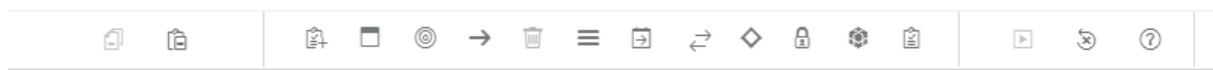


INTEGRATION SUITE PALETTES

Palettes:

Palettes enable one to create I-flow with various different methods for diverse scenarios.



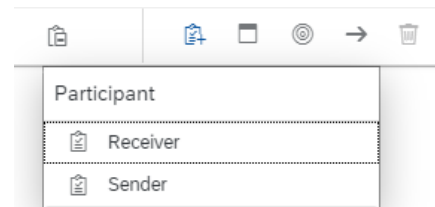
Copy & Paste

These are general copy and paste for the palettes in Integration Suite

Participants

There are two kind of participants available in Integration Suite

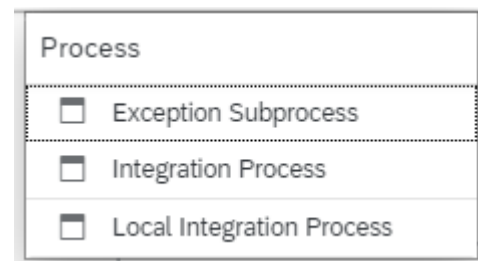
- Sender
- Receiver



Process

There are 3 process available here for three different purposes as follows,

- Exception Subprocess
- Integration Process
- Local Integration Process



Exception Subprocess

It is used when an exception has been thrown by integration flow for handling errors.

Integration Process

It is the basic integration process which contains all the properties of the integration flow

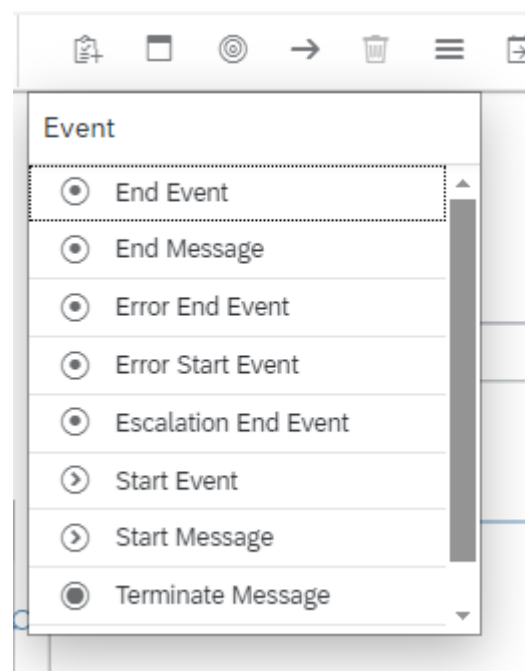
Local Integration Process

It is a kind of an integration Process which can be called by the main Integration Process when it is needed.

Event

Events are used to specify a particular happenings of the integration flow. It contains a set of events as follows,

- End event
- End Message
- Error End Event
- Error Start Event
- Escalation Event
- Start Event
- Start Message
- Terminate Message
- Timer



End event

An End Message event ends a message processing sequence

End Message

It is the end of the message which is processed through the way. It can be mapped with receiver participants using various adapters.

Error End Event

It is used when the i-flow status is failed but it has been handled in exception subprocess successfully.

Error Start Event

It is used to define how to handle errors when message processing fails at runtime. Only can be used in the Exception Subprocess.

Escalation Event

If an Escalation End Event is used in the main integration process, the escalation end event throws an Escalated event and the processing stops afterwards as there are no further steps to execute.

If an integration process is calling a local integration process and an escalation event has been defined in the local integration process, the Escalation Event is thrown, and the processing continues in the calling flow. If there are still steps to process in the main flow, the processing continues normally at this level and the main flow message status will be Completed.

It allows some categories for the use for the Escalation end event to process such as receiver not found.

Terminate Message

A Terminate Message event stops further processing of a message.

Timer

Timer is used to schedule the message process with some specific condition such as run once or run at specific time or run every

time which matches the parameters entered and allows to give some advanced time settings.

Start Event

It denotes the starting position of the event in the local integration process or exception sub-process.

Start Message

It is the start of the message that can be mapped with the sender participant with the use of various adapters.

Connectors

Establish connection between the two flow → steps.

Delete

Delete the i-flow objects

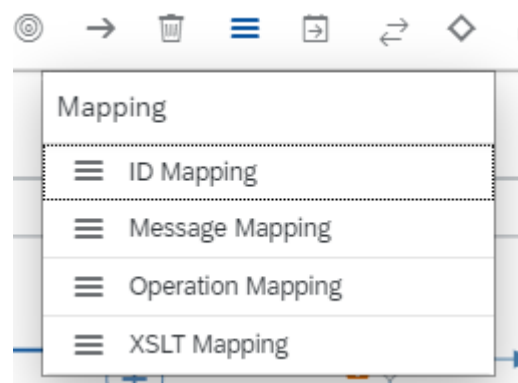


Mapping

Mappings are used to transfer source format into target format.

It supports the following mappings,

- ID mapping
- Message Mapping
- Operation Mapping
- XSLT Mapping



ID mapping

It maps using Message IDs. It maps a Source message id to Target message ID.

Message Mapping

Message Mapping maps source structure to target structure. We can use multiple Mapping functions in Message Mapping.

Operation Mapping

Operation mapping is used to relate an outbound service interface operation with an inbound service interface operation. You can also relate IDoc and RFC interfaces with entities of the same type or with service interface operations.

XSLT Mapping

Maps using XSLT code which can be provided in the editor in i-flow.

Transformation

Transformation option provides the following features,

- Content Modifier
- Converter
- Decoder
- EDI Extractor
- Encoder
- Filter
- Message Digest
- Script
- XML Modifier

Content Modifier

Content Modifier enables modifying the content of the message by changing the MessageHeader and Message body and Exchange Property.

Syntax for call the header element `${header.elementname}`

Attribute for creating content modifier:

- Action : create or delete

- Name : name of the attribute or field value
- Source Type :
 - constant - create a constant value and special characters not allowed.
 - Header - Allows you to specify the name of a Camel header.
 - XPath - Retrieve data from incoming messages using Xpath syntax.
 - Expression- Allows you to enter a Camel Simple Expression Language expression (`${exchangeId}`)
 - Property - Specify an exchange property
 - External Parameter - Integration developer can use externalization and externalize any type of headers or exchange properties
 - Local Variable - to create a local variable
 - Global Variable - to create a global value
 - Number Range - to specify the range of numbers
- Source value - Value for the created row property
- Data Type - Valid Java Data Type.
- Default Value - It is enabled when use local or global data type and used when the value for the field doesn't found at runtime

Converter

It is used to convert one format into another format using specific parameters. Available conversions as follows,

- XML to JSON
- XML to EDI
- XML to CSV
- CSV to XML
- EDI to XML
- JSON to XML

Decoder

Decodes the encoded incoming message using following methods,

- Base64 Decoder
- GZIP Decompression
- MIME Multipart Decoder
- ZIP Decompression

EDI Extractor

- It is used to extract EDI header and transfer to camel headers. It can read flat and xml files.
- It supports EDIFACT, EANCOM, ODETTE, and ASC-X12 documents.

Encoder

Encodes the incoming message to the inbound process using the following methods,

- Base64 Encoder
- GZIP Compression
- MIME Multipart Encoder
- ZIP Compression

Filter

Filter the incoming message using Xpath expression with the value type of Boolean, Integer, Node, Nodelist, String.

Boolean - returns true or false

Integer - returns integer values

Node - returns the first match with the elements

Nodelist - returns all the matches with the elements

String - returns the first match without elements

Message Digest

You can use the Message Digest step to simply check if a message has been changed without checking the payload.

In this process a header will be created and the digest value stored in it.

The value of that specific header will be identical for the same payloads.

Script

It enables the use of JavaScript and Groovy Script in integration flow.

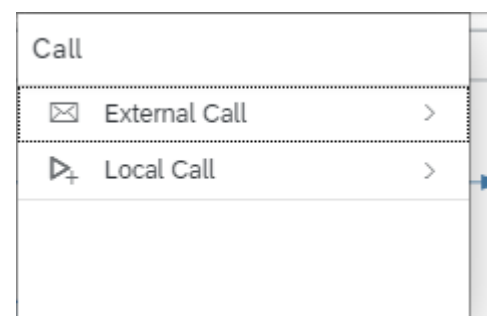
XML Modifier

It is used to modify the xml content by removing external DTD and XML declarations.

Call

Call enables calling the local and external options using the following options.

- External Call
 - Content Enricher
 - Poll Enrich
 - Request Reply
 - Send
- Internal Call
 - Idempotent Process Call
 - Looping Process Call
 - Process Call



External Call

Content Enricher

- Content Enricher is used to enrich the original message using external messages.
- We can combine or enrich the incoming message with the external message such as OData or SuccessFactors.
- In the combine method the external message will be merged with the original message using multimapping.
- In the Enrich method we can look up the external message with the original message by providing the root node and key element of original and look up messages.

Poll Enrich

- Poll Enrich is also used as content Enrich but it is used when the external message is from SFTP only.
- We can use any file here.
- If it is not xml we can send the output as zip or tar file
- For other types we can concatenate the both messages if we need.

Request Reply

It is used to fetch data from external systems to the integration process using various adapters.

Send

It is used to push data from integration processes to the external systems through various adapters.

Internal Call

Idempotent Process Call

- Idempotent Process Call is used to call the local integration process only once using the message id.

- If the same message Id came as input second time the idempotent process call identify it using the message Id and ignore the process call and move it to the next step.

Looping Process Call







Looping process call is used to call a local integration process as a loop until a condition is failed.

Process Call

Process call is used to call a local integration process in the main integration process or exception subprocess or another local integration process.

Routing

- Aggregator
- Gather
- Join
- Multicast
- Router
- Splitter

Routing	
	Aggregator
	Gather
	Join
	Multicast >
	Router
	Splitter >

Aggregator

- Aggregator gets and stores the receiving data from external systems in a specific data store until a condition is met.
- It will send the merged content of the stored data as a single message once the specified condition is met.
- We can set a time to send the data if the condition did not meet but even for the concern of the space of data storage capabilities.

Gather and Join

- Gather and Join are used to merge input from multiple routes.
- Join merges the data from multiple routes without changing the content
- Gather merges the input data using a multi mapping method.\
- If the input is plain text or any other format, gather will merge it and provide the merged output as zip or tar file.

Multicast

- Multicast is the feature that enables us to send data into multiple receivers using multiple branches.
- It has two methods: Parallel Multicast, Sequential Multicast.
 - In Parallel multicast we cannot specify the sequence of the branches.
 - In Sequential Multicast we can specify the sequence of the branches.

Router

- Router is used to divert the way of transport using specific conditions.
- Every router must have a default route.

Splitter

- Splitter is used to split the large message into a series of individual messages.
- It is mandatory to have a gathering step to get all individual splitted messages.
- If splitter is used without gather in a local integration process the output payload from the process call will be

skip the splitter function and send the same input as output.

Security

- Decryptor
- Encryptor
- Signer
- Verifier



Decryptor

Decrypts the encrypted input payload using PGP keyrings or PKCS7 certificates.

Encryptor

Encrypts the input payload using PGP keyrings or PKCS7 certificates.

Signer

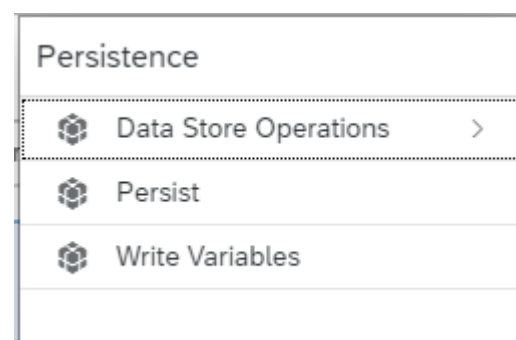
Signs the message using PKCS7, simple and Xml digital signer.

Verifier

Verifier is used to check whether the sign is authentic or not.

Persistence

- Data Store Operations
- Persist
- Write Variables



Data Store Operations

WRITE - Write a new data into a data store

GET - Retrieve the data from the data store using entry id which is processed one by one.

SELECT - Gets the entire Data from the Data Stores.

DELETE - Delete the data from the Data Stores.

Persist

- Persist is used to save data permanently in the tenant database.
- Tenant database has 32 gb space.
- We need to access OData api to access the persisted message

Write Variable

It is used to write global or local variables for later use.

Validator

It is use to validate a xml to EDI file using schema definitions.