Binary Marketdata API Change Log

Document Revision History
(BSE Enhancement in Marketdata)

Symphony Fintech Solutions Pvt. Ltd.
Unit No.301, Everest House, 6 Suren Road, Chakala Andheri (East) , Mumbai - 400093, India.
Tel: 022- 4019 0900 Website: www.symphonyfintech.com

Contents

API Binary Marketdata Changes

API Binary Marketdata Changes

1 : 1501 Event changes

1.1 LastTradedQuantity changes from int32 to long

1.2 TotalBuyQuantity changes from uint32 to long

1.3 TotalSellQuantity changes from uint32 to long

1.4 TotalTradedQuantity changes from uint32 to long

1.5 Size in MarketDepthRow for Bid & Ask changes from int32 to long

1.6 TotalOrders in MarketDepthRow for Bid & Ask changes from int32 to uint32

The Touchline (1501) WebSocket response in Old Build and New Build is elaborated below.

Field	Old Build Response	Change Type	New Build Response
isGzipCompressed	{ “isGzipCompressed” : 1,	No change	{ “isGzipCompressed” : 1,
Header MessageCode	“HeaderMessageCode”: 1501,	No change	“HeaderMessageCode”: 1501,
Header ExchangeSegment	“HeaderExchangeSegment” : 1,	No change	“HeaderExchangeSegment” : 1,
Header ExchangeInstrumentID	“HeaderExchangeInstrumentID” : 2885,	No change	“HeaderExchangeInstrumentID” : 2885,
Header BookType	“HeaderBookType‘: 1,	No change	“HeaderBookType‘: 1,
XmarketType	“XmarketType”: 1,	No change	“XmarketType”: 1,
uncompressedPacketSize	“uncompressedPacketSize”: 180,	No change	“uncompressedPacketSize”: 180,
compressedPacketSize	“compressedPacketSize”: 117,	No change	“compressedPacketSize”: 117,
MessageCode	“MessageCode”: 1501,	No change	“MessageCode”: 1501,
MessageVersion	“MessageVersion”: 4	No change	“MessageVersion”: 4
ApplicationType	“ApplicationType”: 0,	No change	“ApplicationType”: 0,
TokenID	“TokenID”: 0,	No change	“TokenID”: 0,
SequenceNumber	“SequenceNumber”: 170589198,	No change	“SequenceNumber”: 170589198,
SkipBytes	“SkipBytes”: 0,	No change	“SkipBytes”: 0,
ExchangeSegment	“ExchangeSegment”: 1,	No change	“ExchangeSegment”: 1,
ExchangeInstrumentID	“ExchangeInstrumentID”: 2885,	No change	“ExchangeInstrumentID”: 2885,
ExchangeTimeStamp	“ExchangeTimeStamp”: 14328253,	No change	“ExchangeTimeStamp”: 14328253,

Marketdepth.BidData.Size	“Touchline”: {
	“BidInfo”: {
	“Size”: 269,
		Changed data type: int32→ long	“Touchline”: {
			“BidInfo”: {
			“Size”: 269,

Marketdepth.BidData.Price	“Price”: 1422.9,	No change	“Price”: 1422.9,
Marketdepth.BidData.TotalOrders-	“TotalOrders”: 3,	**Changed data type**: int32→ uint32	“TotalOrders”: 3,

Marketdepth.BidData.BuyBackMarketMaker	"BuyBackMarketMaker": 0 },	No change	"BuyBackMarketMaker": 0 },
Marketdepth.AskData.Size	"AskInfo": { "Size": 1858,	**Changed data type**: int32→ long	"AskInfo": { "Size": 1858,
Marketdepth.AskData.Price	"Price": 1423,	No change	"Price": 1423,
Marketdepth.AskData.TotalOrders	"TotalOrders": 17 },	**Changed data type**: int32→ uint32	"TotalOrders": 17 },
Marketdepth.AskData.BackMarketMakerFlag	"BuyBackMarketMaker": 0 },	No change	"BuyBackMarketMaker": 0 },
Marketdepth.LastUpdateTime	"LastUpdateTime": 1432825338,	No change	"LastUpdateTime": 1432825338,
Marketdepth.LastTradedPrice	"LastTradedPrice": 1422.9,	No change	"LastTradedPrice": 1422.9,
Marketdepth.LastTradedQuantity	"LastTradedQunatity": 500,	**Changed data type**: int32→ long	"LastTradedQunatity": 500,
Marketdepth.TotalBuyQuantity	"TotalBuyQuantity": 472449,	**Changed data type**: uint32→ long	"TotalBuyQuantity": 472449,
Marketdepth.TotalSellQuantity	"TotalSellQuantity": 764079,	**Changed data type**: uint32→ long	"TotalSellQuantity": 764079,
Marketdepth.TotalTradedQuantity	"TotalTradedQuantity": 9440532,	**Changed data type**: uint32→ long	"TotalTradedQuantity": 9440532,
Marketdepth.AverageTradedPrice	"AverageTradedPrice": 1421.72,	No change	"AverageTradedPrice": 1421.72,
Marketdepth.LastTradedTime	"LastTradedTime": 1432825338,	No change	"LastTradedTime": 1432825338,
Marketdepth.PercentChange	"PercentChange": -0.829383886,	No change	"PercentChange": -0.829383886,
Marketdepth.Open	"Open": 1426.1,	No change	"Open": 1426.1,
Marketdepth.High	"High": 1437,	No change	"High": 1437,
Marketdepth.Low	"Low": 1413.9,	No change	"Low": 1413.9,
Marketdepth.close	"Close": 1434.8,	No change	"Close": 1434.8,
Marketdepth.TotalValueTraded	"TotalValueTraded": null,	No change	"TotalValueTraded": null,
Marketdepth.BuyBackTotalBuy	"BuyBackTotalBuy": 0,	No change	"BuyBackTotalBuy": 0,
Marketdepth.BuyBackTotalSell	"BuyBackTotalSell": 0,	No change	"BuyBackTotalSell": 0,
Marketdepth.BookType	"BookType": 1,	No change	"BookType": 1,
Marketdepth.MarketType	"MarketType": 1 } }	No change	"MarketType": 1 } }

Latest Build (V2) Touchline (1501) socket response JSON e.g.:

{"isGzipCompressed" : 1,
"HeaderMessageCode": 1501,
"HeaderExchangeSegment" : 1,
"HeaderExchangeInstrumentID" : 2885,
"HeaderBookType": 1,
"XmarketType": 1,
"uncompressedPacketSize": 180,
"compressedPacketSize": 117,
"MessageCode": 1501,

"MessageVersion": 4,
"ApplicationType": 0,
"TokenID": 0,
"SequenceNumber": 170589198,
"SkipBytes": 0,
"ExchangeSegment": 1,
"ExchangeInstrumentID": 2885,
"ExchangeTimeStamp": 14328253,
{
"TOUCHLINE": {
"messageVersion": 4,
"applicationType": 0,
"tokenID": 0,
"sequenceNumber": 1770574240463462,
"skipBytes": 0,
"exchangeSegment": 1,
"exchangeInstrumentId": 2885,
"exchangeTimestamp": 1439391451,
"Bid": {
"size": 375,
"rowprice": 1422.9,
"totalOrders": 3,
"backmarketmakerflag": 0
},
"Ask": {
"size": 750,
"rowprice": 1423,
"totalOrders": 1,
"backmarketmakerflag": 0
},
"lut": 1439391451,
"LTP":1423.8,
"ltq": 75,
"totalBuyQuantity": 388050,
"totalSellQuantity": 312675,
"totalTradedQuantity": 3913200,
"averageTradedPrice":1423.54,
"lastTradedTime": 1439391448,
"percentChange": 0.7724693242120884,
"open": 1426.1,
"high": 1436,
"low": 1431
"close": 1426.1.,
"totalValueTraded": nan,
"bbTotalBuy": 0,
"bbTotalSell": 0,
"BookType": 1,
"MarketType": 1
} }

2: 1502 Event Changes

2.1 LastTradedQuantity changes from int32 to long

2.2 TotalBuyQuantity changes from uint32 to long

2.3 TotalSellQuantity changes from uint32 to long

2.4 TotalTradedQuantity changes from uint32 to long

2.5 Size in MarketDepthRow for Bid & Ask changes from int32 to long

2.6 TotalOrders in MarketDepthRow for Bid & Ask changes from int32 to uint32

The Touchline (1502) WebSocket response in Old Build and New Build is elaborated below.

Field	Old Build Response	Change Type	New Build Response
isGzipCompressed	{ "isGzipCompressed": 1,	No change	{ "isGzipCompressed": 1,
Header MessageCode	"HeaderMessageCode": 1502,	No change	"HeaderMessageCode": 1502,
Header ExchangeSegment	"HeaderExchangeSegment": 1,	No change	"HeaderExchangeSegment": 1,
Header ExchangeInstrumentID	"HeaderExchangeInstrumentID": 2885,	No change	"HeaderExchangeInstrumentID": 2885,
Header BookType	"HeaderBookType": 1,	No change	"HeaderBookType": 1,
XMarketType	"XMarketType": 1,	No change	"XMarketType": 1,
uncompressedPacketSize	"uncompressedPacketSize": 180,	No change	"uncompressedPacketSize": 180,
compressedPacketSize	"compressedPacketSize": 117,	No change	"compressedPacketSize": 117,
MessageCode	"MessageCode": 1502,	No change	"MessageCode": 1502,
MessageVersion	"MessageVersion": 4,	No change	"MessageVersion": 4,
ApplicationType	"ApplicationType": 0,	No change	"ApplicationType": 0,
TokenID	"TokenID": 0,	No change	"TokenID": 0,
SequenceNumber	"SequenceNumber": 170589198,	No change	"SequenceNumber": 170589198,
SkipBytes	"SkipBytes": 0,	No change	"SkipBytes": 0,
ExchangeSegment	"ExchangeSegment": 1,	No change	"ExchangeSegment": 1,
ExchangeInstrumentID	"ExchangeInstrumentID": 2885,	No change	"ExchangeInstrumentID": 2885,
ExchangeTimeStamp	"ExchangeTimeStamp": 14328253,	No change	"ExchangeTimeStamp": 14328253,

Marketdepth.BidData.bidCount	"**Marketdepth**": {
	"**BidInfo**": {
	"bidCount": 5,
		No change	"**Marketdepth**": {
			"**BidInfo**": {
			"bidCount": 5,

Marketdepth.BidData.Size	"Size": 269,	**Changed data type**: int32→ long	"Size": 269,
Marketdepth.BidData.Price	"Price": 1422.9,	No change	"Price": 1422.9,
Marketdepth.BidData.TotalOrders-	"TotalOrders": 3,	**Changed data type**: int32→ uint32	"TotalOrders": 3,

Marketdepth.BidData.BuyBackMarketMaker	"BuyBackMarketMaker": 0 },	No change	"BuyBackMarketMaker": 0 },
Marketdepth.AskData.askCount	"**AskInfo**": { "askCount": 5,	No change	"**AskInfo**": { "askCount": 5,
Marketdepth.AskData.Size	"Size": 1858,	**Changed data type**: int32→ long	"Size": 1858,
Marketdepth.AskData.Price	"Price": 1423,	No change	"Price": 1423,
Marketdepth.AskData.TotalOrders	"TotalOrders": 17 }	**Changed data type**: int32→ uint32	"TotalOrders": 17 }
Marketdepth.AskData.BuyBackMarketMaker	"BuyBackMarketMaker": 0 },	No change	"BuyBackMarketMaker": 0 },
Touchline.BidData.Size	"Size": 269,	**Changed data type**: int32→ long	"Size": 269,
Touchline.BidData.Price	"Price": 1422.9,	No change	"Price": 1422.9,
Touchline.BidData.TotalOrders-	"TotalOrders": 3,	**Changed data type**: int32→ uint32	"TotalOrders": 3,
Touchline.BidData.BuyBackMarketMaker	"BuyBackMarketMaker": 0	No change	"BuyBackMarketMaker": 0
Touchline.AskData.Size	"Size": 1858,	**Changed data type**: int32→ long	"Size": 1858,
Touchline.AskData.Price	"Price": 1423,	No change	"Price": 1423,
Touchline.AskData.TotalOrders	"TotalOrders": 17	**Changed data type**: int32→ uint32	"TotalOrders": 17
Touchline.AskData.BuyBackMarketMaker	"BuyBackMarketMaker": 0	No change	"BuyBackMarketMaker": 0
Marketdepth.TotalBuyQuantity	"TotalBuyQuantity": 472449,	**Changed data type**: uint32→ long	"TotalBuyQuantity": 472449,
Marketdepth.TotalSellQuantity	"TotalSellQuantity": 764079,	**Changed data type**: uint32→ long	"TotalSellQuantity": 764079,
Marketdepth.TotalTradedQuantity	"TotalTradedQuantity": 9440532,	**Changed data type**: uint32→ long	"TotalTradedQuantity": 9440532,
Marketdepth.AverageTradedPrice	"AverageTradedPrice": 1421.72,	No change	"AverageTradedPrice": 1421.72,
Marketdepth.LastTradedTime	"LastTradedTime": 1432825338,	No change	"LastTradedTime": 1432825338,
Marketdepth.PercentChange	"PercentChange": -0.829383886,	No change	"PercentChange": -0.829383886,
Marketdepth.Open	"Open": 1426.1,	No change	"Open": 1426.1,
Marketdepth.High	"High": 1437,	No change	"High": 1437,
Marketdepth.Low	"Low": 1413.9,	No change	"Low": 1413.9,
Marketdepth.close	"Close": 1434.8,	No change	"Close": 1434.8,
Marketdepth.TotalValueTraded	"TotalValueTraded": null,	No change	"TotalValueTraded": null,
Marketdepth.BuyBackTotalBuy	"BuyBackTotalBuy": 0,	No change	"BuyBackTotalBuy": 0,
Marketdepth.BuyBackTotalSell	"BuyBackTotalSell": 0,	No change	"BuyBackTotalSell": 0,
Marketdepth.BookType	"BookType": 1,	No change	"BookType": 1,
Marketdepth.XMarketType	"MarketType": 1 }	No change	"MarketType": 1 }

Latest Build (V2) Touchline (1502) socket response JSON e.g.:

{ "isGzipCompressed": 1,
"HeaderMessageCode": 1502,
"HeaderExchangeSegment": 1,
"HeaderExchangeInstrumentID": 2885,
"HeaderBookType": 1,
"XMarketType": 1,
"uncompressedPacketSize": 180,
"compressedPacketSize": 117,
"MessageCode": 1502,
"MessageVersion": 4,
"ApplicationType": 0,
"TokenID": 0,
"SequenceNumber": 170589198,
"SkipBytes": 0,
"ExchangeSegment": 1,
"ExchangeInstrumentID": 2885,
"ExchangeTimeStamp": 14328253,
{
"MARKETDEPTH": {
"messageVersion": 4,
"applicationType": 0,
"tokenID": 0,
"sequenceNumber": 1770574435529589,
"skipBytes": 0,
"exchangeSegment": 2,
"exchangeInstrumentId": 64103,
"exchangeTimestamp": 1439391451,
"bidCount": 5,
"Bid": [
{
"size": 375,
"rowprice": 1422.9,
"totalOrders": 3,
"backmarketmakerflag": 0
},
{
"size": 825,
"rowprice": 1422.8,
"totalOrders": 3,
"backmarketmakerflag": 0
},
{
"size": 900,

```
        "rowprice": 1422.7,
        "totalOrders": 5,
        "backmarketmakerflag": 0
      },
      {
        "size": 1425,
        "rowprice": 1422.6,
        "totalOrders": 7,
        "backmarketmakerflag": 0
      },
      {
        "size": 1050,
        "rowprice":1422.5,
        "totalOrders": 6,
        "backmarketmakerflag": 0
      }
  ],
  "askCount": 5,
  "Ask": [
      {
        "size": 750,
        "rowprice": 1423,
        "totalOrders": 1,
        "backmarketmakerflag": 0
      },
      {
        "size": 150,
        "rowprice":1423.2,
        "totalOrders": 2,
        "backmarketmakerflag": 0
      },
      {
        "size": 525,
        "rowprice": 1423.4,
        "totalOrders": 4,
        "backmarketmakerflag": 0
      },
      {
        "size": 75,
        "rowprice":1423.5,
        "totalOrders": 1,
        "backmarketmakerflag": 0
      },
      {
```

```
            "size": 1125,
            "rowprice":1423.8,
            "totalOrders": 1,
            "backmarketmakerflag": 0
          }
        ],
        "lut": 1439391451,
        "LTP":1423.8,
        "ltq": 75,
        "totalBuyQuantity": 388050,
        "totalSellQuantity": 312675,
        "totalTradedQuantity": 3913200,
        "averageTradedPrice":1423.54,
        "lastTradedTime": 1439391448,
        "percentChange": 0.7724693242120884,
        "open":1420.0,
        "high":1430.0,
        "low":1411.2,
        "close":1425.1,
        "totalValueTraded": nan,
        "bbTotalBuy": 0,
        "bbTotalSell": 0,
        "BookType": 1,
        "MarketType": 1
    }
}
```

3: 1510 Event Changes

3.1. OpenInterest changes from int32 to long

3.2. UnderlyingTotalOpenInterest changes from int32 to long

The Touchline (1510) WebSocket response in Old Build and New Build is elaborated below.

Field	Old Build Response	Change Type	New Build Response
isGzipCompressed	{ "isGzipCompressed": 1,	No change	{ "isGzipCompressed": 1,
Header MessageCode	"HeaderMessageCode": 1510,	No change	"HeaderMessageCode": 1510,,
Header ExchangeSegment	"HeaderExchangeSegment": 2,	No change	"HeaderExchangeSegment": 2,
Header ExchangeInstrumentID	"HeaderExchangeInstrumentID": 56785,	No change	"HeaderExchangeInstrumentID": 56785,
Header BookType	"HeaderBookType": 1,	No change	"HeaderBookType": 1,
XMarketType	"XMarketType": 1,	No change	"XMarketType": 1,
uncompressedPacketSize	"uncompressedPacketSize": 180,	No change	"uncompressedPacketSize": 180,
compressedPacketSize	"compressedPacketSize": 117,	No change	"compressedPacketSize": 117,
MessageCode	"MessageCode": 1510,	No change	"MessageCode": 1510,
MessageVersion	"MessageVersion": 4,	No change	"MessageVersion": 4,
ApplicationType	"ApplicationType": 0,	No change	"ApplicationType": 0,
TokenID	"TokenID": 0,	No change	"TokenID": 0,
SequenceNumber	"SequenceNumber": 170589198,	No change	"SequenceNumber": 170589198,
SkipBytes	"SkipBytes": 0,	No change	"SkipBytes": 0,
ExchangeSegment	"ExchangeSegment": 2,	No change	"ExchangeSegment": 2,
ExchangeInstrumentID	"ExchangeInstrumentID": 56785,	No change	"ExchangeInstrumentID": 56785,
ExchangeTimeStamp	"ExchangeTimeStamp": 14328253,	No change	"ExchangeTimeStamp": 14328253,
MarketType	"MarketType": "1",	No change	"MarketType": "1",
OpenInterest	"OpenInterest": "1523878",	**Changed data type**: int32→ long	"OpenInterest": "1523878",
UnderlyingExchangeSegment	"UnderlyingExchangeSegment": 1,	No change	"UnderlyingExchangeSegment": 1,
UnderlyingInstrumentID	"UnderlyingInstrumentID": 0,	No change	"UnderlyingInstrumentID": 0,
IsStringExits	"IsStringExits": 1,	No change	"IsStringExits": 1,
StringLength	"StringLength": 10,	No change	"StringLength": 10,
UnderlyingTotalOpenInterest	"UnderlyingTotalOpenInterest": 485431650 }	**Changed data type**: int32→ long	"UnderlyingTotalOpenInterest": 485431650 }

Latest Build (V2) Touchline (1510) socket response JSON e.g.:

{
"isGzipCompressed": 1,
"HeaderMessageCode": 1510,,
"HeaderExchangeSegment": 2,
"HeaderExchangeInstrumentID":56785,
"HeaderBookType": 1,
"XMarketType": 1,
"uncompressedPacketSize": 180,
"compressedPacketSize": 117,
"MessageCode": 1510,
"MessageVersion": 4,
"ApplicationType": 0,
"TokenID": 0,
"SequenceNumber": 170589198,
"SkipBytes": 0,
"ExchangeSegment": 2,
"ExchangeInstrumentID": 56785,
"ExchangeTimeStamp": 14328253,
"MarketType": 1,
"OpenInterest": "1523878",
"UnderlyingExchangeSegment": 1,
"UnderlyingInstrumentID": 0,
"IsStringExits": 1,
"StringLength": 10,
"UnderlyingTotalOpenInterest": 485431650
}

4: Sample python code

```
import zlib
from binary_reader import BinaryReader
import struct
from enum import Enum


sample_data =
b'\x01\xdd\x05\x02\x00g\xfa\x00\x00\x01\x00\x01\x00\xcc\x00\x80\x00\xbb\xcb\xca\xc2\x80\x00i\xbb\x96}\x08\xf6b\x03\xb3\
x99\x18\xd2\x7f10\xdc\x8e;\x1d\n\xe2\x953"T\x1d\xe0\xb9\xe1\xc0\x0cf\xbdc\x82\x88\x18\x03A1\xef\r\x07\x88"\x98\x1e\x98\
xa87T\xdf\xa5\xd7\xac`:\xf9(\xc4\xce\x0f[\xad\xc1\xf47\x8d\x98\xfeG_\xae;\xdc\x80\xea\xdb\xf1n\xca4\xc1\x1d/\xec\xc1v=\xbc\
xee\x00\xa2\x1b\x04n8\x9c=s\xe6\x8c\xcf\xb5\xeb\x0ei@\xe0v\x0f"\xce\xc0\xf0\xe3?\x88d\x04B\x00\x01\xde\x05\x02\x00g\xfa
\x00\x00\x01\x00\x01\x00\xb0\x01\xd3\x00\xbb\xc7\xca\xc2\x80\x00\xa5\xc6\xee\x7f\x82\xbd\xd8\xc0l&\x86\xf4_\x0c\x0c\xb7
\xe3N\x87\x82x\xac@\\\xce\x88Py\x80\xe7\x86\x033\x98e\t\xa1\x18f\xcd\x9c9s\'\\\xb4\x05*j\x0c\x04\x9b\x81\xa2\xac`\xdeD\
x08\xc5p\xf6\xcc\x993k\x80\xa2\xec`\x9e\x14\xd4\ti@\xb0\x0c(\n\xb1\x1f\xa4\xf4\x1d\x13\xc2\x94b\xde\x1b\x0e\x10\x07Lc@\
xd8X\t\x14\x85(\xe2eB\xb8\xae\x01(\n1\xd4\x9b\x01av\x1b\xdc\x84T$O\x1f\x80\x8bb\xf7\x1fv7\xc0\xc2\x05&\n\xb3\xe7\xd2k\
x88\x17\x93\x8fB\xac\xf8\xb0\xd5\x1aL\x7f\xd3\x88\xe9\x7f\xf4\xe5\xba\xc3\r\xa8\xbe\x1d\xef\xa6L\x13\xdc\xf1\xc2\x1el\xd7\
xc3\xeb\x0e`W\x0b\xdcp\x00\x85\x8c\xcf\xb5\xeb\x0e
\xf7\xba\xdd\x83\x8830\xfc\xf8\x0f"\x19\x81\x10\x00\x01\xe6\x05\x02\x00g\xfa\x00\x00\x01\x00\x01\x00P\x006\x00{\xc6\xca\
xc2\x80\x00\xbaA\xee\x7f\x82\xbd\xd8\xc0l&\x86\xf4_\x0c\x0c\xb7\xe3N\x87\x82x\x8c\x0c\x1aW\x98\x18!,\x18`\xe4\xf0\xcbL+
\xa9T05(=\xca-\x07\x11\x03\x00'


def pako_inflate_raw(data):
    decompress = zlib.decompressobj(-15)
    decompressed_data = decompress.decompress(data)
    decompressed_data += decompress.flush()
    return decompressed_data

class ApplicationMessageVersion(Enum):
    Version_1= 1
    Version_1_0_1_0969= 2
    Version_1_0_1_2879= 3
    Version_1_0_1_2983= 4

class MarketDepthEvent():
    def deserialize(reader,count):
        count+= 2
        messageVersion = reader.read_uint16()
        applicationType = reader.read_uint16()
        tokenID = reader.read_uint64()

        count += 8
        if (messageVersion >= ApplicationMessageVersion.Version_1_0_1_2983.value):
            sequenceNumber =reader.read_uint64()
            count += 8
            SkipBytes = reader.read_int32()
            count += 4

        exchangeSegment = int(reader.read_int16())
        count+= 2

        exchangeInstrumentId = reader.read_int32()
        count+= 4

        exchangeTimestamp = reader.read_uint64()
        count+= 8
        bidCount = reader.read_int32()
        count += 4
```

```
bidData = []
for x in range(0, bidCount):
    md = MarketDeptRowInfo(reader, count)
    count, data = md.deserialize()
    bidData.append(data)
askCount = reader.read_int32()
count += 4

askData = []
for x in range(0, askCount):
    md = MarketDeptRowInfo(reader, count)
    count, data = md.deserialize()
    askData.append(data)

md = MarketDeptRowInfo(reader, count)
count = md.deserialize()[0]
md1 = MarketDeptRowInfo(reader, count)
count = md1.deserialize()[0]

lut = reader.read_uint64()
count += 8

LTP = struct.unpack('d', reader.read_bytes(8))[0]
count += 8

ltq = struct.unpack('q', reader.read_bytes(8))[0]
count += 8

totalBuyQuantity = struct.unpack('q', reader.read_bytes(8))[0]
count += 8

totalSellQuantity = struct.unpack('q', reader.read_bytes(8))[0]
count += 8

totalTradedQuantity = struct.unpack('q', reader.read_bytes(8))[0]
count += 8

averageTradedPrice = struct.unpack('d', reader.read_bytes(8))[0]
count += 8

lastTradedTime = reader.read_int64()
count += 8

percentChange = struct.unpack('d', reader.read_bytes(8))[0]
count += 8

open =  struct.unpack('d', reader.read_bytes(8))[0]
count += 8

high = struct.unpack('d', reader.read_bytes(8))[0]
count += 8

low = struct.unpack('d', reader.read_bytes(8))[0]
count += 8

close = struct.unpack('d', reader.read_bytes(8))[0]
count += 8

totalvaluetraded =  struct.unpack('d', reader.read_bytes(8))[0]
count += 8
bbtotalbuy = reader.read_int16()
count += 2
```

```
        bbtotalsell = reader.read_int16()
        count += 2

        Booktype = reader.read_int16()
        count += 2

        MarketType = reader.read_int16()

        return {
          "MARKETDEPTH": {
            "messageVersion": messageVersion,
            "applicationType": applicationType,
            "tokenID": tokenID,
            "sequenceNumber": sequenceNumber,
            "skipBytes": SkipBytes,
            "exchangeSegment": exchangeSegment,
            "exchangeInstrumentId": exchangeInstrumentId,
            "exchangeTimestamp": exchangeTimestamp,
            "bidCount": bidCount,
            "Bid": bidData,
            "askCount": askCount,
            "Ask": askData,
            "lut": lut,
            "LTP": LTP,
            "ltq": ltq,
            "totalBuyQuantity": totalBuyQuantity,
            "totalSellQuantity": totalSellQuantity,
            "totalTradedQuantity": totalTradedQuantity,
            "averageTradedPrice": averageTradedPrice,
            "lastTradedTime": lastTradedTime,
            "percentChange": percentChange,
            "open": open,
            "high": high,
            "low": low,
            "close": close,
            "totalValueTraded": totalvaluetraded,
            "bbTotalBuy": bbtotalbuy,
            "bbTotalSell": bbtotalsell,
            "BookType": bookType,
            "MarketType": MarketType
          }
        }

class MarketDeptRowInfo():
    size = 0
    rowprice = 0.0
    totalOrders = 0
    backmarketmakerflag = 0
    def __init__(self, reader, count):
        self.reader = reader
        self.count = count

    def deserialize(self):
        size = struct.unpack('q', self.reader.read_bytes(8))[0]
        self.count += 8
        rowprice = struct.unpack('d', self.reader.read_bytes(8))[0]
        self.count += 8
        totalOrders =  self.reader.read_uint32()
        self.count += 4
        backmarketmakerflag =  self.reader.read_int16()
        self.count += 2
```

```
        row = {
            "size":size ,
            "rowprice":rowprice,
            "totalOrders":totalOrders ,
            "backmarketmakerflag":backmarketmakerflag
        }
        return self.count, row

class OpenInterest():
    def deserialize(reader,count):
        count+= 2
        messageVersion = reader.read_uint16()
        applicationType = reader.read_uint16()
        tokenID = reader.read_uint64()

        count += 8
        if (messageVersion >= ApplicationMessageVersion.Version_1_0_1_2983.value):
            sequenceNumber =reader.read_uint64()
            count += 8
            SkipBytes = reader.read_int32()
            count += 4

        exchangeSegment = int(reader.read_int16())
        count+= 2

        exchangeInstrumentId = reader.read_int32()
        count+= 4

        exchangeTimestamp = reader.read_uint64()
        count+= 8

        MarketType = reader.read_int16()
        count += 2

        openInterest = struct.unpack('q', reader.read_bytes(8))[0]
        count += 8

        underlyingExchangeSegment = reader.read_int16()
        count += 2

        underlyingInstrumentID = reader.read_uint64()
        count += 8

        isStringExits =reader.read_int8()
        count += 1

        if (isStringExits == 1) :
            stringLength = reader.read_int8()
            count += 1
            count += stringLength

        underlyingTotalOpenInterest = struct.unpack('q', reader.read_bytes(8))[0]
        underlyingTotalOpenInterest =reader.read_uint64()
        count += 8

        return {
        "OpenInterest": {
            "exchangeSegment": exchangeSegment,
            "exchangeInstrumentId": exchangeInstrumentId,
            "exchangeTimestamp": exchangeTimestamp,
            "openInterest": openInterest,
```

```
        "underlyingExchangeSegment": underlyingExchangeSegment,
        "underlyingInstrumentID": underlyingInstrumentID,
        "isStringExits": isStringExits,
        "underlyingTotalOpenInterest": underlyingTotalOpenInterest
    }
    }

class Touchline():
  def deserialize(reader,count):
    count+= 2
    messageVersion = reader.read_uint16()
    applicationType = reader.read_uint16()
    tokenID = reader.read_uint64()

    count += 8
    if (messageVersion >= ApplicationMessageVersion.Version_1_0_1_2983.value):
      sequenceNumber =reader.read_uint64()

      count += 8
      SkipBytes = reader.read_int32()
      count += 4
    exchangeSegment = int(reader.read_int16())
    count+= 2

    exchangeInstrumentId = reader.read_int32()
    count+= 4

    exchangeTimestamp = reader.read_uint64()
    count+= 8

    md = MarketDeptRowInfo(reader,count)
    count, bidData = md.deserialize()


    md1= MarketDeptRowInfo(reader,count)
    count, askData = md1.deserialize()

    lut = reader.read_uint64()
    count += 8

    LTP = struct.unpack('d', reader.read_bytes(8))[0]
    count += 8
    ltq = struct.unpack('q', reader.read_bytes(8))[0]
    count += 8

    totalBuyQuantity = struct.unpack('q', reader.read_bytes(8))[0]
    count += 8

    totalSellQuantity = struct.unpack('q', reader.read_bytes(8))[0]
    count += 8

    totalTradedQuantity = struct.unpack('q', reader.read_bytes(8))[0]
    count += 8


    averageTradedPrice = struct.unpack('d', reader.read_bytes(8))[0]
    count += 8

    lastTradedTime = reader.read_int64()
    count += 8

    percentChange = struct.unpack('d', reader.read_bytes(8))[0]
```

```
    count += 8

    open =  struct.unpack('d', reader.read_bytes(8))[0]
    count += 8

    high = struct.unpack('d', reader.read_bytes(8))[0]
    count += 8

    low = struct.unpack('d', reader.read_bytes(8))[0]
    count += 8

    close = struct.unpack('d', reader.read_bytes(8))[0]
    count += 8


    totalvaluetraded =  struct.unpack('d', reader.read_bytes(8))[0]
    count += 8
    bbtotalbuy = reader.read_int16()
    count += 2

    bbtotalsell = reader.read_int16()
    count += 2

    Booktype = reader.read_int16()
    count += 2

    MarketType = reader.read_int16()

    return {
       "TOUCHLINE": {
          "messageVersion": messageVersion,
          "applicationType": applicationType,
          "tokenID": tokenID,
          "sequenceNumber": sequenceNumber,
          "skipBytes": SkipBytes,
          "exchangeSegment": exchangeSegment,
          "exchangeInstrumentId": exchangeInstrumentId,
          "exchangeTimestamp": exchangeTimestamp,
          "Bid": bidData,
          "Ask": askData,
          "lut": lut,
          "LTP": LTP,
          "ltq": ltq,
          "totalBuyQuantity": totalBuyQuantity,
          "totalSellQuantity": totalSellQuantity,
          "totalTradedQuantity": totalTradedQuantity,
          "averageTradedPrice": averageTradedPrice,
          "lastTradedTime": lastTradedTime,
          "percentChange": percentChange,
          "open": open,
          "high": high,
          "low": low,
          "close": close,
          "totalValueTraded": totalvaluetraded,
          "bbTotalBuy": bbtotalbuy,
          "bbTotalSell": bbtotalsell,
          "BookType": bookType,
          "MarketType": MarketType
       }
    }

a = bytearray(sample_data)
```

```
offset = 0
count = 0
currentsize = 0
isnextpacket = True
datalen=len(a)
packetcount = 0
br = BinaryReader(sample_data)
packetSize = 0
uncompressedPacketSize = 0
nextdata = a
while (isnextpacket):
   nextdata = a[offset:datalen]
   br = BinaryReader(nextdata)
   isGzipCompressed = br.read_int8()
   offset=offset+1
   if (isGzipCompressed == 1):
      nextdata = a[offset:datalen]
      br = BinaryReader(nextdata)
      messageCode = br.read_uint16()
      exchangeSegment = br.read_int16()
      exchangeInstrumentID = br.read_int32()
      bookType = br.read_int16()
      marketType = br.read_int16()
      uncompressedPacketSize = br.read_uint16()
      compressedPacketSize = br.read_uint16()
      offset += 16
      filteredByteArray = a[offset:(offset + compressedPacketSize)]
      inflate = pako_inflate_raw(filteredByteArray)
      result = bytearray(inflate)
      r = BinaryReader(result)
      currentsize = compressedPacketSize  + offset
      if (currentsize < len(a)):
         isnextpacket = True
         packetcount = 1
         offset = currentsize
      else: isnextpacket = False
      messageCode = str(r.read_uint16())
      if ("1501" in str(messageCode)) :
         json_data_1501 = Touchline.deserialize(r,count)
         print(json_data_1501)
      elif ("1502" in str(messageCode)) :
         json_data_1502 = MarketDepthEvent.deserialize(r,count)
         print(json_data_1502)
      elif ("1510" in str(messageCode)):
         json_data_1510 = OpenInterest.deserialize(r,count)
         print(json_data_1510)
   elif (isGzipCompressed == 0):
      messageCode = str(br.read_uint16())
      exchangeSegment = br.read_int16()
      exchangeInstrumentID = br.read_int32()
      bookType = br.read_int16()
      marketType = br.read_int16()
      uncompressedPacketSize = br.read_uint16()
      compressedPacketSize = br.read_uint16()
      offset += 14
      count = offset
      if ("1501" in messageCode) :
         json_data_1501 = Touchline.deserialize(br,count)
         print(json_data_1501)
      elif ("1502" in str(messageCode)) :
         json_data_1502 = MarketDepthEvent.deserialize(r,count)
         print(json_data_1502)
```

```
elif ("1510" in str(messageCode)):
    json_data_1510 = OpenInterest.deserialize(r,count)
    print(json_data_1510)
currentsize = offset+ uncompressedPacketSize
if (currentsize < len(a)):
    isnextpacket = True
    packetcount = 1
    offset = currentsize
else: isnextpacket = False
```

THANK YOU