV20.definitions.transactions as deftransactions
>>> print deftransactions.LimitOrderReason.CLIENT_ORDER
CLIENT_ORDER
>>> c = deftransactions.LimitOrderReason()
>>> print c[c.CLIENT_ORDER]
The Limit Order was initiated at the request of a client
>>> # or
>>> print deftransactions.LimitOrderReason().definitions[c.CLIENT_ORDER]
>>> # all keys
>>> print deftransactions.LimitOrderReason().definitions.keys()
>>> ...
```

```
CLIENT_ORDER = 'CLIENT_ORDER'

REPLACEMENT = 'REPLACEMENT'

__getitem___(definitionID)

return description for definitionID.
```

definitions

readonly property holding definition dict.

CHAPTER 5

oandapyV20.types

The oandapyV20.types module contains the types representing the types that are used in the API-specs of OANDA, check developer.oanda.com. These types offer a convenient interface between Python types and the types used in the REST-API.

Take for instance the *PriceValue* type. It is the string representation of a float.

```
from oandapyV20.types import PriceValue

pv1 = PriceValue(122.345)
pv2 = PriceValue("122.345")
pv1.value
"122.345"
pv1.value == pv2.value
True
```

Regardless the value we instantiate it with, a float or a string, the PriceValue instance will allways be a string value.

The types also validate the values passed. Invalid values will raise an exception.

5.1 AccountID

```
class oandapyV20.types.AccountID(accountID) representation of an AccountID, string value of an Account Identifier.
```

Parameters accountID (string (required)) - the accountID of a v20 account

Example

```
>>> print AccountID("001-011-5838423-001").value
```

A ValueError exception is raised in case of an incorrect value.

```
__init__(accountID)

value

value property.
```

5.2 AccountUnits

```
class oandapyV20.types.AccountUnits (units)
    representation AccountUnits, string value of a float.
    __init__ (units)
    value
        value property.
```

5.3 ClientComment

```
class oandapyV20.types.ClientComment (clientComment)
    representation of ClientComment, a string value of max 128 chars.
    __init__ (clientComment)
    value
        value property.
```

5.4 ClientID

```
class oandapyV20.types.ClientID(clientID)
    representation of ClientID, a string value of max 128 chars.
    __init__(clientID)
    value
        value property.
```

5.5 ClientTag

```
class oandapyV20.types.ClientTag(clientTag)
    representation of ClientTag, a string value of max 128 chars.
    __init__(clientTag)
    value
        value property.
```

5.6 DateTime

```
class oandapyV20.types.DateTime(dateTime) representation of a DateTime as a RFC 3339 string.
```

Parameters

```
• dateTime (string, datetime instance, dict (required)) — the dateTime parameter must be:
```

- a valid RFC3339 string representing a date-time, or
- a dict holding the relevant datetime parts, or
- a datetime.datetime instance
- value property is always RFC3339 datetime string (The) -
- seconds are in microseconds. This compatible with (Fractional)
- datetime.datetime. -

Example

A ValueError exception is raised in case of an invalid value

```
___init___(dateTime)
value
value property.
```

5.7 OrderID

```
class oandapyV20.types.OrderID(orderID) representation of an orderID, string value of an integer.
```

Parameters orderID (integer or string (required)) - the orderID as a positive integer or as a string

Example

```
>>> print OrderID(1234).value

A ValueError exception is raised in case of a negative integer value
__init__(orderID)
value
```

5.8 Orderldentifier

value property.

```
class oandapy V20.types.OrderIdentifier (orderID, clientID) representation of the OrderIdentifier object.
```

5.7. OrderID 103

```
__init__ (orderID, clientID)

value

value property.
```

5.9 OrderSpecifier

```
class oandapyV20.types.OrderSpecifier(specifier)
    representation of the OrderSpecifier.
    __init__(specifier)
    value
        value property.
```

5.10 PriceValue

```
class oandapyV20.types.PriceValue (priceValue)
    representation PriceValue, string value of a float.
    __init__ (priceValue)
    value
        value property.
```

5.11 TradeID

```
class oandapyV20.types.TradeID(tradeID) representation of a tradeID, string value of an integer.
```

Parameters tradeID (integer or string (required)) - the tradeID as a positive integer or as a string

Example

```
>>> print TradeID(1234).value
```

A ValueError exception is raised in case of a negative integer value

```
__init__(tradeID)

value

value property.
```

5.12 Units

```
class oandapyV20.types.Units(units)
    representation Units, string value of an integer.
    __init__(units)
```

value

value property.

5.12. Units 105

oandapyV20.contrib

6.1 Factories

The oandapyV20.contrib.factories module contains several classes / methods that can be used optionally to generate requests.

6.1.1 InstrumentsCandlesFactory

oandapyV20.contrib.factories.InstrumentsCandlesFactory(instrument, params=None) InstrumentsCandlesFactory - generate InstrumentCandles requests.

InstrumentsCandlesFactory is used to retrieve historical data by automatically generating consecutive requests when the OANDA limit of *count* records is exceeded.

This is known by calculating the number of candles between *from* and *to*. If *to* is not specified *to* will be equal to *now*

The *count* parameter is only used to control the number of records to retrieve in a single request.

The *includeFirst* parameter is forced to make sure that results do no have a 1-record gap between consecutive requests.

Parameters

- instrument (string (required)) the instrument to create the order for
- params (params (optional)) the parameters to specify the historical range, see the REST-V20 docs regarding 'instrument' at developer.oanda.com If no params are specified, just a single InstrumentsCandles request will be generated acting the same as if you had just created it directly.

Example

The *oandapyV20.API* client processes requests as objects. So, downloading large historical batches simply comes down to:

```
>>> import json
>>> from oandapyV20 import API
>>> from oandapyV20.contrib.factories import InstrumentsCandlesFactory
>>> client = API(access_token=...)
>>> instrument, granularity = "EUR_USD", "M15"
>>> _from = "2017-01-01T00:00:00Z"
>>> params = {
       "from": _from,
. . .
       "granularity": granularity,
. . .
       "count": 2500,
. . .
...}
>>> with open("/tmp/{}.{}".format(instrument, granularity), "w") as OUT:
       # The factory returns a generator generating consecutive
>>>
        # requests to retrieve full history from date 'from' till 'to'
>>>
        for r in InstrumentsCandlesFactory(instrument=instrument,
>>>
                                            params=params)
>>>
            client.request(r)
            OUT.write(json.dumps(r.response.get('candles'), indent=2))
>>>
```

Note: Normally you can't combine *from*, *to* and *count*. When *count* specified, it is used to calculate the gap between *to* and *from*. The *params* passed to the generated request itself does contain the *count* parameter.

6.2 Generic

The oandapy V20. contrib. generic module contains several classes / methods that serve a generic purpose.

6.2.1 granularity_to_time

```
oandapyV20.contrib.generic.granularity_to_time(s) convert a named granularity into seconds.
```

get value in seconds for named granularities: M1, M5 ... H1 etc.

```
>>> print(granularity_to_time("M5"))
300
```

```
oandapyV20.contrib.generic.secs2time(e) secs2time - convert epoch to datetime.
```

```
>>> d = secs2time(1497499200)
>>> d
datetime.datetime(2017, 6, 15, 4, 0)
>>> d.strftime("%Y%m%d-%H:%M:%S")
'20170615-04:00:00'
```

6.3 Order Classes

The oandapy V20.contrib.requests module contains several classes that can be used optionally when creating Order Requests.

When creating an order to create a position, it is possible to create dependant orders that will be triggered when the position gets filled. This goes typically for *Take Profit* and *Stop Loss*.

These order specifications and additional data that goes with these order specifications can be created by the contrib.requests.*Order* classes and the contrib.requests.*Details classes.

6.3.1 LimitOrderRequest

```
class oandapyV20.contrib.requests.LimitOrderRequest (instrument, units, price, posi-
tionFill='DEFAULT', clientEx-
tensions=None, takeProfitOn-
Fill=None, timeInForce='GTC',
gtdTime=None, stopLossOn-
Fill=None, trailingStopLossOn-
Fill=None, tradeClientExten-
sions=None)
```

Bases: oandapy V20.contrib.requests.baserequest.BaseRequest create a LimitOrderRequest.

LimitOrderRequest is used to build the body for a LimitOrder. The body can be used to pass to the OrderCreate endpoint.

```
__init__ (instrument, units, price, positionFill='DEFAULT', clientExtensions=None, takeProfitOn-
Fill=None, timeInForce='GTC', gtdTime=None, stopLossOnFill=None, trailingStopLos-
sOnFill=None, tradeClientExtensions=None)
Instantiate a LimitOrderRequest.
```

Parameters

- instrument (string (required)) the instrument to create the order for
- units (integer (required)) the number of units. If positive the order results in a LONG order. If negative the order results in a SHORT order
- **price** (float (required)) the price indicating the limit.

Example

```
>>> import json
>>> from oandapyV20 import API
>>> import oandapyV20.endpoints.orders as orders
>>> from oandapyV20.contrib.requests import LimitOrderRequest
>>>
>>> accountID = "..."
>>> client = API(access_token=...)
>>> ordr = LimitOrderRequest(instrument="EUR_USD",
... units=10000, price=1.08)
>>> print(json.dumps(ordr.data, indent=4))
{
    "order": {
        "timeInForce": "GTC",
```

6.3. Order Classes 109

data property.

return the JSON order body

6.3.2 MarketOrderRequest

```
class oandapyV20.contrib.requests.MarketOrderRequest(instrument,
                                                                                 units,
                                                                                         price-
                                                                   Bound=None.
                                                                                      position-
                                                                   Fill='DEFAULT',
                                                                                          clien-
                                                                   tExtensions=None,
                                                                                          take-
                                                                   ProfitOnFill=None,
                                                                                        timeIn-
                                                                   Force='FOK',
                                                                                   stopLossOn-
                                                                                trailingStopLos-
                                                                   Fill=None,
                                                                   sOnFill=None, tradeClientEx-
                                                                   tensions=None)
```

Bases: oandapyV20.contrib.requests.baserequest.BaseRequest

create a MarketOrderRequest.

MarketOrderRequest is used to build the body for a MarketOrder. The body can be used to pass to the Order-Create endpoint.

__init__ (instrument, units, priceBound=None, positionFill='DEFAULT', clientExtensions=None, takeProfitOnFill=None, timeInForce='FOK', stopLossOnFill=None, trailingStopLossOn-Fill=None, tradeClientExtensions=None)
Instantiate a MarketOrderRequest.

Parameters

- instrument (string (required)) the instrument to create the order for
- units (integer (required)) the number of units. If positive the order results in a LONG order. If negative the order results in a SHORT order

```
>>> import json
>>> from oandapyV20 import API
>>> import oandapyV20.endpoints.orders as orders
>>> from oandapyV20.contrib.requests import MarketOrderRequest
>>>
>>> accountID = "..."
>>> client = API(access_token=...)
>>> mo = MarketOrderRequest(instrument="EUR_USD", units=10000)
>>> print(json.dumps(mo.data, indent=4))
```

```
"order": {
        "type": "MARKET",
        "positionFill": "DEFAULT",
        "instrument": "EUR_USD",
        "timeInForce": "FOK",
        "units": "10000"
>>> # now we have the order specification, create the order request
>>> r = orders.OrderCreate(accountID, data=mo.data)
>>> # perform the request
>>> rv = client.request(r)
>>> print(rv)
>>> print(json.dumps(rv, indent=4))
    "orderFillTransaction": {
        "reason": "MARKET_ORDER",
        "pl": "0.0000",
        "accountBalance": "97864.8813",
        "units": "10000",
        "instrument": "EUR_USD",
        "accountID": "101-004-1435156-001",
        "time": "2016-11-11T19:59:43.253587917Z",
        "type": "ORDER_FILL",
        "id": "2504",
        "financing": "0.0000",
        "tradeOpened": {
            "tradeID": "2504",
            "units": "10000"
        "orderID": "2503",
        "userID": 1435156,
        "batchID": "2503",
        "price": "1.08463"
    "lastTransactionID": "2504",
    "relatedTransactionIDs": [
        "2503",
        "2504"
    "orderCreateTransaction": {
        "type": "MARKET_ORDER",
        "reason": "CLIENT_ORDER",
        "id": "2503",
        "timeInForce": "FOK",
        "units": "10000",
        "time": "2016-11-11T19:59:43.253587917Z",
        "positionFill": "DEFAULT",
        "accountID": "101-004-1435156-001",
        "instrument": "EUR_USD",
        "batchID": "2503",
        "userID": 1435156
    }
>>>
```

6.3. Order Classes

data property.

return the JSON body.

6.3.3 MITOrderRequest

```
class oandapyV20.contrib.requests.MITOrderRequest (instrument,
                                                                                 units,
                                                                                            price,
                                                                 priceBound=None,
                                                                                         position-
                                                                 Fill='DEFAULT'.
                                                                                          timeIn-
                                                                                   gtdTime=None,
                                                                 Force='GTC',
                                                                 clientExtensions=None,
                                                                                            take-
                                                                 ProfitOnFill=None,
                                                                                      stopLossOn-
                                                                 Fill=None,
                                                                               trailingStopLossOn-
                                                                 Fill=None,
                                                                                 tradeClientExten-
                                                                 sions=None)
```

Bases: oandapyV20.contrib.requests.baserequest.BaseRequest

create a MarketIfTouched OrderRequest.

MITOrderRequest is used to build the body for a MITOrder. The body can be used to pass to the OrderCreate endpoint.

__init__ (instrument, units, price, priceBound=None, positionFill='DEFAULT', timeInForce='GTC', gtdTime=None, clientExtensions=None, takeProfitOnFill=None, stopLossOnFill=None, trailingStopLossOnFill=None, tradeClientExtensions=None)
Instantiate an MITOrderRequest.

Parameters

- instrument (string (required)) the instrument to create the order for
- units (integer (required)) the number of units. If positive the order results in a LONG order. If negative the order results in a SHORT order
- price (float (required)) the price indicating the limit.

```
>>> import json
>>> from oandapyV20 import API
>>> import oandapyV20.endpoints.orders as orders
>>> from oandapyV20.contrib.requests import MITOrderRequest
>>>
>>> accountID = "..."
>>> client = API(access_token=...)
>>> ordr = MITOrderRequest(instrument="EUR_USD",
                         units=10000, price=1.08)
>>> print(json.dumps(ordr.data, indent=4))
    "order": {
        "timeInForce": "GTC",
        "instrument": "EUR_USD",
        "units": "10000",
        "price": "1.08000",
        "type": "MARKET_IF_TOUCHED",
        "positionFill": "DEFAULT"
```

```
>>> r = orders.OrderCreate(accountID, data=ordr.data)
>>> rv = client.request(r)
>>> ...
```

data property.

return the JSON order body

6.3.4 PositionCloseRequest

create a PositionCloseRequest.

PositionCloseRequest is used to build the body to close a position. The body can be used to pass to the PositionClose endpoint.

```
__init__(longUnits=None, longClientExtensions=None, shortUnits=None, shortClientExtensions=None)
Instantiate a PositionCloseRequest.
```

Parameters

- longUnits (integer (optional)) the number of long units to close
- longClientExtensions (dict (optional)) dict representing longClientExtensions
- shortUnits (integer (optional)) the number of short units to close
- **shortClientExtensions** (dict (optional)) dict representing shortClientExtensions

One of the parameters or both must be supplied.

Example

```
>>> import json
>>> from oandapyV20 import API
>>> import oandapyV20.endpoints.positions as positions
>>> from oandapyV20.contrib.requests import PositionCloseRequest
>>>
>>> accountID = "..."
>>> client = API(access_token=...)
>>> ordr = PositionCloseRequest(longUnits=10000)
>>> print(json.dumps(ordr.data, indent=4))
{
    "longUnits": "10000"
}
>>> # now we have the order specification, create the order request
>>> r = position.PositionClose(accountID,
>>> instrument="EUR_USD", data=ordr.data)
>>> # perform the request
```

6.3. Order Classes 113

```
>>> rv = client.request(r)
>>> print(rv)
>>> ...
```

6.3.5 StopLossOrderRequest

 $Bases: \verb| oandapyV20.contrib.requests.baserequest.BaseRequest| \\$

create a StopLossOrderRequest.

StopLossOrderRequest is used to build the body for a StopLossOrder. The body can be used to pass to the OrderCreate endpoint.

```
__init__(tradeID, price, clientTradeID=None, timeInForce='GTC', gtdTime=None, clientExtensions=None)
Instantiate a StopLossOrderRequest.
```

Parameters

- tradeID (string (required)) the tradeID of an existing trade
- **price** (float (required)) the treshold price indicating the price to close the order

```
>>> import json
>>> from oandapyV20 import API
>>> import oandapyV20.endpoints.orders as orders
>>> from oandapyV20.contrib.requests import StopLossOrderRequest
>>> accountID = "..."
>>> client = API(access_token=...)
>>> ordr = StopLossOrderRequest(tradeID="1234", price=1.07)
>>> print(json.dumps(ordr.data, indent=4))
    "order": {
        "type": "STOP_LOSS",
        "tradeID": "1234",
        "price": "1.07000",
        "timeInForce": "GTC",
>>> # now we have the order specification, create the order request
>>> r = orders.OrderCreate(accountID, data=ordr.data)
>>> # perform the request
>>> rv = client.request(r)
>>> print(json.dumps(rv, indent=4))
>>> ...
```

data property.

return the JSON body.

6.3.6 StopOrderRequest

```
class oandapyV20.contrib.requests.StopOrderRequest (instrument,
                                                                                units.
                                                                                         price,
                                                                 priceBound=None,
                                                                                       position-
                                                                 Fill='DEFAULT',
                                                                                        timeIn-
                                                                 Force='GTC',
                                                                               gtdTime=None,
                                                                 clientExtensions=None,
                                                                                          take-
                                                                 ProfitOnFill=None, stopLossOn-
                                                                 Fill=None, trailingStopLossOn-
                                                                 Fill=None,
                                                                               tradeClientExten-
                                                                 sions=None)
```

Bases: oandapyV20.contrib.requests.baserequest.BaseRequest

create a StopOrderRequest.

StopOrderRequest is used to build the body for an StopOrder. The body can be used to pass to the OrderCreate endpoint.

__init__ (instrument, units, price, priceBound=None, positionFill='DEFAULT', timeInForce='GTC', gtdTime=None, clientExtensions=None, takeProfitOnFill=None, stopLossOnFill=None, trailingStopLossOnFill=None, tradeClientExtensions=None)
Instantiate a StopOrderRequest.

Parameters

- instrument (string (required)) the instrument to create the order for
- units (integer (required)) the number of units. If positive the order results in a LONG order. If negative the order results in a SHORT order
- **price** (float (required)) the treshold price indicating the price to activate the order

Example

```
>>> import json
>>> from oandapyV20 import API
>>> import oandapyV20.endpoints.orders as orders
>>> from oandapyV20.contrib.requests import StopOrderRequest
>>>
>>> accountID = "..."
>>> client = API(access_token=...)
>>> ordr = StopOrderRequest(instrument="EUR_USD",
                            units=10000, price=1.07)
>>> print(json.dumps(ordr.data, indent=4))
    "order": {
        "type": "STOP",
        "price": "1.07000",
        "positionFill": "DEFAULT",
        "instrument": "EUR_USD",
        "timeInForce": "GTC",
```

6.3. Order Classes 115

```
"units": "10000"
}

}
>>> # now we have the order specification, create the order request
>>> r = orders.OrderCreate(accountID, data=ordr.data)
>>> # perform the request
>>> rv = client.request(r)
>>> print(json.dumps(rv, indent=4))
>>> ...
```

data property.

return the JSON body.

6.3.7 TakeProfitOrderRequest

 $Bases: \verb| oandapyV20.contrib.requests.baserequest.BaseRequest| \\$

create a TakeProfit OrderRequest.

TakeProfitOrderRequest is used to build the body for a TakeProfitOrder. The body can be used to pass to the OrderCreate endpoint.

```
__init__ (tradeID, price, clientTradeID=None, timeInForce='GTC', gtdTime=None, clientExtensions=None)
Instantiate a TakeProfitOrderRequest.
```

Parameters

- tradeID (string (required)) the tradeID of an existing trade
- price (float (required)) the price indicating the target price to close the order.

```
}
}
>>> r = orders.OrderCreate(accountID, data=ordr.data)
>>> rv = client.request(r)
>>> ...
```

data property.

return the JSON order body

6.3.8 TradeCloseRequest

```
class oandapyV20.contrib.requests.TradeCloseRequest (units='ALL')
    Bases: oandapyV20.contrib.requests.baserequest.BaseRequest
    create a TradeCloseRequest.
```

TradeCloseRequest is used to build the body to close a trade. The body can be used to pass to the TradeClose endpoint.

```
__init__ (units='ALL')
Instantiate a TradeCloseRequest.
```

Parameters units (integer (optional)) – the number of units to close. Default it is set to "ALL".

Example

```
>>> import json
>>> from oandapyV20 import API
>>> import oandapyV20.endpoints.trades as trades
>>> from oandapyV20.contrib.requests import TradeCloseRequest
>>>
>>> accountID = "..."
>>> client = API(access_token=...)
>>> ordr = TradeCloseRequest(units=10000)
>>> print(json.dumps(ordr.data, indent=4))
   "units": "10000"
>>> # now we have the order specification, create the order request
>>> r = trades.TradeClose(accountID, tradeID=1234,
                          data=ordr.data)
>>> # perform the request
>>> rv = client.request(r)
>>> print(rv)
>>> ...
```

6.3. Order Classes 117

6.3.9 TrailingStopLossOrderRequest

```
class oandapyV20.contrib.requests.TrailingStopLossOrderRequest (tradeID, dis-
tance, client-
TradeID=None,
timeIn-
Force='GTC',
gtdTime=None,
clientExten-
sions=None)
```

Bases: oandapyV20.contrib.requests.baserequest.BaseRequest create a TrailingStopLossOrderRequest.

TrailingStopLossOrderRequest is used to build the body for a TrailingStopLossOrder. The body can be used to pass to the OrderCreate endpoint.

```
__init__ (tradeID, distance, clientTradeID=None, timeInForce='GTC', gtdTime=None, clientExten-
sions=None)
Instantiate a TrailingStopLossOrderRequest.
```

Parameters

- tradeID (string (required)) the tradeID of an existing trade
- distance (float (required)) the price distance

Example

```
>>> import json
>>> from oandapyV20 import API
>>> import oandapyV20.endpoints.orders as orders
>>> from oandapyV20.contrib.requests import TrailingStopLossOrderRequest
>>>
>>> accountID = "..."
>>> client = API(access_token=...)
>>> ordr = TrailingStopLossOrderRequest(tradeID="1234", distance=20)
>>> print(json.dumps(ordr.data, indent=4))
    "order": {
        "type": "TRAILING_STOP_LOSS",
        "tradeID": "1234",
        "timeInForce": "GTC",
        "distance": "20.00000"
>>> # now we have the order specification, create the order request
>>> r = orders.OrderCreate(accountID, data=ordr.data)
>>> # perform the request
>>> rv = client.request(r)
>>> print(json.dumps(rv, indent=4))
>>> ...
```

data

data property.

return the JSON body.

6.4 support classes

The oandapy V20.contrib.requests module contains several classes that can be used optionally when creating Order Requests.

When creating an order to create a position, it is possible to create dependant orders that will be triggered when the position gets filled. This goes typically for *Take Profit* and *Stop Loss*.

These order specifications and additional data that goes with these order specifications can be created by the contrib.requests.*Order* classes and the contrib.requests.*Details classes.

6.4.1 Client Extensions

Client extensions can be used optionally on Order Requests. It allows a client to set a custom ID, Tag and/or Comment.

```
class oandapyV20.contrib.requests.ClientExtensions(clientID=None, clientTag=None, clientComment=None)

Bases: oandapyV20.contrib.requests.baserequest.BaseRequest

Representation of the ClientExtensions.
```

```
__init__(clientID=None, clientTag=None, clientComment=None)
Instantiate ClientExtensions.
```

Parameters

- clientID (clientID (required)) the clientID
- clientTag (clientTag (required)) the clientTag
- clientComment (clientComment (required)) the clientComment

```
>>> import json
>>> from oandapyV20 import API
>>> import oandapyV20.endpoints.orders as orders
>>> from oandapyV20.contrib.requests import (
        MarketOrderRequest, TakeProfitDetails, ClientExtensions)
. . .
>>>
>>> accountID = "..."
>>> client = API(access_token=...)
>>> # at time of writing EUR_USD = 1.0740
>>> # let us take profit at 1.10, GoodTillCancel (default)
>>> # add clientExtensions to it also
>>> takeProfitOnFillOrder = TakeProfitDetails(
        price=1.10,
        clientExtensions=ClientExtensions(clientTag="mytag").data)
>>> print (takeProfitOnFillOrder.data)
    'timeInForce': 'GTC',
    'price": '1.10000',
    'clientExtensions': {'tag': 'mytag'}
>>> ordr = MarketOrderRequest (
       instrument="EUR_USD",
       units=10000,
```

```
takeProfitOnFill=takeProfitOnFillOrder.data
...)
>>> # or as shortcut ...
>>> # takeProfitOnFill=TakeProfitDetails(price=1.10).data
>>> print(json.dumps(ordr.data, indent=4))
>>> r = orders.OrderCreate(accountID, data=ordr.data)
>>> rv = client.request(r)
>>> ...
```

6.4.2 StopLossDetails

```
class oandapy V20.contrib.requests. StopLossDetails (price, timeInForce='GTC', gtdTime=None, clientExtensions=None) Bases: oandapy V20.contrib.requests.onfill.OnFill
```

 $Representation\ of\ the\ specification\ for\ a\ StopLossOrder.$

It is typically used to specify 'stop loss details' for the 'stopLossOnFill' parameter of an OrderRequest. This way one can create the Stop Loss Order as a dependency when an order gets filled.

The other way to create a StopLossOrder is to create it afterwards on an existing trade. In that case you use StopLossOrderRequest on the trade.

```
__init__ (price, timeInForce='GTC', gtdTime=None, clientExtensions=None)
Instantiate StopLossDetails.
```

Parameters

- price (float or string (required)) the price to trigger take profit order
- timeInForce (TimeInForce (required), default TimeInForce.GTC)
 the time in force
- gtdTime (DateTime (optional)) gtdTime is required in case timeInForce == TimeInForce.GTD
- clientExtensions (ClientExtensions (optional)) -

```
>>> import json
>>> from oandapyV20 import API
>>> import oandapyV20.endpoints.orders as orders
>>> from oandapyV20.contrib.requests import (
>>> MarketOrderRequest, StopLossDetails)
>>>
>>> accountID = "..."
>>> client = API(access_token=...)
>>> # at time of writing EUR_USD = 1.0740
>>> # let us take profit at 1.10, GoodTillCancel (default)
>>> stopLossOnFill = StopLossDetails(price=1.06)
>>> print(stopLossOnFill)
{
    "timeInForce": "GTC",
    "price": "1.10000"
}
```

6.4.3 TakeProfitDetails

Bases: oandapyV20.contrib.requests.onfill.OnFill

Representation of the specification for a TakeProfitOrder.

It is typically used to specify 'take profit details' for the 'takeProfitOnFill' parameter of an OrderRequest. This way one can create the Take Profit Order as a dependency when an order gets filled.

The other way to create a TakeProfitOrder is to create it afterwards on an existing trade. In that case you use TakeProfitOrderRequest on the trade.

```
__init__ (price, timeInForce='GTC', gtdTime=None, clientExtensions=None)
Instantiate TakeProfitDetails.
```

Parameters

- price (float or string (required)) the price to trigger take profit order
- timeInForce (TimeInForce (required), default TimeInForce.GTC)
 the time in force
- gtdTime (DateTime (optional)) gtdTime is required in case timeInForce == TimeInForce.GTD

```
>>> import json
>>> from oandapyV20 import API
>>> import oandapyV20.endpoints.orders as orders
>>> from oandapyV20.contrib.requests import (
>>> MarketOrderRequest, TakeProfitDetails)
>>>
>>> accountID = "..."
>>> client = API(access_token=...)
>>> # at time of writing EUR_USD = 1.0740
>>> # let us take profit at 1.10, GoodTillCancel (default)
>>> takeProfitOnFillOrder = TakeProfitDetails(price=1.10)
>>> print(takeProfitOnFillOrder.data)
{
    "timeInForce": "GTC",
    "price": "1.10000"
```

```
>>> ordr = MarketOrderRequest (
       instrument="EUR USD",
>>>
       units=10000,
>>>
       takeProfitOnFill=takeProfitOnFillOrder.data
>>>
>>> )
>>> # or as shortcut ...
>>> # takeProfitOnFill=TakeProfitDetails(price=1.10).data
>>> print(json.dumps(ordr.data, indent=4))
    "order": {
        "timeInForce": "FOK",
        "instrument": "EUR_USD",
        "units": "10000",
        "positionFill": "DEFAULT",
        "type": "MARKET",
        "takeProfitOnFill": {
            "timeInForce": "GTC",
            "price": "1.10000"
    }
>>> r = orders.OrderCreate(accountID, data=ordr.data)
>>> rv = client.request(r)
>>> ...
```

6.4.4 TrailingStopLossDetails

```
class oandapyV20.contrib.requests. TrailingStopLossDetails (distance, timeIn-Force='GTC', gtd-Time=None, clientEx-tensions=None)
```

 $Bases: \verb|oandapyV20.contrib.requests.onfill.OnFill| \\$

Representation of the specification for a TrailingStopLossOrder.

It is typically used to specify 'trailing stop loss details' for the 'trailingStopLossOnFill' parameter of an Order-Request. This way one can create the Trailing Stop Loss Order as a dependency when an order gets filled.

The other way to create a TrailingStopLossOrder is to create it afterwards on an existing trade. In that case you use TrailingStopLossOrderRequest on the trade.

```
__init__ (distance, timeInForce='GTC', gtdTime=None, clientExtensions=None)
Instantiate TrailingStopLossDetails.
```

Parameters

- distance (float or string (required)) the price to trigger trailing stop loss order
- timeInForce (TimeInForce (required), default TimeInForce.GTC)
 the time in force
- gtdTime (DateTime (optional)) gtdTime is required in case timeInForce == TimeInForce.GTD
- clientExtensions (ClientExtensions (optional)) -

```
>>> import json
>>> from oandapyV20 import API
>>> import oandapyV20.endpoints.orders as orders
>>> from oandapyV20.contrib.requests import (
>>>
       MarketOrderRequest, TrailingStopLossDetails)
>>>
>>> accountID = "..."
>>> client = API(access_token=...)
>>> # at time of writing EUR_USD = 1.0740
>>> # let us take profit at 1.10, GoodTillCancel (default)
>>> trailingStopLossOnFill = TrailingStopLossDetails(price=1.06)
>>> print(trailingStopLossOnFill)
    "timeInForce": "GTC",
   "price": "1.10000"
>>> ordr = MarketOrderRequest (
     instrument="EUR_USD",
       units=10000,
>>>
>>>
       trailingStopLossOnFill=trailingStopLossOnFill.data
>>> )
>>> # or as shortcut ...
>>> # ...OnFill=trailingStopLossDetails(price=1.06).data
>>> print(json.dumps(ordr.data, indent=4))
>>> r = orders.OrderCreate(accountID, data=ordr.data)
>>> rv = client.request(r)
>>> ...
```

CHAPTER 7

Examples

Examples can be found in the examples repositiony on github: examples reposition.

7.1 Example for trades-endpoints

Take the script below and name it 'trades.py'. From the shell:

```
hootnot@dev:~/test$ python trades.py list
hootnot@dev:~/test$ python trades.py open
hootnot@dev:~/test$ python trades.py details <id1> [<id2> ...]
hootnot@dev:~/test$ python trades.py close <id1> <numunits> [<id2> <numunits>...]
hootnot@dev:~/test$ python trades.py clext <id1> [<id2> ...]
hootnot@dev:~/test$ python trades.py crc_do <id1> <takeprofit> <stoploss> [<id2> ...]
```

```
if chc == 'open':
  r = trades.OpenTrades(accountID)
  rv = api.request(r)
  print("RESP:\n{} ".format(json.dumps(rv, indent=2)))
  tradeIDs = [o["id"] for o in rv["trades"]]
  print("TRADE IDS: {}".format(tradeIDs))
if chc == 'details':
  for 0 in sys.argv[2:]:
      r = trades.TradeDetails(accountID, tradeID=0)
      rv = api.request(r)
      print("RESP:\n{} ".format(json.dumps(rv, indent=2)))
if chc == 'close':
  X = iter(sys.argv[2:])
  for 0 in X:
      cfg = { "units": X.next() }
      r = trades.TradeClose(accountID, tradeID=0, data=cfg)
      rv = api.request(r)
      print("RESP:\n{} ".format(json.dumps(rv, indent=2)))
if chc == 'cltext':
  for 0 in sys.argv[2:]: # tradeIDs
       cfg = { "clientExtensions": {
              "id": "myID{}".format(0),
               "comment": "myComment",
      r = trades.TradeClientExtensions(accountID, tradeID=0, data=cfg)
       rv = api.request(r)
      print("RESP:\n{} ".format(json.dumps(rv, indent=2)))
if chc == 'crc_do':
  X = iter(sys.argv[2:])
   for 0 in X:
       cfg = {
               "takeProfit": {
                 "timeInForce": "GTC",
                 "price": X.next(),
               },
               "stopLoss": {
                 "timeInForce": "GTC",
                 "price": X.next()
               }
       r = trades.TradeCRCDO(accountID, tradeID=0, data=cfg)
       rv = api.request(r)
      print("RESP:\n{} ".format(json.dumps(rv, indent=2)))
```

CHAPTER 8

Indices and tables

- genindex
- modindex
- search

Python Module Index

0

```
oandapyV20.contrib.factories, 107
oandapyV20.contrib.generic, 108
oandapyV20.definitions.accounts, 79
oandapyV20.definitions.instruments, 80
oandapyV20.definitions.orders, 82
oandapyV20.definitions.pricing, 86
oandapyV20.definitions.trades, 87
oandapyV20.definitions.transactions, 89
```

130 Python Module Index

```
Symbols
                                                            _getitem__() (oandapy V20.definitions.transactions.MarketOrderMarginCl
__getitem__() (oandapyV20.definitions.accounts.AccountFinancingMmethod), 89
                                                           _getitem__() (oandapy V20.definitions.transactions.MarketOrderReason
         method), 79
__getitem__() (oandapyV20.definitions.accounts.PositionAggregationWethed), 92
                                                           _getitem__() (oandapy V20.definitions.transactions.OrderCancelReason
         method), 80
__getitem__() (oandapyV20.definitions.instruments.CandlestickGranufacthod), 99
                                                           _getitem__() (oandapyV20.definitions.transactions.OrderFillReason
         method), 81
 _getitem__() (oandapyV20.definitions.instruments.PriceComponentsmethod), 90
                                                            getitem_() (oandapy V20.definitions.transactions.StopLossOrderReason
         method), 82
__getitem__() (oandapyV20.definitions.instruments.WeeklyAlignmentmethod), 90
                                                           _getitem__() (oandapy V20.definitions.transactions.StopOrderReason
         method), 82
__getitem__() (oandapyV20.definitions.orders.CancellableOrderType method), 92
                                                           _getitem__() (oandapy V20.definitions.transactions.TakeProfitOrderReason
         method), 84
                                                                   method), 94
 getitem () (oandapy V20. definitions. orders. Order Position Fill
                                                            getitem__() (oandapy V20.definitions.transactions.TrailingStopLossOrder
         method), 85
                                                                   method), 100
getitem () (oandapy V20.definitions.orders.OrderState
                                                            getitem__() (oandapy V20.definitions.transactions.TransactionRejectReas
         method), 86
                                                                   method), 98
__getitem__() (oandapy V20.definitions.orders.OrderStateFilter
                                                           _getitem__() (oandapyV20.definitions.transactions.TransactionType
         method), 83
__getitem__() (oandapyV20.definitions.orders.OrderTriggerConditionmethod), 94
                                                         __init__() (oandapyV20.API method), 7
         method), 86
                                                           _init__() (oandapyV20.V20Error method), 8
 _getitem__() (oandapy V20.definitions.orders.OrderType
                                                           init () (oandapy V20.contrib.requests.ClientExtensions
         method), 83
                                                                   method), 119
 _getitem__() (oandapy V20.definitions.orders.TimeInForce
                                                                 () (oandapy V20.contrib.requests.LimitOrderRequest
         method), 85
                                                                   method), 109
 getitem () (oandapy V20. definitions. pricing. Price Status
                                                            _init__() (oandapy V20.contrib.requests.MITOrderRequest
         method), 87
                                                                   method), 112
__getitem__()
               (oandapy V20.definitions.trades.TradePL
                                                            init__() (oandapy V20.contrib.requests.MarketOrderRequest
         method), 87
                                                                   method), 110
 __getitem__() (oandapyV20.definitions.trades.TradeState
                                                            init__() (oandapy V20.contrib.requests.PositionCloseRequest
         method), 88
                                                                   method), 113
__getitem__() (oandapy V20.definitions.trades.TradeStateFilter
                                                                 _() (oandapy V20.contrib.requests.StopLossDetails
         method), 88
__getitem__() (oandapyV20.definitions.transactions.FundingReason method), 120
                                                           _init__() (oandapy V20.contrib.requests.StopLossOrderRequest
         method), 91
__getitem__() (oandapyV20.definitions.transactions.LimitOrderReasomethod), 114
                                                          __init__() (oandapy V20.contrib.requests.StopOrderRequest
         method), 100
__getitem__() (oandapy V20.definitions.transactions.MarketIfTouched Of the Reason
                                                          __init__() (oandapyV20.contrib.requests.TakeProfitDetails
         method), 91
                                                                   method), 121
```

init() (oandapy V20.contrib.requests.TakeProfitOrderRequinsit() (oandapy V20.endpoints.positions.PositionClose
method), 116 method), 52
init() (oandapy V20.contrib.requests.TradeCloseRequestinit() (oandapy V20.endpoints.positions.PositionDetails
method), 117 method), 53
init() (oandapyV20.contrib.requests.TrailingStopLossDetailit() (oandapyV20.endpoints.positions.PositionList
method), 122 method), 54
init() (oandapyV20.contrib.requests.TrailingStopLossOrdenRequest (oandapyV20.endpoints.pricing.PricingInfo
method), 118 method), 56
init() (oandapyV20.endpoints.accounts.AccountChanges_init() (oandapyV20.endpoints.pricing.PricingStream
method), 11 method), 59
init() (oandapyV20.endpoints.accounts.AccountConfigurathitn() (oandapyV20.endpoints.trades.OpenTrades
method), 13 method), 62
init() (oandapy V20.endpoints.accounts.AccountDetailsinit() (oandapy V20.endpoints.trades.TradeCRCDO
method), 14 method), 63
init() (oandapy V20.endpoints.accounts.AccountInstrumeritsit() (oandapy V20.endpoints.trades.TradeClientExtensions
method), 17 method), 65
init() (oandapyV20.endpoints.accounts.AccountListinit() (oandapyV20.endpoints.trades.TradeClose method), 19 method), 66
init() (oandapy V20.endpoints.accounts.AccountSummaryinit() (oandapy V20.endpoints.trades.TradeDetails
method), 19 method), 68
init() (oandapyV20.endpoints.forexlabs.Autochartistinit() (oandapyV20.endpoints.trades.TradesList
method), 21 method), 68
init() (oandapyV20.endpoints.forexlabs.Calendarinit() (oandapyV20.endpoints.transactions.TransactionDetails
method), 22 method), 70
init() (oandapy V20.endpoints.forexlabs.Commitments Of Traitler() (oandapy V20.endpoints.transactions.TransactionIDRange
method), 23 method), 71
init() (oandapy V20.endpoints.forexlabs.HistoricalPositionRittio() (oandapy V20.endpoints.transactions.TransactionList
method), 25 method), 72
init() (oandapy V20.endpoints.forexlabs.OrderbookDatainit() (oandapy V20.endpoints.transactions.TransactionsSinceII
method), 32 method), 74
init() (oandapyV20.endpoints.forexlabs.Spreadsinit() (oandapyV20.endpoints.transactions.TransactionsStream
method), 33 method), 76
init() (oandapy V20.endpoints.instruments.InstrumentsCandles_() (oandapy V20.types.AccountID method), 101
method), 34init() (oandapyV20.types.AccountUnits method),
init() (oandapy V20.endpoints.instruments.InstrumentsOrderBookl 02
method), 36init() (oandapyV20.types.ClientComment method),
init() (oandapy V20.endpoints.instruments.InstrumentsPositionBob®2
method), 39init() (oandapyV20.types.ClientID method), 102
init() (oandapyV20.endpoints.orders.OrderCancelinit() (oandapyV20.types.ClientTag method), 102
method), 43init() (oandapyV20.types.DateTime method), 103
init() (oandapy V20.endpoints.orders.OrderClientExtensionnit() (oandapy V20.types.OrderID method), 103
method), 43init() (oandapyV20.types.OrderIdentifier method),
init() (oandapyV20.endpoints.orders.OrderCreate 103
method), 44init() (oandapyV20.types.OrderSpecifier method),
init() (oandapyV20.endpoints.orders.OrderDetails 104
method), 46init() (oandapyV20.types.PriceValue method), 104
init() (oandapyV20.endpoints.orders.OrderListinit() (oandapyV20.types.TradeID method), 104
method), 47init() (oandapyV20.types.Units method), 104
init() (oandapyV20.endpoints.orders.OrderReplace method), 48
init() (oandapyV20.endpoints.orders.OrdersPending A (oandapyV20.definitions.instruments.PriceComponents
method), 49 attribute), 82 init () (candapy V20 endpoints positions Open Positions + P.G.O.Y. 1777, GV. 16
init() (oandapyV20.endpoints.positions.OpenPositions ABSOLUTE_SUM (oan-
method), 50 dapyV20.definitions.accounts.PositionAggregationMode

attribute), 80			attribute), 91	
ACCOUNT_CONFIGURATION_LOCKED			_CONFIGURE_DATA_MISSING	(oan-
= -	ctionReje	ectReason	dapyV20.definitions.transactions.Transac	ctionRejectReason
attribute), 95			attribute), 95	
ACCOUNT_DEPOSIT_LOCKED			INVALID	(oan-
dapyV20.definitions.transactions.Transac	ctionReje	ectReason	dapyV20.definitions.transactions.Transac	ctionRejectReason
attribute), 95			attribute), 95	
ACCOUNT_LOCKED			ndapyV20.definitions.orders.OrderStateFi	lter at-
dapyV20.definitions.transactions.OrderC	ancelRe		tribute), 83	
attribute), 98		ALL (oa	ndapyV20.definitions.trades.TradeStateFil	lter at-
ACCOUNT_LOCKED	(oan-		tribute), 88	
dapyV20.definitions.transactions.Transac	tionReje	eareoldh		(oan-
attribute), 95			dapyV20.definitions.transactions.Transac	ctionRejectReason
ACCOUNT_NEW_POSITIONS_LOCKED	(oan-		attribute), 95	
dapyV20.definitions.transactions.OrderC	ancelRe	asomOUN	T_MISSING	(oan-
attribute), 98			dapyV20.definitions.transactions.Transac	ctionRejectReason
ACCOUNT_NOT_ACTIVE	(oan-		attribute), 95	
dapyV20.definitions.transactions.Transac	tionReje	e AR Eastm	ss in oandapyV20), 5	
attribute), 95	J		ndapyV20.definitions.orders.OrderTrigger	:Condition
ACCOUNT_ORDER_CANCEL_LOCKED	(oan-		attribute), 86	
		eARtexsloan	rtist (class in oandapyV20.endpoints.fore	xlabs),
attribute), 95	3		20	,,
ACCOUNT_ORDER_CREATION_LOCKED	(oan-	_		
dapyV20.definitions.transactions.OrderC		a B n		
attribute), 98			pyV20.definitions.instruments.PriceComp	onents
ACCOUNT_ORDER_CREATION_LOCKED	(oan-	D (ouncu	attribute), 82	onents
		eÆRÐason	dapyV20.definitions.orders.OrderTrigger	Condition
attribute), 95		- DID (our	attribute), 86	Sondition
ACCOUNT_ORDER_FILL_LOCKED	(oan-	BOUND	S_VIOLATION	(oan-
dapyV20.definitions.transactions.OrderC			dapyV20.definitions.transactions.OrderC	`
attribute), 98	uncente	ason	attribute), 98	uncenteason
ACCOUNT_TRANSFER	(oan-		attribute), 50	
dapyV20.definitions.transactions.Funding		C		
attribute), 91	Siccason		(1 : 1 NOO 1 : (6 11	\ 22
ACCOUNT_WITHDRAWAL_LOCKED	(oan-		(class in oandapyV20.endpoints.forexlabs	
dapyV20.definitions.transactions.Transac	tionRei	Cancella ectReason	bleOrderType (class in	oan-
attribute), 95	tromicj	CANCE	dapy V20.definitions.orders), 84	1 0
AccountChanges (class in	oan-	CANCE	LLED (oandapy V20.definitions.orders.Ord	ierState
dapyV20.endpoints.accounts), 11	Oan	CANGE	attribute), 85	1 C . E''
AccountConfiguration (class in	oan-	CANCE	LLED (oandapy V20. definitions. orders. Ord	ierStateFilter
dapyV20.endpoints.accounts), 13	Oan	G 11	attribute), 83	
AccountDetails (class in	oan-	Candlest	ickGranularity (class in	oan-
dapyV20.endpoints.accounts), 14	Oan-	~	dapyV20.definitions.instruments), 80	,
AccountFinancingMode (class in	oan-	CLIENT	_CONFIGURE	(oan-
dapyV20.definitions.accounts), 79	Oan-		dapyV20.definitions.transactions.Transac	ctionType
AccountID (class in oandapyV20.types), 101		~	attribute), 93	
	000	CLIENT	_CONFIGURE_DATA_MISSING	(oan-
· ·	oan-		dapyV20.definitions.transactions.Transac	ctionRejectReason
dapyV20.endpoints.accounts), 16	oumta)		attribute), 95	
AccountList (class in oandapyV20.endpoints.acco	ounts),	CLIENT	_CONFIGURE_REJECT	(oan-
19 AccountSummers (class in	0.0**		dapyV20.definitions.transactions.Transac	ctionType
AccountSummary (class in	oan-		attribute), 93	
dapyV20.endpoints.accounts), 19		CLIENT	_EXTENSIONS_DATA_MISSING	(oan-
AccountUnits (class in oandapyV20.types), 102	F	4!	dapyV20.definitions.transactions.Transac	ctionRejectReason
ADJUSTMENT (oandapy V20. definitions. transaction	ons.Func	ungkeasc	⁹¹ attribute) 95	

			CLIENT.	_TRADE_TAG_INVALID	(oan-
	dapyV20.definitions.transactions.Funding attribute), 91	Reason		dapy V20.definitions.transactions.Transac attribute), 95	tionRejectReason
CLIENT_0		(oan	ClientCo	mment (class in oandapyV20.types), 102	
				tensions (class in oandapy V20.types), 102	uacte)
	attribute), 100	ucincas	W IIICHTEX	119	uesis),
CLIENT_0		(oan-	ClientID	(class in oandapyV20.types), 102	
	OKDER dapyV20.definitions.transactions.MarketI				
	attribute), 91		_	oandapy V20.definitions.transactions.Trans	eaction Type
CLIENT_C		(oan-	CLOSE (attribute), 93	sactionType
	dapyV20.definitions.transactions.Market(EL CALO		(oan-
	attribute), 92	Jiucikca	жwose_	dapyV20.definitions.transactions.Transac	`
CLIENT_0		(oan-		attribute), 95	Homejeeticason
	dapyV20.definitions.transactions.StopLos		erwee		(oan-
	attribute), 89	SOIGEIN	Caroning	dapyV20.definitions.transactions.Transac	`
CLIENT_C		(oan-		attribute), 95	Homejeeticason
		•	പ് റട	TRADE_UNITS_EXCEED_TRADE_SIZ	'F
	attribute), 92	icircaso	MLOSE_	(oandapy V20. definitions. transactions. Tra	
CLIENT_0		(oan-		attribute), 95	nsactionicjecticason
	dapyV20.definitions.transactions.TakePro		RAGOSE		(oan-
	attribute), 94	intoraci	KULKUIL_	dapyV20.definitions.trades.TradeState	at-
CLIENT_C		(oan-		tribute), 88	ut
	dapyV20.definitions.transactions.Trailing		MidSRe		(oan-
	attribute), 100	отордов	acrocati <u>c</u>	dapyV20.definitions.trades.TradeStateFilt	
	ORDER_COMMENT_INVALID	(oan-		attribute), 88	
			areset	O (oandapy V20. definitions. trades. TradeSta	ate at-
	attribute), 95	oronire je		tribute), 88	
		(oan-	CLOSED	O (oandapy V20.definitions.trades.TradeSta	teFilter
	dapyV20.definitions.transactions.Transact				
	attribute), 95			OUT_POSITION_DOESNT_EXIST	(oan-
CLIENT_0	ORDER_ID_INVALID	(oan-		dapyV20.definitions.transactions.Transac	tionRejectReason
Ċ	dapyV20.definitions.transactions.Transact	tionReje	ctReason	attribute), 95	
	attribute), 95			OUT_POSITION_INCOMPLETE_SPECII	FICATION
	ORDER_TAG_INVALID	(oan-		(oandapyV20.definitions.transactions.Tra	nsactionRejectReason
Ċ	dapyV20.definitions.transactions.Transact	tionReje	ctReason	attribute), 95	
а	attribute), 95		CLOSEC	OUT_POSITION_PARTIAL_UNITS_MIS	SING
CLIENT_I	REQUEST	(oan-		(oandapy V20. definitions. transactions. Tra	nsactionRejectReason
	dapyV20.definitions.transactions.OrderCa			attribute), 95	
	attribute), 99		CLOSEC	OUT_POSITION_REJECT	(oan-
	REQUEST_REPLACED	(oan-		dapyV20.definitions.transactions.Transac	tionRejectReason
	dapyV20.definitions.transactions.OrderCa			attribute), 95	
	attribute), 99		CLOSEC	OUT_POSITION_UNITS_EXCEED_POS	
	TRADE_COMMENT_INVALID	(oan-	_	(oandapy V20. definitions. transactions. Tra	nsactionRejectReason
	dapyV20.definitions.transactions.Transact				
	attribute), 95		Commitn	mentsOfTraders (class in	oan-
	TRADE_ID_ALREADY_EXISTS	(oan-	CDE ATT	dapyV20.endpoints.forexlabs), 23	
	± *	anceikea	ISOMOEALE	E (oandapy V20. definitions. transactions. Tra	nsaction type
	attribute), 99			attribute), 93	
	TRADE_ID_ALREADY_EXISTS	(oan-	D		
	dapyV20.definitions.transactions.Transact				G 1 1
	attribute), 95		D (oanda	pyV20.definitions.instruments.Candlestick	Granularity
	TRADE_ID_INVALID danyV20 definitions transactions Transact	(oan- tionReie	antRana∧∽	attribute), 80	Eineneine M. d
	attribute), 95	nomicje	ANATARAN (A	oandapy V20. definitions. accounts. Account	rmancingivlode
c				attribute), 79	

DAILY_FINANCING (oan-		ns (oandapyV20.definitions.transactions.FundingReason
dapyV20.definitions.transactions.TransactionTyp		attribute), 91
attribute), 93	definition	ns (oandapy V20. definitions. transactions. Limit Order Reason
data (oandapyV20.contrib.requests.LimitOrderRequest		attribute), 100
attribute), 110	definition	ns (oandapy V20.definitions.transactions.MarketIfTouchedOrderRe
data (oandapyV20.contrib.requests.MarketOrderRequest		attribute), 91
attribute), 111	definition	ns (oandapy V20.definitions.transactions.MarketOrderMarginClose
data (oandapyV20.contrib.requests.MITOrderRequest at-		attribute), 89
tribute), 113		ns (oandapy V20.definitions.transactions.MarketOrderReason
data (o and apy V20. contrib. requests. Stop Loss Order Request		attribute), 92
attribute), 114	definition	ns (oandapy V20.definitions.transactions.OrderCancelReason
data (oandapyV20.contrib.requests.StopOrderRequest at-		attribute), 99
tribute), 116		ns (oandapy V20.definitions.transactions.OrderFillReason
data (oandapy V20.contrib.requests.TakeProfitOrderReques		attribute), 90
attribute), 117		ns (oandapy V20.definitions.transactions.StopLossOrderReason
data (o and apy V20. contrib. requests. Trailing Stop Loss Order land to the contribution of the contr		attribute), 90
attribute), 118	definition	ns (oandapy V20.definitions.transactions.StopOrderReason
DateTime (class in oandapyV20.types), 102		attribute), 92
= :	il d efinition	ns (oandapy V20.definitions.transactions.TakeProfitOrderReason
attribute), 84		attribute), 94
	o ndifiniti on	$ns \ (o and apy V20. definitions. transactions. Trailing Stop Loss Order Readers and Stop Readers and Stop Loss $
attribute), 86		attribute), 100
* * * * * * * * * * * * * * * * * * *	nd iafgMiod	ts (oandapy V20.definitions.transactions.TransactionRejectReason
attribute), 80		attribute), 98
definitions (oandapy V20.definitions.accounts.Position Aggregation)	regationt M	
attribute), 80		attribute), 94
definitions (oandapy V20. definitions. instruments. Candlestic	ck iOlEi lnAiYal	
attribute), 81		dapyV20.definitions.transactions.MarketOrderReason
definitions (oandapy V20.definitions.instruments.PriceCom	-	attribute), 92
attribute), 82		ED_TRADE_CLOSURE (oan-
definitions (oandapy V20.definitions.instruments.Weekly Al	ignment	dapyV20.definitions.transactions.TransactionType
attribute), 82		attribute), 93
definitions (o and apy V20. definitions. orders. Cancellable Order and the contract of t	le DilypES IC	
attribute), 84		dapyV20.definitions.transactions.FundingReason
definitions (oandapy V20.definitions.orders.OrderPositionF	ill	attribute), 91
attribute), 85	_	
$definitions\ (o and apy V20. definitions. orders. Order State\ at-$		
tribute), 86		INT (oandapy V20.endpoints.accounts.AccountChanges
definitions (oandapy V20.definitions.orders.OrderStateFilte	r	attribute), 11
attribute), 83	ENDPO	INT (oandapy V20.endpoints.accounts.AccountConfiguration
definitions (oandapy V20.definitions.orders.OrderTriggerCo	ondition	attribute), 13
attribute), 86	ENDPO	INT (oandapy V20.endpoints.accounts.AccountDetails
definitions (oandapyV20.definitions.orders.OrderType at-		attribute), 14
tribute), 83	ENDPO	INT (oandapy V20.endpoints.accounts.AccountInstruments
definitions (oandapy V20. definitions. orders. TimeInForce		attribute), 17
attribute), 85	ENDPO	INT (oandapy V20.endpoints.accounts.AccountList
definitions (oandapy V20. definitions. pricing. Price Status		attribute), 19
attribute), 87	ENDPO	INT (oandapy V20.endpoints.accounts.AccountSummary
$definitions \ \ (o and apy V20. definitions. trades. Trade PL \ \ at-$		attribute), 19
tribute), 87	ENDPO	INT (oandapyV20.endpoints.forexlabs.Autochartist
definitions (oandapyV20.definitions.trades.TradeState at-		attribute), 20
tribute), 88		INT (oandapyV20.endpoints.forexlabs.Calendar
definitions (oandapy V20.definitions.trades.TradeStateFilter attribute), 88	r	attribute), 22

$ENDPOINT\ (o and apy V20. endpoints. for exlabs. Commitmen {\it Endpoints}) and the commitment {\it Endpoints} and {\it Endpoints} are the commitment {\it Endpoints} are the commit$	SOPPOINT (oandapy V20. endpoints. transactions. Tr	ansactionIDRange
attribute), 23	attribute), 71	
ENDPOINT (oandapy V20.endpoints.forexlabs.Historical Pole	ENDRAINT (oandapy V20.endpoints.transactions.Tr	ansactionList
attribute), 24	attribute), 72	
ENDPOINT (oandapy V20.endpoints.forexlabs.Orderbook DE	DADPOINT (oandapy V20. endpoints. transactions. Tr	ansactionsSinceID
attribute), 31	attribute), 74	
ENDPOINT (oandapy V20.endpoints.forexlabs.Spreads I	ENDPOINT (oandapy V20.endpoints.transactions.Tr	ansactionsStream
attribute), 33	attribute), 76	
ENDPOINT (oandapy V20.endpoints.instruments.Instrument		(oan-
attribute), 34	dapyV20.endpoints.accounts.AccountCha	`
ENDPOINT (oandapy V20.endpoints.instruments.Instrument		
		(oan-
ENDPOINT (oandapy V20.endpoints.instruments.Instrument	-	(
attribute), 39	attribute), 13	nguration
ENDPOINT (oandapy V20.endpoints.orders.OrderCancel I		loon
		(oan-
attribute), 42	dapyV20.endpoints.accounts.AccountDeta	uis
ENDPOINT (oandapy V20.endpoints.orders.OrderClientExte		,
		(oan-
ENDPOINT (oandapyV20.endpoints.orders.OrderCreate	dapyV20.endpoints.accounts.AccountInstr	ruments
attribute), 44	attribute), 17	
ENDPOINT (oandapy V20.endpoints.orders.OrderDetails H		(oan-
attribute), 46	dapyV20.endpoints.accounts.AccountList	
ENDPOINT (oandapy V20.endpoints.orders.OrderList at-	attribute), 19	
tribute), 46	EXPECTED_STATUS	(oan-
ENDPOINT (oandapyV20.endpoints.orders.OrderReplace	dapyV20.endpoints.accounts.AccountSum	ımary
attribute), 47	attribute), 19	•
ENDPOINT (oandapy V20.endpoints.orders.OrdersPending I	EXPECTED_STATUS	(oan-
attribute), 49	dapyV20.endpoints.forexlabs.Autochartist	· `
ENDPOINT (oandapy V20.endpoints.positions.OpenPosition		
		(oan-
ENDPOINT (oandapy V20.endpoints.positions.PositionClose		at-
attribute), 51	tribute), 22	ut
ENDPOINT (oandapy V20.endpoints.positions.PositionDetal		(oan-
attribute), 53	dapyV20.endpoints.forexlabs.Commitmen	`
ENDPOINT (oandapy V20.endpoints.positions.PositionList	attribute), 23	asorraders
		loon
		(oan-
ENDPOINT (oandapyV20.endpoints.pricing.PricingInfo	dapyV20.endpoints.forexlabs.HistoricalPo	sitionkatios
attribute), 56	attribute), 24	,
ENDPOINT (oandapy V20.endpoints.pricing.PricingStream F		(oan-
attribute), 59	dapyV20.endpoints.forexlabs.OrderbookD	⁄ata
ENDPOINT (oandapyV20.endpoints.trades.OpenTrades	attribute), 31	
	_	(oan-
ENDPOINT (oandapy V20.endpoints.trades.TradeClientExternation)		at-
attribute), 65	tribute), 33	
ENDPOINT (oandapyV20.endpoints.trades.TradeClose I	EXPECTED_STATUS	(oan-
attribute), 66	dapyV20.endpoints.instruments.Instrumen	ıtsCandles
ENDPOINT (oandapy V20.endpoints.trades.TradeCRCDO	attribute), 34	
attribute), 63	EXPECTED_STATUS	(oan-
ENDPOINT (oandapyV20.endpoints.trades.TradeDetails	dapyV20.endpoints.instruments.Instrumen	ıtsOrderBook
attribute), 68	attribute), 36	
ENDPOINT (oandapyV20.endpoints.trades.TradesList H		(oan-
attribute), 68	dapyV20.endpoints.instruments.Instrumen	\
ENDPOINT (oandapy V20.endpoints.transactions.Transaction		
· · · · · · · · · · · · · · · · · · ·		(oan-

dapyV20.endpoints.orders.OrderCancel	at-	dapyV20.endpoints.trades.TradesList at-
tribute), 42		tribute), 68
EXPECTED_STATUS	(oan-	EXPECTED_STATUS (oan-
dapyV20.endpoints.orders.OrderClientEx	tension	dapyV20.endpoints.transactions.TransactionDetails
attribute), 43		attribute), 70
EXPECTED_STATUS	(oan-	EXPECTED_STATUS (oan-
dapyV20.endpoints.orders.OrderCreate	at-	dapyV20.endpoints.transactions.TransactionIDRange
tribute), 44		attribute), 71
EXPECTED_STATUS	(oan-	EXPECTED_STATUS (oan-
dapyV20.endpoints.orders.OrderDetails	at-	dapy V20. endpoints. transactions. Transaction List
tribute), 46		attribute), 72
EXPECTED_STATUS	(oan-	EXPECTED_STATUS (oan-
dapyV20.endpoints.orders.OrderList attr		dapy V20. endpoints. transactions. Transactions Since ID
46	ioute),	attribute), 74
EXPECTED_STATUS	(oan-	EXPECTED_STATUS (oan-
	(Gail-	_ ,
dapyV20.endpoints.orders.OrderReplace		dapyV20.endpoints.transactions.TransactionsStream
attribute), 47	(attribute), 76
EXPECTED_STATUS	(oan-	F
dapyV20.endpoints.orders.OrdersPending	g	
attribute), 49	,	FIFO_VIOLATION (oan-
EXPECTED_STATUS	(oan-	dapyV20.definitions.transactions.OrderCancelReason
dapyV20.endpoints.positions.OpenPosition	ons	attribute), 99
attribute), 50		FILLED (oandapy V20.definitions.orders.OrderState at-
EXPECTED_STATUS	(oan-	tribute), 85
dapyV20.endpoints.positions.PositionClo	ose	FILLED (oandapyV20.definitions.orders.OrderStateFilter
attribute), 52		attribute), 83
EXPECTED_STATUS	(oan-	FOK (oandapy V20. definitions. orders. TimeInForce
dapyV20.endpoints.positions.PositionDet	tails	attribute), 85
attribute), 53		Friday (oandapy V20. definitions. instruments. Weekly Alignment
EXPECTED_STATUS	(oan-	attribute), 81
dapyV20.endpoints.positions.PositionLis	t	FUNDING_REASON_MISSING (oan-
attribute), 54		dapyV20.definitions.transactions.TransactionRejectReason
EXPECTED_STATUS	(oan-	attribute), 95
dapyV20.endpoints.pricing.PricingInfo	at-	FundingReason (class in oan-
tribute), 56		dapyV20.definitions.transactions), 90
EXPECTED_STATUS	(oan-	dupy v 20.demintions.transactions), 70
dapyV20.endpoints.pricing.PricingStream	`	G
attribute), 59	_	
EXPECTED STATUS	(oan-	GFD (oandapy V20.definitions.orders.TimeInForce
dapyV20.endpoints.trades.OpenTrades	at-	attribute), 85
tribute), 62	uı	granularity_to_time() (in module oan-
EXPECTED_STATUS	(oan-	dapyV20.contrib.generic), 108
	•	GTC (oandapyV20.definitions.orders.TimeInForce
dapyV20.endpoints.trades.TradeClientEx attribute), 65	tensions	attribute), 85
**	(000	GTD (oandapyV20.definitions.orders.TimeInForce at-
EXPECTED_STATUS	(oan-	tribute), 85
dapyV20.endpoints.trades.TradeClose	at-	11
tribute), 66	,	H
EXPECTED_STATUS	(oan-	H1 (oandapyV20.definitions.instruments.CandlestickGranularity
dapyV20.endpoints.trades.TradeCRCDO		attribute), 80
attribute), 63		H12 (oandapy V20.definitions.instruments.CandlestickGranularity
EXPECTED_STATUS	(oan-	attribute), 81
dapyV20.endpoints.trades.TradeDetails	at-	H2 (oandapy V20.definitions.instruments.CandlestickGranularity
tribute), 68		attribute), 81
EXPECTED_STATUS	(oan-	autiouce, or

$H3\ (o and apy V20. definitions. instruments. Can dlestick Grant and the state of the state of$	ıl ing UFFICIENT_MARGIN	(oan-
attribute), 81	dapyV20.definitions.transactions.OrderCa	ancelReason
$H4\ (o and apy V20. definitions. instruments. Can dlestick Grandlestick Grandlest$	ularity attribute), 99	
attribute), 81	INTERNAL_SERVER_ERROR	(oan-
H6 (oandapy V20.definitions.instruments.CandlestickGranu		ancelReason
attribute), 81	attribute), 99	
$H8\ (o and apy V20. definitions. instruments. Can dlestick Grandlestick Grandlest$		(oan-
attribute), 81	dapyV20.definitions.transactions.Transac	tionRejectReason
HEADERS (oandapy V20.endpoints.accounts.AccountCon		
attribute), 13	invalid (oandapyV20.definitions.pricing.PriceStatu	ıs at-
$HEADERS\ (o and apy V20. endpoints. orders. Order Client Extra properties of the p$		
attribute), 43	INVALID_REISSUE_IMMEDIATE_PARTIAL_FI	
HEADERS (oandapyV20.endpoints.orders.OrderCreate	(oandapy V20.definitions.transactions.Tra	nsactionRejectReason
attribute), 44	attribute), 96	
HEADERS (oandapyV20.endpoints.orders.OrderReplace		iggerCondition
attribute), 47	attribute), 86	
HEADERS (oandapy V20.endpoints.positions.PositionClos		e at-
attribute), 52	tribute), 85	
HEADERS (oandapy V20.endpoints.trades.TradeClientExt	ensions	
attribute), 65	L	
HEADERS (oandapyV20.endpoints.trades.TradeClose	LIMIT (oandapy V20. definitions. orders. Cancellable	OrderType
attribute), 66	attribute), 84	
HEADERS (oandapy V20.endpoints.trades.TradeCRCDO	LIMIT (oandapyV20.definitions.orders.Orde	erType
attribute), 63	attribute), 83	
HistoricalPositionRatios (class in oan-	LIMIT_ORDER (oandapyV20.definitions.transaction)	ons.OrderFillReason
dapyV20.endpoints.forexlabs), 24	attribute), 90	
I	LIMIT_ORDER (oandapyV20.definitions.transaction)	ons.TransactionType
1	attribute), 93	
INSTRUMENT_MISSING (oan-	LIMIT_ORDER_REJECT	(oan-
	ectReason dapyV20.definitions.transactions.Transac	tionType
attribute), 95	attribute), 93	
`	LimitOrderReason (class in	oan-
	ectReason dapyV20.definitions.transactions), 100	
attribute), 95	LimitOrderRequest (class in	oan-
INSTRUMENT_PRICE_UNKNOWN (oan-	dapyV20.contrib.requests), 109	,
dapyV20.definitions.transactions.TransactionRej		(oan-
attribute), 96	dapyV20.definitions.transactions.OrderC	ancelReason
INSTRUMENT_UNKNOWN (oan-	attribute), 99	,
dapyV20.definitions.transactions.TransactionRej		(oan-
attribute), 96	dapyV20.definitions.transactions.OrderCa	ancelReason
InstrumentsCandles (class in oan-	attribute), 99	
dapyV20.endpoints.instruments), 34	M	
InstrumentsCandlesFactory() (in module oan-		
dapyV20.contrib.factories), 107	M (oandapy V20.definitions.instruments.Candlestic	kGranularity
InstrumentsOrderBook (class in oan-	attribute), 81	
dapyV20.endpoints.instruments), 36	M (oandapy V20.definitions.instruments.PriceComp	onents
InstrumentsPositionBook (class in oan-	attribute), 82	
dapyV20.endpoints.instruments), 39	M1 (oandapy V20.definitions.instruments.Candlestic	ckGranularity
INSUFFICIENT_FUNDS (oan-	attribute), 81	
dapy V20. definitions. transactions. Transaction Rej	eMResonhdapyV20.definitions.instruments.Candless	tickGranularity
attribute), 96	attribute), 81	
INSUFFICIENT_LIQUIDITY (oan-	M15 (oandapyV20.definitions.instruments.Candlest	tickGranularity
dapyV20.definitions.transactions.OrderCancelRe	eason attribute), 81	
attribute), 99		

```
M2 (oandapy V20. definitions. instruments. Candlestick Granularity
                                                                                                   attribute), 90
              attribute), 81
                                                                                    MARKET ORDER
                                                                                                                                                              (oan-
M30 (oandapy V20. definitions. instruments. Candlestick Granularity
                                                                                                  dapyV20.definitions.transactions.TransactionType
              attribute), 81
                                                                                                   attribute), 93
M4 (oandapy V20.definitions.instruments.CandlestickGranulMARKET ORDER DELAYED TRADE CLOSE
              attribute), 81
                                                                                                  (oandapyV20.definitions.transactions.OrderFillReason
M5 (oandapy V20. definitions. instruments. Candlestick Granularity
                                                                                                   attribute), 90
              attribute), 81
                                                                                    MARKET ORDER MARGIN CLOSEOUT
MARGIN CALL ENTER
                                                                         (oan-
                                                                                                   dapyV20.definitions.transactions.OrderFillReason
              dapyV20.definitions.transactions.TransactionType
                                                                                                   attribute), 90
              attribute), 93
                                                                                    MARKET_ORDER_POSITION_CLOSEOUT
                                                                                                                                                              (oan-
MARGIN_CALL_EXIT
                                                                                                   dapyV20.definitions.transactions.OrderFillReason
                                                                         (oan-
              dapyV20.definitions.transactions.TransactionType
                                                                                                   attribute), 90
                                                                                    MARKET_ORDER_REJECT
              attribute), 93
                                                                                                                                                              (oan-
MARGIN_CALL_EXTEND
                                                                         (oan-
                                                                                                  dapyV20.definitions.transactions.TransactionType
              dapyV20.definitions.transactions.TransactionType
                                                                                                   attribute), 93
              attribute), 93
                                                                                    MARKET_ORDER_TRADE_CLOSE
MARGIN CHECK VIOLATION
                                                                                                   dapyV20.definitions.transactions.OrderFillReason
                                                                         (oan-
              dapy V20.definitions.transactions.MarketOrderMarginClose (authReater)n 90
                                                                                    Market If Touched Order Reason\\
              attribute), 89
                                                                                                                                                     in
                                                                                                                                       (class
                                                                                                                                                               oan-
MARGIN CLOSEOUT
                                                                         (oan-
                                                                                                   dapyV20.definitions.transactions), 91
              dapyV20.definitions.transactions.MarketOrderReaMarketOrderMarginCloseoutReason (class in
                                                                                                                                                               oan-
              attribute), 92
                                                                                                   dapyV20.definitions.transactions), 89
MARGIN_RATE_INVALID
                                                                         (oan- MarketOrderReason
                                                                                                                             (class
                                                                                                                                                               oan-
              dapyV20.definitions.transactions.TransactionRejectReason dapyV20.definitions.transactions), 91
              attribute), 96
                                                                                    MarketOrderRequest
                                                                                                                             (class
                                                                                                                                                               oan-
MARGIN_RATE_WOULD_TRIGGER_CLOSEOUT
                                                                                                   dapyV20.contrib.requests), 110
              (oandapy V20.definitions.transactions.Transaction Rear Relation Rear Relation Rear Relation Rear Relation Relation Rear Relation Rear Relation Rela
                                                                                                                                                              (oan-
              attribute), 96
                                                                                                  dapyV20.definitions.transactions.TransactionRejectReason
MARGIN RATE WOULD TRIGGER MARGIN CALL
                                                                                                   attribute), 96
              (oandapy V20.definitions.transactions.Transaction Reaction Reaction SIDE
              attribute), 96
                                                                                                   dapyV20.definitions.accounts.PositionAggregationMode
MARKET (oandapy V20.definitions.orders.OrderType at-
                                                                                                   attribute), 80
              tribute), 83
                                                                                    METHOD (oandapy V20.endpoints.accounts.AccountChanges
MARKET HALTED
                                                                                                   attribute), 11
                                                                         (oan-
             dapyV20.definitions.transactions.OrderCancelReaM6THOD (oandapyV20.endpoints.accounts.AccountConfiguration
              attribute), 99
                                                                                                   attribute), 13
MARKET IF TOUCHED
                                                                         (oan-
                                                                                    METHOD (oandapy V20.endpoints.accounts.AccountDetails
              dapyV20.definitions.orders.CancellableOrderType
                                                                                                   attribute), 14
              attribute), 84
                                                                                    METHOD (oandapy V20.endpoints.accounts.AccountInstruments
MARKET_IF_TOUCHED
                                                                                                   attribute), 17
                                                                         (oan-
              dapyV20.definitions.orders.OrderType
                                                                                    METHOD (oandapy V20.endpoints.accounts.AccountList
                                                                             at-
              tribute), 83
                                                                                                   attribute), 19
MARKET_IF_TOUCHED_ORDER
                                                                         (oan-
                                                                                    METHOD (oandapy V20.endpoints.accounts.AccountSummary
              dapyV20.definitions.transactions.OrderFillReason
                                                                                                   attribute), 19
              attribute), 90
                                                                                    METHOD (oandapy V20.endpoints.forexlabs.Autochartist
MARKET_IF_TOUCHED_ORDER
                                                                                                   attribute), 21
                                                                         (oan-
              dapyV20.definitions.transactions.TransactionTypeMETHOD
                                                                                                        (oandapyV20.endpoints.forexlabs.Calendar
              attribute), 93
                                                                                                  attribute), 22
MARKET_IF_TOUCHED_ORDER_REJECT
                                                                         (oan- METHOD (oandapy V20.endpoints.forexlabs.CommitmentsOfTraders
              dapyV20.definitions.transactions.TransactionType
                                                                                                   attribute), 23
                                                                                    METHOD\ (o and apy V20. endpoints. for exlabs. Historical Position Ratios
              attribute), 93
MARKET ORDER
                                                                         (oan-
                                                                                                  attribute), 25
              dapyV20.definitions.transactions.OrderFillReasonMETHOD (oandapyV20.endpoints.forexlabs.OrderbookData
```

attribute), 32	attribute), 74
METHOD (oandapyV20.endpoints.forexlabs.Spreads at-	METHOD (oandapy V20.endpoints.transactions.TransactionsStream
tribute), 33	attribute), 76
METHOD (oandapy V20.endpoints.instruments.Instrument	nts Child (exandapy V20. definitions. orders. Order Trigger Condition
attribute), 34	attribute), 86
METHOD (oandapy V20.endpoints.instruments.Instrument	nts OHG READION (oandapy V20. definitions. transactions. Order Cancel Reason
attribute), 36	attribute), 99
METHOD (oandapy V20.endpoints.instruments.Instrument	· · · · · · · · · · · · · · · · · · ·
attribute), 39	dapyV20.contrib.requests), 112
	Monday (oandapy V20.definitions.instruments.WeeklyAlignment
attribute), 42	attribute), 81
METHOD (oandapy V20.endpoints.orders.OrderClientExte	
attribute), 43	N
METHOD (oandapyV20.endpoints.orders.OrderCreate	NEGATIVE (oandapyV20.definitions.trades.TradePL at-
attribute), 44	tribute), 87
METHOD (oandapy V20.endpoints.orders.OrderDetails	NET_SUM (oandapy V20.definitions.accounts.PositionAggregationMode
attribute), 46	attribute), 80
METHOD (oandapy V20.endpoints.orders.OrderList at-	NO_FINANCING (oan-
tribute), 47	dapyV20.definitions.accounts.AccountFinancingMode
METHOD (oandapyV20.endpoints.orders.OrderReplace	,,,
attribute), 48	non_tradeable (oandapyV20.definitions.pricing.PriceStatus
METHOD (oandapy V20.endpoints.orders.OrdersPending	attribute), 87
attribute), 49	
METHOD (oandapy V20.endpoints.positions.OpenPosition	ns O
attribute), 50	oandapyV20.contrib.factories (module), 107
METHOD (oandapy V20.endpoints.positions.PositionClose	valuaby v 20. control generic (module), 100
attribute), 52	oandapy V20.definitions.accounts (module), 79
METHOD (oandapy V20.endpoints.positions.PositionDetai	all Sandapy V20. definitions. instruments (module), 80
attribute), 33	oandapyV20.definitions.orders (module), 82
METHOD (oandapy V20.endpoints.positions.PositionList attribute), 54	valuaby v 20.deminions.pricing (module), 80
METHOD (oandapy V20.endpoints.pricing.PricingInfo	oandapyV20.definitions.trades (module), 87
attribute), 56	oandapyV20.definitions.transactions (module), 89
METHOD (oandapy V20.endpoints.pricing.PricingStream	ON_FILL (oandapyV20.definitions.transactions.StopLossOrderReason
attribute), 59	attribute), 69
METHOD (oandapy V20.endpoints.trades.OpenTrades	ON_FILL (oandapy V20.definitions.transactions.TakeProfitOrderReason
attribute), 62	attribute), 94
METHOD (oandapy V20.endpoints.trades.TradeClientExter	ON_FILL (oandapyV20.definitions.transactions.TrailingStopLossOrderRea
attribute), 65	attribute), 100
METHOD (oandapy V20.endpoints.trades.TradeClose at-	OPEN (oandapyV20.definitions.trades.TradeState at-
tribute), 66	tribute), 88
METHOD (oandapy V20.endpoints.trades.TradeCRCDO	OPEN (oandapyV20.definitions.trades.TradeStateFilter
attribute), 63	attribute), 88
METHOD (oandapyV20.endpoints.trades.TradeDetails	OPEN_ONLY (oandapy V20.definitions.orders.OrderPositionFill
attribute), 68	attribute), 64
METHOD (oandapyV20.endpoints.trades.TradesList at-	OPEN_TRADES_ALLOWED_EXCEEDED (oan-dapyV20.definitions.transactions.OrderCancelReason
tribute), 68	ottmibute) 00
METHOD (oandapy V20.endpoints.transactions.Transaction	on Details on oan-
attribute). /()	1 1/20 1 50
METHOD (oandapy V20.endpoints.transactions.Transaction	on DRange on DRange (class in gandany V20 endpoints trades) 62
attribute), 71	ORDER_CANCEL (oan-
$METHOD\ (o and apy V20. endpoints. transactions. Transactions$	onList dapyV20.definitions.transactions.TransactionType
attribute), 72	attribute), 93
METHOD (oandapy V20.endpoints.transactions.Transaction	onsSinceID and the state of the

	CANCEL_REJECT	(oan-	
d	lapyV20.definitions.transactions.Transa	ctionTyp	peOrderState (class in oandapyV20.definitions.orders), 85
a	ttribute), 93		OrderStateFilter (class in oan-
ORDER_C	CLIENT_EXTENSIONS_MODIFY	(oan-	dapyV20.definitions.orders), 82
d	lapyV20.definitions.transactions.Transa	ctionTyp	peOrderTriggerCondition (class in oan-
a	ttribute), 93		dapyV20.definitions.orders), 86
ORDER_C	CLIENT_EXTENSIONS_MODIFY_RE	EJECT	OrderType (class in oandapyV20.definitions.orders), 83
	oandapyV20.definitions.transactions.Tr		on <u>Ty</u> pe
	ttribute), 93		P [*]
ORDER_D	OOESNT_EXIST	(oan-	PENDING (oandapyV20.definitions.orders.OrderState
d	lapyV20.definitions.transactions.Transa	ctionRej	ejectReason attribute) 86
	ttribute), 96	·	PENDING (oandapy V20.definitions.orders.OrderStateFilter
ORDER_F	TLL (oandapyV20.definitions.transaction	ons.Trans	nsactionType _{attribute}) 83
	ttribute), 93		PENDING_ORDERS_ALLOWED_EXCEEDED (oan-
	TILL_POSITION_ACTION_INVALID	(oan-	
- d	lapyV20.definitions.transactions.Transa	ctionRei	ejectReason attribute) 99
	ttribute), 96	3	PENDING_ORDERS_ALLOWED_EXCEEDED (oan-
	TILL_POSITION_ACTION_MISSING	(oan-	dapyV20.definitions.transactions.TransactionRejectReason
_ d	lapyV20.definitions.transactions.Transa	ctionRei	eiectReason attribute) 96
	ttribute), 96		POSITION_AGGREGATION_MODE_INVALID (oan-
	D_UNSPECIFIED	(oan-	
	lapyV20.definitions.transactions.Transa		
	ttribute), 96		POSITION_CLOSEOUT (oan-
	DENTIFIER_INCONSISTENCY	(oan-	_ ,
	lapyV20.definitions.transactions.Transa		
	ttribute), 96	etromicej	
	ARTIAL_FILL_OPTION_INVALID	(oan-	
	lapyV20.definitions.transactions.Transa		
	ttribute), 96	etromicej	
	ARTIAL_FILL_OPTION_MISSING	(oan-	POSITION_SIZE_EXCEEDED (oan-
	lapyV20.definitions.transactions.Transa		
	ttribute), 96	ctionic	
Orderbook		oan-	PositionAggregationMode (class in oan-
	lapyV20.endpoints.forexlabs), 31	Oan-	dupy v 20.demintions.decounts), 00
	el (class in oandapyV20.endpoints.orde	rc) 12	PositionClose (class in oandapyV20.endpoints.positions),
OrderCance			51
	lapyV20.definitions.transactions), 98	oan-	1 obtronorosertequest (class in our
	tExtensions (class in	oon	dapyV20.contrib.requests), 113
	lapyV20.endpoints.orders), 43	Oan-	PositionDetails (class in oan-
	te (class in oandapy V20.endpoints.orde	re) 11	dapyV20.endpoints.positions), 53
	ils (class in oandapy v20.endpoints.orde		PositionList (class in oandapyV20.endpoints.positions),
			54
OrderFillR	•	oan-	TOSTITYE (building) v 20.deminions.trades.Trader E at
	lapyV20.definitions.transactions), 90		tribute), 87
	lass in oandapyV20.types), 103		PRICE_BOUND_INVALID (oan-
	ifier (class in oandapyV20.types), 103	16	dapyV20.definitions.transactions.TransactionRejectReason
	class in oandapyV20.endpoints.orders)		attribute), 96
OrderPositi	•	oan-	TRICE_BOOT(B_T RECISIOT(_E/TCEEDED (out)
	lapyV20.definitions.orders), 84	\ 17	dapyV20.definitions.transactions.TransactionRejectReason
	ace (class in oandapy V20.endpoints.ord		
OKDEKS_	ON_FILL_DUPLICATE_CLIENT_OF	NDEK_II	IDPRICE_DISTANCE_INVALID (oan-
((oandapy v 20.dennitions.transactions.Tr	ansaction	$on Reject Reason \\ V20. definitions. transactions. Transaction Reject Reason$
	ttribute), 96		attribute), 96
_	ifier (class in oandapyV20.types), 104		PRICE_DISTANCE_MAXIMUM_EXCEEDED (oan-
OrdersPend	ding (class in oandapyV20.endpoints.o	oraers),	dapyV20.definitions.transactions.TransactionRejectReason

attribute), 96		REPLAC	CEMENT	(oan-
PRICE_DISTANCE_MINIMUM_NOT_MET	(oan-		dapyV20.definitions.transactions.Trailing	`
dapyV20.definitions.transactions.Transac	tionRej	ectReason		•
attribute), 96			CING_ORDER_INVALID	(oan-
PRICE_DISTANCE_MISSING	(oan-		dapyV20.definitions.transactions.Transac	tionRejectReason
dapyV20.definitions.transactions.Transac	tionRej			
attribute), 96		REPLAC	CING_TRADE_ID_INVALID	(oan-
PRICE_DISTANCE_PRECISION_EXCEEDED	(oan-	_	dapyV20.definitions.transactions.Transac	tionRejectReason
dapyV20.definitions.transactions.Transac	tionRej			
attribute), 96	(-	(oandapyV20.API method), 8	
PRICE_INVALID			params (oandapyV20.API attribute), 8	(aan
dapyV20.definitions.transactions.Transac attribute), 96	шопкеј	etarsonso <u>n</u>	dapyV20.definitions.transactions.Transac	(oan-
PRICE_MISSING	(oan-		attribute), 93	uontype
dapyV20.definitions.transactions.Transac	`	ectReason		
attribute), 96	riomicj	Š		
PRICE_PRECISION_EXCEEDED	(oan-	S10 (oan	dapyV20.definitions.instruments.Candlesti	ckGranularity
dapyV20.definitions.transactions.Transac		ectReason	attribute) 81	CKOTallularity
attribute), 96	3		dapyV20.definitions.instruments.Candlesti	ckGranularity
PriceComponents (class in	oan-	510 (0411)	attribute), 81	
dapyV20.definitions.instruments), 82		S30 (oand	dapyV20.definitions.instruments.Candlesti	ckGranularity
PriceStatus (class in oandapyV20.definitions.pricin	g), 86	`	attribute), 81	·
PriceValue (class in oandapyV20.types), 104		S5 (oanda	apyV20.definitions.instruments.Candlestic	kGranularity
PricingInfo (class in oandapyV20.endpoints.pricing			attribute), 81	-
PricingStream (class in oandapyV20.endpoints.pr	icing),	Saturday	(o and apy V20. definitions. instruments. Wee	klyAlignment
59			attribute), 82	
R		SECONI	D_BY_SECOND	(oan-
	,		dapyV20.definitions.accounts.AccountFin	nancingMode
REDUCE_FIRST	(oan-		attribute), 79	
dapyV20.definitions.orders.OrderPosition	nFill		e() (in module oandapyV20.contrib.generic	
attribute), 84 REDUCE_ONLY	(oon	SITE_M	IGRATION	(oan-
dapyV20.definitions.orders.OrderPosition	(oan-		dapyV20.definitions.transactions.Funding	gReason
attribute), 85	11 111	Sprands (attribute), 91 (class in oandapy V20.endpoints.forexlabs)	22
REGULATORY_MARGIN_CALL_VIOLATION	(oan-		andapy V20.definitions.orders.Cancellable(
dapyV20.definitions.transactions.Market				order type
attribute), 89			(oandapyV20.definitions.orders.OrderType	e at-
REOPEN (oandapyV20.definitions.transactions.Tra	nsactio		tribute), 83	
attribute), 93			OSS (oandapyV20.definitions.orders.Canc	ellableOrderType
REPLACEMENT	(oan-		attribute), 84	
dapyV20.definitions.transactions.LimitO	rderRea	SSTOP_L	OSS (oandapyV20.definitions.orders.Orde	erType
attribute), 100			attribute), 83	
REPLACEMENT	(oan-	STOP_L	OSS_ON_FILL_CLIENT_ORDER_COM	MENT_INVALID
	IfTouch	edOrderRe	480m dapyV20.definitions.transactions.Tra	nsactionRejectReason
attribute), 91	,		attribute), 96	
REPLACEMENT	(oan-		OSS_ON_FILL_CLIENT_ORDER_ID_A	
dapyV20.definitions.transactions.StopLo	ssOrder.	Reason	(oandapyV20.definitions.transactions.Ord	lerCancelReason
attribute), 90	(oon	CEOD I	attribute), 99	TILL ID
REPLACEMENT dapy V20. definitions. transactions. StopOre	(oan- derReas		OSS_ON_FILL_CLIENT_ORDER_ID_IN	
attribute), 92	ucintas	OII	(oandapy V20.definitions.transactions.Tra attribute), 96	nsacuonkejectkeason
REPLACEMENT	(oan-	STOP I	attribute), 96 OSS_ON_FILL_CLIENT_ORDER_TAG_	INVALID
dapyV20.definitions.transactions.TakePro	•		oandapyV20.definitions.transactions.Tra	
attribute), 94			attribute), 96	

```
STOP LOSS ON FILL GTD TIMESTAMP IN PAST StopLossOrderReason
                                                                                                                                                                                                                    (class
                                                                                                                                                                                                                                                                          oan-
                       (oandapyV20.definitions.transactions.OrderCancelReason dapyV20.definitions.transactions), 89
                       attribute), 99
                                                                                                                                             StopLossOrderRequest
                                                                                                                                                                                                                     (class
                                                                                                                                                                                                                                                                          oan-
STOP_LOSS_ON_FILL_GTD_TIMESTAMP_IN_PAST
                                                                                                                                                                     dapyV20.contrib.requests), 114
                       (oandapy V20. definitions. transactions. Transaction Reject Released as on
                                                                                                                                                                                                            (class
                                                                                                                                                                                                                                               in
                                                                                                                                                                                                                                                                          oan-
                       attribute), 96
                                                                                                                                                                     dapyV20.definitions.transactions), 92
STOP LOSS ON FILL GTD TIMESTAMP MISSING StopOrderRequest
                                                                                                                                                                                                                                                in
                                                                                                                                                                                                                                                                          oan-
                       (oandapy V20. definitions. transactions. Transaction Reject Reastapy V20. contrib. requests), 115
                                                                                                                                             STREAM (oandapy V20.endpoints.pricing.PricingStream
                       attribute), 96
STOP_LOSS_ON_FILL_LOSS
                                                                                                                          (oan-
                                                                                                                                                                     attribute), 59
                       dapyV20.definitions.transactions.OrderCancelReaStiREAM (oandapyV20.endpoints.transactions.TransactionsStream
                       attribute), 99
                                                                                                                                                                     attribute), 76
STOP_LOSS_ON_FILL_PRICE_INVALID
                                                                                                                          (oan-Sunday (oandapy V20. definitions. instruments. Weekly Alignment
                       dapyV20.definitions.transactions.TransactionRejectReason attribute), 82
                       attribute), 97
                                                                                                                                             Т
STOP_LOSS_ON_FILL_PRICE_MISSING
                                                                                                                          (oan-
                       dapy V20. definitions. transactions. Transaction Rejeqt {\tt Reso} {\tt op} ROFIT (oandapy V20. definitions. orders. Cancellable Order Type) {\tt op} {\tt 
                       attribute), 97
                                                                                                                                                                     attribute), 84
STOP_LOSS_ON_FILL_PRICE_PRECISION_EXCEEDEDAKE PROFIT (oandapy V20.definitions.orders.OrderType
                       (oandapy V20. definitions. transactions. Transaction Reject Reasonibute), 83
                       attribute), 97
                                                                                                                                             TAKE_PROFIT_ON_FILL_CLIENT_ORDER_COMMENT_INVALID
STOP LOSS ON FILL TIME IN FORCE INVALID
                                                                                                                                                                     (oandapyV20.definitions.transactions.TransactionRejectReason
                       (oandapy V20.definitions.transactions.TransactionRejectReasonibute), 97
                       attribute), 97
                                                                                                                                             TAKE_PROFIT_ON_FILL_CLIENT_ORDER_ID_ALREADY_EXISTS
STOP LOSS ON FILL TIME IN FORCE MISSING
                                                                                                                                                                     (oandapy V20.definitions.transactions.OrderCancelReason
                       (oandapy V20. definitions. transactions. Transaction Reject Reason ibute), 99
                       attribute), 97
                                                                                                                                             TAKE_PROFIT_ON_FILL_CLIENT_ORDER_ID_INVALID
STOP_LOSS_ON_FILL_TRIGGER_CONDITION_INVALID
                                                                                                                                                                     (oandapy V20.definitions.transactions.TransactionRejectReason
                       (oandapy V20. definitions. transactions. Transaction Reject Reason bute), 97
                       attribute), 97
                                                                                                                                             TAKE PROFIT ON FILL CLIENT ORDER TAG INVALID
STOP_LOSS_ON_FILL_TRIGGER_CONDITION_MISSING
                                                                                                                                                                     (oandapy V20. definitions. transactions. Transaction Reject Reason
                       (oandapyV20.definitions.transactions.TransactionRejectReasonibute), 97
                       attribute), 97
                                                                                                                                             TAKE PROFIT ON FILL GTD TIMESTAMP IN PAST
STOP_LOSS_ORDER
                                                                                                                          (oan-
                                                                                                                                                                     (oandapy V20.definitions.transactions.OrderCancelReason
                       dapyV20.definitions.transactions.OrderFillReason
                                                                                                                                                                     attribute), 99
                       attribute), 90
                                                                                                                                             TAKE PROFIT ON FILL GTD TIMESTAMP IN PAST
STOP LOSS ORDER
                                                                                                                           (oan-
                                                                                                                                                                     (oandapyV20.definitions.transactions.TransactionRejectReason
                       dapyV20.definitions.transactions.TransactionType
                                                                                                                                                                     attribute), 97
                       attribute), 93
                                                                                                                                             TAKE_PROFIT_ON_FILL_GTD_TIMESTAMP_MISSING
STOP_LOSS_ORDER_ALREADY_EXISTS
                                                                                                                          (oan-
                                                                                                                                                                     (oandapy V20.definitions.transactions.TransactionRejectReason
                       dapvV20.definitions.transactions.TransactionRejectReason attribute), 97
                       attribute), 97
                                                                                                                                             TAKE_PROFIT_ON_FILL_LOSS
                                                                                                                                                                                                                                                                        (oan-
STOP LOSS ORDER REJECT
                                                                                                                          (oan-
                                                                                                                                                                     dapyV20.definitions.transactions.OrderCancelReason
                       dapyV20.definitions.transactions.TransactionType
                                                                                                                                                                     attribute), 99
                       attribute), 93
                                                                                                                                             TAKE PROFIT ON FILL PRICE INVALID
STOP_ORDER (oandapy V20.definitions.transactions.OrderFillReason_apy V20.definitions.transactions.TransactionRejectReason_apy V20.definitions.transactions.OrderFillReason_apy V20.definitions.transactions.OrderFillReason_apy V20.definitions.transactions.OrderFillReason_app V20.definitions.OrderFillReason_app V20.defini
                       attribute), 90
                                                                                                                                                                     attribute), 97
STOP_ORDER (oandapy V20.definitions.transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transactions.Transact
                                                                                                                                                                                                                                                                        (oan-
                       attribute), 93
                                                                                                                                                                     dapyV20.definitions.transactions.TransactionRejectReason
STOP_ORDER_REJECT
                                                                                                                          (oan-
                                                                                                                                                                     attribute), 97
                       dapy V20. definitions. Transaction Type {\tt TAKE\_PROFIT\_ON\_FILL\_PRICE\_PRECISION\_EXCEEDED}
                       attribute), 93
                                                                                                                                                                     (oandapyV20.definitions.transactions.TransactionRejectReason
StopLossDetails (class in oandapyV20.contrib.requests),
                                                                                                                                                                     attribute), 97
                       120
                                                                                                                                             TAKE_PROFIT_ON_FILL_TIME_IN_FORCE_INVALID
```

```
(oandapy V20.definitions.transactions.Transaction Release IENT EXTENSIONS MODIFY REJECT
                                                               (oandapyV20.definitions.transactions.TransactionType
         attribute), 97
TAKE PROFIT ON FILL TIME IN FORCE MISSING
                                                               attribute), 94
         (oandapy V20. definitions. transactions. Transaction Report Reaction SE
                                                                                                     (oan-
         attribute), 97
                                                               dapyV20.definitions.transactions.MarketOrderReason
TAKE PROFIT ON FILL TRIGGER CONDITION INVALID
                                                               attribute), 92
         (oandapy V20. definitions. transactions. Transaction REMACLIFE and ESNT EXIST
                                                               dapyV20.definitions.transactions.TransactionRejectReason
TAKE PROFIT ON FILL TRIGGER CONDITION MISSING
                                                               attribute), 97
         (oandapy V20.definitions.transactions.Transaction Repeated UNSPECIFIED
         attribute), 97
                                                               dapyV20.definitions.transactions.TransactionRejectReason
TAKE_PROFIT_ORDER
                                                               attribute), 97
                                               (oan-
        dapyV20.definitions.transactions.OrderFillReasonTRADE_IDENTIFIER_INCONSISTENCY
                                                                                                     (oan-
         attribute), 90
                                                               dapyV20.definitions.transactions.TransactionRejectReason
TAKE_PROFIT_ORDER
                                               (oan-
                                                               attribute), 97
         dapyV20.definitions.transactions.TransactionTypeTRADE_ON_FILL_CLIENT_EXTENSIONS_NOT_SUPPORTED
         attribute), 93
                                                               (oandapyV20.definitions.transactions.TransactionRejectReason
TAKE PROFIT ORDER ALREADY EXISTS
                                                               attribute), 97
         dapyV20.definitions.transactions.TransactionRejectReadle (oandapyV20.definitions.pricing.PriceStatus at-
                                                               tribute), 87
         attribute), 97
TAKE_PROFIT_ORDER_REJECT
                                               (oan- TradeClientExtensions
                                                                                 (class
                                                                                             in
                                                                                                      oan-
        dapyV20.definitions.transactions.TransactionType
                                                               dapyV20.endpoints.trades), 65
         attribute), 93
                                                      TradeClose (class in oandapyV20.endpoints.trades), 66
TakeProfitDetails (class in oandapyV20.contrib.requests),
                                                      TradeCloseRequest
                                                                               (class
         121
                                                               dapyV20.contrib.requests), 117
                                                      TradeCRCDO (class in oandapyV20.endpoints.trades),
TakeProfitOrderReason
                           (class
                                       in
                                                oan-
         dapyV20.definitions.transactions), 94
TakeProfitOrderRequest
                            (class
                                                      TradeDetails (class in oandapyV20.endpoints.trades), 68
                                       in
                                                oan-
        dapyV20.contrib.requests), 116
                                                      TradeID (class in oandapyV20.types), 104
terminate() (oandapy V20.endpoints.pricing.PricingStream TradePL (class in oandapy V20.definitions.trades), 87
                                                      TradesList (class in oandapyV20.endpoints.trades), 68
         method), 62
terminate() (oandapy V20.endpoints.transactions.Transaction Extde State (class in oandapy V20.definitions.trades), 88
         method), 77
                                                      TradeStateFilter (class in oandapyV20.definitions.trades),
Thursday (oandapy V20. definitions. instruments. Weekly Alignment
         attribute), 82
                                                      TRAILING STOP LOSS
TIME IN FORCE EXPIRED
                                                               dapyV20.definitions.orders.CancellableOrderType
                                               (oan-
        dapyV20.definitions.transactions.OrderCancelReason
                                                               attribute), 84
         attribute), 99
                                                      TRAILING_STOP_LOSS
                                                                                                     (oan-
TIME_IN_FORCE_GTD_TIMESTAMP_IN_PAST
                                                               dapyV20.definitions.orders.OrderType
                                                                                                       at-
         (oandapy V20. definitions. transactions. Transaction Reject Reason bute), 83
                                                      TRAILING STOP LOSS ON FILL CLIENT ORDER COMMENT IN
         attribute), 97
TIME IN FORCE GTD TIMESTAMP MISSING
                                                               (oandapyV20.definitions.transactions.TransactionRejectReason
         (oandapy V20.definitions.transactions.TransactionRejectReasotoribute), 97
         attribute), 97
                                                      TRAILING_STOP_LOSS_ON_FILL_CLIENT_ORDER_ID_ALREADY_
TIME_IN_FORCE_INVALID
                                                               (oandapyV20.definitions.transactions.OrderCancelReason
                                               (oan-
         dapyV20.definitions.transactions.TransactionRejectReason attribute), 99
                                                      TRAILING_STOP_LOSS_ON_FILL_CLIENT_ORDER_ID_INVALID
         attribute), 97
TIME_IN_FORCE_MISSING
                                                               (oandapyV20.definitions.transactions.TransactionRejectReason
                                               (oan-
         dapyV20.definitions.transactions.TransactionRejectReason attribute), 97
        attribute), 97
                                                      TRAILING_STOP_LOSS_ON_FILL_CLIENT_ORDER_TAG_INVALID
TimeInForce (class in oandapyV20.definitions.orders), 85
                                                               (oandapyV20.definitions.transactions.TransactionRejectReason
TRADE CLIENT EXTENSIONS MODIFY
                                                               attribute), 97
                                               (oan-
         dapyV20.definitions.transactions.TransactionTypeTRAILING STOP LOSS ON FILL GTD TIMESTAMP IN PAST
        attribute), 93
                                                               (oandapy V20.definitions.transactions.OrderCancelReason
```

attribute), 99		dapyV20.contrib.i	requests), 1	18	
TRAILING_STOP_LOSS_ON_FILL_GTD_TIME	ESTAMP		(class	in	oan-
(oandapyV20.definitions.transactions.Tra			ts.transactio	ons), 70	
attribute), 97		TransactionIDRange	(class	in	oan-
TRAILING_STOP_LOSS_ON_FILL_GTD_TIME	ESTAMP	_MISSIN@apyV20.endpoin	ts.transactio	ons), 71	
(oandapyV20.definitions.transactions.Tra			lass	in	oan-
attribute), 97		dapyV20.endpoin	ts.transactio	ons), 72	
TRAILING_STOP_LOSS_ON_FILL_PRICE_DIS	STANCE		(class	in	oan-
(oandapy V20.definitions.transactions.Tra			•	ions), 94	
attribute), 97		TransactionsSinceID	(class	in	oan-
TRAILING_STOP_LOSS_ON_FILL_PRICE_DIS				ons), 74	
(oandapyV20.definitions.transactions.Tra			(class	in	oan-
attribute), 97		dapyV20.endpoin	*		
TRAILING_STOP_LOSS_ON_FILL_PRICE_DIS	STANCE			in	oan-
(oandapy V20.definitions.transactions.Tra					
attribute), 98		TRANSFER_FUNDS		, ,	(oan-
TRAILING_STOP_LOSS_ON_FILL_PRICE_DIS			ns.transacti	ions Transac	
(oandapy V20.definitions.transactions.Tra					wie in 1 j p c
attribute), 98		TRANSFER_FUNDS_REJ	ECT		(oan-
TRAILING_STOP_LOSS_ON_FILL_PRICE_DIS				ions Transac	\
(oandapy V20.definitions.transactions.Tra			mo.tramoueti	ions. Transac	ation 13 pc
attribute), 98		TRIGGER_CONDITION_I	NVAL ID		(oan-
TRAILING_STOP_LOSS_ON_FILL_TIME_IN_I				ions Transac	\
(oandapyV20.definitions.transactions.Tra			nis.u ansacu	ions. Transac	Monrejeencason
attribute), 98		TRIGGER_CONDITION_I	MISSING		(oan-
TRAILING_STOP_LOSS_ON_FILL_TIME_IN_I				ions Transac	`
(oandapy V20.definitions.transactions.Tra			ms.transaeti	ions. Transac	tionicjecticason
attribute), 98		TRIGGERED (oandapy V20) definitions	orders Ord	erState
TRAILING_STOP_LOSS_ON_FILL_TRIGGER_			.acmintons	s.orucis.oru	erstate
(oandapy V20.definitions.transactions.Tra) definitions	orders Ord	erStateFilter
attribute), 98	ansaction.	attribute), 83	.acmintons	s.orucis.oru	erotater riter
TRAILING_STOP_LOSS_ON_FILL_TRIGGER_	CONDI		itions instri	iments Wee	kly Alignment
(oandapy V20.definitions.transactions.Tra			iitions.instit	aments. wee	Kiy/ Mighillent
attribute), 98	ansaction	rejectiveademioate), 02			
TRAILING_STOP_LOSS_ORDER	(oan-	U			
dapyV20.definitions.transactions.OrderF	(0000		types) 104		
attribute), 90		UNITS_INVALID	.types), 104	•	(oan-
TRAILING_STOP_LOSS_ORDER	(oan-	dapyV20.definition	ne trancacti	ione Trancac	
dapyV20.definitions.transactions.Transac		attribute), 98	nis.u ansacu	ions. Transac	Monicecticason
attribute), 94	• •	UNITS_LIMIT_EXCEEDE	SD.		(oan-
TRAILING_STOP_LOSS_ORDER_ALREADY_I		dapyV20.definition		ione Trancac	•
(oandapy V20.definitions.transactions.Tra		RejectReasonibute) 08	nis.u ansacu	ions. Transac	Monicecticason
attribute), 98		UNITS_MIMIMUM_NOT_	MET		(oon
TRAILING_STOP_LOSS_ORDER_REJECT	(oan-	dapyV20.definition		one Trancac	(oan- etionPaiactPasson
dapyV20.definitions.transactions.Transac	•		nis.u ansacu	ions, mansac	Monkejechkeason
attribute), 94	• •	UNITS_MISSING			(oon
TRAILING_STOP_LOSS_ORDERS_NOT_SUPP		dapyV20.definition	na transaati	one Transac	(oan-
(oandapy V20.definitions.transactions.Tra		RejectReason: hotel	nis.u ansacu	ions. Transac	uonkejeetkeason
attribute), 98			EEDED		(
TrailingStopLossDetails (class in	oan-	UNITS_PRECISION_EXC		T	(oan-
dapyV20.contrib.requests), 122	Jan-	dapyV20.definitio	ms.transacti	ions. i ransac	uonkejecikeason
TrailingStopLossOrderReason (class in	oan-	attribute), 98			
dapyV20.definitions.transactions), 99	oun-	V			
TrailingStopLossOrderRequest (class in	oan-	-	1100)		
Training 5 to process of a citation of the cit	Ouli	V20Frror (class in oandany)	V/2011 ×		

```
value (oandapy V20.types. Account ID attribute), 102
value (oandapy V20.types. Account Units attribute), 102
value (oandapy V20.types.ClientComment attribute), 102
value (oandapyV20.types.ClientID attribute), 102
value (oandapyV20.types.ClientTag attribute), 102
value (oandapyV20.types.DateTime attribute), 103
value (oandapyV20.types.OrderID attribute), 103
value (oandapy V20.types. Order Identifier attribute), 104
value (oandapyV20.types.OrderSpecifier attribute), 104
value (oandapy V20.types. Price Value attribute), 104
value (oandapy V20.types. TradeID attribute), 104
value (oandapyV20.types.Units attribute), 104
W
W (oandapy V20. definitions. instruments. Candlestick Granularity
         attribute), 81
Wednesday (oandapy V20.definitions.instruments.WeeklyAlignment
         attribute), 82
WeeklyAlignment
                          (class
                                         in
                                                    oan-
         dapyV20.definitions.instruments), 81
Ζ
ZERO
         (oandapyV20.definitions.trades.TradePL
                                                     at-
         tribute), 87
```