L2Depth
inherits from MarketDataBase

+date_time
+depth = 5
+bids
+asks

+columns()
+types()
the date is
datetime now
+values()
returns the
values in the
asks and bids
with the common
date

+copy()
+is_diff()

MarketDataBase

+Depth
price, count,
volume

+parse_side()

Trade
inherits from MarketDataBase

+date_time
+trade_id
+trade_price
+trade_volume
+trade_side
+update_date_time

+columns()
+types()
date_time is
date format now
+values()

Snapshot

+columns(): static
same as L2Depth
plus order and
trade time exch
and instmt name

+types(): static
dates are
datetimes now

+values(): static
gives back the
snapshot values

Instrument
instrument per
exchange

+exchange_name
+instmt_name
+instmt_code
+instmt_snapshot_table_name
+order_book_id
+trade_id
+exch_trade_id
+subscribed
+recovered
+l2_depth
+prev_l2_depth
+last_trade
+order_book_channel_id
+trades_channel_id

+getters_for_everything()
+setters_for_everything()
+incr_order_book_id()
+incr_trade_id()

ExchGwBitfinexWs
inherits from
WebSocketApiClien
t

+get_link()
+get_order_book_subscription_string()
+get_trades_subscription_string()
+parse_l2_depth()
+parse_trade()

ExchGwBitfinex
inherits from
ExchangeGateway

+is_local_timestamp
+api_socket
+db_clients
+exch_snapshot_id

+get_exchange_name()
+on_open_handler()
makes
subscription and
logs it

+on_close_handler()
unsubscribes and
logs it

+on_message_handler(instmt,message)
if dict then
either info or
subscription and
logging happens
if list then
checks channel
id for order
book and for
trades and
api_socket
parses the order
or trade,
finally
ExchangeGateway.i
nsert_order or
inser_trade is
called

+start()
instmt.set_l2_dep
th(L2Depth(25))
instmt.set_prev_l
2_dept(L2Depth())
instmt.set_instmt
_snapshot_table_n
ame(

ExchangeGateway
this is the
general class
for the
different
exchange types,
main methods:
init_snapshot_tab
le,
init_instmt_snaps
hot_table,
start,
insert_order_book
,insert_trade

+is_local_timestamp: boolean = True
+api_socket
+db_clients
+exch_snapshot_id = starts from 0

+init(api_socket,db_client,exch_snapshot_id)
it assigns the
db and
api_socket

+get_instmt_snapshot_table_name()
gives back the
generic instmt
table atm

+init_snapshot_table()
instmt snapshot
table w/o ids

+init_instmt_snapshot_table()
it creates
instmt snapshot
table with id

+get_instmt_snapshot_id()
it increments
the
exch_snapshot_id
and gives it
back

+insert_order_book()
if instmt
l2depth is
available then
insert snapshot
and insert
instmt_record

+insert_trade()
if instmt
l2depth and
instmt last
trade are
available then
insert snapshot
and insert
instmt_record