**ALGOTRADING**

AlgoTrading class was written to meet some trading needs(get account info, create market and limit order..vs) over the FTX.

Main methods are described in the following flow.

**get\_available\_currencies:**

**inputs :** market\_type(String) =[‘spot’,’future’,’all’ ]

**outputs:** available\_currency\_list (List)

**description :** gets all available currency pairs for specific market type

**check\_market\_type:**

**inputs :**

* base\_currency (String) = any base currency that listed in FTX
* quote\_currency(String) = any base currency that listed in FTX

**outputs:**

**description :** checks currency pair whether belongs to future market or not. If future markets raise Exception

**create\_market\_order:**

**inputs:**

* action(String) = Market side of order -> [‘buy’,sell’]
* base\_currency(String) = any base currency that listed in FTX
* quote\_currency(String) = any quote currency that listed in FTX
* amount(Float) = Order amount

**outputs:**

* total (Float): Total quantity of quote currency
* price (Float): The per-unit cost of the base currency
* currency (String): The quote currency

**description:** creates market order with respect to inputs

**create\_limit\_order:**

**inputs:**

* action(String) = Market side of order -> [‘buy’,sell’]
* base\_currency(String) = any base currency that listed in FTX
* quote\_currency(String) = any quote currency that listed in FTX
* amount(Float) = Order amount
* Price(Float) = The price of limit order
* Number\_of\_Iceberg\_Order(Integer) = The number of iceberg orders between 1 to 5

**outputs:**

* order\_size (Float): Total quantity of quote currency
* price (Float): The order price
* currency (String): The quote currency

**description:** creates limit order with respect to inputs, if the number of iceberg order is greater than 1, splits the quantity into equal parts and send them separately.

**simulate\_market\_price:**

**inputs:**

* action(String) = Market side of order -> [‘buy’,sell’]
* base\_currency(String) = any base currency that listed in FTX
* quote\_currency(String) = any quote currency that listed in FTX
* amount(Float) = Order amount
* return\_limit\_order\_info(Boolean) = If True returns limit order input with respect to calculation about market orders
* iceberg\_order(Integer) = If return\_limit\_order\_info is True, refers to number of iceberg orders between 1 to 5

**outputs:**

* response:
  + order\_size (Float): Total quantity of quote currency
  + price (Float): The order price
  + currency (String): The quote currency
* limit\_order\_info:
  + action(String) = Market side of order -> [‘buy’,sell’]
  + base\_currency(String) = any base currency that listed in FTX
  + quote\_currency(String) = any quote currency that listed in FTX
  + amount(Float) = Order amount
  + Price(Float) = The price of limit order
  + Number\_of\_Iceberg\_Order(Integer) = The number of iceberg orders between 1 to 5

**description**:

This method is related with simulation and does not send any order to FTX. For example if you wonder that after creating a market order with specific amount of a currency pair what will the average price result of this order, you can find the result with response return.

Moreover, if you have any doubt that creating market order can be affected very quickly via change on orderbook. You can create limit order as a result of simulation. Limit\_order\_info contains inputs for creating limit order that equivalent with market order

**Error Handling**

**\_process\_response** method helps to return FTX API errors ( not enough balance, no such markets ..vs)

**check\_market\_type** method helps to check whether currency pair belongs to future markets or nots

**simulate\_market\_price** method has “try except” block that checks order book in 100 depth whether total quantity in orderbook offset the input amounts or not

**Ongoing work**

There could be some additional controls after creating market and limit orders. For instance, checking balance, positions, open orders ..vs