Configuring FIX endpoints

This section describes how to configure FIX Engine and FIX session.

Configuring FIX Engine

The FIX Engine is based on the FIX Antenna Java library.

FIX Engine's basic settings are stored in the **fixengine.properties** file, located in the **conf** directory. Refer to the B2BITS FIX Antenna Java Programmer's Guide for a complete list of FIX Antenna settings.

The **fixengine.properties** file contains global B2BITS FIX Antenna settings. A user can specify FIX session default parameters here as well.

The **fixedge.properties** file contains settings of the default FIX server port by which the FIX Antenna listens to incoming data.

The currently used default: `server.port=8911`

Additional listening ports could be defined within FIX session configurations (see the `port` property). In this case, the FIX Engine automatically manages their opening and closing according to started sessions.

Custom FIX dictionaries setup

The custom FIX dictionaries setup in the FIXEdge Java (FEJ) product is based on the FIX Antenna custom FIX dictionaries setup (see How to set up custom FIX dictionaries), but there is FIXEdge Java configuration specific.

Follow these steps to set up FIX dictionaries:

1. Place the `customFIXVersions` section into the default **fixengine.properties** file (located in the **conf** directory) to set up custom FIX version info for all FIX sessions.

fixengine.properties

```
# comma separated list of custom FIX dictionary aliases
customFixVersions=FIX44Custom,FIX50Custom

# pair of 'fixVersion' and 'fileName' for each FIX dictionary alias with pattern:
# customFixVersion.<custom FIX version alias>.fixVersion=<base standard FIX version>
# customFixVersion.<custom FIX version alias>.fileName=<custom FIX dictionary file name>

# example of custom FIX dictionary based on FIX.4.4
customFixVersion.FIX44Custom.fixVersion=FIX.4.4
customFixVersion.FIX44Custom.fileName=classpath:fixdic44-custom.xml

# examples of custom FIX dictionary based on FIX.5.0
customFixVersion.FIX50Custom.fixVersion=FIX.5.0
customFixVersion.FIX50Custom.fileName=classpath:fixdic50-custom.xml
```

2. Set the defined dictionary aliases as session's `fixVersion` or `appVersion` in the appropriate FIXEdge Java session configuration file '**s_fix_[SESSION_ID].properties**':

s_fix_session1.properties

```
# FIXEdge Java session configuration properties

fixVersion=FIX44Custom
```

s_fix_session2.properties

```
# FIXEdge Java session configuration properties

fixVersion=FIX50Custom
```

NOTE: Changes, which are made in the **fixengine.properties** file, will be applied only after FIXEdge server restart.

Configuring FIX Session

For establishing and managing FIX sessions, the FIX Antenna Java library is used. FEJ also introduces a per-file base configuration for each FIX session. All such configuration files are placed into subdirectories of the **conf/session** directory and should follow the **s_fix_[SESSION_ID].properties** mask (this behavior can be changed with the `sessionConfigManager` bean into **sysconf/fej-server.xml**). Additionally, each subdirectory may have the **s_fixDefault.properties** configuration file which defines a common option for all groups of the FIX session.

To add a new FIX session, the **s_fix_[SESSION_ID].properties** file is used with new session settings.

To modify the FIX session, the changes to the **s_fix_[SESSION_ID].properties** file should be done.

To apply changes to the FIX session, the following options are available:

- Restart the FIXEdge Java server by using the administrative instruments (all sessions will be initialized in this case)
- Load or reload the session configuration from the FIXICC app (refer to the FIXICC & FEJ Integration User Guide)

NOTE: Changes, which are made in **fixengine.properties** will be applied to the FIX Engine and sessions only after the FIXEdge server restart.

For monitoring available sessions and their states, use the administrative tools (SSH, JMX), or FIXICC.

FIX session configuration options

(You can find the advanced options for the FIX session on the FIX Antenna Java configuration page).

Property name	Default value	Required for session initiator	Required for session acceptor	Description
sessionType	acceptor	Yes	Yes	Session type. If type is not defined, then the session will be resolved as an acceptor. Valid values: acceptor/initiator.
host		Yes		The connecting host for the initiator session
port		Yes	Yes, if SSL is enabled	The connecting port for initiator session or listening port for acceptor session. If the port property is defined for the acceptor, the FIXEdge server tries to open this port during the acceptor session start and close during its stop. The FIXEdge server accepts such a session only on the defined port. The same port may be defined for several acceptor sessions. All of these sessions should contain the same value for the 'enableSSL' property.
senderCompID		Yes	Yes	The assigned value used to identify a firm that sends the message
senderSubID				The assigned value is used to identify a specific message originator (desk, trader, etc.)
senderLocationID				The assigned value used to identify a specific message originator's location (i.e. geographic location and /or desk, trader)
targetCompID		Yes	Yes	The assigned value used to identify a receiving firm
targetSubID				The assigned value used to identify a specific individual or a unit intended to receive the message
targetLocationID				The assigned value used to identify a specific message destination's location (i.e. geographic location and /or desk, trader)
fixVersion		Yes	Yes	A version of the FIX protocol, version of the transport protocol in case of FIX 5.0-FIX 5.0 SP2 (FIXT1.1) or a custom version (see How to set up custom FIX dictionaries) FIX.4.0, FIX.4.1, FIX.4.2, FIX.4.3, FIX.4.4, FIXT.1.1
appVersion				A version of the application-level protocol or a custom version (see How to set up custom FIX dictionaries) Valid values: FIX.4.0, FIX.4.1, FIX.4.2, FIX.4.3, FIX.4.4, FIX.5.0, FIX.5.0SP1, FIX.5.0SP2
backupHost				Backup host for initiator session
backupPort				Backup port for initiator session
incomingSequenceNumber	0			An initial incoming sequence number
outgoingSequenceNumber	0			An initial outgoing sequence number
heartbeatInterval	30			Heartbeat interval (in seconds)
fixFieldList				User-defined fields for messages. If this list is not empty, the Engine adds it to each outgoing message.
outgoingLoginFixFieldList				Additional fields for the outgoing Logon message

groups				A comma-separated list of routing groups
username				The assigned value used to identify a username to send in the Logon message for initiator session and a username to validate with the user name from the Logon request for a session acceptor. The session will be accepted if the username is not defined.
password				The assigned value used to identify a password to send in the Logon message for the session initiator and the password to validate with the password from the Logon request for the session acceptor. The session will be accepted if the password is not defined.
disposeOnDisconnect	false			Disposes the FIX session when another side breaks the connection or in case of disconnection. More details about session statuses can be found in the FIX Antenna Java documentation. **NOTE**: If the property is set to *true*, then the reconnect logic will be suppressed.
startOnload	false	Yes	Yes	Whether a FIX session should be started during FIXEdge Java server initialization.
startTime		Yes	Yes	A cron expression that defines a FIX session start time
stopTime		Yes	Yes	A cron expression that defines a FIX session stop time
scheduleTimeZone		Yes	Yes	A time zone for the start and stop times
resetSequenceOnSchedule	false	Yes	Yes	Whether sequences of a FIX session should be reset at the moment, defined by the startTime expression.

NOTE:

1. The FIX session can be started or stopped only if the `startOnload` option is enabled or if any scheduling is applied. Otherwise, the session will be inactive. Please also refer to the Scheduler configuration section for more complex scheduling definitions.
2. It is possible to use environment variables in the configuration.
   Example: **password = ${ENV_PASSWORD}**, where **ENV_PASSWORD** is the name of the environment variable.

Timestamps in storages

FIX Antenna Java library provides the ability to mark storages and messages inside them with timestamps. This feature is supported only for storages with human-readable formats

- Filesystem Storage - persistent storage, all messages will be saved on disk;
- Sliced File Storage - persistent storage with a defined maximum of file size;
- MMF Storage - persistent file storage that uses the technology of the memory-mapped file.

Incoming and outgoing message storages could contain the timestamp of the FIX sessions start in their names. The following session properties are responsible for such a configuration:

Configuration options

Property name	Default value	Description
storageNameTimestamped	false	Ability to add a session creation timestamp to the storage file name. The timestamp replaces the placeholder {3} from the format defined in the **outgoingLogFile** parameter (see below). If the option is enabled and the storage filename includes a timestamp of the session start, during the backup procedure such file will be moved to a backup folder with the same name. Otherwise, if the filename is simple and without a timestamp, the file will be copied with the timestamp by the FIX Engine backup procedure to support the uniqueness of this file in the backup folder.
storageTimestampFormat	yyyy-MM-dd_HH-mm-ss	The SimpleDateFormat pattern. **NOTE:** Avoid delimiter chars that are not valid for file system file names.

incomingLogFile	{0}.in	Incoming storage file name pattern. • {0} will be replaced with actual sessionID • {1} with actual SenderCompID • {2} with actual TargetCompID • {3} can be replaces with timestamp (see 'storageNameTimestamped') • {4} with actual session qualifier Acceptable values: • {0}-{3}.in • {3}-{0}.in
outgoingLogFile	{0}.out	Outgoing storage file name pattern. • {0} will be replaced with actual sessionID • {1} with actual SenderCompID • {2} with actual TargetCompID • {3} can be replaces with timestamp (see 'storageNameTimestamped') • {4} with actual session qualifier. Acceptable values: • {0}-{3}.out • {3}-{0}.out
timestampsInLogs	true	Ability to write timestamps in the in/out log files.
timestampsPrecisionInLogs	Milli	The desired precision of timestamps in the in/out log files. Valid values: • Milli • Micro • Nano By default, human-readable storages include a millisecond timestamp for each message. This behavior could be flexible changed. For example, it's possible to disable timestamps to minimize the latency and save time for timestamp generation. Or to increase the precision of timestamps if it needs for better statistics.
markIncomingMessageTime	false	Transport will set the additional time mark in nanoseconds for incoming messages right after read data from the socket if this option is set to *true*. The AbstractFIXTransport.getLastReadMessageTimeNano() method could return this value. Note that, by default, messages are marked with a timestamp at the moment of storing them into the storage. But, for incoming messages, it is possible to store the timestamp of the message reading with the enabled `markIncomingMessageTime` option, but this slightly increases the latency.

Secure connection configuration

FIXEdge Java supports configuring secure transport separately for each session.

FIX Sessions can use a mutual store of private keys (KeyStores) and a mutual store of trusted keys (TrustStore).

In some cases (for security reasons, administrative convenience, or flexibility), each session must use independent KeyStore and TrustStore.

Default session settings can be configured in the **s_fixDefault.properties** file (see details here) located in the same directory with the **s_fix<session id>.properties** session configuration file.

Each acceptor can use the dedicated port if this port is defined in the session configuration file. Otherwise, the default settings port is used.

If Secure Connection settings for a session differ, use of a dedicated port is mandatory.

Secure connection parameters

The full list of configuration parameters can be found here: B2BITS FIX Antenna Java Programmer's Guide.

Property	Default value	Description

enableSSL	false	Enable secure transport for the session
keyStorePath		Path to a KeyStore, which contains private keys for a secure connection. One key is to be set per port.
keyStorePassword		KeyStore password
trustStorePath		Path to a TrustStore. Usually contains a chain of trusted certificates.
trustStorePassword		TrustStore password
sslKeystoreType	JKS	The type of a KeyStore. See the KeyStore section in the Java Cryptography Architecture Standard Algorithm Name Documentation for information about standard types. Examples of value: JKS, JCEKS, PKCS12, PKCS11
sslTruststoreType	JKS	The type of a TrustStore. See the KeyStore section in the Java Cryptography Architecture Standard Algorithm Name Documentation for information about standard types. Examples of value: JKS, JCEKS, PKCS12, PKCS11
sslKeystoreKeyAlias		Alias filter for used entities in a KeyStore. The only keys with defined alias will be used for a secure connection if this property is defined.
sslTruststoreKeyAlias		Alias filter for used entities in a TrustStore. The only certificates with defined alias will be used for a secure connection if this property is defined.
sslProtocol	TLSv1.2	Preferred SSL protocol. For the initiator side, it's a protocol, which is used for handshake initialization (but the server may propose another protocol for connection). For an acceptor side, it defined a family of possible protocols. See the SSLContext section in the Java Cryptography Architecture Standard Algorithm Name Documentation for information about standard protocol names. To define certain protocol(s) for communication, please use the `acceptedSslServerProtocols` option.
acceptedSslServerProtocols		Accepted SSL protocols for connection to the server. This option is used to restrict the SSL protocols on which the initiator can connect. Value type: comma-separated string Examples of value: "TLSv1.2", "TLSv1.1, TLSv1.2"
acceptedSslCipherSuites		Accepted SSL cipher suites for connection to server. Value type: comma separated string. Examples of value: "TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256, ..."
keyManagerAlgorithm	SunX509	Key manager factory algorithm name (see Customizing the Default Key Managers and Trust Managers). Possible values are SunX509, PKIX.
keyManagerProvider		Key manager provider. Used to get the KeyManagerFactory instance. Note that the list of registered providers may be retrieved via the Security.getProviders() method.
trustManagerAlgorithm	SunX509	Trust manager factory algorithm name (see Customizing the Default Key Managers and Trust Managers). Possible values are SunX509, PKIX.
sslServerNeedClientAuth	false	Define if authentication is required for the server-side socket. This option is working only for the acceptor sessions.

Example configuration for default acceptor session parameters

conf/sessions/ssl/s_fixDefault.properties

```
# FIXEdge Java SSL default session configuration properties
fixVersion=FIX.4.4
startOnload = true

# Define options for secure connection
port=5555
enableSSL=true
sslProtocol = TLS
acceptedSslServerProtocols = TLSv1.2

# SSL Properties
keyStorePath=conf/ssl/FIXEdge_Server_Keystore1.jks
keyStorePassword=FIXEdgeServer
trustStorePath=conf/ssl/FIX_Client_Truststore1.jks
trustStorePassword=FIXClient
```

Example configuration for a secure acceptor session using default secure properties

conf/sessions/s_fix_session1.properties

```
# FIXEdge Java session configuration properties with default SSL settings
sessionType=acceptor
senderCompID=FIXEdgeJ
targetCompID=FIXClient1
fixVersion=FIX.4.4
```

Example configuration for a secure acceptor session using custom secure properties

conf/sessions/ssl/s_fix_session2.properties

```
# FIXEdge Java session configuration properties with custom SSL settings
sessionType=acceptor
senderCompID=FIXEdgeJ
targetCompID=FIXClient2
fixVersion=FIX.4.4

# Specifing custom port because secure configuration is changed
port=6666
enableSSL=true
sslProtocol = TLS
acceptedSslServerProtocols = TLSv1.2
# SSL Properties
keyStorePath=conf/ssl/FIXEdge_Server_Keystore2.jks
keyStorePassword=FIXEdgeServer
trustStorePath=conf/ssl/FIX_Client_Truststore2.jks
trustStorePassword=FIXClient
```

Example configuration for a secure initiator session using custom secure properties

conf/sessions/ssl/s_fix_session2.properties

```
host=localhost
port=3000
senderCompID=FIXEdgeJ
targetCompID=FIXClient3
fixVersion=FIX.4.4

# Define options for secure connection
enableSSL=true
keyStorePath=conf/ssl/FIXEdge_Server_Keystore3.jks
keyStorePassword=FIXEdgeServer
trustStorePath=conf/ssl/FIX_Client_Truststore3.jks
trustStorePassword=FIXClient
```