



TIBCO ActiveMatrix BusinessWorks™ Plug-in for SWIFT

User Guide

Version 6.10.0 | August 2024

Contents

| | |
|--|-----------|
| Contents | 2 |
| Overview | 5 |
| SWIFT Overview | 7 |
| SWIFT Messages | 8 |
| MT Message | 8 |
| MX Message | 10 |
| CBPR+ Message | 11 |
| Basic Message Flow | 11 |
| Communication with the SWIFT Network | 12 |
| Terminologies and Acronyms | 14 |
| Getting Started | 17 |
| Creating a Project | 17 |
| Creating a Load SWIFT MT Schema Shared Resource | 19 |
| Configuring a Process | 21 |
| Testing a Process | 22 |
| Deploying an Application | 23 |
| Overview of TIBCO Business Studio for BusinessWorks | 24 |
| Creating a Load SWIFT MX Schema Shared Resource | 26 |
| Load SWIFT MT Schema Shared Resource | 29 |
| Load SWIFT MX Schema Shared Resource | 31 |
| SWIFT MT Palette | 37 |
| Parse SWIFT MT Activity | 37 |

| | |
|--|-----------|
| Render SWIFT MT Activity | 45 |
| Route SWIFT MT Activity | 52 |
| Generate SWIFT BICPlusIBAN Activity | 59 |
| Validate SWIFT BICPlusIBAN Activity | 62 |
| Guidelines for Validating the Message Structure | 64 |
| Limitations and Suggestions | 65 |
| SWIFT MX Palette | 66 |
| Parse SWIFT MX Activity | 66 |
| Render SWIFT MX Activity | 72 |
| OpenTelemetry Tracing for TIBCO ActiveMatrix BusinessWorks™ Plug-in for SWIFT Palette | 79 |
| Custom Tags | 79 |
| Configuring Advanced Options | 80 |
| Parsing and Validating SWIFT MT Messages Using SwiftCheck | 80 |
| SwiftCheck Options | 81 |
| Parsing and Validating SWIFT MX/CBPR+ Messages Using SwiftMXCheck | 82 |
| SwiftMXCheck Options | 83 |
| Using Validation Filters | 85 |
| Filter Groups | 85 |
| Filters | 86 |
| Configuring Customized MX Java Rules | 92 |
| MX Java Rule Code Example | 93 |
| MX Message Rule Example | 95 |
| Processing Acknowledgment Messages | 97 |
| Reconciling Acknowledgment Messages | 97 |
| Migrating Projects to the Current SWIFT Standards | 99 |
| Migrating Projects with SWIFT MT Activities | 99 |
| Migrating Projects with SWIFT MX Activities | 100 |
| Migrating from Version 5.23.0 of TIBCO ActiveMatrix BusinessWorks™ Plug-in for | 101 |

| | |
|---|------------|
| SWIFT | |
| Managing Logs | 103 |
| Log Levels | 103 |
| Setting Up Log Levels | 104 |
| Exporting Logs to a File | 106 |
| Error Codes | 107 |
| TIBCO Documentation and Support Services | 109 |
| Legal and Third-Party Notices | 111 |

Overview

TIBCO ActiveMatrix BusinessWorks™ Plug-in for SWIFT provides the functionalities to render, parse, and validate MT and MX messages, as well as route MT messages to different activities based on different message type.

TIBCO ActiveMatrix BusinessWorks Plug-in for SWIFT contains a SWIFT MT palette and a SWIFT MX palette. The activities in the SWIFT MT palette activity are used to handle the MT messages and the activities in the SWIFT MX palettes are used to handle the MX messages.

The following activities are available in the SWIFT MT palette:

- [Parse SWIFT MT Activity](#)
You can use this activity to validate an MT message and parse it to XML format.
- [Render SWIFT MT Activity](#)
You can use this activity to render an MT message from XML format to the MT message format.
- [Route SWIFT MT Activity](#)
You can use this activity to route a message to different activities based on the message types.
- [Generate SWIFT BICPlusIBAN Activity](#)
You can use this activity to generate an IBAN.
- [Validate SWIFT BICPlusIBAN Activity](#)
You can use this activity to validate an IBAN.

i Note: Before parsing or rendering an MT message, you have to load the corresponding message type schema. See [Load SWIFT MT Schema Shared Resource](#) for more information.

The following activities are available in the SWIFT MX palette:

- [Parse SWIFT MX Activity](#)
You can use this activity to validate an MX message and parse it to XML format.
- [Render SWIFT MX Activity](#)

You can use this activity to generate a specific MX message.

i **Note:** Before parsing or rendering an MX message, you have to load the corresponding message type schema. For more information, see [Load SWIFT MX Schema Shared Resource](#).

SWIFT Overview

Society for Worldwide Interbank Financial Telecommunication (SWIFT) is an organization that provides an automated fast and safe means of sending and receiving financial messages between financial institutions worldwide.

SWIFT users include financial institutions such as banks, brokers, dealers, or investment managers. Typical users of SWIFT deal with payments, securities, foreign exchange, money markets, treasury, and trade. All SWIFT users are identified in the SWIFT network by a unique address called a Bank Identifier Code (BIC). To exchange messages over the SWIFT network, every user must have at least one unique BIC.

SWIFT processes information such as data, texts, or commands in the form of messages. Also, SWIFT provides the following two applications that make all messaging functions and facilities available to users:

- General Purpose Application (GPA)
Controls how users communicate within SWIFT.
- Financial Application (MT)
Controls the user-to-user messaging facilities within SWIFT.

SWIFT Messages

The **SWIFT** messages mainly include MT and MX messages. A SWIFT message must conform to SWIFT standards and structure requirements.

MT Message

The SWIFT MT messages start with the letters MT followed by a 3-digit number. The first digit represents the message category, the second digit represents a group of related parts in a transaction life cycle and a third digit represents the message type.

MT Message Types

SWIFT MT messages include the following types:

System Messages

Sent by a user to the SWIFT system (delivery notifications, retrievals) or the other way around (retrieved messages, non-delivery warnings). MT0nn represents the system messages. These are used for system-level information, such as delivery notifications and nondelivery warnings.

User to User Messages

Enable financial transactions and are sent from one user to another. MT1nn through MT9nn is the User to User message representation.

Service Messages

Also known as control messages and related either to system commands LOGIN, SELECT, or QUIT or to message acknowledgments. For example, when you send a message to the SWIFT network, SWIFT accepts the message and sends you an ACK if the syntax of the message is correct. If not, it returns a NAK.

MT Message Structure

All SWIFT MT messages are ASCII text messages that have the following structure:

- Basic header block

{1:} represents the Basic header block. This includes details, such as the application ID, service ID, address of the logical terminal, session number, sequence number, and so on. The application ID in this block helps you identify whether a message is a GPA message (system message) or an MT message (user to user message). For example, 'M' indicates that the message is an MT message and 'A' indicates that the message is a GPA message.
- Application header block

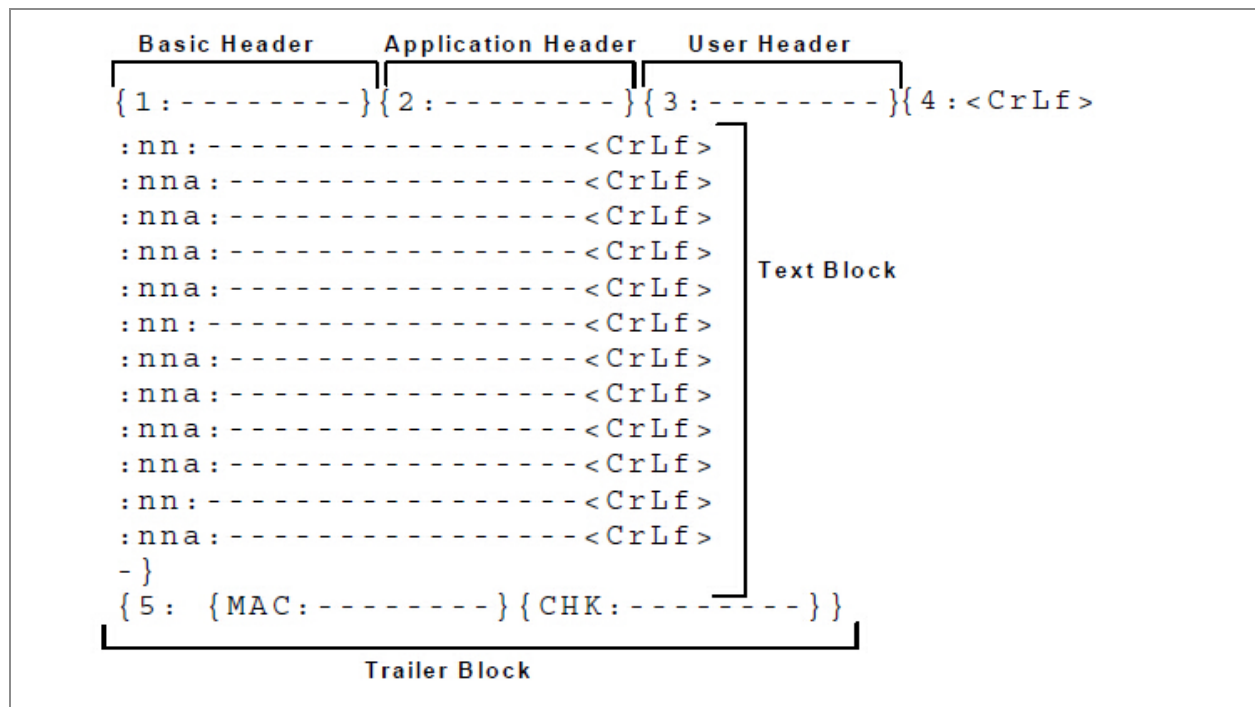
{2:} represents the Application header block. Application headers contain two types: input and output. The structure of the block varies depending on the type of the application header. This header block typically includes details, such as the message type, message priority, delivery monitoring, and so on.
- User header block

{3:} represents the User header block. The block includes details, such as the banking priority code and so on. This is an optional block.
- Text block

{4:} represents the Text block. This block contains the actual MTnnn message. This block includes details, such as the ordering customer, beneficiary customer, amount, currency code, date, and so on. This block consists of field tags of the format:nna: where nn is a number and a is an optional letter, which might be present on selected tags. The symbol CrLf is a control character and it represents Carriage Return or Line Feed. The symbol CrLf is a mandatory delimiter in this block.
- Trailer block

{5:} represents the Trailer block and a message always ends in a trailer block. It is used for control purposes and includes details, such as message authentication code and checksum calculated for all the message types.

The following figure is a graphical representation of a typical SWIFT MT message:



MX Message

SWIFT MX message includes the following types:

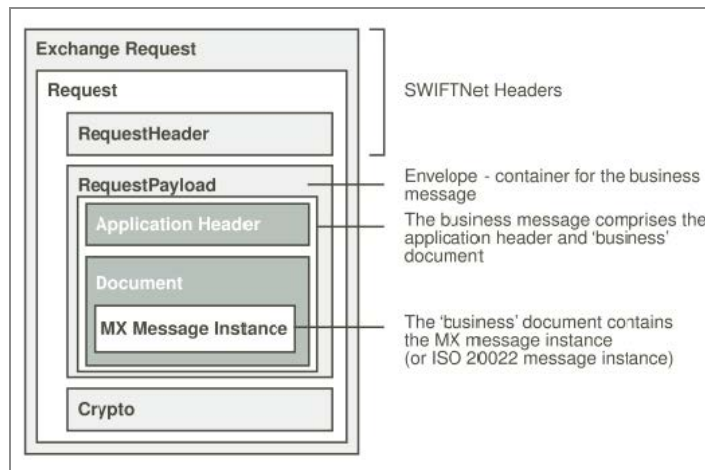
InterAct messages

These MX messages are used by the InterAct messaging service, and also by the key market infrastructures around the world. The InterAct messaging service is used for exchanging XML based financial messages and data between users. The structure of the InterAct messages mainly includes the transport, header, and document.

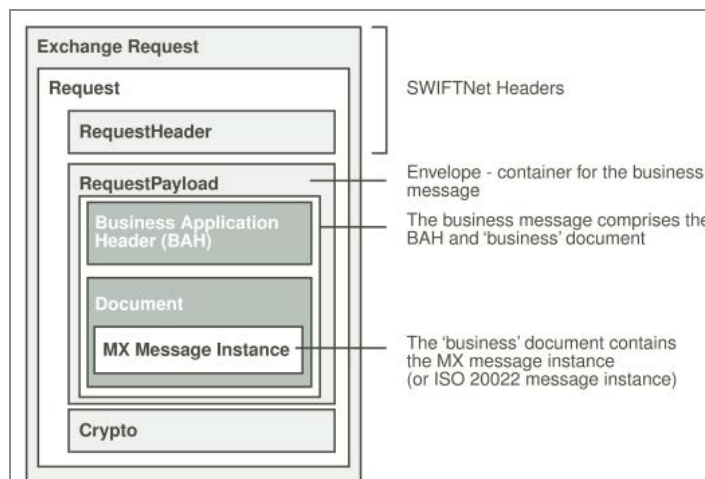
SAA messages

These messages are MX messages delivered to SWIFT Alliance Access (SAA). The structure of the SAA messages mainly includes the transport, header, and document.

The following figure shows the structure of the InterAct MX message blocks with Application Header:



The following figure shows the structure of the InterAct MX message blocks with Business Application Header:



CBPR+ Message

Cross-Border Payments and Reporting Plus (CBPR+) specifications define how the plug-in uses ISO 20022 for cross-border payments and cash reporting on the SWIFT network.

Basic Message Flow

The basic SWIFT message flow involves syntax validation and confirmation of receipt and acceptance.

When you send a message to the SWIFT network, SWIFT validates the syntax of the message. If the message is correct, SWIFT accepts the message, sends you an ACK, and attempts to deliver the message to the receiver. If the message does not comply with the standards, SWIFT rejects it and returns a NAK. The NAK contains an error code, which helps the sender identify the type of error and its location. The sender can then correct the message before resending it to SWIFT.

SWIFT delivers the message as soon as the receiver is logging in to the SWIFT network. The interface at the receiver's end automatically confirms the receipt and acceptance of the message by sending a UAK. When a UAK is received by SWIFT, the message is considered to be delivered. If the message received is corrupted, the interface of the receiver sends a UNK to SWIFT, and SWIFT attempts to redeliver the message.

Communication with the SWIFT Network

TIBCO ActiveMatrix BusinessWorks™ Plug-in for SWIFT does not communicate directly with the SWIFT network. To communicate with the SWIFT network, you must have SWIFT Alliance Access (SAA), SWIFT Alliance Entry (SAE), or any other interface approved by SWIFT.

Messages can be produced on a variety of user applications, but they can only be sent to SWIFT through SAA or SAE. Similarly, messages can be processed on a variety of user applications, but can only be received through SAA or SAE.

You can communicate with SAA by using one of the following interfaces:

- CASmf

CASmf is a software used for communication between SAA and other user applications. CASmf uses a MAPID, a name given to each instance of the messaging application, to establish the communication between the user application and SAA. The MAPID is defined in the user's host and in the SWIFT interface.

CASmf also provides APIs to developers of the user application. With APIs, the user's host environment can communicate with the SWIFT interface, establishing a real-time session. After a real-time session is established, financial messages can be exchanged. APIs can open, close, or abort a session, and send or receive data. The CASmf software uses TCP/IP as the communication protocol.

- FTP

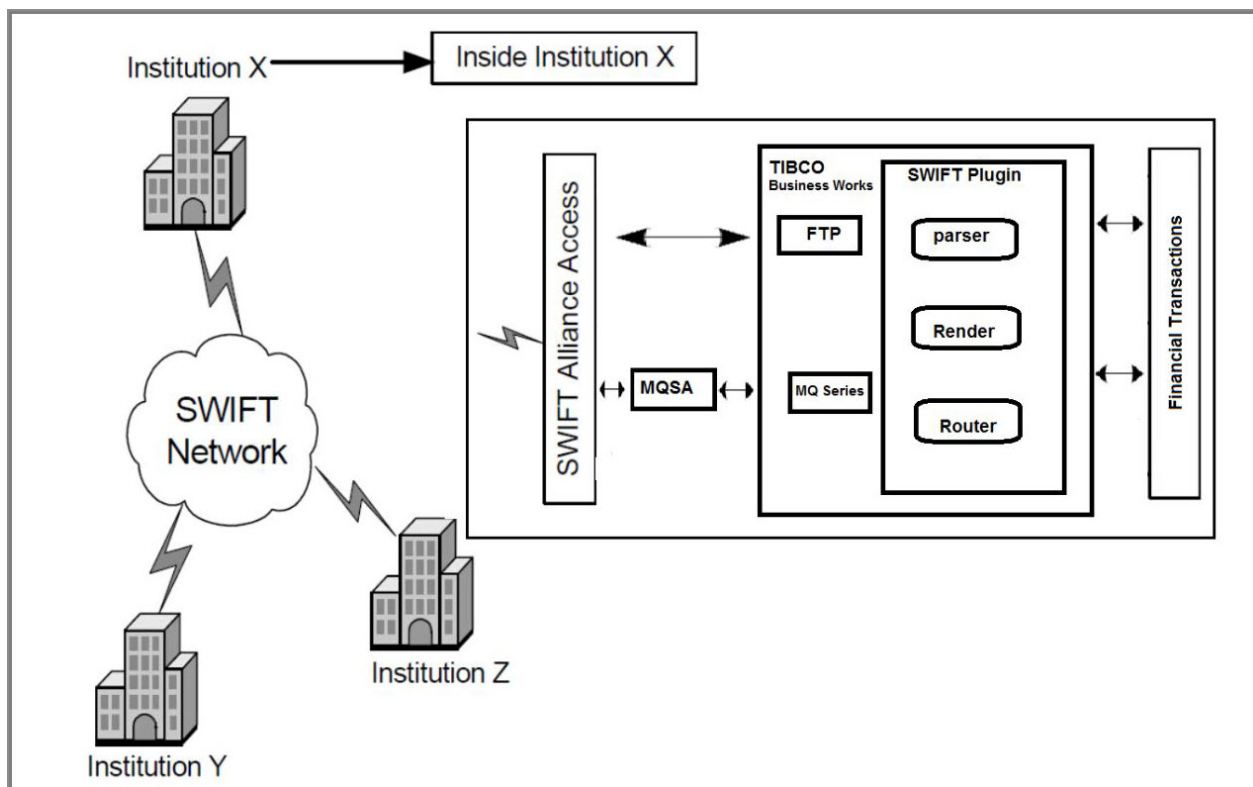
An FTP server can be used for communication between SWIFT interfaces and user

applications. The user application transfers SWIFT messages to a specified directory on the server. SAA picks up the SWIFT messages from this directory, and sends them to the SWIFT network.

- MQSA

MQSA is a software used for communication between SAA and MQSeries. The MQSA interface is based on the SWIFT Alliance Development Kit (ADK). It uses ADK functions to communicate with SWIFT Alliance, and MQSeries functions to access message queuing services.

The following figure shows how the preceding interfaces are used to integrate SAA with the user application:



Terminologies and Acronyms

Terminologies and acronyms used in this manual are provided for your reference.

Terminologies

The following table explains the terminologies used in this manual:

| Terminology | Meaning |
|-----------------|--|
| ADK | Alliance Development Kit. ADK is a SWIFT product that developers of third-parties and financial institutions use to build their own applications for Alliance Access. |
| BIC | Business Identifier Code. BIC is an international standard for identification of institutions within the financial services industry. BICs are used in automated processing. They unambiguously identify a financial institution or a non-financial institution. The ISO 9362 standard specifies the elements and the structure of a BIC. A BIC consists of either eight or eleven contiguous characters. These characters comprise either the first three, or all four, of the following components: party prefix, country code, party suffix, and branch identifier. The International Organization for Standardization has designated SWIFT as the BIC registration authority. |
| CBPR+ | An XML message definition for cross-border payments and cash reporting on the SWIFT network. |
| Delivery Report | The SWIFT network can, either on its own or in response to a request from the user, report the delivery status of a user message. Typically, this message specifies whether the message was successfully delivered, could not be delivered, was |

| Terminology | Meaning |
|---------------------|---|
| | acknowledged or not acknowledged (ACK or NACK) by the receiver or sender. Such a message is called a Delivery Report. |
| MT | A traditional message type for use on the SWIFT network. |
| MX | An XML message definition for use on the SWIFT network. |
| SAA | SWIFT Alliance Access. SAA is a prime multi-platform messaging interface designed to connect business applications to SWIFT messaging services. Customers can use SAA to connect single or multiple destinations to SWIFT with the maximum automation of system management tasks. |
| SAE | SWIFT Alliance Entry. SAE is a messaging interface that SWIFT develops for low-volume customers that use a single destination. SAE offers the flexibility and control of a private infrastructure. |
| SWIFT | Society for Worldwide Interbank Financial Telecommunication. This organization maintains a global store and forward network for secure financial messaging. In addition, it defines standard formats for interbank messages. |
| Transmission Report | The transmission report is a message that states whether the user message was accepted or not (ACK or NACK) by the network. Since the SWIFT network is a store and forward network, every user message is stored by the network before being forwarded to the receiver. At the time of storing, the network also validates the message. |

Acronyms

The following table explains the acronyms used in this manual:

| Acronym | Spelled-out Form |
|---------|--|
| ACK | Positive Acknowledgment |
| API | Application Programming Interface |
| BEI | Business Entity Identifier |
| IBAN | International Bank Account Number |
| JMS | Java Messaging Service |
| JNI | Java Native Interface |
| LTA | Logical Terminal Address |
| NAK | Negative Acknowledgment |
| UAK | User Positive Acknowledgment |
| UNK | User Negative Acknowledgment |
| UETR | Unique End-to-End Transaction Reference. |

Getting Started

This tutorial is designed for the beginners who want to use TIBCO ActiveMatrix BusinessWorks™ Plug-in for SWIFT in TIBCO Business Studio.

All the operations are performed in TIBCO Business Studio. See [Overview of TIBCO Business Studio for BusinessWorks](#) to get familiar with TIBCO Business Studio.

The following procedure takes the Parse SWIFT MT activity as an example to demonstrate how to parse an MT message.

A basic procedure of using TIBCO ActiveMatrix BusinessWorks Plug-in for SWIFT includes:

1. [Creating a Project](#)
2. [Creating a Load SWIFT MT Schema Shared Resource](#)
3. [Configuring a Process](#)
4. [Testing a Process](#)
5. [Deploying an Application](#)

Creating a Project

The first task by using the plug-in is creating a project. After creating a project, you can add resources and processes.

An Eclipse project is an application module configured for TIBCO ActiveMatrix BusinessWorks™. An application module is the smallest unit of resources that is named, versioned, and packaged as part of an application.

Procedure

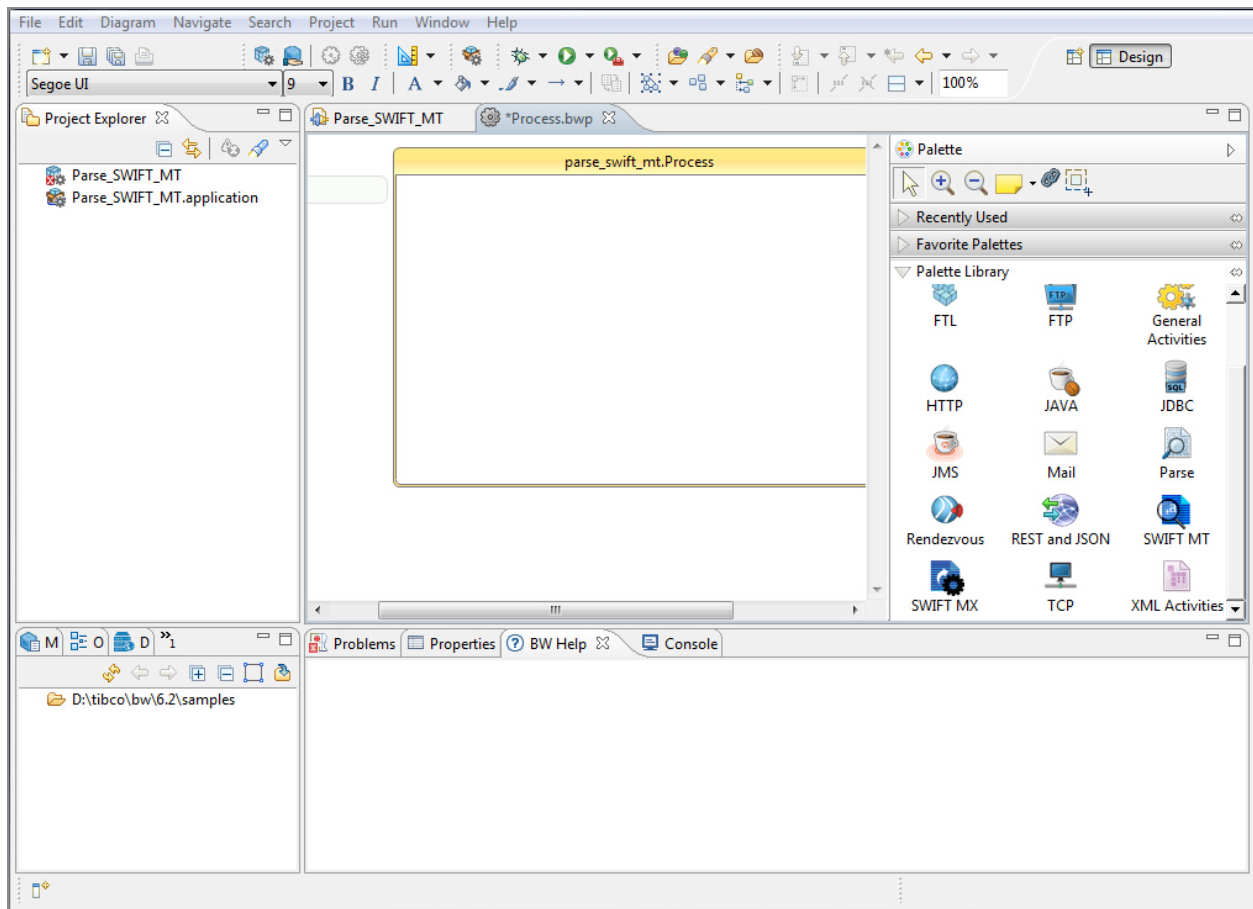
1. Start TIBCO Business Studio using one of the following ways:
 - On Microsoft Windows, click **Start > All Programs > TIBCO > TIBCO_HOME > TIBCO Business Studio *version_number* > Studio for Designers**.

- On Mac OS and Linux, run the TIBCO Business Studio executable file located in the *TIBCO_HOME/studio/version_number/eclipse* directory.
2. From the menu, click **File > New > BusinessWorks Resources** to open the BusinessWorks Resource wizard.
 3. To open the BusinessWorks Resource wizard, in the Select a wizard dialog, click **BusinessWorks Application Module** and click **Next** to open the New BusinessWorks Application Module wizard.
 4. In the Project dialog, configure the project that you want to create:
 - a. In the **Project name** field, enter a project name.
 - b. By default, the created project is in the workspace current in use. If you do not want to use the default location for the project, clear the **Use default location** checkbox and click **Browse** to select a new location.
 - c. Use the default version of the application module, or enter a new version in the **Version** field.
 - d. Keep the **Create empty process** and **Create Application** checkboxes selected to create an empty process and an application when creating the project.
 - e. Select the **Use Java configuration** checkbox if you want to create a Java module.

A Java module provides the Java tooling capabilities.
 - f. Click **Finish** to create the project.

Result

The project with the specified settings is displayed in the Project Explorer view.



To directly access documentation for this product, double-click the following file:

<https://docs.tibco.com/products/tibco-activematrix-businessworks-plugin-for-swift-6-10-0>

Creating a Load SWIFT MT Schema Shared Resource

After creating a project, you can add a **Load SWIFT MT** schema shared resource to select a SWIFT specification and message type. This shared resource is used for the Parse SWIFT MT, Render SWIFT MT, and Route SWIFT MT activities.

Before you begin

The Load SWIFT MT Schema shared resource is available at the **Resources** level. Ensure that you have created a project, as described in [Creating a Project](#).

Procedure

1. Expand the created project in the Project Explorer view.
2. To open the schema loader wizard, right-click the **Resources** folder, and then click **New > Load SWIFT MT Schema**.
3. The resource folder, package name, and resource name of the Load SWIFT MT Schema are provided by default. If you do not want to use the default configurations, change them accordingly. Then, click **Finish**.

The following schema loader editor appears:

MT Schemaloader Configuration

Enable Userheader Validation: ☒

Enable UETR Validation: ☒

Specification: SWIFT November 2023 specification

Message Types:

| Loaded | Message | Description |
|-------------------------------------|---------|--|
| <input type="checkbox"/> | 008 | System Request To Quit |
| <input type="checkbox"/> | 009 | System Request To Logout |
| <input type="checkbox"/> | 010 | Non-Delivery Warning |
| <input type="checkbox"/> | 011 | Delivery Notification |
| <input type="checkbox"/> | 012 | Sender Notification |
| <input type="checkbox"/> | 015 | Delayed NAK |
| <input type="checkbox"/> | 019 | Abort Notification |
| <input type="checkbox"/> | 020 | Retrieval Request (Text and History) |
| <input type="checkbox"/> | 021 | Retrieved Message (Text and History) |
| <input checked="" type="checkbox"/> | 022 | Retrieval Request (History) |
| <input type="checkbox"/> | 023 | Retrieved Message (History) |
| <input type="checkbox"/> | 028 | SWIFTNet FIN Copy Message Status Request |

On creation of the MT Schema Shared Resource, the **Enable Userheader Validation** and **Enable UETR Validation** checkboxes are by default checked.

4. From the **Specification** list, select a SWIFT specification.
5. In the **Message Types** table, find the appropriate message type schema, and then select the checkbox in the **Loaded** column.
6. Click **Load Selected** to load the message type schemas.



Note: To make the message types available or unavailable in the project, click **Load Selected** or **Unload Selected** after selecting or deselecting the message type.

Configuring a Process

After creating a project, an empty process is created. You can add activities to the empty process to complete a task. For example, parse a SWIFT MT message.

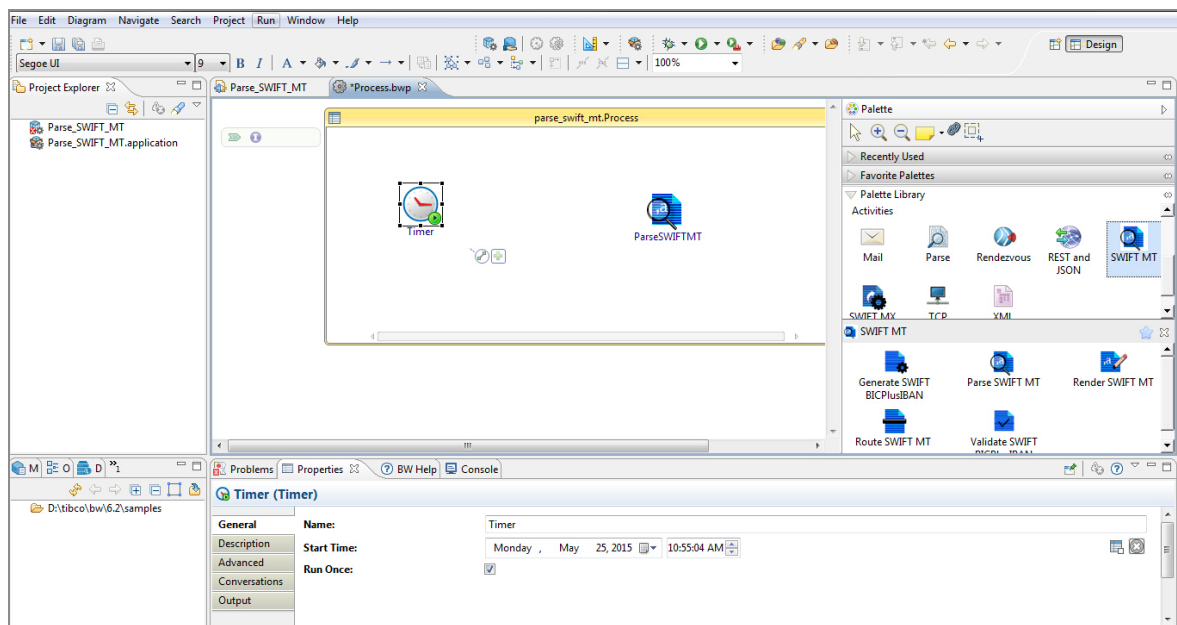
Before you begin


Ensure that you have created an empty process and a Load SWIFT MT Schema shared resource. For more information, see [Creating a Project](#) and [Creating a Load SWIFT MT Schema Shared Resource](#) for more information.

Procedure

1. In the Project Explorer view, click the created project and open the empty process from the Processes folder.
2. Select an activity from the Palette view and drop it in the Process editor.

For example, select and drop the Timer activity from the General Activities palette and the Parse SWIFT MT activity from the SWIFT MT palette.



3. Drag the  icon to create a transition between the added activities. Configure the added activities.
See [SWIFT MT Palette](#) for more information about the configuration parameters.
4. To save the project., click **File > Save**.

Testing a Process

After configuring a process, you can test the process to check whether it completes your task.

Before you begin

Ensure that you have configured a process. See [Configuring a Process](#) for more information.

Procedure

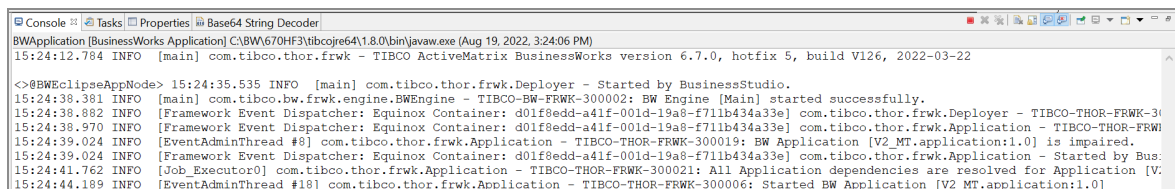
1. On the toolbar, click  **Debug > Debug Configurations**.

2. Click **BusinessWorks Application > BWApplication** in the left panel.

By default, all applications in the current workspace are selected on the **Applications** tab. Ensure that only the application you want to debug is selected on the **Applications** tab in the right panel.

3. Click **Debug** to test the process in the selected application.

TIBCO Business Studio changes to the Debug perspective. The debug information is displayed in the Console view.

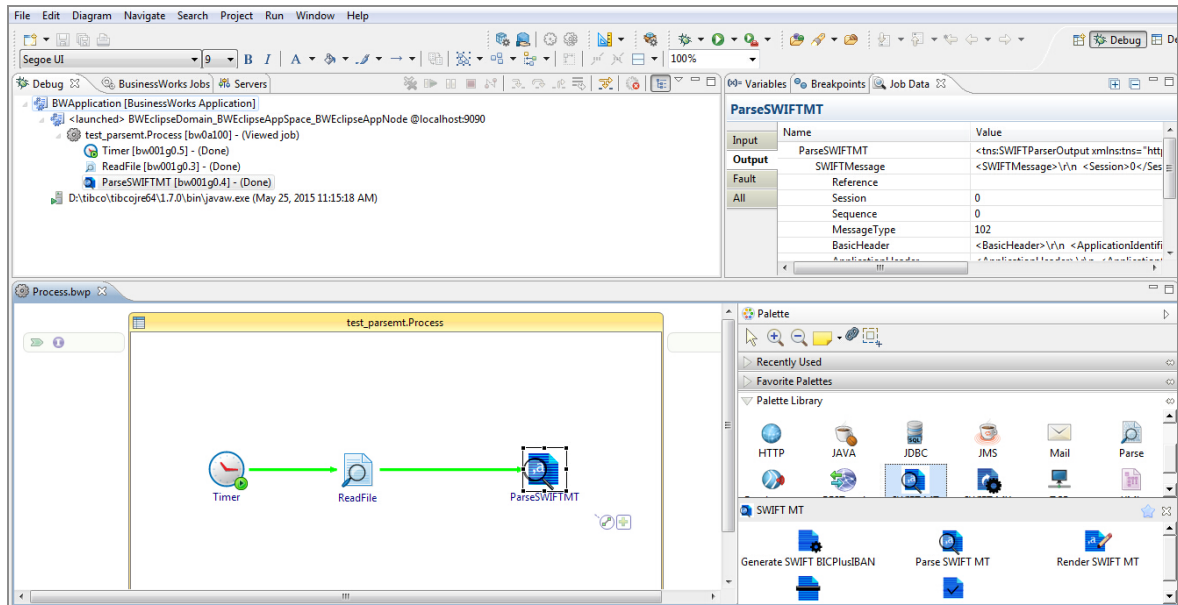


```

Console | Tasks | Properties | Base64 String Decoder
BWApplication [BusinessWorks Application] C:\BW\670HF3\tibcojre64\1.8.0\bin\javaw.exe (Aug 19, 2022, 3:24:06 PM)
15:24:12.784 INFO [main] com.tibco.thor.frwk - TIBCO ActiveMatrix BusinessWorks version 6.7.0, hotfix 5, build V126, 2022-03-22

<>@BWEclipseAppNode> 15:24:35.535 INFO [main] com.tibco.thor.frwk.Deployer - Started by BusinessStudio.
15:24:38.381 INFO [main] com.tibco.bw.frwk.engine.BWEngine - TIBCO-BW-FRWK-300002: BW Engine [Main] started successfully.
15:24:38.882 INFO [Framework Event Dispatcher: Equinox Container: d01f8edd-a41f-001d-19a8-f711b434a33e] com.tibco.thor.frwk.Deployer - TIBCO-THOR-FRWK-300002: BW Engine [Main] started successfully.
15:24:38.970 INFO [Framework Event Dispatcher: Equinox Container: d01f8edd-a41f-001d-19a8-f711b434a33e] com.tibco.thor.frwk.Application - TIBCO-THOR-FRWK-300002: BW Engine [Main] started successfully.
15:24:39.024 INFO [EventAdminThread #8] com.tibco.thor.frwk.Application - TIBCO-THOR-FRWK-300019: BW Application [V2_MT.application:1.0] is impaired.
15:24:39.024 INFO [Framework Event Dispatcher: Equinox Container: d01f8edd-a41f-001d-19a8-f711b434a33e] com.tibco.thor.frwk.Application - Started by Bus:
15:24:41.762 INFO [Job_Executor0] com.tibco.thor.frwk.Application - TIBCO-THOR-FRWK-300021: All Application dependencies are resolved for Application [V:
15:24:44.189 INFO [EventAdminThread #18] com.tibco.thor.frwk.Application - TIBCO-THOR-FRWK-300006: Started BW Application [V2_MT.application:1.0]
  
```

4. On the **Debug** tab, expand the running process and click an activity.
5. In the upper-right corner, click the **Job Data** tab, and then click the **Output** tab to check the activity output.



Deploying an Application

After testing, if the configured process works as expected, you can deploy the application that contains the configured process into a runtime environment, and then use the `bwadmin` utility to manage the deployed application.

Before deploying an application, you must generate an application archive, which is an enterprise archive (EAR) file that is created in TIBCO Business Studio. See [Overview of TIBCO Business Studio for BusinessWorks](#) for more information.

Deploying an application involves the following tasks:








1. Uploading an application archive
2. Deploying an application archive
3. Starting an application

See *TIBCO ActiveMatrix BusinessWorks™ Administration* for more information about how to deploy an application.

Overview of TIBCO Business Studio for BusinessWorks

TIBCO Business Studio™ for BusinessWorks™ is an Eclipse-based integration development environment that is used to design, develop, and test ActiveMatrix BusinessWorks applications. The studio provides a workbench in which you can create, manage, and navigate resources in your workspace. A *workspace* is the central location on your computer where all data files are stored.

The following table introduces the workbench UI elements:

| UI Element | Description |
|---------------------|--|
| Menu | Contains menu items such as File, Edit, Navigate, Search, Project, Run, Window, and Help. |
| Toolbar | <p>Contains the following buttons for frequently used commands:</p> <ul style="list-style-type: none"> • New  • Save  • Enable/Disable Business Studio capabilities  • Create a new BusinessWorks Application Module  • Debug As  • Run As  |
| Perspectives | <p>Contains an initial set and layout of views that are required to perform a certain task. TIBCO Business Studio for BusinessWorks launches the Design perspective by default. Use the Design perspective when designing a process and the Debug perspective when testing and debugging a process. To change the perspective, select Window > Open Perspective > <i>perspective_name</i> from the main menu. Alternatively, click the Open Perspective  button from the top upper right of the workbench and select the perspective.</p> |

| UI Element | Description |
|----------------|---|
| Views | Lists the resources and helps you navigate within the workbench. For example, the Project Explorer view displays the ActiveMatrix BusinessWorks applications, modules, and other resources in your workspace, and the Properties view displays the properties for the selected resource. To open a view, select Window > Show View > <i>view_name</i>view_name from the main menu. |
| Editors | Provides a canvas to configure, edit, or browse a resource. Double-click a resource in a view to open the appropriate editor for the selected resource. For example, double-click on a process (<code>MortgageAppConsumer.bwp</code>) in the Project Explorer view to open the process in the editor. |
| Palette | Contains a set of widgets and a palette library. A <i>palette</i> groups activities that perform similar tasks, and provides quick access to activities when configuring a process. |

Creating a Load SWIFT MX Schema Shared Resource

You can use the Load SWIFT MX Schema shared resource to load the SWIFT MX schema.

Before you begin

Before creating the Load SWIFT MX Schema shared resource, you have to download the following files, and then save them to the *TIBCO_HOME/bw/palettes/swift/version_number/bin/xsd/specification_year* directory:


















- The required MX message .xsd files, which can be downloaded from https://www2.swift.com/uhbonline/books/a2z/standards_mx.htm
- The required CBPR+ message .xsd files, which can be downloaded from https://www2.swift.com/mystandards/#/group/Cross_Border_Payments

After downloading the .xsd files, you need to rename as shown in the example:

For example: You must rename CBPRPlus_ISO_20022_Portfolio_November_2022_Release_CBPRPlus-pacs_009_001_08_COV_FinancialInstitutionCreditTransfer_20220208_1214_iso15.xsd to

CBPRPlus-pacs_009_001_08_COV.xsd.

You can view the files as shown in the following image after renaming all CBPR+ .xsd files.

| | |
|--|----------|
|  CBPRPlus-camt_029_001_09 | XSD File |
|  CBPRPlus-camt_052_001_08 | XSD File |
|  CBPRPlus-camt_053_001_08 | XSD File |
|  CBPRPlus-camt_054_001_08 | XSD File |
|  CBPRPlus-camt_056_001_08 | XSD File |
|  CBPRPlus-camt_057_001_06 | XSD File |
|  CBPRPlus-camt_060_001_05 | XSD File |
|  CBPRPlus-pacs_002_001_10 | XSD File |
|  CBPRPlus-pacs_004_001_09 | XSD File |
|  CBPRPlus-pacs_008_001_08 | XSD File |
|  CBPRPlus-pacs_008_001_08_STP | XSD File |
|  CBPRPlus-pacs_009_001_08 | XSD File |
|  CBPRPlus-pacs_009_001_08_ADV | XSD File |
|  CBPRPlus-pacs_009_001_08_COV | XSD File |
|  CBPRPlus-pacs_010_001_03 | XSD File |
|  CBPRPlus-pain_001_001_09 | XSD File |
|  CBPRPlus-pain_002_001_10 | XSD File |

- The header and app header .xsd files, including \$ahV10.xsd, Sw.xsd, SwInt.xsd, SwGbl.xsd, SwSec.xsd, Doc.xsd, and head.001.001.01.xsd, which can be downloaded from <https://www2.swift.com/dlc/pages/search.faces>
- The SAA .xsd file, that is SAA_XML_v2_0_3.xsd, which can be downloaded from [Tip-5017588 Where can I find the xsd schemas for Alliance Lite2?](#)

Procedure

1. Expand the created project in the Project Explorer view.
2. To open the mxschemaloader wizard, right-click the **Resources** folder, and then click **New > Load SWIFT MX Schema**.
3. The resource folder, package name, and resource name are provided by default. If you do not want to use the default configurations, change them accordingly. Click **Finish** to open the mxschemaloader editor.

When you select MX as message type, the plug-in displays all the available SWIFT MX schemas in the dialog and when you select CBPR+ as message type, the plug-in displays all the available SWIFT CBPR+ schemas in the dialog. You can use this shared resource in your project.

mxschemaloader Configuration

Specification: SWIFT November 2023 specification

Message Type: MX

| Message | Description |
|-----------------|--|
| SwInt | InterAct |
| \$ahV10 | AppHeader |
| head.001.001.01 | ISO AppHeader V1 |
| head.001.001.02 | ISO AppHeader V2 |
| SAA_XML_v2_0_2 | SAA (SWIFT ALLIANCE ACCESS) v2_0_2 |
| SAA_XML_v2_0_3 | SAA (SWIFT ALLIANCE ACCESS) v2_0_3 |
| acmt.001.001.08 | AccountOpeningInstructionV08 |
| acmt.002.001.08 | AccountDetailsConfirmationV08 |
| acmt.003.001.08 | AccountModificationInstructionV08 |
| acmt.004.001.06 | GetAccountDetailsV06 |
| acmt.005.001.06 | RequestForAccountManagementStatusReportV06 |
| acmt.006.001.07 | AccountManagementStatusReportV07 |

mxschemaloader Configuration

Specification: SWIFT November 2023 specification

Message Type: CBPR+

| Message | Description |
|------------------------------|--|
| CBPRPlus-camt_029_001_09 | ResolutionOfInvestigationV09 |
| CBPRPlus-camt_052_001_08 | BankToCustomerAccountReportV08 |
| CBPRPlus-camt_053_001_08 | BankToCustomerStatementV08 |
| CBPRPlus-camt_054_001_08 | BankToCustomerDebitCreditNotificationV08 |
| CBPRPlus-camt_056_001_08 | FItoFIPaymentCancellationRequestV08 |
| CBPRPlus-camt_057_001_06 | NotificationToReceiveV06 |
| CBPRPlus-camt_060_001_05 | AccountReportingRequestV05 |
| CBPRPlus-pacs_002_001_10 | FItoFIPaymentStatusReportV10 |
| CBPRPlus-pacs_004_001_09 | PaymentReturnV09 |
| CBPRPlus-pacs_008_001_08 | FItoFICustomerCreditTransferV08 |
| CBPRPlus-pacs_008_001_08_STP | STPFItoFICustomerCreditTransferV08 |
| CBPRPlus-pacs_009_001_08 | FinancialInstitutionCreditTransferV08 |

Note: If no .xsd file is matched for the message type in the **Message** column, the message type is disabled.

4. Save this shared resource.

Load SWIFT MT Schema Shared Resource

You can use the Load SWIFT MT Schema shared resource to select a SWIFT Specification and load the message type schemas. The schemas loaded using the Load SWIFT MT shared resource are used by the Parse SWIFT MT, Route SWIFT MT, and Render SWIFT MT activities. You can add only one Load MT Schema shared resource in a project.

General

The following table lists the configurations in the **General** panel of the Load SWIFT MT Schema shared resource:

| Field | Module Property? | Description |
|--------------------|------------------|--|
| Package | No | The name of the package where the new shared resource is added. |
| Name | No | The name to be displayed as the label for the shared resource in the process. By default the name displayed is <i>MTSchemaLoaderResource</i> . |
| Description | No | A short description for this shared resource. |

MT Schemaloader Configuration

The following table lists the configurations in the **MT Schemaloader Configuration** panel of the Load SWIFT MT Schema shared resource:

| Field | Description |
|-------------------------------------|---|
| Enable Userheader Validation | Select the checkbox to enforce validation rule that at least one tag out of 103, 113, 108, 119 or 115 must be present in the user header. |

| Field | Description |
|---|---|
| | <p>Note: Missing user header is legal if there is no header treating it as valid (Except in UETR case where it is validated based on the configuration).</p> |
| Enable UETR Configuration | Select the checkbox to enforce a validation of the UETR (121) field for MT's 103, 103 REMIT, 103 STP, 202, 202 COV, 205, and 205 COV. |
| Specification | <p>You can select a SWIFT specification from the Specification list. Specification 2023 and specification 2024 are supported. The default value is SWIFT November 2024 specification.</p> |
| Message Types | <p>Lists all the message types supported by the selected SWIFT specification.</p> <p>Select the checkboxes next to the message types, and then click Load Selected to load message schemas to a project. To unload the message schemas, click Unload Selected.</p> <p>Note: When another schema is initiated, If you do not select all the schemas at once, then the previous selection might be removed from the project.</p> |
| <p>i Note: See Creating a Load SWIFT MT Schema Shared Resource to create the Load SWIFT MT Schema shared resource.</p> | |

Load SWIFT MX Schema Shared Resource

The Load SWIFT MX Schema shared resource is used to select a SWIFT Specification and load or unload the MX message schemas. The Parse SWIFT MX and Render SWIFT MX activities use the schemas that are loaded by the Load SWIFT MX Schema shared resource. You can add only one Load MX Schema shared resource in a project.

General

The following table lists the configurations in the **General** panel of the Load SWIFT MX Schema shared resource:

| Field | Module Property? | Description |
|--------------------|------------------|---|
| Package | No | The name of the package where the new shared resource is added. |
| Name | No | The name to be displayed as the label for the shared resource in the process. |
| Description | No | A short description for this shared resource. |

MX Schemaloader Configuration

The following table lists the configurations in the **MX Schemaloader Configuration** panel of the Load SWIFT MX Schema shared resource:

| Field | Description |
|----------------------|--|
| Specification | <p>You can select a SWIFT specification from the Specification list. Specification 2023 and specification 2024 are supported. The default value is SWIFT November 2024 specification.</p> <p>This lists all the message types supported by the SWIFT specification.</p> |

| Field | Description |
|---------------------|---|
| Message Type | You can select the MX or CBPR+ message type from the dropdown. After you select the message type, the system displays the supported messages. |

Note: See [Creating a Load SWIFT MX Schema Shared Resource](#) to create the Load SWIFT MT Schema shared resource.

Supported MX Messages

The plug-in supports part of SWIFT MX Standard Release 2023 and 2024, and therefore, supports a certain set of MX messages that conform to SWIFT MX Standard Release 2023 and 2024.

The plug-in supports extended rules validation for the following MX messages:

```
<!--acmt 2024 -->
acmt.001.001.08
acmt.002.001.08
acmt.003.001.08
acmt.004.001.06
acmt.005.001.06
acmt.006.001.07

<!-- camt 2024 -->
camt.029.001.12
camt.040.001.04
camt.041.001.04
camt.042.001.04
camt.043.001.04
camt.044.001.03
camt.045.001.03
camt.050.001.06
camt.052.001.11
camt.053.001.11
camt.054.001.11
camt.056.001.10
camt.057.001.07
camt.060.001.06

<!-- pacs 2024 -->
pacs.002.001.13
pacs.003.001.10
```



```
pacs.004.001.12
pacs.007.001.12
pacs.008.001.11
pacs.009.001.10

<!-- pain 2024 -->
pain.001.001.11

<!-- reda 2024 -->
reda.001.001.04
reda.002.001.04
reda.004.001.06
reda.005.001.03

<!-- secl 2024 -->
secl.010.001.03

<!-- semt 2024 -->
semt.001.001.04
semt.002.001.02
semt.003.001.02
semt.004.001.02
semt.005.001.02
semt.006.001.02
semt.007.001.02
semt.012.001.01
semt.013.001.06
semt.014.001.07
semt.015.001.09
semt.017.001.12
semt.018.001.13
semt.019.001.10
semt.021.001.08
semt.023.001.01

<!-- sese 2024 -->
sese.001.001.09
sese.002.001.09
sese.003.001.09
sese.004.001.09
sese.005.001.09
sese.006.001.09
sese.007.001.09
sese.008.001.09
sese.009.001.08
sese.010.001.07
sese.011.001.09
```

```
sese.012.001.11  
sese.013.001.11  
sese.014.001.09  
sese.018.001.09  
sese.019.001.08  
sese.020.001.07  
sese.023.001.11  
sese.024.001.12  
sese.025.001.11  
sese.027.001.07  
sese.028.001.10  
sese.029.001.06  
sese.030.001.09  
sese.031.001.09  
sese.038.001.09  
sese.039.001.06  
sese.040.001.04  
  
<!-- setr 2024 -->  
setr.001.001.04  
setr.002.001.04  
setr.003.001.04  
setr.004.001.04  
setr.005.001.04  
setr.006.001.04  
setr.007.001.04  
setr.008.001.04  
setr.009.001.04  
setr.010.001.04  
setr.011.001.04  
setr.012.001.04  
setr.013.001.04  
setr.014.001.04  
setr.015.001.04  
setr.016.001.04  
setr.017.001.04  
setr.018.001.04  
setr.027.001.03  
setr.029.001.01  
setr.030.001.01  
setr.044.001.02  
setr.047.001.02  
setr.048.001.01  
setr.049.001.02  
setr.050.001.01  
setr.051.001.02  
setr.052.001.01
```

```
setr.053.001.02  
setr.054.001.01  
setr.055.001.02  
setr.056.001.01  
setr.057.001.02  
setr.058.001.02  
setr.059.001.01  
setr.060.001.01  
setr.061.001.01  
setr.062.001.01  
setr.064.001.01  
setr.065.001.01  
setr.066.001.01  
  
<!-- seev 2024 -->  
seev.001.001.11  
seev.006.001.10  
seev.007.001.10  
seev.008.001.09  
seev.031.001.14  
seev.033.001.12  
seev.034.001.14  
seev.035.001.15  
seev.036.001.15  
seev.037.001.15  
seev.038.001.08  
seev.039.001.12  
seev.040.001.12  
seev.041.001.13  
seev.042.001.12  
seev.044.001.12  
seev.047.001.03
```

Supported CBPR+ Messages

The plug-in supports part of SWIFT CBPR+ Standard Release 2023, and therefore, supports a certain set of CBPR+ messages that conform to SWIFT CBPR+ Standard Release 2023.

```
CBPRPlus-camt_029_001_09  
CBPRPlus-camt_052_001_08  
CBPRPlus-camt_053_001_08  
CBPRPlus-camt_054_001_08
```

```
CBPRPlus-camt_056_001_08  
CBPRPlus-camt_057_001_06  
CBPRPlus-camt_060_001_05  
CBPRPlus-pacs_002_001_10  
CBPRPlus-pacs_004_001_09  
CBPRPlus-pacs_008_001_08  
CBPRPlus-pacs_008_001_08_STP  
CBPRPlus-pacs_009_001_08  
CBPRPlus-pacs_009_001_08_ADV  
CBPRPlus-pacs_009_001_08_COV  
CBPRPlus-pacs_010_001_03  
CBPRPlus-pain_001_001_09  
CBPRPlus-pain_002_001_10
```



Note: You should keep CBPR+ .xsd file name consistent with the CBPR+ message name displayed in MX Schema resource.

SWIFT MT Palette

The SWIFT MT palette contains the following activities:

- [Parse SWIFT MT Activity](#)
- [Render SWIFT MT Activity](#)
- [Route SWIFT MT Activity](#)
- [Generate SWIFT BICPlusIBAN Activity](#)
- [Validate SWIFT BICPlusIBAN Activity](#)


Parse SWIFT MT Activity


You can use the Parse SWIFT MT activity to validate the MT message and parse it to XML format.

General

On the **General** tab, you can specify a message schema and select multiple validation types to parse and validate an MT message.

The following table lists the configurations on the **General** tab of the Parse SWIFT MT activity:


| Field | Module Property? | Visual Diff? | Description |
|----------------------------|------------------|--------------|--|
| Name | No | Yes | Specify the name to be displayed as the label for the activity in the process. |
| SWIFT Specification | Yes | Yes | Click  to select a Load SWIFT MT Schema shared resource. |




| Field | Module Property? | Visual Diff? | Description |
|-----------------------------------|------------------|--------------|--|
| | | | If no matching Load SWIFT MT Schema shared resource is found, click Create Shared Resource to create one. For more details, see Creating a Load SWIFT MT Schema Shared Resource . |
| SWIFT Message Schema | No | Yes | Click  to select a message type schema which you want to configure for the activity. |
| Select All Validations | No | Yes | Select this checkbox to enable all the validations. |
| Validate Message Structure | No | Yes | Select this checkbox to validate the SWIFT message structure. See Guidelines for Validating the Message Structure for more information. |
| Validate Field Format | No | Yes | Select this checkbox to validate the SWIFT message format. This format specification includes character set membership, mandatory subfield presence, fixed and maximum length subfield enforcement, and multiple line subfield line count enforcement. All SWIFT text and header fields must follow their format specifications exactly. Note: Certain field format errors are considered parse errors and they occur even when this validation feature is disabled. |

| Field | Module Property? | Visual Diff? | Description |
|--------------------------------------|------------------|--------------|--|
| | | | <p>For example, the format of the field 20C of an MT 500 is 4!c//16x , which means the string must be as follows:</p> <ul style="list-style-type: none"> • 4 alphabetic characters fixed length • double forward slash (//) • not more than 16 characters which belong to the X S.W.I.F.T. character set |
| Validate Data Types | No | Yes | <p>Select this checkbox to validate the SWIFT message data type. SWIFT header and text subfields, such as message types, currency codes, amounts, data, and times are all validated for correct format. The syntactic BIC and LTA validations are part of this feature. Validation against the BIC database requires the use of the Validate BIC/BEI field.</p> |
| Validate Qualifier Code Words | No | Yes | <p>Select this checkbox to validate the qualifier code word. SWIFT code words are various qualifier-dependent values used in various text subfields. The SWIFT network is currently in a transition for code word validation, and not all customers might want to validate code words.</p> |
| Validate BIC/BEI | No | Yes | <p>Select this checkbox to validate the SWIFT message BIC/BEI codes against a BIC directory. SWIFT BIC and BEI codes are entity identifiers that are</p> |

| Field | Module Property? | Visual Diff? | Description |
|---------------------------------------|------------------|--------------|--|
| | | | <p>displayed in various subfields of the text block. Validating them takes a lot of time and memory, because data must be searched in a very large database.</p> <p>At the initialization, an FI.dat BIC directory is provided to the validator, and then full BIC/BEI validation is performed against the provided directory. If no BIC directory is provided, the plug-in uses the default BIC file.</p> <div> <p>Note:</p> <ul style="list-style-type: none"> • Use the latest BIC file provided by SWIFT for up-to-date BIC or BEI validation. • An FI.dat BIC directory is provided by the plug-in as the default BIC file, but you can also provide the most recent BIC file from Swift to overwrite the default file in validation. </div> |
| Validate Field-Level Semantics | No | Yes | <p>Select this checkbox to validate the SWIFT text blocks field-level semantic. Field-level rules are SWIFT network validation rules that relate data in the subfields of a single field. Examples include "field 22H must have one of subfield Narrative, or Amount, but not both".</p> |

| Field | Module Property? | Visual Diff? | Description |
|---|------------------|--------------|--|
| | | | <p>Note: This validation is not performed for SWIFT-to-user system messages even if this feature is enabled. This situation happens because these messages are always from SWIFT, and they are assumed to be valid.</p> |
| Validate Message-Level Semantics | No | Yes | <p>Select this checkbox to enable message-level semantic validation of SWIFT text blocks. Message-level rules are SWIFT network validation rules that relate data in different parts of a message. For example, "if field 22H in sequence A has value X, then field 98A in sequence B must be present".</p> <p>These rules are complex and slow the validation process. Furthermore, customers frequently enforce these rules downstream of the validator in their own business logic, so it is not always necessary to enable this sort of validation.</p> <p>Note: This validation is not performed for SWIFT-to-user system messages even if this feature is enabled. This situation happens because these messages are always from SWIFT, and they are assumed to be valid.</p> |
| Validate Structured Narrative | No | Yes | <p>Select this checkbox to validate the structure narrative of SWIFT message headers. Various SWIFT narrative tags,</p> |

| Field | Module Property? | Visual Diff? | Description |
|---------------------------------|------------------|--------------|---|
| | | | for example, 71B and 72 are required structures in various messages. The structure typically consists of keywords, continuation characters, or both, occasionally in a particular order. |
| Validate Advanced Header | No | Yes | <p>Select this checkbox to enable advanced validation of SWIFT message headers. Such validation includes:</p> <ul style="list-style-type: none"> • Enforcing the order of tags in the user header section • Enforcing the order of tags in the trailer section • Enforcing relationships between delivery options, message types, and obsolescence periods in the application header |
| BIC Code File(FI.dat) | Yes | Yes | <p>Click  and navigate to the location of the BIC or BICPlus file. Select the file and click Open to load it.</p> |

| Field | Module Property? | Visual Diff? | Description |
|--|------------------|--------------|--|
| | | | <p>Note:</p> <ul style="list-style-type: none"> The BIC or BICPlus file is extremely large and, if specified, causes the increase of plug-in startup time. If this file is not specified, the default FI.dat file is used. An FI.dat BIC directory is provided by the plugin as the default BIC file, but you can also provide the most recent BIC file from Swift to overwrite the default file in validation. |
| ISO3166 Country Code File(CT.dat) | Yes | Yes | <p>Click  to navigate to the location of the country code data file. Select the file and click Open to load it.</p> <p>Note: The ISO3166 country codes are used if the file is specified here. Otherwise, an internal database of country codes is used.</p> |
| ISO4217 Currency Code File(CU.dat) | Yes | Yes | <p>Click  to navigate to the location of the currency code data file. Select the file and click Open to load it.</p> <p>Note: The ISO4217 currency codes are used if the file is specified here. Otherwise an internal database of currency codes is used.</p> |
| Validation Filter File (ValidationFilter.xml) | Yes | Yes | <p>Click  to navigate to the location of</p> |

| Field | Module Property? | Visual Diff? | Description |
|-------|------------------|--------------|---|
| | | | <p>the <code>ValidationFilter.xml</code> file. Select the file and click Open to load it.</p> <p>See Using Validation Filters for more information about the validation filters feature.</p> |



Note: You can specify the .dat files in the dat folder located at `<TIBCO_HOME>/bw/palettes/swift/<version>/bin`. Ensure that the .dat files only have `FI.dat`, `CI.dat` and `CT.dat` as file names.

Description

On the **Description** tab, you can enter a short description for the Parse SWIFT MT activity.

Input

The following table lists the input element on the **Input** tab of the Parse SWIFT MT activity:

| Input Item | Data Type | Description |
|------------|-----------|------------------------------|
| FINMessage | String | MT message in String format. |

Output

On the **Output** tab, you can find the parse results.

The following table lists the output element on the **Output** tab of the Parse SWIFT MT activity:

| Output Item | Data Type | Description |
|--------------|-----------|---|
| SWIFTMessage | XML | XML representation of the input MT message. |

| Output Item | Data Type | Description |
|-------------|-----------|--|
| | | The contents of the schema are determined by the SWIFT Message schema specified in the General tab. |

Fault

On the **Fault** tab, you can find the error code and error message of the Parse SWIFT MT activity. See [Error Codes](#) for more detailed explanation of errors.

The following table lists the error schema elements on the **Fault** tab of the Parse SWIFT MT activity:

| Error Schema Element | Data Type | Description |
|----------------------|-----------|--|
| ValidationException | String | The MT messages have validation errors when the validation does not follow the selected SWIFT specification. |
| SwiftException | String | The activity cannot locate the metadata directory or it fails to initialize the metadata. |



Render SWIFT MT Activity

You can use the Render SWIFT MT activity to render an MT message from XML format to the MT message format.

General

On the **General** tab, you can specify a message schema and select multiple validation types to render and validate the MT message in XML format.




The following table lists the configurations on the **General** tab of the Render SWIFT MT activity:


| Field | Module Property? | Visual Diff? | Description |
|-----------------------------------|------------------|--------------|--|
| Name | No | Yes | Specify the name to be displayed as the label for the activity in the process. |
| SWIFT Specification | Yes | Yes | <p>Click  to select a Load SWIFT MT Schema shared resource.</p> <p>If no matching Load SWIFT MT Schema shared resource is found, click Create Shared Resource to create one. For more details, see Creating a Load SWIFT MT Schema Shared Resource.</p> |
| SWIFT Message Schema | No | Yes | Click  to select a message type schema which you want to configure for the activity. |
| Select All Validations | No | Yes | Select this checkbox to enable all the validations. |
| Validate Message Structure | No | Yes | <p>Select this checkbox to validate the SWIFT message structure.</p> <p>See Guidelines for Validating the Message Structure for more information.</p> |
| Validate Field Format | No | Yes | Select this checkbox to validate the SWIFT message format. This format specification includes character set membership, mandatory subfield presence, fixed and maximum length subfield enforcement, and multiple line subfield line count enforcement. All SWIFT text and header fields must |

| Field | Module Property? | Visual Diff? | Description |
|--------------------------------------|------------------|--------------|--|
| | | | <p>follow their format specifications exactly.</p> <p>Note: Certain field format errors are considered parse errors and they occur even when this validation feature is disabled.</p> |
| Validate Data Types | No | Yes | <p>Select this checkbox to validate the SWIFT message data type. SWIFT header and text subfields, such as message types, currency codes, amounts, data, and times are all validated for correct format. The syntactic BIC and LTA validations are part of this feature. Validation against the BIC database requires the use of the Validate BIC/BEI field.</p> |
| Validate Qualifier Code Words | No | Yes | <p>Select this checkbox to validate the qualifier code word. SWIFT code words are various qualifier-dependent values used in various text subfields. The SWIFT network is currently in a transition for code word validation, and not all customers might want to validate code words.</p> |
| Validate BIC/BEI | No | Yes | <p>Select this checkbox to validate the SWIFT message BIC/BEI codes against a BIC directory. SWIFT BIC and BEI codes are entity identifiers that are displayed in various subfields of the text block. Validating them takes a lot of time and memory, because data</p> |

| Field | Module Property? | Visual Diff? | Description |
|---|------------------|--------------|--|
| | | | <p>must be searched in a very large database.</p> <p>At the initialization, an FI.dat BIC directory is provided to the validator, and then full BIC/BEI validation is performed against the provided directory. If no BIC directory is provided, the plug-in uses the default BIC file.</p> <p>Note: Use the latest BIC file provided by SWIFT for up-to-date BIC or BEI validation.</p> |
| Validate Field-Level Semantics | No | Yes | <p>Select this checkbox to validate the SWIFT text blocks field-level semantic. Field-level rules are SWIFT network validation rules that relate data in the subfields of a single field. For example, Examples include "field 22H must have one of subfield Narrative, or Amount, but not both".</p> <p>Note: This validation is not performed for SWIFT-to-user system messages even if this feature is enabled. This situation happens because these messages are always from SWIFT, and they are assumed to be valid.</p> |
| Validate Message-Level Semantics | No | Yes | <p>Select this checkbox to enable message-level semantic validation of SWIFT text blocks. Message-level rules are SWIFT network validation rules</p> |

| Field | Module Property? | Visual Diff? | Description |
|--------------------------------------|------------------|--------------|---|
| | | | <p>that relate data in different parts of a message. For example, "if field 22H in sequence A has value X, then field 98A in sequence B must be present".</p> <p>These rules are complex and slow the validation process. Furthermore, customers frequently enforce these rules downstream of the validator in their own business logic, so it is not always necessary to enable this sort of validation.</p> <div> <p>Note: This validation is not performed for SWIFT-to-user system messages even if this feature is enabled. This situation happens because these messages are always from SWIFT, and they are assumed to be valid.</p> </div> |
| Validate Structured Narrative | No | Yes | <p>Select this checkbox to validate the structure narrative of SWIFT message headers. Various SWIFT narrative tags, for example, 71B and 72 are required structures in various messages. The structure typically consists of keywords, continuation characters, or both, occasionally in a particular order.</p> |
| Validate Advanced Header | No | Yes | <p>Select this checkbox to enable advanced validation of SWIFT message headers. Such validation includes:</p> <ul style="list-style-type: none"> Enforcing the order of tags in the user header section |

| Field | Module Property? | Visual Diff? | Description |
|---|------------------|--------------|---|
| | | | <ul style="list-style-type: none"> Enforcing the order of tags in the trailer section Enforcing relationships between delivery options, message types, and obsolescence periods in the application header |
| BIC Code File(FI.dat) | Yes | Yes | <p>Click  and navigate to the location of the BIC or BICPlus file. Select the file and click Open to load it.</p> <p>Note: The BIC or BICPlus file is extremely large and, if specified, causes the increase of plug-in startup time. If this file is not specified, the default FI.dat file is used.</p> |
| ISO3166 Country Code File(CT.dat) | Yes | Yes | <p>Click  to navigate to the location of the country code data file. Select the file and click Open to load it.</p> <p>Note: The ISO3166 country codes are used if the file is specified here. Otherwise, an internal database of country codes is used.</p> |
| ISO4217 Currency Code File(CU.dat) | Yes | Yes | <p>Click  to navigate to the location of the currency code data file. Select the file and click Open to load it.</p> |

| Field | Module Property? | Visual Diff? | Description |
|--|------------------|--------------|--|
| | | | <p>Note: The ISO4217 currency codes are used if the file is specified here. Otherwise an internal database of currency codes is used.</p> |
| Validation Filter File (ValidationFilter.xml) | Yes | Yes | <p>Click  to navigate to the location of the ValidationFilter.xml file. Select the file and click Open to load it.</p> <p>See Using Validation Filters for more information about the validation filters feature.</p> |

i Note: You can specify the .dat files in the dat folder located at <TIBCO_HOME>/bw/palettes/swift/<version>/bin. Ensure that the .dat files only have FI.dat, CI.dat and CT.dat as file names.

Description

On the **Description** tab, you can enter a short description for the Render SWIFT MT activity.

Input

The following table lists the input element on the **Input** tab of the Render SWIFT MT activity:

| Input Item | Data Type | Description |
|--------------|-----------|--|
| SWIFTMessage | XML | <p>XML representation of the input MT message.</p> <p>The contents of the schema are determined by the SWIFT Message schema specified in the General tab.</p> |

Output

On the **Output** tab, you can find the render results.

The following table lists the output element on the **Output** tab of the Render SWIFT MT activity:

| Output Item | Data Type | Description |
|-------------|-----------|------------------------------|
| FINMessage | String | MT message in String format. |

Fault

On the **Fault** tab, you can find the error code and error message of the Render SWIFT MT activity. See [Error Codes](#) for more detailed explanation of the error.

The following table lists the error schema elements on the **Fault** tab of the Render SWIFT MT activity:

| Error Schema Element | Data Type | Description |
|----------------------|-----------|--|
| ValidationException | String | The MT messages have validation errors when the validation does not follow the selected SWIFT specification. |
| SwiftException | String | The activity cannot locate the metadata directory or it fails to initialize the metadata. |


Route SWIFT MT Activity

You can use the Route SWIFT MT activity to route messages to different activities based on the message types.

General

On the **General** tab, you can specify a message schema and select multiple validation types to route and validate an MT message.




The following table lists the configurations on the **General** tab of the Route SWIFT MT activity:


| Field | Module Property? | Visual Diff? | Description |
|-----------------------------------|------------------|--------------|--|
| Name | No | Yes | Specify the name to be displayed as the label for the activity in the process. |
| SWIFT Specification | Yes | Yes | Click  to select a Load SWIFT MT Schema shared resource. If no matching Load SWIFT MT Schema shared resource is found, click Create Shared Resource to create one. For more details, see Creating a Load SWIFT MT Schema Shared Resource . |
| Select All Validations | No | Yes | Select this checkbox to enable all the validations. |
| Validate Message Structure | No | Yes | Select this checkbox to validate the SWIFT message structure. See Guidelines for Validating the Message Structure for more information. |
| Validate Field Format | No | Yes | Select this checkbox to validate the SWIFT message format. This format specification includes character set membership, mandatory subfield presence, fixed and maximum length subfield enforcement, and multiple line subfield line count enforcement. All SWIFT text and header fields must follow their format specifications exactly. |

| Field | Module Property? | Visual Diff? | Description |
|--------------------------------------|------------------|--------------|---|
| | | | Note: Certain field format errors are considered parse errors and they occur even when this validation feature is disabled. |
| Validate Data Types | No | Yes | Select this checkbox to validate the SWIFT message data type. SWIFT header and text subfields, such as message types, currency codes, amounts, data, and times are all validated for correct format. The syntactic BIC and LTA validations are part of this feature. Validation against the BIC database requires the use of the Validate BIC/BEI field. |
| Validate Qualifier Code Words | No | Yes | Select this checkbox to validate the qualifier code word. SWIFT code words are various qualifier-dependent values used in various text subfields. The SWIFT network is currently in a transition for code word validation, and not all customers might want to validate code words. |
| Validate BIC/BEI | No | Yes | Select this checkbox to validate the SWIFT message BIC/BEI codes against a BIC directory. SWIFT BIC and BEI codes are entity identifiers that are displayed in various subfields of the text block. Validating them takes a lot of time and memory, because data must be searched in a very large database. |

| Field | Module Property? | Visual Diff? | Description |
|---|------------------|--------------|---|
| | | | <p>At the initialization, an FI.dat BIC directory is provided to the validator, and then full BIC/BEI validation is performed against the provided directory. If no BIC directory is provided, the plug-in uses the default BIC file.</p> <p>Note: Use the latest BIC file provided by SWIFT for up-to-date BIC or BEI validation.</p> |
| Validate Field-Level Semantics | No | Yes | <p>Select this checkbox to validate the SWIFT text blocks field-level semantic. Field-level rules are SWIFT network validation rules that relate data in the subfields of a single field. Examples include "field 22H must have one of subfield Narrative, or Amount, but not both".</p> <p>Note: This validation is not performed for SWIFT-to-user system messages even if this feature is enabled. This situation happens because these messages are always from SWIFT, and they are assumed to be valid.</p> |
| Validate Message-Level Semantics | No | Yes | <p>Select this checkbox to enable message-level semantic validation of SWIFT text blocks. Message-level rules are SWIFT network validation rules that relate data in different parts of a message. For example, "if field 22H in sequence A has value X, then field 98A</p> |

| Field | Module Property? | Visual Diff? | Description |
|--------------------------------------|------------------|--------------|--|
| | | | <p>in sequence B must be present".</p> <p>These rules are complex and slow the validation process. Furthermore, customers frequently enforce these rules downstream of the validator in their own business logic, so it is not always necessary to enable this sort of validation.</p> <div> <p>Note: This validation is not performed for SWIFT-to-user system messages even if this feature is enabled. This situation happens because these messages are always from SWIFT, and they are assumed to be valid.</p> </div> |
| Validate Structured Narrative | No | Yes | <p>Select this checkbox to validate the structure narrative of SWIFT message headers. Various SWIFT narrative tags, for example, 71B and 72 are required structures in various messages. The structure typically consists of keywords, continuation characters, or both, occasionally in a particular order.</p> |
| Validate Advanced Header | No | Yes | <p>Select this checkbox to enable advanced validation of SWIFT message headers. Such validation includes:</p> <ul style="list-style-type: none"> • Enforcing the order of tags in the user header section • Enforcing the order of tags in the trailer section |

| Field | Module Property? | Visual Diff? | Description |
|---|------------------|--------------|---|
| | | | <ul style="list-style-type: none"> Enforcing relationships between delivery options, message types, and obsolescence periods in the application header |
| BIC Code File(FI.dat) | Yes | Yes | <p>Click  and navigate to the location of the BIC or BICPlus file. Select the file and click Open to load it.</p> <p>Note: The BIC or BICPlus file is extremely large and, if specified, causes the increase of plug-in startup time. If this file is not specified, the default FI.dat file is used.</p> |
| ISO3166 Country Code File(CT.dat) | Yes | Yes | <p>Click  to navigate to the location of the country code data file. Select the file and click Open to load it.</p> <p>Note: The ISO3166 country codes are used if the file is specified here. Otherwise, an internal database of country codes is used.</p> |
| ISO4217 Currency Code File(CU.dat) | Yes | Yes | <p>Click  to navigate to the location of the currency code data file. Select the file and click Open to load it.</p> <p>Note: The ISO4217 currency codes are used if the file is specified here. Otherwise an internal database of currency codes is used.</p> |

| Field | Module Property? | Visual Diff? | Description |
|--|------------------|--------------|--|
| Validation Filter File (ValidationFilter.xml) | Yes | Yes | <p>Click  to navigate to the location of the ValidationFilter.xml file. Select the file and click Open to load it.</p> <p>See Using Validation Filters for more information about the validation filters feature.</p> |

i Note: You can specify the .dat files in the dat folder located at <TIBCO_HOME>/bw/palettes/swift/<version>/bin. Ensure that the .dat files only have FI.dat, CI.dat and CT.dat as file names.

Description

On the **Description** tab, you can enter a short description for the Route SWIFT MT activity.

Input

The following table lists the input element on the **Input** tab of the Route SWIFT MT activity:

| Input Item | Data Type | Description |
|------------|-----------|------------------------------|
| FINMessage | String | MT message in String format. |

Output

On the **Output** tab, you can find the route results.

The following table lists the output elements on the **Output** tab of the Route SWIFT MT activity:

| Output Item | Data Type | Description |
|-------------|-----------|--------------------------------|
| MessageType | String | Message type in String format. |
| FINMessage | String | MT message in String format. |

Fault

On the **Fault** tab, you can find the error code and error message of the Route SWIFT MT activity. See [Error Codes](#) for more detailed explanation of the error.

The following table lists the error schema elements on the **Fault** tab of the Route SWIFT MT activity:

| Error Schema Element | Data Type | Description |
|----------------------|-----------|--|
| ValidationException | String | The MT messages have validation errors when the validation does not follow the selected SWIFT specification. |
| SwiftException | String | The activity cannot locate the metadata directory or it fails to initialize the metadata. |


Generate SWIFT BICPlusIBAN Activity

You can use the Generate SWIFT BICPlusIBAN activity to generate an IBAN.

General

On the **General** tab, you can specify the BANK Directory and IBAN Plus data file to generate an IBAN.

The following table lists the configurations on the **General** tab of the Generate SWIFT BICPlusIBAN activity:

| Field | Module Property? | Visual Diff? | Description |
|-------------------------------------|------------------|--------------|--|
| Name | No | Yes | Specify the name to be displayed as the label for the activity in the process. |
| BANK Directory and IBAN Plus | Yes | Yes | <p>Click  and navigate to the directory where the BANK Directory and IBAN Plus data file is located. Select the file and click OK to load it.</p> <div> <p>Note: You have to download the required BANK Directory and IBAN Plus data file from the SWIFT official website, and ensure that the following files exist inside:</p> <ul style="list-style-type: none"> • IBANPLUS_V*_FULL_year**** file (IBAN information), required. • IBANSTRUCTURE_FULL_year**** file (IBAN structure information), required. • EXCLUSIONLIST_V*_FULL_year**** file (National IDs cannot be in IBANs), required. • COUNTRY_CODE_year**** file (Country code information), required. • BANKDIRECTORYPLUS_V*_FULL_year**** file (Bank information), required. </div> |

Description

On the **Description** tab, you can enter a short description for the Generate SWIFT BICPlusIBAN activity.

Input

The following table lists the input elements on the **Input** tab of the Generate SWIFT BICPlusIBAN activity:

| Input Item | Data Type | Description |
|-------------|-----------|--|
| CountryCode | String | Specify the 2-character country code that is required to generate an IBAN. |
| BBAN | String | Specify the BBAN that is required to generate an IBAN. |

Output

On the **Output** tab, you can find the generated results.

The following table lists the output element on the **Output** tab of the Generate SWIFT BICPlusIBAN activity:

| Output Item | Data Type | Description |
|-------------|-----------|------------------------|
| IBAN | String | IBAN in String format. |

Fault

On the **Fault** tab, you can find the error code and error message of the Generate SWIFT BICPlusIBAN activity. See [Error Codes](#) for more detailed explanation of errors.

The following table lists the error schema elements on the **Fault** tab of the Generate SWIFT BICPlusIBAN activity:

| Error Schema Element | Data Type | Description |
|----------------------|-----------|--|
| ValidationException | String | The MT messages have validation errors when the validation does not follow the selected SWIFT specification. |
| SwiftException | String | The activity cannot locate the metadata directory or it fails to initialize the metadata. |


Validate SWIFT BICPlusIBAN Activity

You can use the Validate SWIFT BICPlusIBAN activity to validate an IBAN.

General

On the **General** tab, you can specify the BANK Directory and IBAN Plus data file to validate an IBAN.

The following table lists the configurations on the **General** tab of the Validate SWIFT BICPlusIBAN activity:

| Field | Module Property? | Visual Diff? | Description |
|-------------------------------------|------------------|--------------|--|
| Name | No | Yes | Specify the name to be displayed as the label for the activity in the process. |
| BANK Directory and IBAN Plus | Yes | Yes | <p>Click  and navigate to the directory where the BANK Directory and IBAN Plus data file is located. Select the file and click OK to load it.</p> <div> <p>Note: You have to download the required BANK Directory and IBAN Plus data file from the SWIFT official website, and ensure that the following files exist inside:</p> <ul style="list-style-type: none"> • IBANPLUS_V*_FULL_year**** file (IBAN information), required. • IBANSTRUCTURE_FULL_year**** file (IBAN structure information), required. • EXCLUSIONLIST_V*_FULL_year**** file (National IDs cannot be in IBANs), required. • COUNTRY_CODE_year**** file (Country code information), required. • BANKDIRECTORYPLUS_V*_FULL_year**** file (Bank information), required. </div> |

Description

On the **Description** tab, you can enter a short description for the Validate SWIFT BICPlusIBAN activity.

Input

On the **Input** tab, you can specify the IBAN that you want to validate.

The following table lists the input elements on the **Input** tab of the Validate SWIFT BICPlusIBAN activity:

| Input Item | Data Type | Description |
|------------|-----------|---|
| IBAN | String | Specify the IBAN that you want to validate. |
| BIC | String | Specify the BIC that you want to validate. |
| BranchCode | String | Specify the BranchCode that you want to validate. |

Output

On the **Output** tab, you can find the validation results.

The following table lists the output element on the **Output** tab of the Validate SWIFT BICPlusIBAN activity:

| Output Item | Data Type | Description |
|-------------------|-----------|---------------------------------------|
| BICPlusIBANRecord | XML | Displays the IBAN validation results. |

Fault

On the **Fault** tab, you can find the error code and error message of the Search Entry activity. See [Error Codes](#) for more detailed explanation of errors.

The following table lists the error schema elements on the **Fault** tab of the Validate SWIFT BICPlusIBAN activity:

| Error Schema Element | Data Type | Description |
|----------------------|-----------|--|
| Validation Exception | String | The MT messages have validation errors when the validation does not follow the selected SWIFT specification. |
| Swift Exception | String | The activity cannot locate the metadata directory or it fails to initialize the metadata. |

Guidelines for Validating the Message Structure

When validating an MT message structure, the following guidelines are used:

- The fields of all required basic and application headers have the correct length.
- The basic header and service identifiers are valid.
- Application input and output indicators are valid.
- Application input delivery monitoring and obsolescence cannot be displayed in the message unless the priority is displayed.
- Application input obsolescence cannot be displayed in the message unless the delivery monitoring is displayed.
- The user header is displayed if required.
- Message types requiring a user header field validation tag 119 have such a tag.
- No duplicate user header or trailer tags are found.
- No invalid user header, trailer, or system acknowledgment tags are found.
- The lengths of all user header, trailer, and system acknowledgment tag values are correct.

- Message types requiring user header field validation tags to select correct message metadata (types 102, 103, and 574) have validation tags with valid values.
- All mandatory text block sequences and fields are displayed in the message.
- The text block contains no invalid fields.

Limitations and Suggestions

When using the SWIFT MT palette of the plug-in, for message types containing more than one Qualifier Groups, the order of the output MT message being displayed in the Qualifier Groups can be different from that of the input MT message. This is not an error. The Qualifier Groups can be displayed in any order according to the SWIFT specification.

SWIFT MX Palette

The SWIFT MX palette contains two activities, which are used to handle the SWIFT MX messages and the CBPR+ messages:

- [Parse SWIFT MX Activity](#)
- [Render SWIFT MX Activity](#)


Parse SWIFT MX Activity


You can use the Parse SWIFT MX activity to validate a SWIFT MX message and parse it to XML format.

General


On the **General** tab, you can select the MX schema to parse the SWIFT MX message.




The following table lists the configurations on the **General** tab of the Parse SWIFT MX activity:


| Field | Module Property? | Visual Diff? | Description |
|---------------------|------------------|--------------|--|
| Name | No | Yes | Specify the name to be displayed as the label for the activity in the process. |
| SWIFT Specification | Yes | Yes | Click  to select a Load SWIFT MX Schema shared resource. If no matching Load SWIFT MX Schema shared resource is found, click Create Shared Resource to create one. For more details, see Creating a Load SWIFT MX Schema Shared Resource . |

| Field | Module Property? | Visual Diff? | Description |
|-------------------------|------------------|--------------|--|
| Schema | No | Yes | <p>The following schemas are available in the Schema list:</p> <ul style="list-style-type: none"> • MX Schema • All Schema |
| Transport Schema | No | Yes | <p>Click  to select the transport schema that you want to parse.</p> <p>Note: This field is displayed when you select All Schema from the Schema list.</p> |
| Element | No | Yes | <p>Select a message type from the Element list.</p> <p>Note: This field is displayed when you select All Schema from the Schema list. It provides a list of message element types after you select a transport schema in the Transport Schema field.</p> <p>The following items are available in the Element list after you select an InterAct type transport schema:</p> <ul style="list-style-type: none"> • ExchangeRequest • ExchangeResponse • HandleRequest • HandleResponse <p>Only the DataPDU item is available in the Element list after you select an SAA type transport schema.</p> |

| Field | Module Property? | Visual Diff? | Description |
|----------------------------------|------------------|--------------|---|
| AppHeader Schema | No | Yes | <p>Click  to select the AppHeader schema that you want to parse.</p> <p>Note: This field is displayed when you select All Schema from the Schema list.</p> |
| MX Schema | No | Yes | <p>Click  to select an MX schema which is loaded in the Load SWIFT MX Schema shared resource.</p> |
| Select All Validations | No | Yes | Select this checkbox to enable all the validations. |
| Validate Transport Schema | No | Yes | <p>Select this checkbox to enable message transport validation in SWIFT messages against the SWIFT transport schema.</p> <p>Note: This field is displayed when you select All Schema from the Schema list.</p> |
| Validate AppHeader Schema | No | Yes | <p>Select this checkbox to enable message header validation in SWIFT messages against the SWIFT AppHeader schema.</p> <p>Note: This field is displayed when you select All Schema from the Schema list.</p> |
| Validate MX Schema | No | Yes | Select this checkbox to enable syntax validation in SWIFT messages against |

| Field | Module Property? | Visual Diff? | Description |
|-------------------------------------|------------------|--------------|--|
| | | | the SWIFT MX schema. |
| Validate MX Rules | No | Yes | Select this checkbox to enable validation for extended rules and business logic in SWIFT messages against the MX rules. |
| Validate BIC/BEI | No | Yes | <p>Select this checkbox to validate the SWIFT message BIC/BEI codes against a BIC directory. SWIFT BIC and BEI codes are entity identifiers that are displayed in various subfields of the text block. Validating them takes a lot of time and memory, because data must be searched in a very large database.</p> <p>At the initialization, an FI.dat BIC directory is provided to the validator, and then full BIC/BEI validation is performed against the provided directory. If no BIC directory is provided, the plug-in uses the default BIC file.</p> <div> Note: Use the latest BIC file provided by SWIFT for up-to-date BIC or BEI validation. </div> |
| Validate IBAN/BBAN | No | Yes | Select this checkbox to enable IBAN/BBAN validation in SWIFT messages against an IBAN/BBAN directory. |
| BANK Directory and IBAN Plus | No | Yes | Click  and navigate to the location |

| Field | Module Property? | Visual Diff? | Description |
|---|------------------|--------------|---|
| | | | <p>of the BANK directory and IBAN plus file. Select the file and click Open to load it.</p> <p>Note: This field is available when you select the Validate IBAN/BBAN checkbox.</p> |
| BIC Code File(FI.dat) | Yes | Yes | <p>Click  and navigate to the location of the BIC or BICPlus file. Select the file and click Open to load it.</p> <p>Note: The BIC or BICPlus file is extremely large and, if specified, causes the increase of plug-in startup time. If this file is not specified, the default FI.dat file is used.</p> |
| ISO3166 Country Code File(CT.dat) | Yes | Yes | <p>Click  to navigate to the location of the country code data file. Select the file and click Open to load it.</p> <p>Note: The ISO3166 country codes are used if the file is specified here. Otherwise, an internal database of country codes is used.</p> |
| ISO4217 Currency Code File(CU.dat) | Yes | Yes | <p>Click  to navigate to the location of the currency code data file. Select the file and click Open to load it.</p> |

| Field | Module Property? | Visual Diff? | Description |
|--|------------------|--------------|--|
| | | | <p>Note: The ISO4217 currency codes are used if the file is specified here. Otherwise an internal database of currency codes is used.</p> |
| Validation Filter File (ValidationFilter.xml) | Yes | Yes | <p>Click  to navigate to the location of the ValidationFilter.xml file. Select the file and click Open to load it.</p> <p>See Using Validation Filters for more information about the validation filters feature.</p> |

i Note: You can specify the .dat files in the dat folder located at <TIBCO_HOME>/bw/palettes/swift/<version>/bin. Ensure that the .dat files only have FI.dat, CI.dat and CT.dat as file names.

Description

On the **Description** tab, you can enter a short description for the Parse SWIFT MX activity.

Input

The following table lists the input element on the **Input** tab of the Parse SWIFT MX activity:

| Input Item | Data Type | Description |
|------------|-----------|------------------------------|
| MXMessage | String | MX message in String format. |

Output

On the **Output** tab, you can find the parse results.

The following table lists the output element on the **Output** tab of the Parse SWIFT MX activity:

| Output Item | Data Type | Description |
|--------------|-----------|---|
| SWIFTMessage | XML | MX message in XML format. The contents of the schema are determined by the SWIFT Message schema specified in the General tab. |

Fault

On the **Fault** tab, you can find the error code and error message of the Parse SWIFT MX activity. See [Error Codes](#) for more detailed explanation of errors.

The following table lists the error schema elements on the **Fault** tab of the Parse SWIFT MX activity:

| Error Schema Element | Data Type | Description |
|----------------------|-----------|--|
| ValidationException | String | The MX messages have validation errors when the validation does not follow the selected SWIFT specification. |
| SwiftException | String | The activity cannot locate the metadata directory or it fails to initialize the metadata. |



Render SWIFT MX Activity



You can use the Render SWIFT MX activity to generate specific MX messages.

General



On the **General** tab, you can select the MX schema to render a SWIFT MX message.




The following table lists the configurations on the **General** tab of the Render SWIFT MX activity:

| Field | Module Property? | Visual Diff? | Description |
|----------------------------|------------------|--------------|--|
| Name | No | Yes | Specify the name to be displayed as the label for the activity in the process. |
| SWIFT Specification | Yes | Yes | <p>Click  to select a Load SWIFT MX Schema shared resource.</p> <p>If no matching Load SWIFT MX Schema shared resource is found, click Create Shared Resource to create one. For more details, see Creating a Load SWIFT MX Schema Shared Resource.</p> |
| Schema | No | Yes | <p>The following schemas are available in the Schema list:</p> <ul style="list-style-type: none"> • MX Schema • All Schema |
| Transport Schema | No | Yes | <p>Click  to select the transport schema that you want to parse.</p> <div> <p>Note: This field is displayed when you select All Schema from the Schema list.</p> </div> |
| Element | No | Yes | Select a message type from the Element list. |

| Field | Module Property? | Visual Diff? | Description |
|-------------------------------|------------------|--------------|---|
| | | | <p>Note: This field is displayed when you select All Schema from the Schema list. It provides a list of message element types after you select a transport schema in the Transport Schema field.</p> <p>The following items are available in the Element list after you select an InterAct type transport schema:</p> <ul style="list-style-type: none"> • ExchangeRequest • ExchangeResponse • HandleRequest • HandleResponse <p>Only the DataPDU item is available in the Element list after you select an SAA type transport schema.</p> |
| AppHeader Schema | No | Yes | <p>Click  to select the AppHeader schema that you want to parse.</p> <p>Note: This field is displayed when you select All Schema from the Schema list.</p> |
| MX Schema | No | Yes | <p>Click  to select an MX schema which is loaded in the Load SWIFT MX Schema shared resource.</p> |
| Select All Validations | No | Yes | Select this checkbox to enable all the validations. |
| Validate Transport | No | Yes | Select this checkbox to enable |

| Field | Module Property? | Visual Diff? | Description |
|----------------------------------|------------------|--------------|--|
| Schema | | | <p>message transport validation in SWIFT messages against the SWIFT transport schema.</p> <p>Note: This field is displayed when you select All Schema from the Schema list.</p> |
| Validate AppHeader Schema | No | Yes | <p>Select this checkbox to enable message header validation in SWIFT messages against the SWIFT AppHeader schema.</p> <p>Note: This field is displayed when you select All Schema from the Schema list.</p> |
| Validate MX Schema | No | Yes | <p>Select this checkbox to enable syntax validation in SWIFT messages against the SWIFT MX schema.</p> |
| Validate MX Rules | No | Yes | <p>Select this checkbox to enable validation for extended rules and business logic in SWIFT messages against the MX rules.</p> |
| Validate BIC/BEI | No | Yes | <p>Select this checkbox to validate the SWIFT message BIC/BEI codes against a BIC directory. SWIFT BIC and BEI codes are entity identifiers that are displayed in various subfields of the text block. Validating them takes a lot of time and memory, because data must be searched in a very large database.</p> |

| Field | Module Property? | Visual Diff? | Description |
|-------------------------------------|------------------|--------------|---|
| | | | <p>At the initialization, an FI.dat BIC directory is provided to the validator, and then full BIC/BEI validation is performed against the provided directory. If no BIC directory is provided, the plug-in uses the default BIC file.</p> <p>Note: Use the latest BIC file provided by SWIFT for up-to-date BIC or BEI validation.</p> |
| Validate IBAN/BBAN | No | Yes | Select this checkbox to enable IBAN/BBAN validation in SWIFT messages against an IBAN/BBAN directory. |
| BANK Directory and IBAN Plus | No | Yes | <p>Click  and navigate to the location of the BANK directory and IBAN plus file. Select the file and click Open to load it.</p> <p>Note: This field is available when you select the Validate IBAN/BBAN checkbox.</p> |
| BIC Code File(FI.dat) | Yes | Yes | Click  and navigate to the location of BANK directory and IBAN plus file. Select the file and click Open to load it. |

| Field | Module Property? | Visual Diff? | Description |
|--|------------------|--------------|---|
| | | | <p>Note: The BIC or BICPlus file is extremely large and, if specified, causes the increase of plug-in startup time. If this file is not specified, the default FI.datfile is used.</p> |
| ISO3166 Country Code File(CT.dat) | Yes | Yes | <p>Click  to navigate to the location of the country code data file. Select the file and click Open to load it.</p> <p>Note: The ISO3166 country codes are used if the file is specified here. Otherwise, an internal database of country codes is used.</p> |
| ISO4217 Currency Code File(CU.dat) | Yes | Yes | <p>Click  to navigate to the location of the currency code data file. Select the file and click Open to load it.</p> <p>Note: The ISO4217 currency codes are used if the file is specified here. Otherwise an internal database of currency codes is used.</p> |
| Validation Filter File (ValidationFilter.xml) | Yes | Yes | <p>Click  to navigate to the location of the ValidationFilter.xml file. Select the file and click Open to load it.</p> <p>See Using Validation Filters for more information about the validation filters feature.</p> |

i Note: You can specify the .dat files in the dat folder located at <TIBCO_HOME>/bw/palettes/swift/<version>/bin. Ensure that the .dat files only have FI.dat, CI.dat and CT.dat as file names.

Description

On the **Description** tab, you can enter a short description for the Render SWIFT MX activity.

Input

The following table lists the input element on the **Input** tab of the Render SWIFT MX activity:

| Input Item | Data Type | Description |
|--------------|-----------|---|
| SWIFTMessage | XML | MX message in XML format. The contents of the schema are determined by the SWIFT Message schema specified in the General tab. |

Output

On the **Output** tab, you can find the render results.

The following table lists the output element on the **Output** tab of the Render SWIFT MX activity:

| Output Item | Data Type | Description |
|-------------|-----------|------------------------------|
| MXMessage | String | MX message in String format. |

Fault

On the **Fault** tab, you can find the error code and error message of the Render SWIFT MX activity. See [Error Codes](#) for more detailed explanation of the error.

The following table lists the error schema elements on the **Fault** tab of the Render SWIFT MX activity:

| Error Schema Element | Data Type | Description |
|----------------------|-----------|--|
| ValidationException | String | The MX messages have validation errors when the validation does not follow the selected SWIFT specification. |
| SwiftException | String | The activity cannot locate the metadata directory or it fails to initialize the metadata. |

OpenTelemetry Tracing for TIBCO ActiveMatrix BusinessWorks™ Plug-in for SWIFT Palette

OpenTelemetry is an open-source and vendor-neutral standard for distributed systems. OpenTelemetry can be used to track the current state of the job. For more information, see the OpenTelemetry section in the *TIBCO ActiveMatrix BusinessWorks™ Administration Guide*.

Custom Tags

To add custom tags for OpenTelemetry, use the **Tags** tab of each activity in TIBCO ActiveMatrix BusinessWorks™. You can add expressions such as hard-coded values or XPath expressions. To avoid predefined engine tags from being overridden, add an asterisk (*) symbol as a prefix before the custom tags. For more information, see the Custom Tags for OpenTelemetry section in the *TIBCO ActiveMatrix BusinessWorks™ Administration Guide*.

Configuring Advanced Options

This plug-in provides the following advanced configuration options:

- [Parsing and Validating SWIFT MT Messages Using SwiftCheck](#)
- [Parsing and Validating SWIFT MX/CBPR+ Messages Using SwiftMXCheck](#)
- [Using Validation Filters](#)
- [Configuring Customized MX Java Rules](#)

Parsing and Validating SWIFT MT Messages Using SwiftCheck

You can use the SwiftCheck utility to parse and validate one or more SWIFT MT message files. The utility can be used for low-level testing of SWIFT MT message files.

Procedure

1. Navigate to the `TIBCO_HOME/bw/palettes/swift/version_number/bin` directory.
2. Copy the `SwiftCheck.tra` file, rename the copied file as `mySwiftCheck.tra`, and open it in the text editor.
3. Change the `application.args` property to include the metadata and validation options to use. For example:

```
application.args -data TIBCO_HOME/bw/palettes/swift/version_number/bin/data -set  
specification_year -v TIBCO_HOME/bw/palettes/swift/version_  
number/samples/SampleMT.txt
```

Note: For more information about the options, see [SwiftCheck Options](#).

4. Save the `mySwiftCheck.tra` file and exit the text editor.
5. Open a command line and change to the `TIBCO_HOME/bw/palettes/swift/version_number/bin` directory:


```
cd TIBCO_HOME/bw/palettes/swift/version_number/bin
```

6. Type the following command to start the utility:

```
SwiftCheck --run --propFile mySwiftCheck.tra
```

SwiftCheck Options

You can use the metadata and validation options to parse and validate SWIFT MT message files.

Metadata Options

The following table shows the descriptions of the metadata options:

| Option | Description |
|--------------|--------------------------------|
| -data <file> | metadata directory |
| -set <file> | metadata set |
| -fi <file> | bankdata file (BIC codes) |
| -ff <file> | validation warning filter file |
| -cu <file> | currency code file |
| -ct <file> | country code file |

Validation Options

The following table shows the descriptions of the validation options:

| Option | Description |
|--------|--------------------------|
| -v | validate using all rules |

| Option | Description |
|--------|--|
| -vc | turn on format validation |
| -vt | turn on data type validation |
| -vw | turn on code word validation |
| -vb | turn on BIC code validation |
| -vf | turn on field level semantics validation |
| -vm | turn on message level semantics validation |
| -vn | turn on narrative validation |
| -va | turn on advanced header validation |
| -vs | turn on structural validation |

Parsing and Validating SWIFT MX/CBPR+ Messages Using SwiftMXCheck

You can use the SwiftMXCheck utility to validate one or more SWIFT MX message files. This utility can be used for low-level testing of SWIFT MX message files.

Procedure

1. Navigate to the `TIBCO_HOME/bw/palettes/swift/version_number/bin` directory.
2. Copy the `SwiftMXCheck.tra` file, rename the copied file as `mySwiftMXCheck.tra`, and then open it in the text editor.
3. Change the `application.args` property to include the metadata and validation options to use.

For example,
`application.args`

```
-mxdata TIBCO_HOME/bw/palettes/swift/version_number/bin/mxdata -set year
-txsd TIBCO_HOME/bw/palettes/swift/version_number/bin/xsd/year/SwInt.xsd
-axsd TIBCO_HOME/bw/palettes/swift/version_number/bin/xsd/year/$ahV10.xsd
-mxsd TIBCO_HOME/bw/palettes/swift/version_number/bin/xsd/year/ -ib IBAN_
dir
-v TIBCO_HOME/bw/palettes/swift/version_number/samples/SampleMX.xml
```

For more information about the options, see [SwiftMXCheck Options](#).

Note: For the CBPR+ message, the argument mxsd should contain the full path of the .xsd file, that is -mxsd TIBCO_HOME/bw/palettes/swift/version_number/bin/xsd/year/xsd_name.xml.

The path separator should be OS-specific. For Windows (using backward slash (\) in the path), Linux (using forward slash (/) in the path) and MAC OS (using forward slash (/) in the path).

For example, if you are using the **SwiftMXCheck** utility for a cbpr+ message CBPRPlus-pacs_009_001_08_ADV. Then the mxsd argument should be -mxsd TIBCO_HOME/bw/palettes/swift/version_number/bin/xsd/year/CBPRPlus-pacs_009_001_08_ADV.xml.

4. Save the mySwiftMXCheck.tra file and exit the text editor.
5. Open a command line and navigate to the TIBCO_HOME/bw/palettes/swift/version_number/bin directory by using the following command:


```
cd TIBCO_HOME/bw/palettes/swift/version_number/bin
```
6. Type the following command to start the utility:


```
SwiftMXCheck --run --propFile mySwiftMXCheck.tra
```

SwiftMXCheck Options

You can use the metadata options and validation options to validate SWIFT MX message files.

Metadata Options

The following table shows the descriptions of the metadata options:

| Option | Description |
|---------------------|--------------------------------|
| -mxdata <file> | metamxdata directory |
| -set <file> | mxmetadata set |
| -schema <Int> | Schema |
| -txsd <file> | Transport XSD File |
| -axsd <file> | AppHdr XSD File |
| -mxsd <file or dir> | MX XSD File or Dir |
| -ib <dir> | IBAN/BBAN Dir |
| -fi <file> | bankdata file (BIC codes) |
| -cu <file> | currency code file |
| -ct <file> | country code file |
| -ff <file> | validation warning filter file |



Note: In the metadata options, the Schema option accepts one and three integer values: whereas one is for MX schema validation and three is for all schema validation.

Validation Options

The following table shows the description of the validation options:

| Option | Description |
|--------|-------------------------------------|
| -v | validate using all rules |
| -vt | turn on Transport schema validation |

| Option | Description |
|--------|-----------------------------------|
| -va | turn on AppHdr schema validation |
| -vm | turn on MX schema validation |
| -vr | turn on MX rules validation |
| -vb | turn on BIC code validation |
| -vi | turn on IBAN/BBAN code validation |

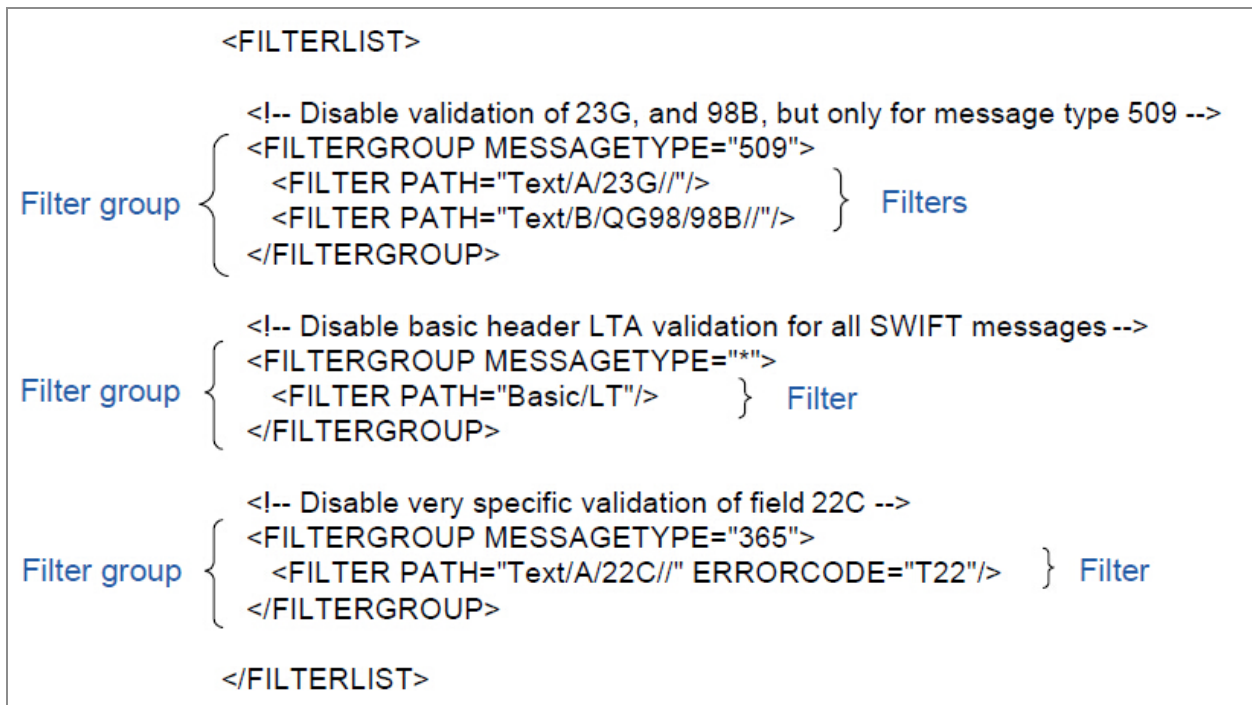
Using Validation Filters

When processing a message, you can customize a validation filter file to ignore some specified validation errors.

Filter Groups

The `ValidationFilter.xml` file consists of a list of filter groups. Each filter group has zero or more filters.

The following figure shows an example list of MT filter groups in the `ValidationFilter.xml` file:



For more details, see the `ValidationFilter.xml` file in the `TIBCO_HOME/bw/palettes/swift/version_number/samples/ValidationFilter_for_MT` or `TIBCO_HOME/bw/palettes/swift/version_number/samples/ValidationFilter_for_MX` directory.

Each filter group has a specification that defines the **SWIFT** message type or types to which the group's filters apply. A wildcard "*" character is permitted as the entire message type specification, or can be used after one or more message type characters. The specification "*", for example, applies its filter group to all **SWIFT** message types. The specification "5*" applies its filter group to all **SWIFT** message types whose first digit is five ; the specification "sese.023.001*" applies its filter group to all **SWIFT** message types whose first prefix is sese.023.001, and so on.

Filters

Each filter group has zero or more filters. With these filters, you can have access to even finer-grained control of validation.

Filtering does not apply to parse errors, only to validation errors. If a SWIFT message is invalid at a fundamental structural level and cannot be parsed, such parse errors cannot be filtered out.

Filters for MT messages are different from those for MX messages.

MT Filters

Each MT filter contains a disabled path specification and an optional error code.

A path specification is an XPath-like string that identifies one or more fields in a SWIFT MT message. See [Path Specification](#) for details on how to construct path specifications. All fields that match the MT filter's path specification have their validation warnings ignored.

If an MT filter also contains a SWIFT error code, then in addition to a path match with a field, only those validation errors with that particular error code are ignored. With this feature, you can have access to even finer-grained control of validation.

Consider the following filter for MT messages:

```
<FILTER PATH="Text/A/22C//"/>
```

The preceding filter ignores validation errors for all occurrences of field 22C in sequence A, for all messages in the filter's filter group. Another example:

```
<FILTER PATH="Text/A/22C//" ERRORCODE="T22"/>
```

This filter is more fine-grained than the previous filter. It only ignores validation errors on field 22C in sequence A with SWIFT error code T22. All other validation errors for this field pass through the filter.

Not all MT filters have to apply to the text block of a SWIFT MT message. The following filter disables validation of the basic header's Logical Terminal Address:

```
<FILTER PATH="Basic/LT"/>
```

The following XML contains examples of how to disable various kinds of validation:

```
<FILTERLIST>
<!--Disable validation of 23G, and 98B, but only for message type 509 --
>
<FILTERGROUP MESSAGETYPE="509">
<FILTER PATH="Text/A/23G//" />
<FILTER PATH="Text/B/QG98/98B//" />
</FILTERGROUP>
<!-- Disable basic header LTA validation for all SWIFT messages-->
<FILTERGROUP MESSAGETYPE="*">
<FILTER PATH="Basic/LT" />
</FILTERGROUP>
<!-- Disable very specific validation of field 22C for error code T22 --
>
<FILTERGROUP MESSAGETYPE="365">
<FILTER PATH="Text/A/22C//" ERRORCODE="T22" />
</FILTERGROUP>
</FILTERLIST>
```

For more details, see the `ValidationFilter.xml` file in the `TIBCO_HOME/bw/palettes/swift/version_number/samples/ValidationFilter_for_MT` directory.

Path Specification

An MT filter uses a subset of the XPath grammar with a few additional abbreviations to identify one or more elements of a message.

The XPath like syntax for the identifying path is composed of tokens separated by the slash (/) character. Each token is composed of an element name or the wildcard character (*). Optionally a predicate can be added inside square brackets immediately following the element name. The predicates supported are a simple one based index to identify the position of an element and an expression to identify an attribute of the element. You can only use one predicate in square brackets for each element name. In addition, only tag fields support the attribute value expression. You can use an additional abbreviation with the attribute matching expression. Multiple values can be entered separately by a vertical bar to denote that either of the supplied values must match.

The slash separator character is used to mark the separation of element names. They are also used to identify characteristics of the start and end of the path. A slash at the start of a path identifies the path as an absolute path. Since all paths are evaluated from the message, the path can be a relative path that does not start with a slash or an absolute path that does start with a slash. As an addition to XPath, a slash at the end of the path

indicates that the path must match the last name in the path as well as any immediate children of that element. Two slashes indicate all descendants of the element must match.

The following table shows the description of the path specification abbreviations:

| Field | Description |
|---------|--|
| Text | The message Text block as sequences |
| Basic | The message Basic Header: AppId, Service, LT, Session, Sequence |
| App | The message Application Header: <ul style="list-style-type: none"> • io: IO flag, either "I" or "O". • I: Input Header: Type, Receiver, Priority, Delivery, Obsolescence. • O: Output Header: Type, InputTime, MIR, OutputDate, OutputTime, Priority. |
| User | The message User Header |
| Tags | The message Text block as a flat list of tag fields |
| Trailer | The message Trailer |
| S | The message SWIFT Alliance trailer if it exists |

Valid Paths Example

The following table lists examples of valid paths:

| Example | Description |
|--------------|---|
| /Basic/AppId | The AppId in the Basic Header |
| /Basic/* | Any subfield of the Basic Header |
| Basic/*[2] | The second subfield in the Basic Header |

| Example | Description |
|---|--|
| Basic/2 | The second subfield in the Basic Header |
| /Text/A/A1[2]/A1a/16R/1 | The first subfield of a 16R |
| /Text/A/*[2]/16R/Qualifier | Any second sequence in A |
| Text/A/*/57D/2-3 | Subfield 2 to 3 of tag field 57D in any sequence in A |
| /Tags/16R/Qualifier | Used to identify the subfield by name |
| Tags/*/1 | The first subfield of the any tag field |
| /App/O/MIR | MIR in the Output Header of the Application Header |
| /App/io | The io value for the Application Header |
| /User/119 | Subfield 119 of the User Header |
| Trailer/MAC | Subfield MAC of the Trailer |
| /Text/A/ | A and any immediate sequences, qualifier groups, or fields |
| Text/A// | A and any descendent sequences, qualifier groups, or fields |
| /Text/A/A1[2]/QG22/22F [@Qualifier=MICO] | A 22F tag field where the Qualifier subfield is 'MICO' |
| Text/A/A1[2]/QG22/22F [@Qualifier=MICO FORM] | A 22F tag field where the Qualifier subfield is 'MICO' or 'FORM' |
| // | The message and any element in the message |

Invalid Paths

The following table lists the invalid paths:

| Field | Description |
|-------------------------------------|---|
| Text/A/A1[2]/A1a/16RS/ Qualifier | Multiple option characters for tags with different field names are not supported. |
| Invalid/12A/2 | Invalid part of the message. |

MX Filters

Each MX filter contains a SWIFT error code. All MX messages that match the error code of the MX filter have their validation warnings ignored.

The following code is an example of MX validation filter:

```
<FILTERLIST>
<!-- Disables validation all sese.023.001* message's error code D00008 -
->
<FILTERGROUP MESSAGETYPE="sese.023.001*">
<FILTER ERRORCODE="D00008"/>
</FILTERGROUP>
</FILTERLIST>
```

This filter ignores validation errors in MX messages with the prefix sese.023.001* and SWIFT error code D00008. All other validation errors pass through the filter.

For more details, see the `ValidationFilter.xml` file in the `TIBCO_HOME/bw/palettes/swift/version_number/samples/ValidationFilter_for_MX` directory.

Configuring Customized MX Java Rules

You can configure customized MX Java rules for the plug-in to support additional MX message type schemas.

Procedure

1. Open a code writing software to write the Java code. See [MX Java Rule Code Example](#) for more information.



Note: When writing an MX Java rule code:

- You have to extend the `MXValidationRule` Java class and override the `eval()` method.
- The Java class name must be the same as the rule name in the SWIFT Standard MX file.
- The Java rule code depends on the `dom4j-2.1.3.jar` file and the `com.tibco.bw.palette.swift.common_6.x.x.0xx.jar` file, which are located in the `TIBCO_HOME/bw/palettes/swift/version_number/tools/lib` and `TIBCO_HOME/bw/palettes/swift/version_number/runtime/plugins` directory.

2. After writing the Java code, export the Java code to a JAR file and save it to the `TIBCO_HOME/bw/palettes/swift/version_number/lib` directory.
3. Add a customized MX message type to the `KnownMessages.xml` file, which is located in the `TIBCO_HOME/bw/palettes/swift/version_number/bin/mxdata/year` directory.
4. Configure a customized MX message rule in a specific XML file in the `TIBCO_HOME/bw/palettes/swift/version_number/bin/mxdata/year` directory based on the message type.

Note: The prefix of the MX message type name in an MX message rule must be the same as the suffix of a specific XML file in the *TIBCO_HOME/bw/palettes/swift/version_number/bin/mxdata/year* directory.

For example,

- The semt.021.001.08 MX message rule must be configured in the *MXsemt.xml* file, which is located in the *TIBCO_HOME/bw/palettes/swift/version/bin/mxdata/year* directory.
- The pacs.003.001.09 MX message rule must be configured in the *MXpacs.xml* file, which is located in the *TIBCO_HOME/bw/palettes/swift/version/bin/mxdata/year* directory.
- The acmt.001.001.08 MX message rule must be configured in the *MXacmt.xml* file, which is located in the *TIBCO_HOME/bw/palettes/swift/version/bin/mxdata/year* directory.

For detailed information about an example of the MX message rule, see [MX Message Rule Example](#).

5. Restart TIBCO Business Studio.

Result

The customized MX Java rule becomes available, and the plug-in supports the corresponding MX message.

MX Java Rule Code Example

An example of the MX Java rule code is provided for your reference.

```
package com.tibco.swift2.validate.mx.rules.spec2017;
import java.util.List;
import org.dom4j.Element;
import com.tibco.swift2.msg.MXFinder;
import com.tibco.swift2.msg.SwiftMXMessage;
import com.tibco.swift2.util.SwiftException;
import com.tibco.swift2.validate.ArgResolver;
import com.tibco.swift2.validate.MXValidationRule;
import com.tibco.swift2.validate.MXValidator;
```

```

public class PhysicalDeliveryDetails1Rule extends MXValidationRule {
public PhysicalDeliveryDetails1Rule(String name,String xpath,,String
overrideErrorSeverity, String
overrideErrorCode,String overrideErrorText,
String[] args, ArgResolver resolver) throws SwiftException {
super(name, xpath, overrideErrorSeverity, overrideErrorCode,
overrideErrorText, args, resolver, 0, 0, new
Class[0]);
}
@Override
public boolean eval(SwiftMXMessage msg, MXValidator validator, int
validationLevel) {
MXFinder finderParent=new MXFinder(getParam_Parent());
boolean isValid = true;
List<Element> elementsParent = finderParent.getElements(msg);
String physDlvryIndTag=getParam_PhysDlvryInd().replace("//", "");
String physDlvryDtlsTag=getParam_PhysDlvryDtls().replace("//", "");
for(int i=0;i<elementsParent.size();i++) {
Element elementParent=elementsParent.get(i);
Element elementPhysDlvryInd=elementParent.element(physDlvryIndTag);
Element elementPhysDlvryDtls=elementParent.element(physDlvryDtlsTag);
if(elementPhysDlvryInd!=null) {
if(elementPhysDlvryInd != null){
if(elementPhysDlvryInd.getStringValue().equals("false") ||
elementPhysDlvryInd.getStringValue().equals("0")){
if(elementPhysDlvryDtls!=null) {
msg.addWarning(elementParent, getName(), getOverrideErrorCode
(),getOverrideErrorText());
isValid = false;
}
}
}
}
}
return isValid;
}
public String getParam_PhysDlvryInd(){
return getParam().get("PhysDlvryInd");
}
public String getParam_PhysDlvryDtls(){
return getParam().get("PhysDlvryDtls");
}
public String getParam_Parent(){
return getParam().get("Parent");
}
}

```

MX Message Rule Example

An example of the MX message rule is provided for your reference.

```
<MESSAGE TYPE="setr.001.001.04" NAME="RedemptionBulkOrderV04">
<RULE NAME="C3" EXPRESSION="ActiveCurrencyRule()" XPATH="//*/@Ccy |
//ReqdSttlmCcy"
  ERRORTTEXT="Invalid currency code" ERRORCODE="D00005"/>
<RULE NAME="C4" EXPRESSION="ActiveOrHistoricCurrencyRule()"
  XPATH="//*/@Ccy | //UnitCcy |
  //ReqdNAVCcy | //QtdCcy" ERRORTTEXT="Invalid currency code"
  ERRORCODE="D00006"/>
<RULE NAME="C5" EXPRESSION="AnyBICRule()" XPATH="//BICorBEI | //Pty"
  ERRORTTEXT="Invalid
  BIC" ERRORCODE="D00008"/>
<RULE NAME="C6" EXPRESSION="BICFIRule()" XPATH="//BICFI"
  ERRORTTEXT="Invalid FI BIC."
  ERRORCODE="D00001"/>
<RULE NAME="C8" EXPRESSION="CountryRule()" XPATH="//Ctry | //DmstIdSrc"
  ERRORTTEXT="Invalid
  Country Code" ERRORCODE="D00004"/>
<RULE NAME="C9" EXPRESSION="CurrencyAmountRule()" XPATH="//*/@Ccy"
  ERRORTTEXT="Invalid
  currency code or too many decimal digits" ERRORCODE="D00007"/>
<RULE NAME="C10" EXPRESSION="CurrencyAmountRule()" XPATH="//*/@Ccy"
  ERRORTTEXT="Invalid
  currency code or too many decimal digits" ERRORCODE="D00007"/>
<RULE NAME="C12" EXPRESSION="spec2016.DeliverersIntermediary1DetailsRule
  ()"
  ERRORTTEXT="DeliverersIntermediary1Details must be present."
  ERRORCODE="X00395">
<PARAM NAME="Parent" XPATH="//DlvrgSdDtls"/>
<PARAM NAME="DlvrrsIntrmy1Dtls" XPATH="//DlvrrsIntrmy1Dtls"/>
<PARAM NAME="DlvrrsIntrmy2Dtls" XPATH="//DlvrrsIntrmy2Dtls"/>
</RULE>
<RULE NAME="C13" EXPRESSION="spec2017.DiscountElementRule()"
  ERRORTTEXT="Amount Or Rate Or
  Basis must be present." ERRORCODE="X00410">
<PARAM NAME="Parent" XPATH="//DscntDtls"/>
<PARAM NAME="Amt" XPATH="//Amt"/>
<PARAM NAME="Rate" XPATH="//Rate"/>
<PARAM NAME="Bsis" XPATH="//Bsis"/>
</RULE>
<RULE NAME="C17" EXPRESSION="IBANRule()" XPATH="//IBAN"
  ERRORTTEXT="Invalid IBAN format or
```

```

invalid check digits" ERRORCODE="D00003"/>
<RULE NAME="C23" EXPRESSION="spec2017.PhysicalDeliveryDetails1Rule()"
ERRORTEXT="PhysicalDeliveryDetails is not allowed." ERRORCODE=" X00396">
<PARAM NAME="Parent" XPATH="//IndvOrdrDtls"/>
<PARAM NAME="PhysDlvryInd" XPATH="//PhysDlvryInd"/>
<PARAM NAME="PhysDlvryDtls" XPATH="//PhysDlvryDtls"/>
</RULE>
<RULE NAME="C24" EXPRESSION="spec2017.PhysicalDeliveryDetails2Rule()"
ERRORTEXT="PhysicalDeliveryDetails must be present." ERRORCODE="X00397">
<PARAM NAME="Parent" XPATH="//IndvOrdrDtls"/>
<PARAM NAME="PhysDlvryInd" XPATH="//PhysDlvryInd"/>
<PARAM NAME="PhysDlvryDtls" XPATH="//PhysDlvryDtls"/>
</RULE>
<RULE NAME="C26" EXPRESSION="spec2016.ReceiversIntermediary1DetailsRule
()"
ERRORTEXT="ReceiversIntermediary1Details must be present."
ERRORCODE="X00378">
<PARAM NAME="Parent" XPATH="//RcvgSdDtls"/>
<PARAM NAME="RcvrsIntrmy1Dtls" XPATH="//RcvrsIntrmy1Dtls"/>
<PARAM NAME="RcvrsIntrmy2Dtls" XPATH="//RcvrsIntrmy2Dtls"/>
</RULE>
</MESSAGE>

```


Processing Acknowledgment Messages

The acknowledgment is sent by the MT service to the Logical Terminal to confirm the receipt of a message and its safe storage by the service.

If an ACK is returned, the message is accepted by the MT service for delivery to its destination. If a NAK is returned, the message is safely stored, but failed to deliver. The structure of an ACK or a NAK message is the same, except that the value of tag 451 is in the text block of an Acknowledgment message. 0 indicates the ACK, and 1 indicates the NAK.

MT acknowledgment messages are parsed using the FIN_Acknowledgment schema. This schema loads automatically after you create a Load SWIFT MT schema shared resource. The plug-in supports the parsing and rendering of the ACK and NAK messages.

Procedure

1. Open the process that you want to process the acknowledge messages.



Note: Ensure that you have created a Load SWIFT MT schema shared resource in this process.

2. Click the **General** tab of the Parse SWIFT MT activity or Render SWIFT MT activity.
3. In the **SWIFT Message Schema** field, click and select the Message_FIN_Acknowledgement message schema.
4. Save the process.

Reconciling Acknowledgment Messages

The acknowledgment message from the SWIFT network contains field 108. The field carries Message User Reference (MUR) information of the original message. This information can be presented in one of the following ways:

- MUR information is present in the user header of the original message.
- If no MUR is present in the original message, contents of Field 20 of the original

message or (for Category 5 messages only) the contents of Field 20C, with the code word SEME, but only when all the letters of the alphabets are uppercase.

- Contents of Field 20C.

For reconciling, you must have the reference information of the MT message sent and the value of the tag 108 of the acknowledgment message received. Compare both information. If they match, the reconciliation is successful.

Migrating Projects to the Current SWIFT Standards

Every year, SWIFT releases new message standards. This release of the plug-in supports the standard release 2024. Therefore, you have to migrate existing projects to the current SWIFT standards.

To migrate projects created in version 6.9.0 (2023 specification) of TIBCO ActiveMatrix BusinessWorks™ Plug-in for SWIFT, you can import the projects directly to version 6.10.0 (2023 specification) of the plug-in, or open the projects directly with version 6.10.0 (2023 specification) of the plug-in.

The migrations to the current SWIFT standards are different for projects with SWIFT MT activities and projects with SWIFT MX activities.

Migrating Projects with SWIFT MT Activities

You can migrate your projects that use SWIFT MT activities and the previous SWIFT standards to the current SWIFT standards.



Note: The known issues section of the release notes lists issues that might occur when migrating from version 6.9.0 to 6.10.0 of the plug-in. Use the workaround mentioned in the release notes to fix the migration issue manually.

Procedure

1. Uninstall the previous version of the plug-in.
2. Install the new version of the plug-in.
3. Start **TIBCO Business Studio**.
4. Click **File > Import**
5. In the Import window, select **General > Existing Studio Projects into the Workspace**, and click **Next**.

6. In the Import Projects window, you can select projects from the **Select root directory** or **Select archive files**.
7. Click **Finish** to import the projects.
8. In the **General** tab of the Load SWIFT MT Schema shared resource, click **Unload Selected** to unload message type schemas of the previous SWIFT MT standard release, and then select the new SWIFT specification. For example, **SWIFT November 2024** specification, from the Specification list.
9. In the **General** tab, select the checkboxes next to the corresponding message type schemas as required in the existing project.
10. Click **Load Selected** to reload the schema.




Note: In SWIFT Standards Release 2024, structural changes in a **SWIFT** message can cause the mappings from the existing projects to break. Restore the broken links manually.

Migrating Projects with SWIFT MX Activities

You can migrate your projects that use the previous SWIFT MX activities and the previous SWIFT standards to the current SWIFT standards.

Procedure

1. Uninstall the previous version of the plug-in.
2. Install the new version of the plug-in.
3. Download required XSD files from the SWIFT website, and save them in the TIB_BWPLUGINSWIFT_HOME\bin\xsd\year directory.
The XSD files must correspond to the message type schemas that are used in the existing project.
4. Start **TIBCO Business Studio**.
5. Click **File > Import**
6. In the Import window, select **General > Existing Studio Projects into the Workspace** and click **Next**.

7. In the Import Projects window, you can select projects from the **Select root directory** or **Select archive files**.
8. Click **Finish** to import the projects.
9. In the **General** tab of the Load SWIFT MX Schema shared resource, click **Unload Selected** to unload message type schemas of the previous SWIFT MX standard release and then select the new SWIFT specification. For example, **SWIFT November 2024 specification**, from the Specification list.
10. In the **General** tab of the Parse SWIFT MX or Render SWIFT MX activity, configure the following fields as required:
 - a. To clear the previous SWIFT specification, click **Clear value**.
 - b. Click **Choose/Create Default Resource**  next to the SWIFT Specification field to reload the SWIFT specification.
 - c. To remove the dependencies that are generated from the SWIFT 2023 specification, click **BW processes** property.
11. From the menu bar, click **Project > Save** to save the project.

i Note: If an MX message used in existing projects is updated, you must update the corresponding MX message file accordingly.

Migrating from Version 5.23.0 of TIBCO ActiveMatrix BusinessWorks™ Plug-in for SWIFT

You can migrate TIBCO ActiveMatrix BusinessWorks™ Plug-in for SWIFT 5.23.0 projects to TIBCO ActiveMatrix BusinessWorks™ Plug-in for SWIFT 6.10.0 by performing the following steps:

Procedure

1. Start **TIBCO Business Studio**.
2. Click the **Project** tab, and then select **Migrate BW Projects**.
3. In the Project Migration Wizard, click **Browse** in the **Select Project(s) to be Migrated** field, and select the project you want to migrate.

The default selection is **Migrate Single BusinessWorks 5.x Project**. Click **Migrate Multiple BusinessWorks 5.x Projects** to migrate multiple 5.x projects.

4. Click **Migrate Project**.
5. Click **Start Migration**.
6. Click **Finish** after the migration is complete.



Note: The known issues section of the release notes lists issues that might occur when migrating from version 5.23.0 to 6.10.0 of the plug-in. Use the workaround mentioned in the release notes to fix the migration issue manually.

Managing Logs

When an error occurs, you can check logs to trace and troubleshoot the plug-in exceptions.

By default, when you run a process in the debug mode, error logs are displayed in the Console view. You can change the log level of the plug-in to trace different messages and export logs to a file. Different log levels correspond to different messages, as described in [Log Levels](#).

Log Levels

Different log levels include different information.

The plug-in supports the following log levels:

| Log Level | Description |
|-----------|---|
| Trace | Includes all information regarding the running process. |
| Debug | Indicates a developer-defined tracing message. |
| Info | Indicates normal plug-in operations. No action is required. A tracing message tagged with Info indicates that a significant processing step is reached, and logged for tracking or auditing purposes. Only info messages preceding a tracking identifier are considered as significant steps. |
| Warn | Indicates that an abnormal condition occurred. Processing continues, but special attention from an administrator is recommended. |
| Error | Indicates that an unrecoverable error occurred. Depending on the severity of the error, the plug-in might continue with the next operation or might stop. |

Setting Up Log Levels

You can configure a different log level for the plug-in and plug-in activities to trace different messages.

By default, the plug-in uses the log level configured for TIBCO ActiveMatrix BusinessWorks. The default log level of TIBCO ActiveMatrix BusinessWorks is Error.

Procedure

1. Navigate to the *TIBCO_HOME/bw/version_number/config/design/logback* directory and open the *logback.xml* file.
2. Add the following node in the **BusinessWorks Palette and Activity loggers** area to specify a log level for the plug-in.

For the activities in the SWIFT MT palette, add the following node:

```
<logger name="com.tibco.bw.palette.swift.runtime">
  <level value="DEBUG"/>
</logger>
```

For the activities in the SWIFT MX palette, add the following node:

```
<logger name="com.tibco.bw.palette.swiftmx.runtime">
  <level value="DEBUG"/>
</logger>
```

The value of the `level` element can be Error, Info, or Debug.

i Note: If you set the log level to Debug, the input, and output for the plug-in activities are also displayed in the Console view. See [Log Levels](#) for more details regarding each log level.

3. Add the following node in the **BusinessWorks Palette and Activity loggers** area to specify a log level for an activity:

For the activities in the SWIFT MT palette, add the following node:

```
<logger name="com.tibco.bw.palette.swift.runtime.ActivityNameActivity">
  <level value="DEBUG"/>
</logger>
```


i Note: For each activity, the *ActivityName* is:

- Parse SWIFT MT: Parser
- Render SWIFT MT: Renderer
- Route SWIFT MT: Router
- Generate SWIFT BICPlusIBAN: BICPlusIBANGenerator
- Validate SWIFT BICPlusIBAN: BICPlusIBANValidator

For the activities in the SWIFT MX palette, add the following node:

```
<logger
name="com.tibco.bw.palette.swiftmx.runtime.ActivityNameActivity">
  <level value="DEBUG"/>
</logger>
```

i Note: For each activity, the *ActivityName* is:

- Parse SWIFT MX: Mxparser
- Render SWIFT MX: Mxrenderer

For example, add the following node to set the log level of the Parse SWIFT MT activity to Debug:

```
<logger name="com.tibco.bw.palette.swift.runtime.ParserActivity">
  <level value="DEBUG"/>
</logger>
```

i Note: The activities that are not configured with specific log levels use the log level configured for the plug-in.

4. Save the file.

Exporting Logs to a File

You can update the `logback.xml` file to export plug-in logs to a file.

Procedure

1. Navigate to the `TIBCO_HOME/bw/version_number/config/design/logback` directory and open the `logback.xml` file.

Note: After deploying an application in TIBCO Enterprise Administrator, navigate to the `TIBCO_HOME/bw/version_number/domains/domain_name/appnodes/space_name/node_name` directory to find the `logback.xml` file.

2. Add the following node to specify the file where the log is exported:

```
<appender name="FILE" class="ch.qos.logback.core.FileAppender">
  <file>c:/bw6-swift.log</file>
  <encoder>
    <pattern>%d{HH:mm:ss.SSS} [%thread] %-5level %logger
{36}-%msg%n</pattern>
  </encoder>
</appender>
```

The value of the `file` element is the absolute path of the file that stores the exported log.

3. Add the following node to the root node at the bottom of the `logback.xml` file:

```
<logger name="com.tibco.bw.palette.swift">
  <appender-ref ref="STDOUT" />
  <appender-ref ref="FILE" />
  <level value="DEBUG"/>
</logger>
```

4. Save the file.

Error Codes

The following table lists error codes, a detailed explanation of each error, and where applicable, ways to solve different errors.

| Error Code and Error Message | Role | Category | Description | Solution |
|--|-------|------------|--|--|
| TIBCO-BW-PALETTE-SWIFT-200001 {0} | debug | BW-Plug-in | Shows the input or output and configuration parameters of an activity in XML format. | This is a debug message and resolution is not applicable. |
| TIBCO-BW-PALETTE-SWIFTMX-200001 {0} | debug | BW-Plug-in | Shows the input or output and configuration parameters of an activity in XML format. | This is a debug message and resolution is not applicable. |
| TIBCO-BW-PALETTE-SWIFT-500004 ERROR_ FORMAT2={0} {1}{2} | error | BW-Plug-in | When an MT message does not follow the SWIFT specification, or an activity cannot locate the metadata directory or fails to initialize the metadata. | Check errors listed in the error message, and take appropriate action. |
| TIBCO-BW-PALETTE-SWIFT-500005 ERROR_ | error | BW-Plug-in | After the validation of an MT message, an error occurs when you transfer the message. | Check errors listed in the error message, and take appropriate |

| Error Code and Error Message | Role | Category | Description | Solution |
|--|-------|------------|--|--|
| FORMAT3={0} {1} | | | | action. |
| TIBCO-BW- PALETTE- SWIFTMX- 500004 ERROR_ FORMAT2={0} {1}{2} | error | BW-Plug-in | When an MX message does not follow the SWIFT specification, or an activity cannot locate the metadata directory or fails to initialize the metadata. | Check errors listed in the error message, and take appropriate action. |
| TIBCO-BW- PALETTE- SWIFTMX- 500005 ERROR_ FORMAT3={0} {1} | error | BW-Plug-in | After the validation of an MX message, an error occurs when you transfer the message. | Check errors listed in the error message, and take appropriate action. |

TIBCO Documentation and Support Services

For information about this product, you can read the documentation, contact TIBCO Support, and join TIBCO Community.

How to Access TIBCO Documentation

Documentation for TIBCO products is available on the [Product Documentation website](#), mainly in HTML and PDF formats.

The [Product Documentation website](#) is updated frequently and is more current than any other documentation included with the product.

Product-Specific Documentation

The documentation for this product is available on the [TIBCO ActiveMatrix BusinessWorks™ Plug-in for SWIFT Product Documentation](#) page.

How to Contact Support for TIBCO Products

You can contact the Support team in the following ways:

- To access the Support Knowledge Base and getting personalized content about products you are interested in, visit our [product Support website](#).
- To create a Support case, you must have a valid maintenance or support contract with a Cloud Software Group entity. You also need a username and password to log in to the [product Support website](#). If you do not have a username, you can request one by clicking **Register** on the website.

How to Join TIBCO Community

TIBCO Community is the official channel for TIBCO customers, partners, and employee subject matter experts to share and access their collective experience. TIBCO Community offers access to Q&A forums, product wikis, and best practices. It also offers access to extensions, adapters, solution accelerators, and tools that extend and enable customers to gain full value from TIBCO products. In addition, users can submit and vote on feature

requests from within the [TIBCO Ideas Portal](#). For a free registration, go to [TIBCO Community](#).

Legal and Third-Party Notices

SOME CLOUD SOFTWARE GROUP, INC. (“CLOUD SG”) SOFTWARE AND CLOUD SERVICES EMBED, BUNDLE, OR OTHERWISE INCLUDE OTHER SOFTWARE, INCLUDING OTHER CLOUD SG SOFTWARE (COLLECTIVELY, “INCLUDED SOFTWARE”). USE OF INCLUDED SOFTWARE IS SOLELY TO ENABLE THE FUNCTIONALITY (OR PROVIDE LIMITED ADD-ON FUNCTIONALITY) OF THE LICENSED CLOUD SG SOFTWARE AND/OR CLOUD SERVICES. THE INCLUDED SOFTWARE IS NOT LICENSED TO BE USED OR ACCESSED BY ANY OTHER CLOUD SG SOFTWARE AND/OR CLOUD SERVICES OR FOR ANY OTHER PURPOSE.

USE OF CLOUD SG SOFTWARE AND CLOUD SERVICES IS SUBJECT TO THE TERMS AND CONDITIONS OF AN AGREEMENT FOUND IN EITHER A SEPARATELY EXECUTED AGREEMENT, OR, IF THERE IS NO SUCH SEPARATE AGREEMENT, THE CLICKWRAP END USER AGREEMENT WHICH IS DISPLAYED WHEN ACCESSING, DOWNLOADING, OR INSTALLING THE SOFTWARE OR CLOUD SERVICES (AND WHICH IS DUPLICATED IN THE LICENSE FILE) OR IF THERE IS NO SUCH LICENSE AGREEMENT OR CLICKWRAP END USER AGREEMENT, THE LICENSE(S) LOCATED IN THE “LICENSE” FILE(S) OF THE SOFTWARE. USE OF THIS DOCUMENT IS SUBJECT TO THOSE SAME TERMS AND CONDITIONS, AND YOUR USE HEREOF SHALL CONSTITUTE ACCEPTANCE OF AND AN AGREEMENT TO BE BOUND BY THE SAME.

This document is subject to U.S. and international copyright laws and treaties. No part of this document may be reproduced in any form without the written authorization of Cloud Software Group, Inc.

TIBCO, the TIBCO logo, the TIBCO O logo, ActiveMatrix BusinessWorks, Business Studio, and TIBCO Business Studio are either registered trademarks or trademarks of Cloud Software Group, Inc. in the United States and/or other countries.

All other product and company names and marks mentioned in this document are the property of their respective owners and are mentioned for identification purposes only. You acknowledge that all rights to these third party marks are the exclusive property of their respective owners. Please refer to Cloud SG’s Third Party Trademark Notices (<https://www.cloud.com/legal>) for more information.

This document includes fonts that are licensed under the SIL Open Font License, Version 1.1, which is available at: <https://scripts.sil.org/OFL>

Copyright (c) Paul D. Hunt, with Reserved Font Name Source Sans Pro and Source Code Pro.

Cloud SG software may be available on multiple operating systems. However, not all operating system platforms for a specific software version are released at the same time. See the “readme” file for the availability of a specific version of Cloud SG software on a specific operating system platform.

THIS DOCUMENT IS PROVIDED “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT.

THIS DOCUMENT COULD INCLUDE TECHNICAL INACCURACIES OR TYPOGRAPHICAL ERRORS. CHANGES ARE PERIODICALLY ADDED TO THE INFORMATION HEREIN; THESE CHANGES WILL BE INCORPORATED IN NEW EDITIONS OF THIS DOCUMENT. CLOUD SG MAY MAKE IMPROVEMENTS AND/OR CHANGES IN THE PRODUCT(S), THE PROGRAM(S), AND/OR THE SERVICES DESCRIBED IN THIS DOCUMENT AT ANY TIME WITHOUT NOTICE.

THE CONTENTS OF THIS DOCUMENT MAY BE MODIFIED AND/OR QUALIFIED, DIRECTLY OR INDIRECTLY, BY OTHER DOCUMENTATION WHICH ACCOMPANIES THIS SOFTWARE, INCLUDING BUT NOT LIMITED TO ANY RELEASE NOTES AND "README" FILES.

This and other products of Cloud SG may be covered by registered patents. For details, please refer to the Virtual Patent Marking document located at <https://www.cloud.com/legal>.

Copyright © 2001-2024. Cloud Software Group, Inc. All Rights Reserved.