

# BL Scripting with JavaScript

- Introduction
- Business Rule scripting with JavaScript
  - JS FIX functionality
  - Error handling during script execution
  - Logging in JavaScript
    - print(<text>)
  - Operations with session
    - startSession(<sessionName>)
    - startSession(<senderCompld>,<targetCompld>)
    - startSession(<senderCompld>,<targetCompld>,<sessionQualifier>)
    - disconnectSession(<sessionName>,<reason>)
    - disconnectSession(<senderCompld>,<targetCompld>,<reason>)
    - disconnectSession(<senderCompld>,<targetCompld>,<sessionQualifier>,<reason>)
    - terminateSession(<sessionName>,<reason>)
    - terminateSession(<senderCompld>,<targetCompld>,<reason>)
    - terminateSession(<senderCompld>,<targetCompld>,<sessionQualifier>,<reason>)
  - Operations with group fields
    - getGroup(<fieldTag>)
    - getGroup(<parentGroupHandle>, <entry>, <fieldTag>)
    - removeField(<group handle>, <entry>, <fieldTag>)
    - isGroupValid(<handle>)
  - Operations with message fields
    - removeField(<fieldTag>)
    - createReject("session")
    - isSupportedField(<fieldTag>)
    - isSupportedField(<group handle>, <fieldTag>)
    - getNumField(<fieldTag>)
    - getNumField(<group handle>, <entry>, <fieldTag>)
    - getDateField(<fieldTag>, <format>)
    - getDateField(<group handle>, <entry>, <fieldTag>, <format>)
    - getStringField(<fieldTag>)
    - getStringField(<group handle>, <entry>, <fieldTag>)
    - getMStringField(<fieldTag>)
    - getMStringField(<group handle>, <entry>, <fieldTag>)
    - getRawDataField(<fieldTag>)
    - getRawDataField(<group handle>, <entry>, <fieldTag>)
    - setNumField(<fieldTag>, <new value>)
    - setDateField(<fieldTag>, <format>, <new value>)
    - setStringField(<fieldTag>, <new value>)
    - setMStringField(<fieldTag>, <new value>)
    - setRawDataField(<fieldTag>, <new value>)
    - setNumField(<group handle>, <entry>, <fieldTag>, <new value>)
    - setDateField(<group handle>, <entry>, <fieldTag>, <format>, <new value>)
    - setStringField(<group handle>, <entry>, <fieldTag>, <new value>)
    - setMStringField(<group handle>, <entry>, <fieldTag>, <new value>)
    - setRawDataField(<group handle>, <entry>, <fieldTag>, <new value>)
    - swapFields(<fieldTag1>, <fieldTag2>)
    - swapFields(<group handle field 1>, <entry field 1>, <fieldTag1>, <fieldTag2>)
    - swapFields(<group handle field 1>, <entry field 1>, <fieldTag1>, <group handle field 2>, <entry field 2>, <fieldTag2>)
  - Additional Date operations
  - Create date from current date or using passed arguments.
    - getCurrentDate(<format>)
    - createDate(<format>, <val1>, <val2>, ...<valn>)
    - getCurrentDateStr(<format>)
    - createDateStr(<format>, <val1>, <val2>, ...<valn>)
  - Convert from date to string
    - dateToString(<date handle>)
    - dateToString(<date handle>, <format>)
  - Get part of date - year, month, day, etc.
    - getYear(<date handle>)
    - getMonth(<date handle>)
    - getWeek(<date handle>)
    - getDay(<date handle>)
    - getHour(<date handle>)
    - getMinutes(<date handle>)
    - getSec(<date handle>)
    - getMSec(<date handle>)
    - getNSec(<date handle>)
  - Set date part - year, month, day, etc.
    - setYear(<date handle>, <new val>)
    - setMonth(<date handle>, <new val>)
    - setWeek(<date handle>, <new val>)
    - setDay(<date handle>, <new val>)
    - setHour(<date handle>, <new val>)
    - setMinutes(<date handle>, <new val>)

- setSec(<date handle>, <new val>)
- setMSec(<date handle>, <new val>)
- setNSec(<date handle>, <new val>)
- **Change** date part with automatic date recalculation
  - changeYear(<date handle>, <new val>)
  - changeMonth(<date handle>, <new val>)
  - changeDay(<date handle>, <new val>)
  - changeHour(<date handle>, <new val>)
  - changeMinutes(<date handle>, <new val>)
  - changeSec(<date handle>, <new val>)
  - changeMSec(<date handle>, <new val>)
  - changeNSec(<date handle>, <new val>)
- **Routing** functions
  - getSourceClientId()
  - getSourceSessionId()
  - getSourceSessionQualifier()
  - isSessionActive(<Session Name>)
  - convert(<source protocol>, <target protocol>)
  - transform(<target protocol>, <target message type>)
  - getMsgBySeqNum(<SenderCompld>, <TargetCompld>, <SeqNum>)
  - getMsgBySeqNum(<SenderCompld>, <TargetCompld>, <SessionQualifier>, <SeqNum>)
  - send(<SenderCompld>, <TargetCompld>)
  - send(<SenderCompld>, <TargetCompld>, <SessionQualifier>)
  - send(<session source identifier>)
  - handler(<handler name>, <array of additional parameters>)
  - processNetStatusRequest()
  - createNotification (<category>, <reason>)
  - strategySend (<strategyName>)
  - strategyReject (<historyName>, <createBusinessRejectIfFailed>)
  - serializeMessage()
  - serializeMessage(<tagDelimiter>)
  - parseMessage(<msg>)
  - parseMessage(<msg>, <version>)
- **History** functions
  - saveToHistory (<historyName>, <keyFields>, <commonFields>, <expireDateTime>)
  - updateHistory (<historyName>, <keyFields>, <commonFields>, <expireDateTime>)
  - getFromHistory (<historyName>, <keyFields>, <columnName>)
  - getRecordFromHistory (<historyName>, <keyFields>)
  - removeRecordFromHistory (<historyName>, <keyFields>)
  - removeRecordFromHistoryByCompositeKey (<historyName>, <keyFields>)
- **Security** functions
  - decryptString(<text>)
  - hashString(<text>)
- **Predefined FIX date formats**

## Introduction

FIXEdge offers a flexible solution for Business Rules purposed to provide routing, transferring, and data manipulations for FIX messages that go through the Business Layer of FIXEdge. In addition to XML Rules, there are two ways to enhance the flexibility of message transferring and data manipulation. Those ways are scripting with JavaScript and XSLT. To use this feature properly, scripts should be written in one of the scripting languages and assigned to the Script instruction of Action section in a Business Rule.

The scripting subsystem gives the user the following advantages:

- Saves time spent on business logic implementation
- Flexibility and simplicity of modification
- Does not bring significant performance breakdown



FIXEdge uses [SpiderMonkey](#) JavaScript engine

## Business Rule scripting with JavaScript

It is enough to mention that the script is a JavaScript in the Script tag, in order to run the script and do the proper transformations. This structure facilitate user with all the standard Javascript features and functionality. In addition to this, it enables user to use functions that provide access to the FIX message inside data, allows manipulations with fields and groups as well as whole FIX messages.

### JS FIX functionality

A trivial example of a complete JS script for a business rule is shown below:

```

//take the value the 11th tag from the fix message and get the last 4 characters from it
//assign the new value for the tag 11 with format BBBBSSSSCCYYMMDD
//where
//BBBB - any 4 characters
//SSSS - last 4 characters from the old value of field 11
//CC - century
//YY- year
//MM - month
//DD - day

field_11 = getStringField(11);
if(field_11.length() < 4)
    return;
last4 = field_11.slice(field_11.length() - 4);
currDate = getCurrentDateStr(YYYYMMDDUtc);
century = currDate.slice(1, 3);
century++;
result = "bbbb" + last4 + century + currDate.substr(3);
setStringField(11, result);

```

## Error handling during script execution

In case of syntax error in script - script execution will be terminated and failed. There is no way to handle syntax errors in script during script execution. User able to stop script execution using throw statement, the result in this case will be the same. User must be able to verify and handle logical errors using verification functions (like isNaN, isMessageValid(), etc) in condition statements.

## Logging in JavaScript

### print(<text>)

Example Code

```
print (getCurrentDateStr(DATETIMEUtc));
```

Execution of this line results in the following record in the application log:

```
[NOTE] 20131023-08:28:53.651 [4320] [../FixEdge/conf/test.js] - JavaScript
'../FixEdge/conf/test.js' output: 20131023-08:28:53
```

## Operations with session

### startSession(<sessionName>)

Starts session "sessionName"

Example Code

```
startSession( "StartByEvent" )
```

Starts the "StartByEvent" session

### startSession(<senderCompId>,<targetCompId>)

Starts session where SenderCompID = "senderCompID" and TargetCompID = "targetCompID"

Example Code

```
startSession( "FIXEdge", "SimpleClient" )
```

Starts a session where SenderCompID = "FIXEdge" and TargetCompID = "SimpleClient"

### startSession(<senderCompId>,<targetCompId>,<sessionQualifier>)

Starts session where SenderCompID = "senderCompID",TargetCompID = "targetCompID",SessionQualifierValue = "sessionQualifier"

Example Code

```
startSession( "FIXEdge", "SimpleClient", "Q1" )
```

Starts a session where SenderCompID = "FIXEdge", TargetCompID = "SimpleClient", SessionQualifierValue = "Q1"

**disconnectSession(<sessionName>,<reason>)**

Disconnects session "sessionName" with reason "reason"

**Example Code**

```
disconnectSession("DisconnectByEvent", "Disconnect by message")
```

Disconnects the "DisconnectByEvent" session with the "Disconnect by message" reason

**disconnectSession(<senderCompId>,<targetCompId>,<reason>)**

Disconnects session where SenderCompID = "senderCompId" and TargetCompID = "targetCompId" with reason "reason"

**Example Code**

```
disconnectSession("FIXEdge", "SimpleClient", "Disconnect by message")
```

Disconnects a session where SenderCompID = "FIXEdge" and TargetCompID = "SimpleClient" with reason "Disconnect by message"

**disconnectSession(<senderCompId>,<targetCompId>,<sessionQualifier>,<reason>)**

Starts session where SenderCompID = "senderCompID", TargetCompID = "targetCompID", SessionQualifierValue = "sessionQualifier" with reason "reason"

**Example Code**

```
disconnectSession("FIXEdge", "SimpleClient", "Disconnect by message", "Q1")
```

Disconnects a session where SenderCompID = "FIXEdge", TargetCompID = "SimpleClient", SessionQualifierValue = "Q1" with reason "Disconnect by message"

**terminateSession(<sessionName>,<reason>)**

Terminates session "sessionName" with reason "reason"

**Example Code**

```
terminateSession("TerminateByEvent", "Terminate by message")
```

Terminates the "TerminateByEvent" session with the "Terminate by message" reason

**terminateSession(<senderCompId>,<targetCompId>,<reason>)**

Terminates session where SenderCompID = "senderCompID" and TargetCompID = "targetCompID" with reason "reason"

**Example Code**

```
terminateSession("FIXEdge", "SimpleClient", "Terminate by message")
```

Terminates a session where SenderCompID = "FIXEdge" and TargetCompID = "SimpleClient" with reason "Terminate by message"

**terminateSession(<senderCompId>,<targetCompId>,<sessionQualifier>,<reason>)**

Terminates session where SenderCompID = "senderCompID", TargetCompID = "targetCompID", SessionQualifierValue = "sessionQualifier"

**Example Code**

```
terminateSession("FIXEdge", "SimpleClient", "Q1", "Terminate by message")
```

Terminates a session where SenderCompID = "FIXEdge", TargetCompID = "SimpleClient", SessionQualifierValue = "Q1" with reason "Terminate by message"

**Please note:**

1. The session will not be visible in FIXICC or manageable (start, disconnect, etc) from FIXICC/BL/JS/scheduler.
2. The termination and the disconnection occur with async way after rule completion. In case of fail <OnRuleFailEvent> will be called.

## Operations with group fields

### getGroup(<fieldTag>)

Returns group handle for <fieldTag> group field

**Example Code**

```
hdl = getGroup(78);  
isGroupValid(hdl)
```

Gets a handle on group 78 and tests whether handle 78 is valid

### getGroup(<parentGroupHandle>, <entry>, <fieldTag>)

Returns group handle for <fieldTag> group field from <parentGroupHandle>[<entry>] group



Numeration of <entry> starts with 0.

**Example Code**

```
hdl = getGroup(552);  
isGroupValid(hdl);  
  
hdl2 = getGroup(hdl, 0, 518);  
isGroupValid(hdl2);  
  
if(hdl2 != hdl){  
    setNumField(552, 4);  
}
```

Gets a handle on group 552 and tests whether the handle 552 is valid

Gets a handle from the first record of 552 group on group 518 and tests whether the handle 518 is valid

If the group handles are not identical, then increase the number of group 552 elements up to 4

### removeField(<group handle>, <entry>, <fieldTag>)

Removes field from group field



Numeration of <entry> starts with 0.

**Example Code**

```
hdl = getGroup(78);  
isGroupValid(hdl);  
removeField(hdl, 0, 79);
```

Gets handle on group 78 and tests whether the handle 78 is valid

Removes tag 79 from group 78

### isGroupValid(<handle>)

Returns true or false depending on the handle value

**Example Code**

```
//see the example above
```

## Operations with message fields

### removeField(<fieldTag>)

Removes field from message

Example Code

```
removeField(58);
```

Removes tag 58 from the message

### createReject("session")

Creating reject

Example code

```
createReject("Target42sohSender42");
```

Creating reject and sending status message 35=3

### isSupportedField(<fieldTag>)

Returns true if the field is defined in the dictionary, i.e. it can be presented in the message. False means the tag is not expected in this message type

Example Code:

```
if ( isSupportedField( 58 ) )
    removeField( 58 );
```

The example shows how to remove tag 58 from the message if it is defined in the dictionary

The script checks if tag 58 is defined for the message  
And remove it from the message.

### isSupportedField( <group handle> , <fieldTag>)

Returns true if the field is defined in the dictionary for the specific group, i.e. it can be presented in the repeating group of the message. False means the tag is not expected in this specific group for this message type

Example Code

```
partiesGrp = getGroup(448);
if ( isGroupValid(partiesGrp) &&
    isSupportedField(partiesGrp, 2376) )
    removeField( partiesGrp, 0 ,
    2376 );
```

The example shows how to remove non required tag 2376 from the Parties group entry #0.

The script gets the repeating group 448 handle.  
Then checks if the group is valid and present in the message. And if tag 2376 is defined in the dictionary.  
Then removes tag 2376 from the group entry #0

### getNumField(<fieldTag>)

Returns numeric value of the <fieldTag> field

Example Code

```
getNumField(78);
```



Returns a numeric value of field 78



### getNumField(<group handle>, <entry>, <fieldTag>)

Returns number value of the <fieldTag> field from group <group handle>[<entry>]





Numeration of <entry> starts with 0.

<p>Example Code</p> <pre>hndl = getGroup(552); isGroupValid(hndl); setNumField(552,getNumField(hndl, 0, 581));</pre>	<p>Gets handle on group 552 and tests whether the handle 552 is valid</p> <p>Gets a numeric value of field 581 and assigns it to field 552</p>
<p><b>getDateField(&lt;fieldTag&gt;, &lt;format&gt;)</b></p> <p>Returns date handle to the &lt;fieldTag&gt; field value</p>	
<p>Example Code</p> <pre>getDateField(64, YYYYMMDDLmd);</pre>	<p>Gets a date value of the field 64 in YYYYMMDD format</p>
<p><b>getDateField(&lt;group handle&gt;, &lt;entry&gt;, &lt;fieldTag&gt;, &lt;format&gt;)</b></p> <p>Returns date handle to the &lt;fieldTag&gt; field value from group &lt;group handle&gt;[&lt;entry&gt;]</p>	
<p> Numeration of &lt;entry&gt; starts with 0.</p>	
<p>Example Code</p> <pre>hndl = getGroup(778); isGroupValid(hndl); getDateField(hndl, 0, 779, YYYYMMDDLmd);</pre>	<p>Gets handle on group 778 and tests whether the handle 778 is valid</p> <p>Gets a date value of the field 779 in the YYYYMMDD format from the first element of group 778</p>
<p><b>getStringField(&lt;fieldTag&gt;)</b></p> <p>Returns string value of the &lt;fieldTag&gt; field</p>	
<p>Example Code</p> <pre>getStringField(78);</pre>	<p>Returns a string value of field 78</p>
<p><b>getStringField(&lt;group handle&gt;, &lt;entry&gt;, &lt;fieldTag&gt;)</b></p> <p>Returns string value of the &lt;fieldTag&gt; field from group &lt;group handle&gt;[&lt;entry&gt;]</p>	
<p> Numeration of &lt;entry&gt; starts with 0.</p>	
<p>Example Code</p> <pre>hndl = getGroup(552); isGroupValid(hndl); setStringField(552,getStringField(hndl, 0, 581));</pre>	<p>Gets a handle on group 552 and tests whether handle 552 is valid</p> <p>Gets a string value of field 581 and assigns it to field 552</p>
<p><b>getMStringField(&lt;fieldTag&gt;)</b></p> <p>Returns array that contains multi string value of the &lt;fieldTag&gt; field</p>	

<p>Example Code</p> <pre>tmp = getMStringField(18);  if(3 == tmp.length){ setStringField(55, tmp[0]); setStringField(65, tmp[1]); }</pre>	<p>Gets a multiple string value of field 18 and assigns it to the tmp variable</p> <p>If the number of strings in the tmp variable equals to 3, then the value of the first string is assigned to field 55 and the value of the second string is assigned to field 65</p>
<p><b>getMStringField(&lt;group handle&gt;, &lt;entry&gt;, &lt;fieldTag&gt;)</b></p> <p>Returns array that contains multi string value of the &lt;fieldTag&gt; field from group &lt;group handle&gt;[&lt;entry&gt;]</p> <div style="border: 1px solid #ccc; padding: 5px; background-color: #fff9c4;">  Numeration of &lt;entry&gt; starts with 0. </div>	
<p>Example Code</p> <pre>hndl = getGroup(552); isGroupValid(hndl); tmp = getMStringField(hndl, 0, 18);</pre>	<p>Gets a handle on group 552 and tests whether handle 552 is valid</p> <p>Gets a multiple string value of field 18 found in the first entry of group 552 and assigns it to the tmp variable</p>
<p><b>getRawDataField(&lt;fieldTag&gt;)</b></p> <p>returns array that contains raw data value of the &lt;fieldTag&gt; field</p>	
<p>Example Code</p> <pre>tmp = getRawDataField(91);</pre>	<p>Gets a raw data value of field 91 and assigns it to the tmp variable</p>
<p><b>getRawDataField(&lt;group handle&gt;, &lt;entry&gt;, &lt;fieldTag&gt;)</b></p> <p>Returns array that contains raw data value of the &lt;fieldTag&gt; field from group &lt;group handle&gt;[&lt;entry&gt;]</p> <div style="border: 1px solid #ccc; padding: 5px; background-color: #fff9c4;">  Numeration of &lt;entry&gt; starts with 0. </div>	
<p>Example Code</p> <pre>hndl = getGroup(5521); isGroupValid(hndl); tmp = getRawDataField(hndl, 0, 91);</pre>	<p>Gets a handle on group 5521 and tests whether handle 5521 is valid</p> <p>Gets a raw data value of field 91 found in the first entry of group 5521 and assigns it to the tmp variable</p>
<p><b>setNumField(&lt;fieldTag&gt;, &lt;new value&gt;)</b></p> <p>assigns value &lt;new value&gt; to the &lt;fieldTag&gt; field as a NUMBER</p>	
<p>Example Code</p> <pre>//see the sample for the getNumField();</pre>	
<p><b>setDateField(&lt;fieldTag&gt;, &lt;format&gt;, &lt;new value&gt;)</b></p> <p>Assigns value &lt;new value&gt; to the &lt;fieldTag&gt; field as a DATE</p>	



<p>Example Code</p> <pre>tmp = getDateField(64, YYYYMMDDLmd); setYear(tmp, 1979); setDateField(64, YYYYMMDDLmd, tmp);</pre>	<p>Gets a date value of field 64 in the YYYYMMDD format and assigns it to the tmp variable</p> <p>Sets the year of the date to 1979 in the tmp variable</p> <p>Assigns a date from the tmp variable back to field 64</p>
<p><b>setStringField(&lt;fieldTag&gt;, &lt;new value&gt;)</b></p> <p>Assigns value &lt;new value&gt; to the &lt;fieldTag&gt; field as a STRING</p>	
<p>Example Code</p> <pre>// see the example for getMStringField();</pre>	
<p><b>setMStringField(&lt;fieldTag&gt;, &lt;new value&gt;)</b></p> <p>Assigns value &lt;new value&gt; to the &lt;fieldTag&gt; field as a MString</p>	
<p>Example Code</p> <pre>tmp = ["aa", "abb", "bbbc"]; setMStringField(18, tmp);</pre>	<p>Sets an array of strings named tmp</p> <p>Assigns the array of strings to multiple string field 18</p>
<p><b>setRawDataField(&lt;fieldTag&gt;, &lt;new value&gt;)</b></p> <p>Assigns value &lt;new value&gt; to the &lt;fieldTag&gt; field as a RawData</p>	
<p>Example Code</p> <pre>tmp = [12, 56, 0, 34]; setRawDataField(91, tmp);</pre>	<p>Sets an array of numbers named tmp</p> <p>Assigns the array of numbers to raw data field 91</p>
<p><b>setNumField(&lt;group handle&gt;, &lt;entry&gt;, &lt;fieldTag&gt;, &lt;new value&gt;)</b></p> <p>Assigns value &lt;new value&gt; to the &lt;fieldTag&gt; field from group &lt;group handle&gt;[&lt;entry&gt;] as a NUMBER</p> <div data-bbox="142 1234 1481 1314" style="border: 1px solid #ccc; background-color: #fff9c4; padding: 5px;">  Numeration of &lt;entry&gt; starts with 0. </div>	
<p><b>Example Code</b></p> <pre>hndl = getGroup(552); isGroupValid(hndl); setNumField(hndl, 1, 519, 4);</pre>	<p><b>Explanations</b></p> <p>Gets a handle on group 552 and tests whether handle 552 is valid</p> <p>Field 519 from the second entry of group 552 is assigned to 4</p>
<p><b>setDateField(&lt;group handle&gt;, &lt;entry&gt;, &lt;fieldTag&gt;, &lt;format&gt;, &lt;new value&gt;)</b></p> <p>Assigns value &lt;new value&gt; to the &lt;fieldTag&gt; field from group &lt;group handle&gt;[&lt;entry&gt;] as a DATE</p> <div data-bbox="142 1680 1481 1759" style="border: 1px solid #ccc; background-color: #fff9c4; padding: 5px;">  Numeration of &lt;entry&gt; starts with 0. </div>	

**Example Code**

```
tmp = getDateField(64, YYYYMMDDLmd);  
  
hndl = getGroup(5521);  
isGroupValid(hndl);  
setDateField(hndl, 0, 64, YYYYMMDDLmd, tmp);
```

Gets a date in the YYYYMMDD format from field 64 and assigns the obtained value to the tmp variable

Gets a handle on group 5521 and tests whether handle 5521 is valid

Field 64 from the first entry of group 5521 is assigned the value from the tmp variable

**setStringField(<group handle>, <entry>, <fieldTag>, <new value>)**

Assigns value <new value> to the <fieldTag> field from group <group handle>[<entry>] as a STRING



Numeration of <entry> starts with 0.

**Example Code**

```
tmp = getStringField(581);  
  
hndl = getGroup(5521);  
isGroupValid(hndl);  
setDateField(hndl, 0, 581, tmp);
```

Gets a string value from field 581 and assigns it to the tmp variable

Gets a handle on group 5521 and tests whether handle 5521 is valid

Field 581 from the first entry of group 5521 is assigned the value from the tmp variable

**setMStringField(<group handle>, <entry>, <fieldTag>, <new value>)**

Assigns value <new value> to the <fieldTag> field from group <group handle>[<entry>] as a MString



Numeration of <entry> starts with 0.

**Example Code**

```
tmp = getMStringField(18);  
  
hndl = getGroup(5521);  
isGroupValid(hndl);  
setMStringField(hndl, 0, 18, tmp);
```

Gets a multiple strings value from field 18 and assigns it to the tmp variable

Gets a handle on group 5521 and tests whether handle 5521 is valid

Field 18 from the first entry of group 5521 is assigned the value from the tmp variable

**setRawDataField(<group handle>, <entry>, <fieldTag>, <new value>)**

Assigns value <new value> to the <fieldTag> field from group <group handle>[<entry>] as a RawData



Numeration of <entry> starts with 0.

<p>Example Code</p> <pre>tmp = getRawDataField(91);  hndl = getGroup(5521); isGroupValid(hndl); setRawDataField(hndl, 0, 91, tmp);</pre>	<p>Gets a multiple raw data value from field 91 and assigns it to the tmp variable</p> <p>Gets a handle on group 5521 and tests whether handle 5521 is valid</p> <p>Field 91 from the first entry of group 5521 is assigned the value from the tmp variable</p>
--	---

### swapFields(<fieldTag1>, <fieldTag2>)

Exchanges values of fields <fieldTag1> and <fieldTag2>

<p>Example Code</p> <pre>swapFields(63, 573);</pre>	<p>Values of field 63 and field 573 were exchanged</p>
---	--

### swapFields(<group handle field 1>, <entry field 1>, <fieldTag1>, <fieldTag2>)

Exchanges values of fields <fieldTag1> (from group <group handle field 1>[<entry field 1>]) and <fieldTag2> (which can be both from group <group handle field 1>[<entry field 1>] and outside of it)



Numeration of <entry field 1> starts with 0.

<p>Example Code</p> <pre>hndl = getGroup(552); isGroupValid(hndl); swapFields(hndl, 1, 11, 572);]</pre>	<p>Gets a handle on group 552 and tests whether handle 552 is valid</p> <p>Exchanges values of <b>field 11</b> in the second entry of group 552 <b>and field 572</b> outside of it</p>
---	--

### swapFields(<group handle field 1>, <entry field 1>, <fieldTag1>, <group handle field 2>, <entry field 2>, <fieldTag2>)

Exchanges values of fields <fieldTag1> (from group <group handle field 1>[<entry field 1>]) and <fieldTag2> (from group <group handle field 2>[<entry field 2>])



Numeration of <entry field 1> and <entry field 2> starts with 0.

<p>Example Code</p> <pre>hndl = getGroup(552); isGroupValid(hndl); swapFields(hndl, 1, 11, hndl, 1, 37);</pre>	<p>Gets a handle on group 552 and tests whether handle 552 is valid</p> <p>Exchanges values of field 11 in the second entry and field 37 in the first entry of group 552</p>
--	--

## Additional Date operations

### Create date from current date or using passed arguments.

#### getCurrentDate(<format>)

Returns handle that encapsulates current date in format <format>

Example Code <pre>getCurrentDate(DATETIMEUtc);</pre>	Creates the date data of the DATETIMEUtc format
<b>createDate(&lt;format&gt;, &lt;val1&gt;, &lt;val2&gt;, ...&lt;valn&gt;)</b>	
Returns handle that encapsulates date in format <format> constructed using values <val1>, <val2>, ...<valn>	
Example Code <pre>tmp = createDate(YYYYMMDD, 1995, 12, 23);  setStringField(55, dateToString(tmp));</pre>	Creates a date of the YYYYMMDD format with corresponding values and assigns it to the tmp variable  Converts the value of the tmp variable to string and assigns this string to be the value of field 55
<b>getCurrentDateStr(&lt;format&gt;)</b>	
Returns current date in format <format>	
Example Code <pre>setStringField(55, getCurrentDateStr(DATETIMEUtc));</pre>	Gets a current date as a string and assigns this string to be the value of field 55
<b>createDateStr(&lt;format&gt;, &lt;val1&gt;, &lt;val2&gt;, ...&lt;valn&gt;)</b>	
Returns date in format <format> constructed using values <val1>, <val2>, ...<valn>	
Example Code <pre>tmp = createDateStr(YYYYMMDD, 1995, 12, 23);]</pre>	Creates a date of YYYYMMDD format with corresponding values returns this date as string and assigns it to the tmp variable
<b>Convert from date to string</b>	
<b>dateToString(&lt;date handle&gt;)</b>	
Returns date in format that was mentioned in getCurrentDate() or createDate() call	
Example Code <pre>//see the example of createDate()</pre>	
<b>dateToString(&lt;date handle&gt;, &lt;format&gt;)</b>	
Returns date in format <format>	
Example Code <pre>tmp = createDate(YYYYMMDD, 1995, 12, 23);  setStringField(55, dateToString(tmp, YYYYMM));</pre>	Creates a date from given data in one format and assigns it to the tmp variable  Field 55 is assigned a value of the tmp variable
<b>Get part of date - year, month, day, etc.</b>	
<b>getYear(&lt;date handle&gt;)</b>	
Returns year of date <date handle>	

<p>Example Code</p> <pre>tmp = getCurrentDate(DATETIMEUtc); tmp2 = getYear(tmp); setDateField(64, YYYYMMDDLmd, tmp2);</pre>	<p>Creates a date handle of the current date</p> <p>Extracts the year from the tmp variable and assigns it to the tmp2 variable</p> <p>Sets the value of field 64 to the beginning of current year</p>
<p><b>getMonth(&lt;date handle&gt;)</b></p> <p>Returns month of date &lt;date handle&gt;</p>	
<p>Example Code</p> <pre>tmp = getCurrentDate(DATETIMEUtc); tmp3 = getMonth(tmp);</pre>	<p>Creates a date handle of the current date</p> <p>Extracts the month value from the tmp variable and assigns it to the tmp3 variable</p>
<p><b>getWeek(&lt;date handle&gt;)</b></p> <p>Returns weekof date &lt;date handle&gt;</p>	
<p>Example code</p> <pre>tmp = createDate(YYYYMMWW, 1995, 12, 2); setStringField(55, getWeek(tmp));</pre>	<p>Creates a date handle &lt;date handle&gt;</p> <p>Sets the value of the tmp variable to field 55</p>
<p><b>getDay(&lt;date handle&gt;)</b></p> <p>Returns day of date &lt;date handle&gt;</p>	
<p>Example Code</p> <pre>tmp = getCurrentDate(DATETIMEUtc); tmp4 = getDay(tmp);</pre>	<p>Creates a date handle of the current date</p> <p>Extracts the day value from the tmp variable and assigns it to the tmp4 variable</p>
<p><b>getHour(&lt;date handle&gt;)</b></p> <p>Returns hour of date &lt;date handle&gt;</p>	
<p>Example Code</p> <pre>tmp = getCurrentDate(DATETIMEUtc); tmp5 = getHour(tmp);</pre>	<p>Creates a date handle of the current date</p> <p>Extracts the hour value from the tmp variable and assigns it to the tmp5 variable</p>
<p><b>getMinutes(&lt;date handle&gt;)</b></p> <p>Returns minutes of date &lt;date handle&gt;</p>	
<p>Example Code</p> <pre>tmp = getCurrentDate(DATETIMEUtc); tmp6 = getMinutes(tmp);</pre>	<p>Creates a date handle of the current date</p> <p>Extracts the minutes value from the tmp variable and assigns it to the tmp6 variable</p>
<p><b>getSec(&lt;date handle&gt;)</b></p> <p>Returns seconds of date &lt;date handle&gt;</p>	
<p>Example Code</p> <pre>tmp = getCurrentDate(DATETIMEUtc); tmp7 = get Sec(tmp);</pre>	<p>Creates a date handle of the current date</p> <p>Extracts the seconds value from tmp variable and assigns it to the variable tmp7</p>
<p><b>getMSec(&lt;date handle&gt;)</b></p> <p>Returns milliseconds of date &lt;date handle&gt;</p>	

<p>Example Code</p> <pre>tmp = getCurrentDate(DATETIMEUtc); tmp7 = getMSec(tmp);</pre>	<p>Creates a date handle of the current date</p> <p>Extracts the mile seconds value from the tmp variable and assigns it to the tmp7 variable</p>
--	---

### getNSec(<date handle>)

Returns nanoseconds of date <date handle>. Function is available since **FIXEdge 6.2.0**.

<p>Example Code</p> <pre>tmp = getCurrentDate (DATETIMENanoUtc); tmp7 = getNSec(tmp);</pre>	<p>Creates a date handle of the current date</p> <p>Extracts the nanoseconds value from the tmp variable and assigns it to the tmp7 variable</p>
---	--

## Set date part - year, month, day, etc.

### setYear(<date handle>, <new val>)

Assign new value <new val> to year of date <date handle>



#### Validation rule

New value must be greater than 1970 and less than 2038

<p>Example Code</p> <pre>tmp = getDateField(64, YYYYMMDDLmd); setYear(tmp, 1979); setDateField(64, YYYYMMDDLmd, tmp);</pre>	<p>Gets the date value of field 64 in the YYYYMMDD format and assigns it to the tmp variable</p> <p>Sets the year of the date to 1979 in the tmp variable</p> <p>Assigns the date from the tmp variable back to field 64</p>
---	--

### setMonth(<date handle>, <new val>)

Assign new value <new val> to month of date <date handle>



#### Validation rule

New value must be greater than or equal 1 and less than or equal 12

<p>Example Code</p> <pre>tmp = getDateField(64, YYYYMMDDLmd); setMonth(tmp, 06); setDateField(64, YYYYMMDDLmd, tmp);</pre>	<p>Gets the date value of field 64 in the YYYYMMDD format and assigns it to the tmp variable</p> <p>Sets the month of the date to 06 in the tmp variable</p> <p>Assigns the date from the tmp variable back to field 64</p>
--	---




### setWeek(<date handle>, <new val>)

Assign new value <new val> to week of date <date handle>



#### Validation rule

New value must be greater than or equal 1 and less than or equal 5

<p>Example Code</p> <pre>tmp = createDate(YYYYMMWW, 1995, 12, 2); setWeek(tmp, 4); setStringField(55, dateToString(tmp));</pre>	<p>Gets the date value of field 64 in the YYYYMMWW format and assigns it to the tmp variable</p> <p>Sets the week of the date to 4 in the tmp variable</p> <p>Assigns the date from the tmp variable to field 55 in the string format</p>
<p><b>setDay(&lt;date handle&gt;, &lt;new val&gt;)</b></p> <p>Assign new value &lt;new val&gt; to day of date &lt;date handle&gt;</p> <div data-bbox="142 420 1481 525" style="border: 1px solid #ccc; padding: 5px; background-color: #fff9c4;"> <p> <b>Validation rule</b></p> <p>New value must be greater than or equal 1 and less than or equal 31</p> </div>	
<p>Example Code</p> <pre>tmp = getDateField(64, YYYYMMDDLmd); setDay(tmp, 27); setDateField(64, YYYYMMDDLmd, tmp);</pre>	<p>Gets the date value of field 64 in the YYYYMMDD format and assigns it to the tmp variable</p> <p>Sets the day of the date to 27 in the tmp variable</p> <p>Assigns the date from the tmp variable back to field 64</p>
<p><b>setHour(&lt;date handle&gt;, &lt;new val&gt;)</b></p> <p>Assign new value &lt;new val&gt; to hour of date &lt;date handle&gt;</p> <div data-bbox="142 898 1481 1003" style="border: 1px solid #ccc; padding: 5px; background-color: #fff9c4;"> <p> <b>Validation rule</b></p> <p>New value must be greater than or equal 0 and less than 24</p> </div>	
<p>Example Code</p> <pre>tmp = getDateField(64, DATETIMEUtc); setHour(tmp, 10); setDateField(64, DATETIMEUtc, tmp);</pre>	<p>Gets the date value of field 64 in the DATETIMEUtc format and assigns it to the tmp variable</p> <p>Sets the hour of the date to 10 in the tmp variable</p> <p>Assigns the date from the tmp variable back to field 64</p>
<p><b>setMinutes(&lt;date handle&gt;, &lt;new val&gt;)</b></p> <p>Assign new value &lt;new val&gt; to minutes of date &lt;date handle&gt;</p> <div data-bbox="142 1402 1481 1507" style="border: 1px solid #ccc; padding: 5px; background-color: #fff9c4;"> <p> <b>Validation rule</b></p> <p>New value must be greater than or equal 0 and less than or equal 59</p> </div>	
<p>Example Code</p> <pre>tmp = getDateField(64, DATETIMEUtc); setMinutes(tmp, 20); setDateField(64, DATETIMEUtc, tmp);</pre>	<p>Gets the date value of field 64 in the DATETIMEUtc format and assigns it to the tmp variable</p> <p>Sets the minutes of the date to 20 in the tmp variable</p> <p>Assigns the date from the tmp variable back to field 64</p>

### setSec(<date handle>, <new val>)

Assign new value <new val> to seconds of date <date handle>



#### Validation rule

New value must be greater than or equal 0 and less than or equal 59

#### Example Code

```
tmp = getDateField(64,
DATETIMEUtc);

setSec(tmp, 20);

    setDateField(64, DATETIMEUtc,
tmp);
```

Gets the date value of field 64 in the DATETIMEUtc format and assigns it to the tmp variable

Sets the seconds of the date to 20 in the tmp variable

Assigns the date from the tmp variable back to field 64

### setMSec(<date handle>, <new val>)

Assign new value <new val> to milliseconds of date <date handle>



#### Validation rule

New value must be greater than or equal 0 and less than or equal 59

#### Example Code

```
tmp = getDateField(64,
DATETIMEUtc);

setMSec(tmp, 50);

setDateField(64, DATETIMEUtc, tmp);
```

Gets the date value of field 64 in the DATETIMEUtc format and assigns it to the tmp variable

Sets the milliseconds of the date to 50 in the tmp variable

Assigns the date from the tmp variable back to field 64

### setNSec(<date handle>, <new val>)

Assign new value <new val> to nanoseconds of date <date handle>. Function is available since **FIXEdge 6.2.0**.

#### Example Code

```
tmp = getDateField(64,
DATETIMENanoUtc);

setNSec(tmp, 50);

setDateField(64, DATETIMENanoUtc,
tmp);
```

Gets the date value of field 64 in the DATETIMENanoUtc format and assigns it to the tmp variable

Sets the nanoseconds of the date to 50 in the tmp variable

Assigns the date from the tmp variable back to field 64

## Change date part with automatic date recalculation

### changeYear(<date handle>, <new val>)

Add value <new val> to year of date <date handle>

#### Example Code

```
tmp = getDateField(64,
DATETIMEUtc);

changeYear(tmp, 2);

setDateField(64, DATETIMEUtc, tmp);
```

Gets the date value of field 64 in the DATETIMEUtc format and assigns it to the tmp variable

Adds 2 years to the value of the date in the tmp variable

Assigns the date from the tmp variable back to field 64

### changeMonth(<date handle>, <new val>)

Add value <new val> to month of date <date handle>



<p>Example Code</p> <pre>tmp = getDateField(64, DATETIMEUtc);  changeMonth(tmp, 5);  setDateField(64, DATETIMEUtc, tmp);</pre>	<p>Gets the date value of field 64 in the DATETIMEUtc format and assigns it to the tmp variable</p> <p>Adds 5 months to the value of the date in the tmp variable</p> <p>Assigns the date from the tmp variable back to field 64</p>
<p><b>changeDay(&lt;date handle&gt;, &lt;new val&gt;)</b></p> <p>Add value &lt;new val&gt; to day of date &lt;date handle&gt;</p>	
<p>Example Code</p> <pre>tmp = getDateField(64, DATETIMEUtc);  changeDay(tmp, 10);  setDateField(64, DATETIMEUtc, tmp);</pre>	<p>Gets the date value of field 64 in the DATETIMEUtc format and assigns it to the tmp variable</p> <p>Adds 10 days to the value of the date in the tmp variable</p> <p>Assigns the date from the tmp variable back to field 64</p>
<p><b>changeHour(&lt;date handle&gt;, &lt;new val&gt;)</b></p> <p>Add value &lt;new val&gt; to hour of date &lt;date handle&gt;</p>	
<p>Example Code</p> <pre>tmp = getDateField(64, DATETIMEUtc);  changeHour(tmp, 24);  setDateField(64, DATETIMEUtc, tmp);</pre>	<p>Gets the date value of field 64 in the DATETIMEUtc format and assigns it to the tmp variable</p> <p>Adds 24 hours to the value of the date in the tmp variable</p> <p>Assigns the date from the tmp variable back to field 64</p>
<p><b>changeMinutes(&lt;date handle&gt;, &lt;new val&gt;)</b></p> <p>Add value &lt;new val&gt; to minutes of date &lt;date handle&gt;</p>	
<p>Example Code</p> <pre>tmp = getDateField(64, DATETIMEUtc);  changeMinutes(tmp, 600);  setDateField(64, DATETIMEUtc, tmp);</pre>	<p>Gets the date value of field 64 in the DATETIMEUtc format and assigns it to the tmp variable</p> <p>Adds 600 minutes to the value of the date in the tmp variable</p> <p>Assigns the date from the tmp variable back to field 64</p>
<p><b>changeSec(&lt;date handle&gt;, &lt;new val&gt;)</b></p> <p>Add value &lt;new val&gt; to seconds of date &lt;date handle&gt;</p>	
<p>Example Code</p> <pre>tmp = getDateField(64, DATETIMEUtc);  changeSec(tmp, 3600);  setDateField(64, DATETIMEUtc, tmp);</pre>	<p>Gets the date value of field 64 in the DATETIMEUtc format and assigns it to the tmp variable</p> <p>Adds 3600 seconds to the value of the date in the tmp variable</p> <p>Assigns the date from the tmp variable back to field 64</p>
<p><b>changeMSec(&lt;date handle&gt;, &lt;new val&gt;)</b></p> <p>Assign new value &lt;new val&gt; to milliseconds of date &lt;date handle&gt;</p>	
<p>Example Code</p> <pre>tmp = getDateField(64, DATETIMEUtc);  changeMSec(tmp, 36);  setDateField(64, DATETIMEUtc, tmp);</pre>	<p>Gets the date value of field 64 in the DATETIMEUtc format and assigns it to the tmp variable</p> <p>Adds 36 milliseconds to the value of the date in the tmp variable</p> <p>Assigns the date from the tmp variable back to field 64</p>

### changeNSec(<date handle>, <new val>)

Assign new value <new val> to nanoseconds of date <date handle>. Function is available since **FIXEdge 6.2.0**.

#### Example Code

```
tmp = getDateField(64,
DATETIMENanoUtc);

changeNSec(tmp, 36);

setDateField(64, DATETIMENanoUtc,
tmp);
```

Gets the date value of field 64 in the DATETIMENanoUtc format and assigns it to the tmp variable

Adds 36 nanoseconds to the value of the date in the tmp variable

Assigns the date from the tmp variable back to field 64

## Routing functions

### getSourceClientId()

Returns Client ID of source

#### Example Code

```
clientId = getSourceClientId();
```

Gets the TA client ID of the session context where script is applied

**Note:** Returns NULL value, if session is not TA client

### getSourceSessionId()

Returns Session ID of source

#### Example Code

```
clientId = getSourceSessionId();
```

Gets the FIX session ID of the session context where script is applied

**Note:** Returns NULL value, if the session is not FIX session.

### getSourceSessionQualifier()

Returns the name of SessionQualifier (String) in case if SessionQualifier is available for source session

#### Example Code

```
sessionQualifier =
getSourceSessionQualifier();
```

Gets the SessionQualifier name of the session context where script is applied

### isSessionActive(<Session Name>)

Checks is FIX Session with name <Session Name> active

#### Example Code

```
active = isSessionActive
("SOME_SESSION");

if (active)

    print("SOME_SESSION
active");
```

Checks if FIX Session with the "SOME\_SESSION" name is active

Prints a message

### convert(<source protocol>, <target protocol>)

Convert message from <source protocol> to <target protocol>

#### Example Code

```
convert("FIX.4.4", "FIX.4.2");
```

Converts message in the given script context, replaces a source FIX44 message with converted to FIX42.

**Note:** please refer to BusinessLayer.dtd to see applicable names of protocols

### transform(<target protocol>, <target message type>)

Transform message from current script context to<target protocol> <target message type> message

#### Example Code

```
transform("FIX.4.2", "8");
```

Transforms message in the given script context, replaces a source message with transformed to FIX42 MsgType="8".

**Note:** refer to BusinessLayer.dtd to see applicable names of protocols

### getMsgBySeqNum(<SenderCompId>, <TargetCompId> , <SeqNum>)

Search for message in session identified as <SenderCompId> and <TargetCompId> pair logs by < SeqNum>

### getMsgBySeqNum(<SenderCompId>, <TargetCompId>, <SessionQualifier>, <SeqNum>)

Search for message in session identified as <SenderCompID>/<TargetCompID>/<SessionQualifier> set logs by < SeqNum>

Example Code

```
getMsgBySeqNum( "TestSender" ,  
"TestTarget" , 12345);
```

Searches for message with given SeqNum in the session identified by the 'sender|target' pair.

**Note:** FIX Session where search occurs must be active and use persistent storage.

### send(<SenderCompID>, <TargetCompID>)

Send a message into session identified as <SenderCompID> and <TargetCompID> pair.

### send(<SenderCompID>, <TargetCompID>, <SessionQualifier>)

Send a message into session identified as <SenderCompID>/<TargetCompID>/<SessionQualifier> set.

### send(<session source identifier>)

Send a message into session that identified by <sessions source identifier>.

Example Code

```
send( "sender" , "target" );  
  
send( "exchange" );
```

Sends a message into the session identified by the 'sender|target' pair.

Sends a message into the session identified by the 'exchange' source identifier.

### handler(<handler name>, <array of additional parameters>)

Route message into handler <handler name> with array of additional parameters as JSONObject (optional)

Example Code

```
params = [{"Storage" , "file.txt"}, {"ClearTime" ,  
"22:00"}]  
  
handler( "CMEHandler" , params );
```

Routes a message in the "CMEHandler" handler and sets additional parameters for processing:

Storage = file.txt, ClearTime = 22:00

### processNetStatusRequest()

Response Network Status Response message ("BD") on request

Example Code

```
processNetStatusRequest();
```

Replaces a Network Status Request message ("BC") with Response ("BD") in the routing

### createNotification (<category>, <reason>)

Creates notification message ("C") with <category>, <reason> then puts it into the routing

Example Code

```
createNotification( "E" , "Session is  
rejected" );
```

Creates a notification message ("C") with "Status = ERROR", "Code=201", "Reason = Session is rejected" in three entries of group (58).

**Note:** <category> applies values "N" – NOTE, "W" – WARNING, "E" – ERROR, "F" - FATAL

### strategySend (<strategyName>)

Processes a message by strategy that identified by <strategyName>

<p>Example Code</p> <pre>strategySend( "OrderFlow5" );</pre>	<p>Processes a message by the default routing strategy with the "OrderFlow5" name.</p> <p><b>Note:</b> only default routing strategies can be used for processing: deliveryToStrategy, OrderFlowStrategy and WheelOrderFlowStrategy.</p>
<p><b>strategyReject (&lt;historyName&gt;, &lt;createBusinessRejectIfFailed&gt;)</b></p> <p>Creates reject on message that was stored in history &lt;historyName&gt;, or BusinessReject (MsgType = 'j'), if &lt;createBusinessRejectIfFailed&gt; is 'true' and rejecting has fail. &lt;createBusinessRejectIfFailed&gt; accepts 'false' by default if it is absent in parameters.</p>	
<p>Example Code</p> <pre>strategyReject( "orders", true );</pre>	<p>Rejects a message that was stored by some strategy in the 'orders' history. Creates 'j' Business Reject when message rejecting has failed.</p> <p><b>Notes:</b> The method is used only in the default routing. Searching in history processes by the 'ClOrdID' key in the original message if its type: 'D', 'AB' or by 'OrigClOrdID' key in original message if its type: 'AC', 'F', 'G'. Inapplicable for other messages.</p>
<p><b>serializeMessage()</b></p> <p>Returns the string view of the current message. SOH will be used as a delimiter.</p>	
<p>Example Code</p> <pre>msg = serializeMessage();</pre>	<p>Returns the string view of the message where SOH will be used as a delimiter and saves it into the "msg" variable.</p>
<p><b>serializeMessage(&lt;tagDelimiter&gt;)</b></p> <p>Returns the string view of the current message. Specified symbol will be used as a delimiter.</p>	
<p>Example Code</p> <pre>msg = serializeMessage( " " );</pre>	<p>Returns the string view of the message where " " will be used as a delimiter and saves it into the "msg" variable.</p>
<p><b>parseMessage(&lt;msg&gt;)</b></p> <p>Parses the specified string message and replaces the current message with the parsed one. After that the old message will be lost. &lt;msg&gt; is a string view of the message. Only messages with SOH as a delimiter can be parsed.</p>	
<p>Example Code</p> <pre>parseMessage( "8=FIX.4.4 \x019=56\x0135=b\x0149=TESTI\x0156= TESTA\x0134=14\x0152=20030204-09: 25:43\x01297=0\x0110=139\x01" );</pre>	<p>Parses a message from the specified string and replaces the current message with it.</p>
<p><b>parseMessage(&lt;msg&gt;, &lt;version&gt;)</b></p> <p>Parses the specified string message using specified FIX protocol and replaces the current message with the parsed one. After that the old message will be lost. &lt;msg&gt; is a string view of the message and &lt;version&gt; is a session version in FIXEdge.properties. Only messages with SOH as a delimiter can be parsed.</p>	

### Example Code

```
parseMessage("8=FIX.4.4
\x019=56\x0135=b\x0149=TESTI\x0156=
TESTA\x0134=14\x0152=20030204-09:
25:43\x01297=0\x0110=139\x01", "FIX
44");
```

Parses a message from the specified string using the FIX44 protocol and replaces the current message with it.



If the 1128 tag is specified in the message for parseMessage(<msg>, <version>) function, then the value of the tag has higher priority during parsing than the version value. For instance:

```
1128=7 and FIXT11:FIX44 and the message has 1092 tag from FIX
5.0:
parseMessage("8=FIXT.1.1
\x019=114\x0135=D\x011128=7\x0149=TESTI\x0156=TESTA\x0134=2\x0152
=20081023-17:03:
05\x0111=90001008\x0155=IBM\x0154=1\x0160=20080717-10:00:
00\x0138=4000\x0140=1\x011092=0\x0110=171\x01", "FIXT11:FIX44");
- the message is parsed successfully as FIX 5.0.
```

```
1128=6 and FIXT11:FIX50 and the message has 1092 tag from FIX
5.0:
parseMessage("8=FIXT.1.1
\x019=114\x0135=D\x011128=6\x0149=TESTI\x0156=TESTA\x0134=2\x0152
=20081023-17:03:
05\x0111=90001008\x0155=IBM\x0154=1\x0160=20080717-10:00:
00\x0138=4000\x0140=1\x011092=0\x0110=171\x01", "FIXT11:FIX50");
- the following error appears:
2017-09-13 11:28:01,558 UTC ERROR [JS_interpreter] 3760
C:\B2BITS\FIXEdge\FIXEdge1\conf\parseMessage.js: failed with
error: Tag 1092 is not defined for this message type. Parsing
stopped at column: 136 in message D with sequence number 2.
```



Please note: the next action in the rule will receive the replaced message!

## History functions

### saveToHistory (<historyName>, <keyFields>, <commonFields>, <expireDateTime>)

Creates a record in history <historyName> with content: <keyFields> – array of composite key or single key, <commonFields> - array of common data fields, <expireDateTime> - timestamp with an expire date-time for the ClearHistory procedure. Returns true if the function succeeded and returns false otherwise.

#### Example Code:

```
key = new Array("snd", "tgt");

common = new Array("ClOrdID", "
dest");

saveToHistory("orders", key,
common, "20101010-10:52:00");
```

Creates new record in the "orders" history with fields set: (snd,tgt,ClOrdID,dest,20101010-10:52:00). The record will be deleted automatically when the ClearHistory task is executed after date-time 20101010-10:52:00.



- If the <expireDateTime> field is not defined in the history definition, use an empty string for it in the script so that it works.
- If the <expireDateTime> is defined in the history definition and it is an empty string, then it will be treated as NULL and the [ClearHistory](#) procedure will delete the record.



If <keyFields> is composite, then all fields in the key set should be filled.

### updateHistory (<historyName>, <keyFields>, <commonFields>, <expireDateTime>)

Modifies fields in the history <historyName> with values specified in <commonFields> and <expireDateTime>. Record for update is searching by <keyFields> criteria. Returns true if the function succeeded and returns false otherwise.

**Example Code:**

```
key = new Array("snd", "tgt");
common = new Array("", "dest2");
updateHistory("orders", key,
common, "20121221-10:52:00");
```

If a record with keys set (snd, tgt) was created by the previous sample for the saveToHistory command, then the 'dest' field value replaces with 'dest2', the '20101010-10:52:00' field value replaces with '20121221-10:52:00'.



- If the <expireDateTime> field is not defined in the history definition, use an empty string for it in the script so that it works.
- If the <expireDateTime> is defined in the history definition and it is an empty string, then it will be treated as NULL and the [ClearHistory](#) procedure will delete the record.



If <keyFields> is composite, then all fields in the key set should be filled. Empty fields in <commonFields> or <expireDateTime> will not be changed.

### getFromHistory (<historyName>, <keyFields>, <columnName>)

Retrieves field value in the history <historyName>. Searching criteria is the <keyFields> for a record and <columnName> for column where value reads. Returns target field value, but if such value wasn't found, returns 'undefined'.

**Example Code:**

```
key = new Array("snd", "tgt");
time = getFromHistory("orders",
key, "ExpireDateTime");
```

Returns the '20121221-10:52:00' value from the record with key (snd,tgt) that has been updated by previous sample.

**Notes:** the column which is created by the <expireDateTime> parameter always has the "ExpireDateTime" name in a history. Other columns have names as described in history description in BL\_Config.xml or as a tag number in string representation if the column name is absent in history description.

### getRecordFromHistory (<historyName>, <keyFields>)

Retrieves fields array from the record with key <keyFields>, located in the history <historyName>. Returns target record, but if such record wasn't found, returns 'undefined'.

**Example Code:**

```
key = new Array("snd", "tgt");
record = getFromHistory("orders",
key);
print(record);
```

Returns a fields array of the record with the key (snd,tgt), then prints content into log. The result will show "snd, tgt, CIOrdID, dest2, 20121221-10:52:00" if the record (snd, tgt) has been saved and updated by previous samples.

**Notes:** Empty fields have NULL values and printing represents it as "<null>". If <keyFields> is composite, then all fields in the key set should be filled.

### removeRecordFromHistory (<historyName>, <keyFields>)

Removes entire record with key <keyFields>, located in the history <historyName>. Returns a number of deleted records.

**Example Code:**

```
key = new Array("snd", "tgt");
removeRecordFromHistory("orders",
key);
```

Removes a record with a key (snd, tgt) from the 'orders' history.

**Notes:** If <keyFields> is composite, then all fields in the key set should be filled.

### removeRecordFromHistoryByCompositeKey (<historyName>, <keyFields>)

Removes entire record by matching of composite key with <keyFields>, located in the history <historyName>. Returns a number of deleted records.

**Example Code:**

```
key = new Array("snd", "");
ndeleted =
removeRecordFromHistoryByCompositeKey
("orders", key);
```

Removes all records with keys which have first the "snd" field in a key.

**Notes:** the composite <keyFields> set should contain at least one field. All empty fields in a key will be ignored in searching.

## Security functions



Available since FIXEdge 5.10.1

### decryptString(<text>)

Decrypts string using AES 128-bit encryption mechanism. It is useful for encrypted tags (554, 925) in Logon messages.

Example Code:

```
encryptedPass = getStringField  
( "554" );  
pass = decryptString  
(encryptedPass);
```

Decrypts tag 554

### hashString(<text>)

Calculates SHA256 hash for the sting.

Example Code:

```
hash = hashString("MyPassword");
```

Getting hash for the "MyPassword" string.

**Note:** string for calculating hash should be not "null".

## Predefined FIX date formats

1. **HHMMSSUtc** for the UTC "HH:MM:SS" date format
2. **HHMMSSsssUtc** or **HHMMSSMilliUtc** for the UTC "HH:MM:SS.sss" date format
3. **HHMMSSMicroUtc** for the UTC "HH:MM:SS.ssssss" date format
4. **HHMMSSNanoUtc** for the UTC "HH:MM:SS.ssssssss" date format
5. **YYYYMM** for the "YYYYMM" date format
6. **YYYYMMDD** for the "YYYYMMDD" date format
7. **YYYYMMWW** for the "YYYYMMWW" date format, where WW - number of week "w1"..."w4"
8. **DATETIMEUtc** for the UTC "YYYYMMDD-HH:MM:SS" date format
9. **DATETIMEsssUtc** or **DATETIMEMilliUtc** for the UTC "YYYYMMDD-HH:MM:SS.sss" date format
10. **DATETIMEMicroUtc** for the UTC "YYYYMMDD-HH:MM:SS.ssssss" date format
11. **DATETIMENanoUtc** for the UTC "YYYYMMDD-HH:MM:SS.ssssssss" date format
12. **YYYYMMDDLmd** for the local market "YYYYMMDD" date format