FIX Antenna Java Release Notes

FIX Antenna Java 2.21.0

September 04, 2019

New Features and Improvements

- Added mechanism for customization of outgoing *Logon* and *Logout* messages.
  - *FIXSession* interface is extended with a method *addOutSessionLevelMessageListener()* for registering interceptors of outgoing session-level messages.
- Added mechanism for monitoring *LastMsgSeqNumProcessed(369)* tag in incoming messages.
  - New implementations of global user handlers are added into the package (read more about LastMsgSeqNumProcessed(369) processing)
    - *LastProcessedSlowConsumerCheckerMessageHandler* controls a gap between outgoing sequences and received *LastMsgSeqNumProcessed(369)* tag value and report about the slow consumer in case of exceeding the threshold.
    - *LastProcessedSequenceSyncMessageHandler* can be used to automatically synchronize sequences based on tag 369 on incoming Logout. This implementation should be used carefully because it can lead to loss of outgoing messages.
- Secure FIX server with scheduler is included into the package.
  - Implementation of *ScheduledSSLFIXServer*, which accepts secure SSL/TLS connection and supports timetables for FIX sessions is available out of the box.

Critical bugs and fixes

- FIX session with no-queue sending mode might send a bunch of messages with higher than expected *MsgSeqNum(34)* right after reconnect.
  - Problem description: FIX session with no-queue sending mode didn't wait for the finish of sequences synchronization procedure and started to send application messages immediately after *Logon* message. It might lead to sending a bunch of messages with sequence numbers higher than expected and receiving o lot of *ResendRequests(2)* from others side.
  - Solution: Awaiting for an answer from the other side after reconnect during some reasonable period was introduced.

FIX Antenna Java 2.20.2

August 15, 2019

New Features and Improvements

- Don't print exception stack traces in case of logging level “INFO”, “WARN” or “ERROR”.
  - Changed messages with meaningful problem descriptions. Exception stack traces still can be accessed with DEBUG and TRACE logging levels.

Critical bugs and fixes

- CRITICAL. Incorrect serialization of messages, which was taken from the internal message pool.
  - Problem description: Application, which uses prepared messages, under certain circumstances corrupted messages in the pool. It led to the error and broke the connection.
  - Solution: Logic for management of messages in the pool was fixed.
- Problem with parsing iLink Session Access Secret Key file, which is used for CME Secure Logon support.
  - Problem description: File with access keys for iLink created via CME's website was missing "Environment" entry. Such file caused an error when the adapter was parsing it.
  - Solution: Parsing the iLink Session Access Secret Key was improved and made "Environment" entry optional in this file.
- FIX Engine sends redundant *Logout(5)* messages if there is no answer for *TestRequest(1)* during some reasonable time.
  - Problem description: If the application has not received any data for (*HeartBtInt(108)* + “some reasonable transmission time”) seconds, it should transmit a *TestRequest(1)* message. If there is still no *Heartbeat(0)* message answer received then the connection should be considered lost and the session should be closed. There is no reason to send *Logout(5)* message for such FIX session.
  - Solution: Recovery logic was improved and sending the redundant *Logout(5)* message was removed.
- FIX engine sends *Heartbeat(0)* message right after FIX session reconnect.
  - Problem description: FIX engine sent *Heartbeat(0)* message as a first message instead of *Logon(A)* message. It led to error on the other side and unexpected closing of the FIX session.
  - Solution: FIX Antenna Java support message queuing for outgoing messages. Session-level messages, normally, should be sent out of turn. Such queue may contain session-level messages, like *Heartbeat(0)* or *TestRequest(1)*, at the moment of a broken connection. After the reconnect they are outdated but they are still in a queue and in front of new *Logon(A)* message. To solve this issue, additional cleaning of such outdated out-of-turn messages was added right before session connect.
- FIX engine doesn't generate a warning message if receives *SequenceReset(4) (Reset)* message with *NewSeqNo(36)* value which is equal to the expected sequence number.
  - Problem description: According to FIX specification, when FIX engine receives *SequenceReset(4) (Reset)* message with *NewSeqNo(36)* = to the expected sequence number, it has to accept such reset but "generate a "warning" condition in test output". Otherwise, FIX engine doesn't write such working messages in some cases.
  - Solution: Printing a warning message to engine log was added for such cases.
- FIX engine increments the expected sequence when process *SequenceReset(4) (Reset)* message with *NewSeqNo(36)* that is less than the expected sequence number.
  - Problem description: According to FIX specification, when FIX engine receives *SequenceReset(4) (Reset)* message with *NewSeqNo(36)* < then expected sequence number, it has to reject such reset with reason *SessionRejectReason(373)* = "Value is incorrect (out of range) for this tag" and doesn't change expected sequence number. Otherwise, FIX engine accepted *SequenceReset(4) (Reset)*, sent *Reject(3)* for it but incremented expected sequence number.
  - Solution: Processing of *SequenceReset(4) (Reset)* message was improved.
- FIX engine accepts duplicates of *SequenceReset(4) (Gap Fill)* message with *MsgSeqNum(34)* that is less than the expected sequence number.
  - Problem description: According to FIX specification, when FIX engine receives duplicate *SequenceReset(4) (Gap Fill)* message with *MsgSeqNum(34)* < than expected sequence number and *PossDupFlag(43)* = “Y” it have to ignore such message. Otherwise, FIX engine accepted it and change the result expected sequence number.
  - Solution: Processing of *SequenceReset(4) (Gap Fill)* message was improved.
- Some of JMS and Kafka samples could not be started in Linus.
  - Problem description: Scripts for start samples didn't work due to syntax errors.
  - Solution: Bash scripts for running samples were fixed.

FIX Antenna Java 2.20.1

April 16, 2019

New Features and Improvements

- New methods for sequence management were introduced.
  - FIX session extended with a set of methods for separate set and get session' sequences:
    - *FIXSession.setInSeqNum(long)*
    - *FIXSession.getInSeqNum()*
    - *FIXSession.setOutSeqNum(long)*
    - *FIXSession.getOutSeqNum()*
- Improved notification about delivering messages to Kafka in Kafka Adaptor.
  - New implementation notifies about successful delivery only when Kafka confirms such operation.

FIX Antenna Java 2.20.0

March 06, 2019

New Features and Improvements

- Added JMX interface for FIX session management.
  - Implemented JMX interface allows managing FIX session with standard Java mechanism. JMX interface provides the same volume of operation for FIX Antenna as FIXICC has.
- Added Kafka adapter.
  - The new adapter for communication with Kafka is added. It provides pub/sub mechanism for Kafka topics. Package includes samples, which help to route FIX messages from FIX Session to Kafka.
- Default engine logging mechanism migrated to Log4j2 implementation.
  - Log4j2 is a low-garbage solution with a better logging performance than Log4j 1.x, better support and it has more abilities for managing the logging output.

FIX Antenna Java 2.19.7

December 18, 2019

New Features and Improvements

- The sample for CME iLink adapter was updated.
  - CME iLink sample was update to provide easier interaction with CME Autocert+ certification platform.

Critical bugs and fixes

- CRITICAL. The setting of double type may corrupt the FIX message.
  - Problem description: The problem was reproduced in parsing of prepared messages when setting double type.
    1) the sum of size of an integer part and fractional part (precision parameter) exceed the reserved space for this tag in a message;
    2) the fractional part of double values might be rounded.
  - Solution: Serialization of doubles to internal message' buffers was fixed.
- *FIXSession.setSequenceNumbers()* method doesn't change sequence number after the FIX session was disconnect.
  - Problem description: FIX Session had an issue with updating its persistent state and, therefore, in disconnected state *setSequenceNumbers()* method didn't provide the correct result.
  - Solution: the session state updating logic was fixed.
- Scheduled session uses invalid TimeZone if FIX session has to start during initialization.
  - Problem description: Scheduled session may be configured to used custom TimeZone for scheduling session' start/stop actions.But, due to the bug, scheduled session always used default local TimeZone if the session should be started during its initialization.if the session should be started during its initialization.
  - Solution: Scheduled session initialization logic was fixed to use correct TimeZone.
- FIX session with *sync_noqueue* mode sends application message before *Logon(A)* on connect.
  - Problem description: There were cases when a new connection is established FIX Antenna sent application message, which wasn't successfully sent during the previous connection session, before *Logon(A)* message.
  - Solution: Initialization of FIX sessions with *sync_noqueue* mode was fixed. User's sending threads will be blocked till the engine successfully sends *Logon(A)* message.
- *ignoreResetSeqNumFlagOnReset* setting did not work at session start for Acceptor.
  - Problem description: Even if *ignoreResetSeqNumFlagOnReset* was set to *true*, acceptor FIX session still sent *ResetSeqNumFlag(141=Y)* in answer to counterparty.
  - Solution: Processing of *ignoreResetSeqNumFlagOnReset* by acceptor was fixed.
- FIX Antenna ignores incoming *TestRequest(1)* with higher sequence number than expected.
  - Problem description: FIX Antenna ignored incoming *TestRequest(1)* with a higher sequence number than expected. As a result, counterparty never receives an answer for *TestRequest(1)* and can break the connection.
  - Solution: The order of session' handlers, which process the incoming message, was changed to process *TestRequest(1)* before the engine marks it as an out-of-order message.
- FIX Antenna writes incorrect timestamp into storage files.
  - Problem description: FIX Antenna wrote incorrect timestamp for incoming messages when option *markIncomingMessageTime* is enabled.
  - Solution: The bug with timestamp calculation was fixed.

FIX Antenna Java 2.19.4

November 12, 2018

New Features and Improvements

- A new fan-out sample is added to the package.
  - A sample, which demonstrates the principles from the article How to implement fan-out case, is added into the benchmark subfolder of the distribution package.
- Accuracy of timestamps in incoming FIX transactions logs is improved.
  - If the option *markIncomingMessageTime* is enabled, incoming log stores a timestamp of the moment FIX message was read from the socket. The default behavior takes timestamp of writing message to log file (after it was processed by an application).

Critical bugs and fixes

- Initiator session sends incorrect *BeginSeqNo(7)* in *Resend Request(2)* message in case of multiple session reconnects.
  - Problem description: Session state management error.
  - Solution: The handling of internal session state was fixed for this case.
- FIX Antenna forcibly rejects messages from internal message queue.
  - Problem description: In some edge cases (received message with too low sequence number, received invalid *Logon(A)* message), when a session is forcibly closed due to detected problems, messages are rejected from the queue even if option *enableMessageRejecting* is disabled. This may lead to losing messages in such cases.
  - Solution: Session closing logic was fixed. Messages can be rejected from the internal queue (passed to registered *RejectMessageListene* or removed from the queue) only when option *enableMessageRejecting* is enabled.
- An API call to disconnect of an auto-reconnect FIX session doesn't change session state.
  - Problem description: If *FIXSession.disconect()* method is called when the session is waiting for next reconnect try, session stops attemps to reconnect but doesn't change its state from *RECONNECTING* to *DISCONNECTED*. It may be confused for custom code, which handles the state of a session.
    Solution: Switching session to *DISCONNECTED* state was implemented for the described case.
- CME iLink adapter sends *Logout(5)* for invalid incoming *Logon(A)* message.
  - Problem description: During passing CME Autocert+ tests with FIX Antenna Java CME iLink adapter it was discovered that CME doesn't expect any answer for invalid *Logon(A)* message. CME iLink adapter inherits processing of invalid incoming *Logon(A)* message from FIX Antenna Java engine and sends *Logout(5)* before disconnect. Such behavior is unexpected by CME and fails some Autocert+ tests.

    - Solution: New session handler *ILinkQuietLogonModeHandler* was implemented and integrated into the processing chain to prevent sending *Logout(5)* for invalid incoming *Logon(A)*.
- *FIX Integrated Control Center* doesn't display correct timezone for FIX Antenna Java engine instance.
    - Problem description: FIX Antenna Java doesn't provide information about timezone for *FIX Integrated Control Center*.
    - Solution: Administrative module sends timezone in 10003 tag of outgoing *Logon(A)* message for administrative connection.
- *FIX Integrated Control Center* doesn't display *SessionQualifier* tag information for FIX Antenna sessions.
    - Problem description: FIX Antenna doesn't process *SessionQualifier* tag, which *FIXICC* sends in session create command.
    - Solution: Implemented processing of SessionQualifier tag for the administrative module of FIX Antenna Java.
- Setting of 'Active Connection' property on *FIXICC* UI form is not working.

    - Problem description: *FIXICC* UI form for creating FIX session has an option '*Active Connection*' for connection settings. This option allows to define which connection (backup or primary) will be used during session start. But FIX Antenna Java ignores this option when creates new FIX session via *FIXICC*.
    - Solution: The processing of '*Active Connection*' property was restored in the administrative module of FIX Antenna Java.

FIX Antenna Java 2.19.3

August 22, 2018

New Features and Improvements

- Two new examples were added.
    - EchoServer and ConnectToGateway examples demonstrate acceptor and initiator sides and connection between them.
- Option "resetQueueOnLowSequence" was added.
    - It turns off/on the resetting of session's outgoing queue when a client connects with sequence that is low than expected. Outgoing queue is cleaned by default.
- Option "disconnectOnLogonHeartbeatMismatch" was added.

    - Checking and disconnecting session if logon answer contains different from configured value of HeartBtInt(108). Client will be disconnected by default if they use different from configured interval.

Critical bugs and fixes

- FIXServer did not accept configured sessions.
    - Problem description: FIXServer could not match a configured session in case when session name is not anything but sender-target pair.
    - Solution: Session matching logic was fixed.
- setSequenceNumbers did not work when session was not in "connected" state.
    - Problem description: sequences were not set if method setSequenceNumbers was called before connecting a session.
    - Solution: Method behavior was fixed.
- Option "markIncomingMessageTime" tracked incorrect time for incoming messages.
    - Problem description: FIXMessageChopper (validateGarbledMessage=true) set last read message time for the incoming messages at the end of reading the previous ones.
      Solution: Timestamp is set just before message read from transport.
- FIX Antenna's Connection was broken during long resend request processing.
    - Problem description: FIX Antenna did not read heartbeats while servicing long resend requests.
    - Solution: The re-sending logic was fixed. Test requests are not sent while processing resend request.
- Working with a large number of messages (> 100M) in MMF indexed storage.
    - Problem description: MMF indexed storage could not process indexes for more than 100 millions messages.
    - Solution: Incorrect opening of index file was fixed.
- Indexes in MMF storage did not work without backup in case of sequence reset.
    - Problem description: MMF storage did not properly index messages in case of sequence reset.
    - Solution: Index logic of MMF storage was changed to track all messages including sequence reset case.
- The size of MMF storage file was incorrectly set.
    - Problem description: MMF storage increased its file size after each opening.
    - Solution: Initialization logic was fixed.
- FIXICC did not show parameters of the sessions.
    - Problem description: Admin module in fix antenna java could not find session parameters in its list.
    - Solution: Session searching logic in admin module was fixed.
- Internal object pool generated garbage.
    - Problem description: Returned objects were not returned into the pool.
    - Solution: Logic for returning objects into the pool was fixed.
- Erroneous error message in the log ("Unable to connect with attempt: 0 while maxAttempts: -1")

    - Problem description: Erroneous error message was in the log after unsuccessful connection attempts even if autoreconnect mode was off.
    - Solution: Logging of the error was disabled if autoreconnect mode was off.

FIX Antenna Java 2.19.2

June 29, 2018

New Features and Improvements

- Optimized serialization of FIX messages.
    - Implemented lightweight *MsgType(35)* validation in message serialization process without extracting its tag value.

Critical bugs and fixes

- Fixed validation of *HeartBtInt(108)* in incoming *Logon(A)* message by initiator session.
    - Problem description: Initiator FIX session successfully accepts *Logon(A)* answer with tag value for *HeartBtInt(108)* which is different from the defined for the session. But, according to FIX specification, "*The HeartBtInt value should be agreed upon by the two firms and specified by the Logon initiator and echoed back by the Logon acceptor*".
    - Solution: Improved validation of incoming *HeartBtInt(108)* tag value. If *Logon(A)* answer contains another value that it was defined for the session, *Reject(3)* message with reason "Incorrect or undefined heartbeat interval" will be sent back.

FIX Antenna Java 2.19.1

May 22, 2018

New Features and Improvements

- Optimized memory usage by *FIXFieldList*.
    - *FIXFieldList* produced some garbage during its serialization. Such behavior was introduced by improvements from FIX Antenna 2.17.4 version. *FIXFieldList* was refactored to reuse allocated memory.
- Added API for TCP transport customization.
    - Added *SocketContextFactory* interface for customization sockets of initiator and acceptor sessions. Implementation of this interface may be introduced for FIX Antenna as socketContextFactory configuration option.

Critical bugs and fixes

- Fixed parsing of double values.
    - Problem description: *FIXFieldList.getTagValueAsDouble()* returned invalid precision for some values. For example, method returned 99.6 719999998 double for tag value "44=99.672".
    - Solution: parsing algorithm for double values was fixed.
- Restored access to protected members for storage classes.
    - Problem description: Obfuscator hid protected members of classes from API in com.epam.fixengine.storage package.
    - Solution: Access to protected members is restored.

FIX Antenna Java 2.19.0

April 26, 2018

New Features and Improvements

- CME Secure logon API support.
    - CME Secure Logon support has been added. Refer to How to connect to CME Globex using Secure Logon API for more information.
- Getting BigDecimal from FIXFieldList.
    - Method getTagValueAsBigDecimal has been added in FIXFieldList. It is used for getting decimal numbers with high precision.

FIX Antenna Java 2.18.3

April 18, 2018

Critical bugs and fixes

- Fixed auto reconnection bug.
    - Problem description: FIX engine didn't do pauses between reconnection tries and ignored value of 'autoreconnectDelayInMs' property. This autoreconnection behavior was corrupted started from implementation of 'Round-robin reconnect option to multiple backup destinations' feature in 2.18.1 release. But it was possible to use timeout value from 'connectionSwitchDelayInMs' property instead of 'autoreconnectDelayInMs' with the same result.
    - Solution: Fixed autoreconnect logic to use 'autoreconnectDelayInMs' value as timeout period. 'connectionSwitchDelayInMs' property is deprecated.
- Fixed ResendRequest processing issue with in-memory storage.
    - Problem description: Initiator FIX session sent Sequence Reset - Gap Fill(4) message with lower NewSeqNum(36) than it had during initial sequence reconciliation. As a result, acceptor had to send one additional ResendRequest to synchronize sequences.
    - Solution: Logic for Resend Request processing was fixed.
- Fixed deadlock on FIX session disconnect/dispose
    - Problem description: We found a possibility of deadlock in a case of simultaneous shutdown FIX session from different threads. It may manifest when the user tries to close the session, which should be also closed due to internal reasons(invalid sequences, missing heartbeats, etc.).
    - Solution: Deadlock is resolved.
- Fix to avoid removal of ApplVerID(1128) tag during serialization
    - Problem description: FIXSession.sendWithChanges() method with ChangesType.DELETE_AND_ADD_SMH_AND_SMT mode removed SMH fields and there is no way to send to counterparty tags like ApplVerID(1128) or MessageEncoding(347).
    - Solution: Serialization is improved and such send call removes or replaces only tags, which are directly related to routing and may be filled by the engine itself (BeginString(8), BodyLength(9), MessageSeqNum(34), SendingTime(52), SenderCompID(49), TargetCompID (56), SenderLocationID(142), TargetLocationID(143), SenderSubID(50), TargetSubID(57), LastMsgSeqNumProcessed(369), CheckSum (10)).

FIX Antenna Java 2.18.2

March 28, 2018

Critical bugs and fixes

- Fixed NIO transport problem.
    - Problem description: FIX engine can't instantiate session with NIO transport due to ClassCastException. The problem was introduced with the latest changes.
    - Solution: Inheritance of transport classes was restored.

FIX Antenna Java 2.18.1

March 23, 2018

New Features and Improvements

- Slow consumer detection and notification is implemented.
    - *FIXSessionSlowConsumerListener* callback for *FIXSession* was implemented and several configuration's options to control notifications about slow consumers were added.
- Throttling controller is implemented for FIXSession.
    - With enabled throttling option engine counts a number of messages for a given message with defined message type (e.g. NewOrder) during a specified period of time and disconnects session with the reason THROTTLING if the set threshold is exceeded.
- Chronicle Queue bases storage for FIX Session.
    - Implemented new type of persistent message storage for FIX Session (*ChronicleStorageFactory*), which is based on OpenHFT Chronicle Queue (https://github.com/OpenHFT/Chronicle-Queue) implementation. Chronicle Queue is a project which aims to build a persisted low-latency GC-free solutions for high performance and critical applications.
- Implemented new synchronous sending algorithm - *sync_noqueue*.
    - The current sending mechanism (*sync*), used by default in FIX antenna, is designed for low latency and consistency operations. It attempts to send messages synchronously most of the time.
      It also allows queueing messages if the session is disconnected and it can be switched to asynchronous message sending mode to handle messages in the queue.
      The new *sync_noqueue* mode provides a strict synchronous mechanism as an option. If this sending mode is enabled, the session sends messages only synchronously, in the same thread. This mode does not support queuing(e.g., it will be impossible to pass messages to the disconnected session)
- Round-robin reconnect option to multiple backup destinations for an initiator session has been added.
    - Option to switch to a backup destination, selecting from the list of available servers in round-robin fashion has been added to session initiator.
- Added new API to create Acceptor session in a declarative form
    - Using this API an acceptor session will be created and later on initialized when a connection arrives (#createAcceptorSession())
- Missed sequence reset detection was implemented in this version.
    - FIX Antenna can automatically handle cases when counterparty ignores sequence reset and continues to send old sequence numbers. Such behavior (when FIX Antenna resets its sequences but counterparty does not) in general can lead to requesting a lot of old messages due to gap detection in sequences.This feature was added to mitigate these cases. (check resetThreshold configuration option for details)
- Notification about sequence synchronization was extended.
    - *FIXSessionOutOfSyncListener* API was extended with new methods to get more control over sequence gap detection and ResetRequest handling procedures.
- Optionally ignore compID validation for FIX messages.
    - Implemented optional switching off validation of *SenderCompId(49)* and *TargetCompID(56)* fields in FIX message's header. This feature may be used for message routing purposes. (check senderTargetIdConsistencyCheck configuration option for details)
- Updated MMF storage implementation to conform Java 8 codebase.
    - Deprecated SDK calls inside MMF storage implementation were updated with a more flexible solution.

Critical bugs and fixes

- Fixed algorithms for asynchronous sending.
    - Problem description: *FIXSession.sendWithChanges()* method ignored *optionMark* parameter and sent messages always synchronous.
    - Solution: Fixed sending logic for sendWithChanges method.
- NPE in RG API was fixed.
    - Problem description: RG API produced NPE exception for incorrect FIX messages when counter tag of the repeating group inside the FIX message had a lower value than the reals number of entries.
    - Solution: Forced validation for counter tags was added. The *InvalidLeadingTagValueException* will be thrown if the repeating group has an invalid counter tag.

FIX Antenna Java 2.17.11

February 14, 2018

Critical bugs and fixes

- Prepared messages format incorrectly negative double values.
    - Problem description: Prepared message tries to use preallocated space for values. If a numeric value uses less space then was reserved, a prepared message added '0' before the value to get the required size. But, in case of negative doubles, such trailing zeros were added before minus sign (00000-0.05).
    - Solution: Formatting of negative values was fixed.

FIX Antenna Java 2.17.10

February 08, 2018

New Features and Improvements

- Added feature to set send and receive socket buffer sizes for TCP transport.
    - Added API methods in transport and configuration options ("tcpSendBufferSize" and "tcpReceiveBufferSize") to set TCP socket buffers.

Critical bugs and fixes

- Fixed deadlock in an object pool for RG API objects.
    - Problem description: RG API reuses instances of RepeatingGroup and Entry classes to reduce garbage producing. These instances are organized in object pools. Due to an error in pool implementation, reusing of FIX messages with embedded repeating groups with concurrent threads may lead to deadlock.
    - Solution: Object pool implementation was fixed and tested in a concurrent environment.
- Fixed problem with sending Logout message in response to receiving Logon message with lower than expected sequence number.
    - Problem description: In case of receiving Logon message with lower than expected sequence number, acceptor doesn't send back Logout with the description of a problem. This problem was introduced by a fix in 2.15 branch for processing corrupted Logon messages.
    - Solution: Fixed acceproor's behavior.
- Fixed clone method in SessionParameter class.
    - Problem description: Changes in a cloned copy of SessionParameter object affects senderCopmId and targetCompId properties of an original object.
    - Solution: Fixed by copying of all internal objects.

FIX Antenna Java 2.17.9

January 18, 2018

Critical bugs and fixes

- Fixed processing of errors during synchronous message sending.
    - Problem description: IO error during storing a message to an outgoing storage may lead to losing the message without appropriate notification of customer code. This problem is actual for synchronous sending mode.
    - Solution: If it's impossible to store a message to outgoing storage, the message is added to internal session's outgoing queue for asynchronous delivering later. In case of problems with adding to the queue, RuntimeException is thrown to customers code.
- Fixed validation of duplicated fields inside FIX message.
    - Problem description: Sequence of repeating groups in FIX message, which follow one by one, may lead to improperly failing validation of duplicated fields in the message.
    - Solution: Fixed processing of repeating groups during validation.

FIX Antenna Java 2.17.8

December 28, 2017

New Features and Improvements

- License validation process changes.
    - Added closing of FIX session in case of receiving restricted by license message.

Critical bugs and fixes

- Fixed parsing of repeating groups with zero in leading tag value.
    - Problem description: RG API throws exception if it tries to index leading tag of repeating group with '0' value.
    - Solution: Fixed indexing of repeating groups.
- Fixed loading of stored session configuration on session creation.

- Problem description: FIX session loads its persistent runtime state on initialization/connection stage. But sometimes, loaded session runtime information might be necessary to change after creating session and before the connection.
- Solution: Added preload of runtime configuration on session creation.

FIX Antenna Java 2.17.7

December 13, 2017

New Features and Improvements

- Custom configuration for *SSLFIXServer*.
  - Added new constructor with *Configuration* object as parameter. It allows creating secure FIX server with custom configuration.
- *FIXVersionContainer* instance using only dictionary file.
  - Added new builder within FIXVersionContainer class. It allows building and initialization of *FIXVersionContainer* instance only with FIX dictionary file. It simplifies customization of FIX sessions.

Critical bugs and fixes

- Fixed processing of *NextExpectedSeqNum(789)* tag in Logon.
  - Problem description: Counterparty sends *Logon(A)* message with *NextExpectedSeqNum(789)* tag lower than the actual session's outgoing sequence number. FIX Antenna's acceptor session sent *SequenceReset(4)* message with invalid lower *NewSeqNo(36)* tag during processing such *Logon(A)*.
  - Solution: Improved processing of *NextExpectedSeqNum(789)* for acceptor FIX session.
- Fixed the mechanism that processes incoming TCP connections.
  - Problem description: Queuing of incoming TCP connections and discarding them by timeout. Multiple clients establishing TCP connections simultaneously (with some clients not sending *Logon(A)*).
  - Solution: Changed implementation of threadpool, which services handlers for incoming connections.

FIX Antenna Java 2.17.6

November 28, 2017

Critical bugs and fixes

- Fixed parsing of high-precision timestamps with partial fractions of seconds for *SendingTime(52)* and *OriginalSendingTime(122) tags*.
  - Problem description: While generic parser, which is used for accessing timestamps in clients code was fixed in FIXAJ 2.17.5, system handlers still used old parsing mechanism and timestamps with partial fractions of seconds in *SendingTime(52)* and *OriginalSendingTime (122)* which led to errors.
  - Solution: System handlers were updated to use new fixed parsing mechanism for timestamps.

FIX Antenna Java 2.17.5

November 24, 2017

Critical bugs and fixes

- Fixed parsing of high-precision timestamps with partial fractions of seconds.
  - Problem description: Some vendors may send timestamps with partial fractions of seconds, for example, 12:30:30.5 (only 1 digit for milliseconds instead of 3). Such timestamps led to parsing errors.
  - Solution: Parser for timestamps was improved to correctly process such timestamps.

FIX Antenna Java 2.17.4

November 15, 2017

Critical bugs and fixes

- Fixed high-precision timestamps in storages.
  - Problem description: High-precision timestamps in outgoing storage were invalid and greater than *SendingTime(52)* of stored messages for 1-2 seconds.
  - Solution: ode, which calculated micro and nanoseconds timestamps for storages, was fixed.
- Fixed serialization of FIX messages when using *FIXSession#sendWithChanges()* method.
  - Problem description: After fix in 2.17.3, *FIXSession#sendWithChanges()* method call for synchronous sending led to removing *SMH* and *SMT* tags in original message.
  - Solution: Synchronous serialization of messages was improved. *SMH* and *SMT* may be updated with actual sending values or left unchanged depending on ChangesType parameter.
- Improved processing of T*estReques(1)* messages.

    - Problem description: If counterparty sends no messages during heartbeat interval, FIX engine, according to FIX protocol, sends *TestRequest(1)*. Then it waits for another heartbeat interval for response (*Hearbeat(0)* message with *TestReqId(112)* tag). If response is not received, engine closes session unconditionally, even if communication is restored and counterparty starts sending other messages.
    - Solution: Strict condition for waiting *TestRequest(1)* response was updated and any incoming message cancels countdown counter now.
- Restored processing of *NextExpectedSeqNum(789)* tag in Logon.
    - Problem description: FIXAJ ignored *NextExpectedSeqNum(789)* tag in Logon(A) messages.
    - Solution: Restored logic for *handleSeqNumAtLogon* option. This option allows to handle *NextExpectedSeqNum(789)* tag in incoming *Logon(A)* and sets this tag for outgoing *Logon(A)* message for sessions with FIX 4.4 protocol and above.

FIX Antenna Java 2.17.3

November 2, 2017

Critical bugs and fixes

- Fixed FIXSession#sendWithChanges() method and made it possible to send messages synchronously.
    - Problem description: FIXSession#sendWithChanges() always sent messages asynchronously, even when preferredSendingMode option was set to sync.
    - Solution: Improved behaviour of FIXSession#sendWithChanges() method.

FIX Antenna Java 2.17.2

October 26, 2017

Critical bugs and fixes

- Fixed parsing of FIX messages with nested repeating groups for RG API.
    - Problem description: There was ArrayIndexOutOfBoundsException exception during a parsing of FIX message with nested repeating groups (with 3 nested levels). Problem was related to Repeating Group API functional.
    - Solution: Resizing strategy for stuctures used for parsing repeating groups was fixed.

FIX Antenna Java 2.17.1

September 13, 2017

Critical bugs and fixes

- Fixed exception raised when updating String fields in prepared message
    - Problem description: There was invalid field value in the message after the series of its updates with shorter and longer strings.
    - Solution: Fixed methods for updating character values

FIX Antenna Java 2.17.0

September 1, 2017

New Features and Improvements

- Implemented options to configure CPU affinity.
    - New session's configuration parameters recvCpuAffinity, sendCpuAffinity and cpuAffinity allow pinning session's thread to selected CPU.

Critical bugs and fixes

- Fixed exception during the update of String fields in prepared message.
    - Problem description: There was IndexOutOfBoundsException during an update of the field in prepared message. The exception occurred in the case when new value had lower size than the old one.
    - Solution: Fixed methods for character values update.

FIX Antenna Java 2.16.5

January 10, 2018

Critical bugs and fixes

- Fixed validation of duplicated fields inside FIX message.
    - Problem description: Sequence of repeating groups in FIX message, which follow one by one, may lead to improperly failing validation of duplicated fields in the message.
    - Solution: Fixed processing of repeating groups during validation.

FIX Antenna Java 2.16.4

August 18, 2017

New Features and Improvements

- Obfuscation has been disabled for default global and per-type message handlers.
    - lients become able to extend default logic and build customized solution without braking down default behaviour.
- Scripts for roundtrip benchmark has been splited to run client's and server's parts separatelly.
    - To get the real result it better to run different parts of this benchmark on different boxes. New running scripts allows to do this easier.

Critical bugs and fixes

- Improved accepting several connections simultaneously
    - Problem description: In case of large amount of slow connections (means that they send Logon(A) with some delay after establishing connection) some of them can be discarded. FIX server processed all connections in single thread and, by default configuration, it can wait up to 5 second for Logon for each connection. It was lead to queuing new incomming connections and even to their discardic by timeout.
    - Solution: Processing of incomming connections were reorganized to work with Executors.

FIX Antenna Java 2.16.3

August 4, 2017

New Features and Improvements

- Scripts for installing FIXAJ libraries into client's local repository has been added to package
    - Added Maven build configuration script for deploying FIX Antenna libraries into client's local repository
    - Added description for maven dependency section with FIX Antenna libraries (can be copied to client's project configuration after importing artifacts)
- Round trip benchmark has been added
    - The new benchmark samples, which used for measuring latest results, were added to package
- An ability to leverage Java 8 notation for functional interface to load handlers for message processing in FIX sessions
    - FIXAJ used only java reflection to load messages handlers. Such strategy was a problem to link custom client's handlers with other application infrastructure (Spring integration for example). Custom loader, for example, allows to extract the existing handler from application context.

Critical bugs and fixes

- Fixed FIX session message reading mechanism for sessions with disabled validateGarbledMessage option.
    - Problem description: Sometimes, during reading big messages (3K and above) by FIX session with disabled validateGarbledMessage option, massage had tags with wrong length.
    - Solution: Fixed indexes in message reader during remapping of internal buffer.

FIX Antenna Java 2.16.2

July 20, 2017

Critical bugs and fixes

- Fixed API backward compatibility
    - Fixed backward compatibility in API for building timestamps for storages

FIX Antenna Java 2.16.1

July 11, 2017

Critical bugs and fixes

- Fixed messages serialization with *ChangesType.UPDATE_SMH_AND_SMT_EXCEPT_COMPIDS*
    - Updated behavior: *SubIds* and *LocationIds*, configured in session parameters, will be added to message only of there is no correcponding *CompId*

FIX Antenna Java 2.16.0

July 5, 2017

MIFID II support

- Timestamps with increased precision can be validated in incoming messages (in UTCTimestamp, UTCTimeOnly, TZTimestamp, TZTimeOnly)
- Timestamps in *SendingTime* (52), *OrigSendingTime* (122) are filled with configurable precision (*timestampsPrecisionInTags* option)
- Public API is extended to operate timestamps with additional precision
- Timestamps in message storages are filled with configurable precision (*timestampsPrecisionInLogs*, *backupTimestampsPrecision* options)

For more MIFID II compliance details see https://www.b2bits.com/regulatory/mifid-ii.html.

Migrated source code to Java 8

- Now FIXAJ libraries are built with Java 8. Previous version of JRE may not be supported.

FIX Message API and RG API improvements

- Allowed to parse RG with custom dictionaries
  - Public API is extended to use *FIXVersionContainer* as well as *FIXVersion* for indexing repeating groups inside the message
- Optimized prepared messages
  - Problem description: Prepared messages didn't handle string values in optimal way – it splitted internal buffer for every string tag
  - Solution: Leave internal buffer unsplitted till the new string values length is smaller than reserved in buffer and fill the string end with spaces
- Allowed to get the FIX version for read message
  - Added fixVersion property to FIXFielList class. This property is filled with session's FIX version to allow indexing repeating groups with correct dictionary

Logging improvements

- Improve warn message logging
- Updated logged license description (contacts)
- Changed IgnoreMessageHandler to warn rejected messages
- Changed the logging level for duplicated incoming sessions from ERROR to WARN

JMS Adapter

- Fixed JMS Adaptor reconnection
- Changed ILinkRejectMessageHandler to warn rejected messages

Other improvements

- Updated internal security (license) mechanism
- FIX dictionaries were updated to 1.5.48
- Allow to throw *IOException* by initiator for *FIXSession.connect()*
  - Problem description: Autoreconnect initiator session not always notify called about connection problems
  - Solution: Fixed connect logic – now all *IOException* will be thrown up to caller
- Initialization of *InMemoryStorageFactory* was fixed
  - Problem description: Warning was printed to log during *InMemoryStorageFactory* initialization. Engine tries to initialize storage with constructor and pass its configuration as a parameter to constructor. If storage doesn't have parametrized constructor, the warning is printed to log.
  - Solution: Parametrized constructor as added to *InMemoryStorageFactory* class.
- Made *FIXFieldListWithTypeFactory* visible and able to use
  - Problem description: Reusing of Queue implementations sometimes requires access to *FIXFieldListWithTypeFactory*
  - Solution: Removed obfuscation for *FIXFieldListWithTypeFactory* class
- Suppress sending session qualifier tag in *Logon*(A) message
  - Problem description: *SessionParameters.sessionQualifier* property sent in *Logon*(A) message if defined as 9012 tag by default (*logonMessageSessionQualifierTag* property).
  - Solution: New '*suppressSessionQualifierTagInLogonMessage*' property as added to suppress sending session qualifier tag in *Logon*(A) message if this property set to true.
- Improved processing of connection abrupt during sending message
  - Problem description: When connection abrupt during sending message, this message may be lost.
  - Solution: In case of connection abruption during sending message, engine throws *TransportMessagesNotSentException* with message content to notify caller about the possible problems.
- Added possibility to process sequence gap open/close events and *ResendRequest* processing events
  - *FIXSessionOutOfSyncListener* was added to *ExtendedFIXSession* API
- Allowed to prevent sending of *ResetSeqNumFlag*(141) in *Logon*(A) message after resetting sequences
  - Added new configuration option *IGNORE_RESET_SEQ_NUM_FLAG_ON_RESET*

FIX Message API and RG API bugfixes

- Fixed problem with RG API after adding/removing tags before RG
  - Problem description: Accessing to RG entries throwed *IllegalArgumentException* after removing or adding tags before RG.
  - Solution: Fixed operations with internal RG index.
- Fixed NPE in RG API
  - Problem description: NPE occurred during reusing the instance of *FIXFieldList*.
  - Solution: Fixed initialization procedure for internal RG index
- Fixed message deep copy operation
  - Problem description: There was NPE during accessing RG in cloned message with indexed repeating groups
  - Problem description: There was NPE during cloning message with invalidated RG index
  - Solution: Fixed deep copy operation for FIX message

- Fixed FIXFieldList reinitialization
  - Problem description: Sometimes after reusing of *FIXFieldList* instance, it contained wrong RG index.
  - Solution: Fixed *FIXFieldList.clear()* method to clean RG index too
- Fixed parsing and validating RG with nested groups
  - Problem description: There was exception during indexing RGs with nested groups
  - Solution: Indexing of nested repeating groups was fixed
- Fixed removing tags from repeating groups
- Fixed problems in *FIXFieldList* with creating/removing RG
  - Problem description: There was exception during removing just added empty repeating group
  - Problem description: There was exception with adding again just removed empty repeating group
  - Solution: Fixed internal RG index
- Fixed problem with caching FIXField instances within *FIXFieldList*
  - Problem description: There was invalid *FIXField* objects for removed tags after removing whole repeating group via RG API
  - Solution: Fixed removing tags operations to update the *FIXFied* cache

Scheduller bugfixes

- Fixed session's scheduler to use correct time zone
  - Problem description: Scheduler always used default Time Zone for its tasks.
  - Solution: Fixed scheduler to use Time Zone from *ScheduledSessionParameters*.
- Added method for closing scheduller
  - Problem description: *ScheduledFIXServer.stop()* didn't stop the scheduler thread
  - Solution: *ScheduledFIXServer.stop()* has been updated to stop the scheduler too

Oher critical bugs and fixes

- Fix messages serialization with *ChangesType.UPDATE_SMH_AND_SMT_EXCEPT_COMPIDS*
  - Added *EncryptMethod*(98) for *Logon*(A)
  - Fixed *BodyLength* (9) value
- Fixed garbled messages parsing
  - Problem description: Extra garbage character after FIX message in stream led to *IOException* and invalid reader state
  - Solution: Fixed parser to skip extra chanracters as a garbage
  - Problem description: Empty *MsgSeqNum*(34) tag value in incoming *Logon*(A) message led to unexpected exception.
  - Solution: Added validation for empty value and marking such message as garbled
- Fixed reinit sequence numbers after session's initialization
- Fixed transport closing on session's abnormal disconnect
  - Problem description: During abnormal disconnect due to error, session's transport was left opened and this led to writing queued message to it by pumper thread and to losing them.
  - Solution: Added closing transport on abnormal disconnect
- Fixed to ignore buffered messages with lower seq numbers after resend request fill gap
  - Problem description: In case of processing sequence gap for incoming messages, new messages may be buffered by message reader. In case of processing several gaps one-by-one, answer for ResendRequest may include buffered messages and such messages in buffer should be skipped from further processing.
  - Solution: Added extra condition for processing buffered messages in message reader to skip processed messages
- Fixed NPE in TCPTransport
  - Problem description: NPE is thrown when case when accessing address and port properties for closed transport
  - Solution: Updated methods to return configured host and port instead the runtime socket parameters

FIX Antenna Java 2.12.29

FIX Antenna Java 2.12.29 released on 2 June, 2015.

Bugfixes

- Potential conflict causing duplication sequence numbers (if reset defined for session in configuration and called explicitly) was resolved.

  Solution**:** Internal flag was added for control and prevent second sequence reset.

- Issue with sending corrupted messages was fixed.

```
1  FIXFieldList orderMsg = newFIXFieldList();
2  orderMsg.addTag(11, "123456789");
3  orderMsg.addTag(55, "IBM");
4  //update in arena storage
5  //original message is “11=123456789 55=IBM”
6  orderMsg.set(11, "123456789");
7
8  //but copy is coruupted: “11=12345678955=”
9  FIXFieldList copy = orderMsg.deepClone(true, false);
10
```

  Solution: Internal message structure was fixed and new verification test was added.
- Obfuscation from QuietLogonModeHandler was removed.

  Solution: This class was excluded from obfuscation procedure.

FIX Antenna Java 2.12.28

FIX Antenna Java 2.12.29 released on 15 May, 2015.

Bugfixes

- Backup file overwriting issue was fixed.
  Problem description: Backup file name contains timestamp with seconds. If the backup procedure processed twice at the same second, it may lead to overwriting of existing backup file (same name with the same timestamp). For example, duplicate call for reset sequences within the same second may lead to overwriting of new backup file.
  Solution: Verification for uniqueness of file name during creating backup. New unique index to backup file name in case of duplication was added.

- Issue with unnecessary backup file creation in some specific cases was fixed.
  Problem description: If the backup procedure was called for session at the moment when storage files are empty (there were no stored messages yet), then empty files will be backuped. For example, duplicate sequence reset call within short period of time may lead to copying of empty files.
  Solution: Check for the file size during backup of storage files was added.