AAPI On-Boarding FAQ

Version 1.3

1. **Encoding**

The encoding over the wire should be utf-8.

2. **The websocket is wrapped in sockjs**

Yes, this is to support http polling fall-back for web clients that cannot establish or don't support a websocket connection. While it would not be expecting that any Retail API consumers of PUSH to be using the fallback option. In theory any IETF standard websocket (RFC 6455) client can connect.

3. **Is this intentional - it seems like you try quite hard to keep the number of bytes you send quite low, and sockjs adds some overhead on top of this?**

We do try to keep the message size low, though the message format is more to do with the historical evolution of the product. The overhead of sockjs is not considered an issue.

4. **Do you ever delete topics in the topic hierarchy?**

Yes, when Events and Markets are settled or in the unusual sceanrio of the Events and Markets being undisplayed on the exchange. We do not currently inform the client when topics are deleted. This function is current scheduled for release in Q1 2018.

5. **If there's no topic deletion, is there some way of avoiding the structure from growing forever?**

The lifecycle of a market is the same whether they are polling for information via External API or pushed via push. As flagged above, the functionality to push a message when a topic is deleted will be added in Q1 2018, however the client system could never rely solely on this. As the socket connection is not a forever event. It will occasionally drop and need to be reset and a message may not be received.

e.g. if a client is not connected when a market is settled the client would never be informed of a Topic deletion. The client would therefore need to deduce a deletion by either the lifecycle of changes (completion etc) or the marketId being returned in a subscription response message as unknown.

6. **What currency should be specified (should this just be e.g. "GBP"?)**

The currency must be specified but is not used for API clients and therefore can be anything. Irrespective of the value specified here, the currency of the session will always be that of the account concerned. Best practise would be to specific the same currency and the Betdaq Account establishing the session.

7. What is the current AAPI Version?

“2.0” is the current supported value

8. What is specified in the clientSpecifiedGuid?

The clientSpecifiedGUId is a UUID variant 4 and with no "{}". The clientSpecifiedGUId is created by the client to uniquely identify an execution of the client. It should be generated for each distinct connection.

e.g. "85bb329c-b5c4-4cf3-9e30-2d24f64c8c09" be valid?

9. What is specified in the channelInformation and how is it used?

This is a unique identifier set by the client for their application and should include a version number [*String (256)*]. This is to allow both the client and Betdaq easily identify the calls from an application when trouble shooting any issues. This would have a similar purpose to the applicationIdentifier in the ExternalApiHeader.

10. What is granularChannelType used for?

The granular ChannelType to identify the channel an interaction with a client has occurred though – this covers all possible channels including mobile, browsers etc. It is an enum and the domain values are specified below. The Retail API consumers of PUSH are set this as API (9).

GranularChannelType

Enumeration defining the channel through which an interaction occurred. Domain values are:

MobileBrowser (1)

The interaction occurred through a browser on a mobile device.

TabletBrowser (2)

The interaction occurred through a browser on a tablet device.

MobileApplication (3)

The interaction occurred through a native application on a mobile device.

TabletApplication (4)

The interaction occurred through a native application on a tablet device.

InternetBrowser (5)

The interaction occurred placed through a browser on a desktop or laptop PC.

InternetApplication (6)

The interaction occurred through a trading application (other than a trading application specific to the Punter concerned) on a desktop or laptop PC.

LiveOperator (7)

The interaction occurred with the assistance of a live operator (for example, a telebet or cash desk operative).

SelfService (8)

The interaction occurred through a self-service device.

API (9)

The interaction occurred through a bespoke trading application that directly utilises the API (that is, through a trading application that is specific to the Punter concerned).

11. Topic Index numbering. Please provide more information around the numbering used in lists of prices for selectionIds. In the initial topic load, a selection has an index that is different in subsequent updates e.g.

In the initial topic load, a particular selectionId has an index of 95.

```
TopicName = AAPI/3/E/E_1/E/E_100006/E/E_191561/E/E_4579782/M/E_11460642/MEI/MDP/10_10_0_GBP_1
MessageIdentifier =  (Unknown)
isTopicLoad = T
...
1V95-1 => 72673427
1V95-2V1-1 => 4.1
1V95-2V1-2 => 29.75
1V95-2V2-1 => 3.3
1V95-2V2-2 => 44.63
1V95-2V3-1 => 5.6
1V95-2V3-2 => 19.83
...
```

Then in a later update we get an index of 1:

```
TopicName = AAPI/3/E/E_1/E/E_100006/E/E_191561/E/E_4579782/M/E_11460642/MEI/MDP/10_10_0_GBP_1
MessageIdentifier =  (Unknown)
isTopicLoad = F
1V1-1 => 72673427
1V1-2V1-1 => 4.1
1V1-2V1-2 => 26.19
1V1-2V2-1 => 3.3
1V1-2V2-2 => 39.29
1V1-2V3-1 => 5.6
1V1-2V3-2 => 17.46
```

Is this expected?

Yes - this is expected behaviour.

Should this index using these numbers at all in our representation of the tree or are they arbitrarily numbered for each message? If it's arbitrary what is the unique identifier we should use to know which parts of the tree we should update, for all of the variable attributes?

The variable number of a parameter is unique to a single message relative to the number of items in that message. So on the initial topic load you get all selections, so 95 indicated it is the 95th selection in that message, it does not indicate that selection id 72673427 will always be the 95 element of every subsequent message.

When you get a delta message you only get informed of what has changed. So in the case above it is probable that only one selection has changed or at least the first selection you are being notified of in this message is 72673427. It does not imply any change in any other selection.

As a general rule the number immediately after a V indicates a variable number of items whereas the first number or the first number after a dash is fixed and corresponds to the attribute as set out in the specification for the topic message concerned.

In your example we know from the specification that 1v1-1 is selection Id, 72673427 uniquely identifies the selection concerned.

- 1V1-2 specifies it is back prices.
- 1V1-2V1 specifies it is the first back price in the message
- 1V1-2V1-1 specifies it is the display price
- 1V1-2V1-2 specifies the amount available to back at this price.

So to uniquely identify a price that has changed you would use a combination of selection id and display price and whether it is back or lay price.

It is important to emphasise that the only information included in the message information on actually changed. For example, in the above example if only the amount available to match @ 3.3 had changed and nothing else then the expected message would be like the following:

```
1V1-1 => 72673427
1V1-2V1-1 => 3.3
1V1-2V1-2 => 39.29
```

i.e. the prices @ 4.1, 5.6 are still valid.

Similarly if a new price is available 20.23 to be matched 3 the message would be like the following:

```
TopicName = AAPI/3/E/E_1/E/E_100006/E/E_191561/E/E_4579782/M/E_11460642/MEI/MDP/10_10_0_GBP_1
MessageIdentifier =  (Unknown)
isTopicLoad = F
1V1-1 => 72673427
1V1-2V1-1 => 3
1V1-2V1-2 => 20.23
```

If a price on a selection is no longer available we send a message with the price but no amount eg

```
TopicName = AAPI/3/E/E_1/E/E_100006/E/E_191561/E/E_4579782/M/E_11460642/MEI/MDP/10_10_0_GBP_1
MessageIdentifier =  (Unknown)
```

```
isTopicLoad = F
1V1-1 => 72673427
1V1-2V1-1 => 4.1
```

Or the above two changes could be sent together if they occurred at the same time or during the same refresh period

```
TopicName = AAPI/3/E/E_1/E/E_100006/E/E_191561/E/E_4579782/M/E_11460642/MEI/MDP/10_10_0_GBP_1
MessageIdentifier =  (Unknown)
isTopicLoad = F
1V1-1 => 72673427
1V1-2V1-1 => 3
1V1-2V1-2 => 20.23
1V1-2V2-1 => 4.1
```

12. When subscribing to some event classifiers to e.g. horse racing the implementation gets disconnected before the initial Topic Load has finished (but not consistently). This is different behaviour attempting to subscribe to Soccer which contains more markets than the account limit is currently allowed to subscribe to. In this case there is an error response telling stating MaximumSubscribedMarketsReached. Is this a known issue?

This is expected behaviour. When you issue a subscription command you will get pushed all topic loads for every topic in the subscription - if the subscription is valid/allowed based on account limits.

If the subscription is at a high enough point in the hierarchy – i.e. to get all the markets in Horse Racing - in one go then the server needs to send the initial topic load (i.e. the current state of the topic) for each market being subscribing to. This can be a substantial volume of data for the calling implementation and connection to consume (hundreds of messages in one go). If the client implementation cannot consume the data fast enough or the internet connection cannot transfer the data fast enough, the data builds up on a server side buffer and \ or buffers of intermediate network devices on the connection. Either the server or any of the intermediate devices will drop the connection if the buffer exceeds pre-configured levels or it takes too long to write data down the connection.

This is one of the reasons why GBET recommend that client implementions subscribe in a graduated manner at a rate the client and connection can sustain rather than attempt to subscriber to a large volume of markets in one go. i.e. subscribe at a rate the client connection can consume, eg 50-100 markets at a time. When the response message is then received after all the topic loads, it is possible to subscribe to the next batch of markets, in this way it is possible to protect against overloading the implementation or connection.

As regards attempt to subscribe to soccer markets from the top level Soccer event classifier, it would exceed the max allowable subscribed markets limits, in this case the client is not subscribed to anything and is sent back a single error response message with return code RC961 MaximumSubscribedMarketsReached