TIBCO ActiveMatrix BusinessWorks™ Plug-in for Apache Kafka

User's Guide

Version 6.4.1

October 2020

Contents

TIBCO Documentation and Support Services

How to Access TIBCO Documentation

Documentation for TIBCO products is available on the TIBCO Product Documentation website, mainly in HTML and PDF formats.

The TIBCO Product Documentation website is updated frequently and is more current than any other documentation included with the product. To access the latest documentation, visit https://docs.tibco.com.

Product-Specific Documentation

The following documents for this product can be found on the TIBCO Documentation site: https://docs.tibco.com/products/tibco-activematrix-businessworks-plug-in-for-apache-kafka.

- *TIBCO ActiveMatrix BusinessWorks Plug-in for Kafka Release Notes*
- *TIBCO ActiveMatrix BusinessWorks Plug-in for Kafka Installation*
- *TIBCO ActiveMatrix BusinessWorks Plug-in for Kafka User's Guide*

How to Contact TIBCO Support

You can contact TIBCO Support in the following ways:

- For an overview of TIBCO Support, visit http://www.tibco.com/services/support.
- For accessing the Support Knowledge Base and getting personalized content about products you are interested in, visit the TIBCO Support portal at https://support.tibco.com.
- For creating a Support case, you must have a valid maintenance or support contract with TIBCO. You also need a user name and password to log in to https://support.tibco.com. If you do not have a user name, 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 https://community.tibco.com.

Plug-in Overview

Apache Kafka is a distributed messaging system, providing fast, highly scalable, and redundant messaging through a publisher-subscriber model.

It is available and resilient to node failures and supports automatic recovery. Apache Kafka has robust queues that can handle a high volumes of data and has an enabler to pass on the messages from one endpoint to another. Apache Kafka is suitable for both offline and online message consumption. Apache Kafka is built on top of the Apache ZooKeeper™ synchronization service. All Kafka messages are organized into topics.

TIBCO ActiveMatrix BusinessWorks™ Plug-in for Apache Kafka plugs into TIBCO ActiveMatrix BusinessWorks™. The Apache Kafka palette can be used to create producers, consumers, and perform send message and receive message operations.

The plug-in provides the following main features:

- Kafka Connection Shared Resource

  Kafka connection shared resource is used to connect and fetch the list of topics from the Kafka server. The shared resource is used by the other activities for configuring the connection.
- Kafka Send Message Activity

  This activity is used to send messages to the Kafka consumer. It performs as a Kafka producer, which sends the message to a specified topic, and consumers can fetch the message from the specified topics.
- Kafka Receive Message Activity

  This activity is a process starter activity that starts the process execution on receiving a Kafka message.
- Avro Schema Registry

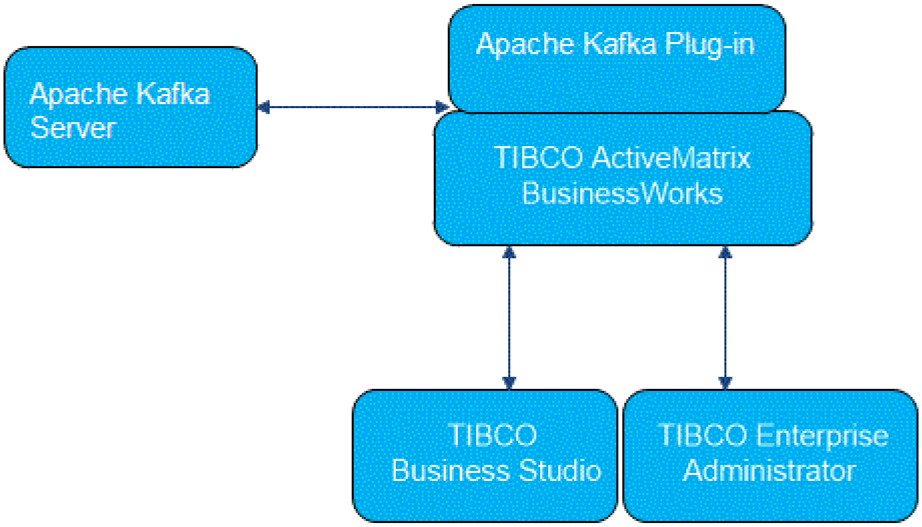
  Avro schema is used with or without registry to configure Send and Receive activities to use supplied avro schema as a message format.
- Kafka Transactions

  Kafka Transactions is used to support exactly once semantics and Kafka Transaction API.
- Kafka Get Messages Activity

  This activity is used to consume messages from Kafka topics. The activity also allows you to seek partition offset to beginning, end or custom offset value.

Plug-in Architecture

The following figure describes the relationship between the Apache Kafka Server, TIBCO ActiveMatrix BusinessWorks™ Plug-in for Apache Kafka, and TIBCO ActiveMatrix BusinessWorks™.


Apache Kafka Server
Apache Kafka Plug-in
TIBCO ActiveMatrix BusinessWorks
TIBCO Business Studio
TIBCO Enterprise Administrator

The following list describes the relationship between different products in the previous figure.

- TIBCO ActiveMatrix BusinessWorks™ Plug-in for Apache Kafka communicates with the Kafka server instance.
- TIBCO ActiveMatrix BusinessWorks™ Plug-in for Apache Kafka plugs into TIBCO ActiveMatrix BusinessWorks™ and connects to a Kafka server instance.
- TIBCO ActiveMatrix BusinessWorks™ is an easy-to-use integration product suite for enterprise applications.
- TIBCO Business Studio™ is the graphical user interface (GUI) used by TIBCO ActiveMatrix BusinessWorks™ and the plug-in to design business processes, and it is also the process engine used to execute them.
- TIBCO Enterprise Administrator™ provides a centralized administrative interface to manage and monitor the plug-in applications deployed in an enterprise.

Getting Started

This section is designed to get users started with TIBCO ActiveMatrix BusinessWorks™ Plug-in for Apache Kafka in TIBCO Business Studio.

All the operations are performed in TIBCO Business Studio. Refer to *TIBCO ActiveMatrix BusinessWorks™ Concepts* Guide to get familiar with TIBCO Business Studio.

The basic steps to create and deploy an application by using TIBCO ActiveMatrix BusinessWorks™ Plug-in for Apache Kafka include:

1. Creating a Project
2. Creating a Kafka Connection
3. Configuring a Process
4. Testing a Process
5. Deploying an Application

Creating a Project

The first task for using a plug-in is creating a project. You can add resources and processes after creating a project.
An Eclipse project is an application module configured for TIBCO ActiveMatrix BusinessWorks™. An application module is a resource unit that is named, versioned, and packaged as part of an application.

Procedure

1. Start TIBCO Business Studio™ by using one of the following ways:
   - Microsoft Windows: click **Start** > **All Programs** > **TIBCO** > ***TIBCO_HOME*** > **TIBCO Business Studio *version_number*** > **Studio for Designers**.
   - 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. In the Select a wizard dialog box, click **BusinessWorks Application Module** and click **Next** to open the New BusinessWorks Application Module wizard.
4. In the Project dialog box, 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 located in the workspace currently in use. If you do not want to use the default location for the project, clear the **Use default location** check box 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** check boxes selected to automatically create an empty process and an application when creating the project.
   e) Select the **Use Java configuration** check box, 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.

Creating a Kafka Connection

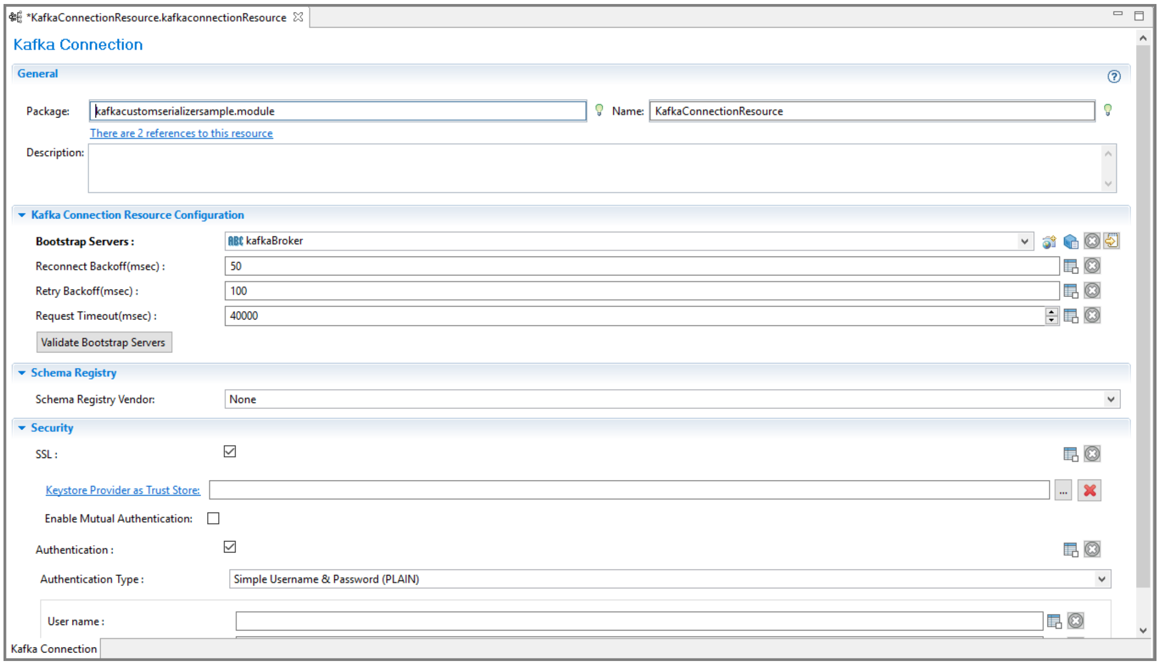
After creating a project, you must create a Kafka Connection shared resource to connect to a Kafka server instance.

Prerequisites

Kafka Connection shared resource is available at the **Resources** level. Create the project as described in Creating a Project.

Procedure

1. Expand the created project in the **Project Explorer** view.
2. Right-click the **Resources** folder and click **New** > **Kafka Connection** to open the Kafka Connection wizard. The resource folder, package name, and resource name of the Kafka connection are provided by default, which can be customized.
3. Click **Finish** to open the **Kafka Connection editor**.
4. Configure the Kafka Connection shared resource in the **Kafka Connection editor**.
5. Click **Validate Bootstrap Servers** to validate the connection.



*KafkaConnectionResource.kafkaconnectionResource

Kafka Connection

General

Package: kafkacustomserializersample.module

Name: KafkaConnectionResource

There are 2 references to this resource

Description:

Kafka Connection Resource Configuration

Bootstrap Servers : kafkaBroker

Reconnect Backoff(msec) : 50

Retry Backoff(msec) : 100

Request Timeout(msec) : 40000

Validate Bootstrap Servers

Schema Registry

Schema Registry Vendor: None

Security

SSL : ☑

Keystore Provider as Trust Store:

Enable Mutual Authentication: ☐

Authentication : ☑

Authentication Type : Simple Username & Password (PLAIN)

User name :

Kafka Connection

Configuring a Process

An empty process is created by default when creating a project. You can add activities to the empty process.

Prerequisites

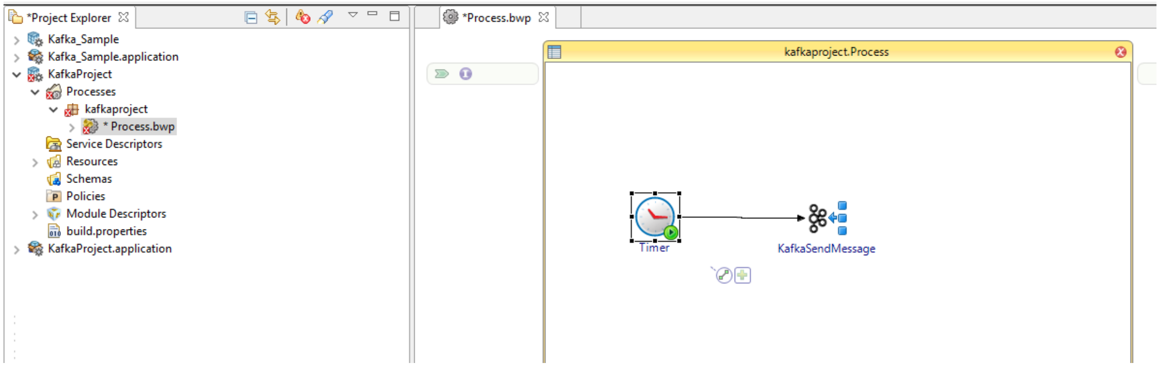
Ensure that you have created an empty process when creating a project. See Creating a Project for more information.

Procedure

1. In the **Project Explorer** view, click the created project and open the empty process from the **Processes** folder.

2. Select activities from the **Palette** view and drop them in the Process editor.

   For example, select and drop the **Timer** activity from the **General Activities** palette, and the **KafkaSendMessage** activity from the Kafka palette library.



3. Click an activity in the Process editor and drag the icon to create a transition between the added activities.
4. Configure the added Kafka activities, as described in Kafka Palette.
5. Click **File** > **Save** to save the process.

Testing a Process

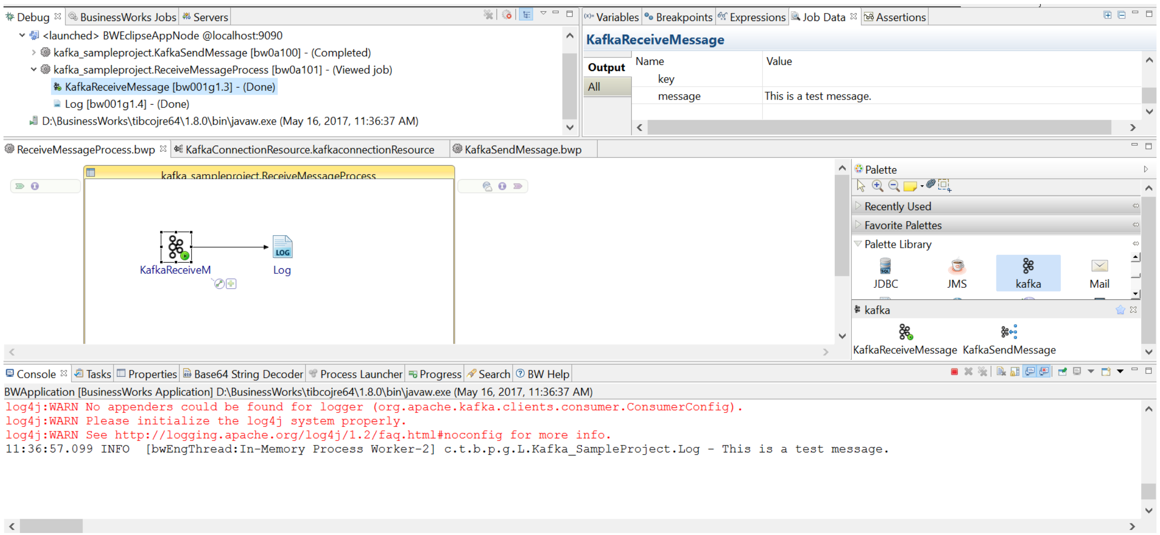
After configuring, you might debug or test a process.

Prerequisites

Ensure that you have configured a process, as described in Configuring a Process.

Procedure

1. Open the process you have configured in the TIBCO Business Studio™.
2. On the toolbar, click **Debug** > **Debug Configurations**.
3. Click **BusinessWorks Application** > **BWApplication** in the left panel.
   By default, all the 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.
4. 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.
5. On the **Debug** tab, expand the debugged process and click an activity, such as the KafkaReceiveMessage process.
6. In the upper-right panel, click **Job Data** view, and then click the **Output** tab to check the activity output.



```
<launched> BWEclipseAppNode @localhost:9090
  kafka_sampleproject.KafkaSendMessage [bw0a100] - (Completed)
  kafka_sampleproject.ReceiveMessageProcess [bw0a101] - (Viewed job)
    KafkaReceiveMessage [bw001g1.3] - (Done)
    Log [bw001g1.4] - (Done)
D:\BusinessWorks\tibcojre64\1.8.0\bin\javaw.exe (May 16, 2017, 11:36:37 AM)
```

KafkaReceiveMessage

Name	Value
key	
message	This is a test message.

```
BWApplication [BusinessWorks Application] D:\BusinessWorks\tibcojre64\1.8.0\bin\javaw.exe (May 16, 2017, 11:36:37 AM)
log4j:WARN No appenders could be found for logger (org.apache.kafka.clients.consumer.ConsumerConfig).
log4j:WARN Please initialize the log4j system properly.
log4j:WARN See http://logging.apache.org/log4j/1.2/faq.html#noconfig for more info.
11:36:57.099 INFO  [bwEngThread:In-Memory Process Worker-2] c.t.b.p.g.L.Kafka_SampleProject.Log - This is a test message.
```

Deploying an Application

After testing, 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.

Prerequisites

Before deploying an application, generate an application archive, which is an enterprise archive (EAR) file created in TIBCO Business Studio™ .

Deploying an application involves:

Procedure

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

For more details about how to deploy an application, see *TIBCO ActiveMatrix BusinessWorks™ Administration Guide*.

Apache Kafka Palette

This plug-in contains one shared connection and two activities.

- Kafka Connection Shared Resource
- Kafka Send Message activity
- Kafka Receive Message activity
- Kafka Get Messages Activity

All Apache Kafka Plug-in activities can use the same Kafka Shared Connection resource and can work independently.

Kafka Connection Shared Resource

Kafka Connection resource is used to specify the configuration details of the Kafka server hosted across nodes. Typically, a producer would publish the messages to a specific topic hosted on a server node of a Kafka cluster and consumer can subscribe to any specific topic to fetch the data.

General

On the **General** tab, you can specify the required parameters before you use this activity. The **General** tab contains the following fields:

Field	Description
Package	Package path to be added
Name	The name to be displayed as the label for the resource
Description	A short description of the shared resource

Kafka Connection Resource Configuration1

The Kafka Connection Configuration section has the following fields: anbbd c

Field	Literal Value/Module Property?	Description
Bootstrap Servers	Yes	A list of host/port pairs to use for establishing the initial connection to the Kafka cluster.
Reconnect Backoff(msec)	Yes	This is the amount of time to wait before attempting to reconnect to a given host. This avoids repeatedly connecting to a host in a tight loop. The default value is 50.

Field	Literal Value/Module Property?	Description
Retry Backoff(msec)	Yes	The amount of time to wait before attempting to retry a failed fetch request to a given topic partition. This avoids repeated fetching-and-failing in a tight loop. The default value is 100.
Request Timeout(msec)	Yes	The maximum amount of time the client waits for the response of a request. If the response is not received before the timeout elapses the client resends the request if necessary or the request fails the retries are exhausted. The default value is 30000.

Schema Registry

The Schema Registry section has the following fields:

Condition Applicable	Field	Literal Value/ Module Property ?	Description
N/A	**Schema Registry Vendor**	No	The vendor that provides schema management through schema registry. There are two types of Schema Registry Vendors: • TIBCO Schema Registry • Confluent Schema Registry To use TIBCO Schema Registry, for an existing application module, right-click the project name in the Project Explorer view and select **Configure** > **Convert to Java project**. Add `tibftl-kafka-avro-<version>`.jar into your application module `lib` folder from `<TIBCO_HOME>\akd\repo\<version>\lib`.

Condition Applicable	Field	Literal Value/Module Property?	Description
Available only when **Schema Registry Vendor** is selected.	**Schema Registry URL**	Yes	The URL for the Schema Registry. For the Confluent Schema Registry, the list of registry server URLs are separated by a comma (,) and for the TIBCO Schema Registry they are separated by a pipe (|) For TIBCO Schema Registry use the FTL realm URL.
Available only when **Schema Registry Vendor** is selected.	**Username**	Yes	The username to access the Schema Registry
Available only when **Schema Registry Vendor** is selected.	**Password**	Yes	The password to access the Schema Registry
Available for both **Confluent Schema Registry** and **TIBCO Schema Registry**.	**SSL**	No	Select this check box to use the SSL authentication to verify the user and the schema registry server. This flag enables or disables transport layer security for the connection. Only trust store configuration is used while connecting with the TIBCO Schema Registry. You can import FTL trust certificate (PEM file) in the configured trust store.
Available only when **SSL** check box is selected.	**SSL Client Configuration**	No	This field is available when the SSL checkbox is selected. Select the SSL Client Shared Resource to setup the connection from the plug-in to the schema registry.
Available only when **Schema Registry Vendor** is selected.	**Test Connection**	No	Click this button to validate the connection to the schema registry.

Security

Add required SSL properties in the `server.properties` file to enable SSL.

The Security section has the following fields:

Field	Literal Value/Module Property	Description
SSL	Yes	Select this check box to use the SSL authentication to verify the user and the server.
Keystore Provider as Trust Store	None	This field is used to create KeystoreProviderResource and then provide trust store URL and password. Available only when **SSL** check box is selected.
Enable Mutual Authentication	None	Select this check box to provide two-way SSL authentication. Available only when **SSL** check box is selected.
Identity Store Provider	None	This field is used to create KeystoreProviderResource and then provide key store URL and password. Available only when **Enable Mutual Authentication** check box is selected.
Key Password	Yes	Specify the key password. Available only when **Enable Mutual Authentication** check box is selected.

Authentication

In the **Authentication** section, you can select and configure an authentication type based on the level of security and encryption needed. The Kerberos security authentication can be combined with SSL to provide SASL_SSL.

Before enabling Kerberos authentication, ensure that `krb5.conf` or`krb5.ini` file (depending on your operating system) is placed in a default location. If you want to place the `krb5.conf` or`krb5.ini` file in a custom location, configure `java.security.krb5.conf` system property. For more information, see "SECURITY" section in Apache Kafka Documentation.

Kafka plug-in accepts "kafka" as a service principal name to connect with kerberized server. If the user creates a service name other than "kafka", the following error message is displayed:

```
Authentication failure
```

You can overwrite the Kerberos service name by adding the following property in Kafka producer/ consumer "Advanced Properties" section:

```
sasl.kerberos.service.name=<service-name>.
```

The Authentication section has the following fields:

Condition Applicable	Field	Literal Value/ Module Property	Description
N/A	**Authentication**	Yes	Select this check box to use the authentication to authenticate a user to connect to a server.

Condition Applicable	Field	Literal Value/ Module Property	Description
Available only when **Authentication** check box is selected.	**Authentication Type**	Yes	There are following four types of authentication you can use: • Simple Username & Password (PLAIN) It is a good practice to use SSL with Simple Username & Password (PLAIN) type of authentication. • Modern Username & Password with Challenge (SCRAM-SHA-256) • Modern Username & Password with Challenge (SCRAM-SHA-512) • Kerberos (GSSAPI) • OAuthBearer

Available only when **OAuth Bearer** is selected in the **Authentication Type** field.	**JAAS Config File**	Yes	Select the JAAS configuration file using the Resource Picker.
			JAAS configuration example,
			`KafkaClient {`
			`org.apache.kafka.common.sec`
			`urit`
			`y.oauthbearer.OauthBearerLo`
			`ginModulerequired`
			`unsecuredLoginStringClaim_s`
			`ub="`
			`alice";`
			`};`

Available only when **OAuth Bearer** is selected in the **Authentication Type** field.	**Login Callback Handler**	Yes	Select a class which provides implementation of "org.apache.kafka.common.security.auth.AuthenticateCallbackHandler" interface. Please refer Kafka documentation for more details.
Not available when you select the authentication type as **Kerberos (GSSAPI)**.	**User name**	Yes	Enter the user name for authentication.
	Password	Yes	Enter the password for authentication.

Condition Applicable	Field	Literal Value/ Module Property	Description
Available only when you select authentication type as **Kerberos (GSSAPI)**.	**Use Ticket Cache**	Yes	Select this check box to use the ticket, which is present in the Kerberos cache. Else, select **Keytab File** and **Principal Name** options. Ensure that latest version of JCE Policy is installed, to use **Use Ticket Cache** option.
	Keytab File	Yes	Select the Keytab file for authentication by using **Resource Picker**.
	Principal Name	Yes	Fill the principal name such as admin@KAFKASECURE, depending on the environment setup.

Kafka Send Message Activity

The Kafka Send Message activity is used to send or publish messages to consumers through Kafka brokers.

Kafka transactions can be applied to Kafka Send Message activity. To apply transactions:

1. In the application process, right-click and create a group named local transactions.
2. Add the Kafka Send Message activity inside the newly created local transaction group.
3. On the **General** tab of the local transaction, select **Transaction Transport** as Kafka.
4. Add one or more Kafka Send Message activities which participates in the transaction inside Local Transaction group. Configure Transaction ID and ensure Enable Idempotence is selected.

> The Kafka Send Message activities (inside the transaction group) which are configured with the same shared resource and the Transaction ID participates in a transaction identified by the provided Transaction ID. In such cases, the Kafka producer instance is initialized in the first Kafka Send Message activity within the transaction group and the same producer instance is used by subsequent Kafka Send Message activities. The transaction gets committed or aborted as execution progress towards the end of transaction group.
>
> Ensure that multiple Kafka Send Message activities participating in the same transaction should be configured with identical configuration. For example, if the first Kafka Send activity is using a string value serializer, then all subsequent activities should be configured with the string value serializer. However, partition or topic name can be provided differently for each activity.

General

On the **General** tab, you can specify the required parameters before you use this activity. The **General** tab contains the following fields:

Field	Literal Value/ Module Property?	Description
Name	None	The name to be displayed as the label for the activity in the process.
Kafka Connection	Yes	The Kafka connection resource provides the connection details to communicate with a Kafka server Instance.
Topic Name	Yes	Provide the topic name where the Kafka cluster stores streams of records.
Assign Custom Partition	None	This is a check box to select if **Partition ID** field has to be entered. You can select the check box to override Kafka default partition assignment behavior.
Partition ID	Yes	Sequence ID of the custom partition to which the message is sent. The default value is 0. This field is enabled only if **Assign Custom Partition** is checked.
Use Registry	No	This check box is available when **Avro Schema** is selected in the **Key Serializer** or **Value Serializer** field. This check box enables you to use the Avro Schema with the Schema Registry. If a registry and any subject name strategy except TopicName strategy is used, provide the strategy name through the **Additional properties**.
Avro Encoding Type	Yes	This field is available when the Avro Schema is selected in the Key Serializer or Value Serializer field and the Use Registry check box is not enabled. Three types of encoding are available in the dropdown: • JSON Encoding • Binary Encoding • Single Object Encoding For more information on the Avro Encoding types, please refer to Avro documentation.
Key Serializer	No	This is a serializer class for key that implements the serializer interface.
Key Avro Schema Name	Yes	This field is available when **Avro Schema** is selected in the **Key Serializer** field and the **Use Registry** check box is enabled. Provide the name and version of the avro schema to be used in the following format. <<subject-name>> : <<version>>

Field	Literal Value/ Module Property?	Description
Select Avro Schema	No	This button is available when **Use Registry** check box is enabled. This button selects the Avro Schema that is used for the serialization of the data. On selecting the **Select Avro Schema** a wizard is displayed from where the user can select the stored schema and the version. From the wizard, you can view and select the Subject and Version of the Avro schema and the selected avro schema and the sample json payload for that schema. The Sample JSON payload can be used to create an XML Schema file using BW Schemas tool XML Schema File from JSON Payload. This XSD file can be used to map and create the JSON string input.
Key Avro Schema File	Yes	This field is available when **Avro Schema** is selected in the **Key Serializer** field and the **Use Registry** check box is not enabled. Specifies the path to the `.avsc` file which contains the avro schema to be used. Select the file through the resource picker.
Sample Avro Data	No	This button is available when **Use Registry** check box is not enabled. On selecting the Sample Avro Data, a wizard is displayed. The wizard displays the avro schema and sample JSON payload for that schema. The sample JSON payload can be used to create XML schema file using BW schema tool, **XML Schema File from JSON Payload**. This XSD file is used to map and create the JSON string input.
Value Serializer	No	Value for the serializer interface.
Value Avro Schema Name	Yes	This field is available when **Avro Schema** is selected in the **Value Serializer** field and the **Use Registry** check box is enabled. Provide the name and version of the avro schema to be used in the following format. <<subject-name>> : <<version>>
Select Avro Schema	No	This button is available when **Use Registry** check box is enabled. This button selects the Avro Schema that is used for the serialization of the data. On selecting the **Select Avro Schema** a wizard is displayed from where the user can select the stored schema and the version. From the wizard, you can view and select the Subject and Version of the Avro schema and the selected avro schema and the sample json payload for that schema. The Sample JSON payload can be used to create an XML Schema file using BW Schemas tool XML Schema File from JSON Payload. This XSD file can be used to map and create the JSON string input.

Field	Literal Value/ Module Property?	Description
Value Avro Schema File	Yes	This field is available when the Avro Schema is selected in the **Value Serializer** field and the **Use Registry** check box is not enabled. Specifies the path to the `.avsc` file which contains the avro schema to be used. Select the file through the resource picker.
Sample Avro Data	No	This button is available when **Use Registry** check box is not enabled. On selecting the Sample Avro Data, a wizard is displayed. The wizard displays the avro schema and sample JSON payload for that schema. The sample JSON payload can be used to create XML schema file using BW schema tool, **XML Schema File from JSON Payload**. This XSD file is used to map and create the JSON string input.
Acks	Yes	When writing to Kafka, producers can choose whether they wait for the message to be acknowledged by 0, 1 or all replicas. This controls the durability of records that are sent. This configuration controls the criteria under which producer requests are considered complete. • acks=0. The producer does not wait for any acknowledgment from the Kafka server. The sent record is added to buffer and considered sent. In this scenario, no guarantee can be made that server has received the record, and retries do not take any effect. • acks=1. The record is added to the log. In this case if the leader has written the record to its local log, it responds without waiting for follower's acknowledgment. • acks=all. An acknowledgment of the record is sent if the data is committed by all the in-sync replicas. This is the strongest available guarantee.
Buffer Memory	Yes	This is total amount of memory available to the producer to buffer records waiting to be sent to the server. The default value is 33554432.
Compression Type	Yes	If enabled, data is compressed by the producer, written in compressed format on the server, and decompressed by the consumer. The default is none.
Retries	Yes	Specifies the number of retries that are made by a client to resend any record in event of a transient error.

Description

On the **Description** tab, you can enter a short description for the Send Message activity.

Field	Literal Value/Process Property/Module Property?	Description
Description	None	A description of the activity.

Advanced

The following table describes the fields on the **Advanced** tab of the Send Message activity.

Field	Literal Value/ Module Property?	Description
Batch Size	None	Records are batched together by producer whenever multiple records are sent to the same partition. No attempt is made to batch the records larger than this size. The default value is 16384.
Client ID	Yes	A String client ID is passed to the server while making the requests to track the source of the requests.
Linger	Yes	You can set linger.ms to something greater than 0 to instruct the producer to wait up to that number of milliseconds before sending a request in hope that more records arrive to fill up the same batch. The default behavior is to send messages immediately even if there is additional space in the buffer.
Max Request Size	Yes	Limits the number of record batches the producer sends in a single request to avoid sending huge requests. The default value is 1048576.
Override Transaction Behavior	No	When this check box is selected, the Send Message activity does not participate in the transaction even if it is a part of the transaction group. The default value is a boolean value.
Transactional ID	Yes	This field is available only when the **Override Transaction Behavior** check box is unselected. It is a unique String ID which defines the Kafka transaction. For more information on transactions, please refer to the Kafka documentation.
Enable Idempotence	No	This field enables exactly once semantics for the Kafka producer. In the producer activity when the value is set to **true**, exactly one copy of each of the message is written in the stream. If the value is set to **false**, the producer retries to connect because of broker failures and so on, that may write duplicate copies of the retried message in the stream. The default value is false.

Field	Literal Value/ Module Property?	Description
Properties	Yes	You can set the properties name and value for the producer. - All additional producer properties can be configured here. For example, to set buffer memory value, set property name as buffer.memory and its value. - Module properties of only type string is supported.
Interceptors	No	The Interceptor class intercepts the sent message to modify or read it before publishing it to a topic or channel. You can add the interceptor classes using ➕ icon and delete the classes using ✖ icon. The interceptor classes are executed in the order in which you specify them. The order of execution can be managed using ⇧ and ⇩ icons. To add user-defined properties to the interceptor class, you must specify those property in the **Properties** field.

Input

The following table describes the fields on the **Input** tab of the Send Message activity.

Field	Data Type	Description
ProducerConfig (All fields in this section are optional.)		
Topic	String	Topic name where Kafka cluster stores streams of records.
Partition ID	Number	Sequence ID of the partition to which the message is sent. Sequence ID of the partition can be entered here or in the **Partition ID** field under **General** tab. The **Assign Custom Partition** check box under **General** tab must be selected to enter a value here.
Batch Size	Number	Batch size detail to batch the records sent to same partition.
Client ID	String	Client ID passed to the server while making the request.
Linger.ms	Number	Time in millisecond to add artificial delay while sending the message from producer.
Max Request Size	Number	Maximum size of a request in bytes. Use it to limit the number of record batches the producer sends in a single request to avoid sending huge requests.

Field	Data Type	Description
Additional properties		
• Key • Value	• String • String	• Provide key for additional properties. • Provide value for specified key in additional properties.
schemaRegistryConfig		
KeySubjectNameVersion	String	Provide the name and version of the avro schema to be used in the following format: <<subject-name>> : <<version>> This field is available when **Avro Schema** is selected and the **Use Registry** check box is enabled.
subjectNameVersion	String	Provide the name and version of the avro schema to be used in the following format: <<subject-name>> : <<version>> This field is available when **Avro Schema** is selected and the **Use Registry** check box is enabled.
Messages		
• Key • Message • Headers – Key – Value	String	• Optional. Key for the message to be sent. • Required. The actual message as value for the specified key. • Optional. User defined custom headers to be added in the outgoing message. – Provide key for the headers to be sent. – Provide value for specified key in headers. The value is provided in base64binary

Use `-Dcom.tibco.plugin.kafka.new.producer` JVM property to create a new producer for every job:

- If the property is not set or set to false, only one KafkaProducer is created in init() method when the application starts and the producer gets closed when the application stops.
- In the above case, topic from the Input tab is overridden. All other producer properties are derived from the General and Advanced tabs and the Input tab configurations are ignored.
- If the property is set to true, then properties specified in the Input tab override the properties in the General tab and Advanced tab, and a new KafkaProducer is created for every job.

Output

The following table describes the fields on the **Output** tab of the Send Message activity.

Output Item	Data Type	Description
KafkaSendMessageOutput	complex	The complete output of the Send Message activity.
result	complex	Information about the content of the sent and failed messages.
status	string	Status of the message sent by a producer.
SendSuccess	complex	Information about the content of the sent message.
topic	string	The topic name for publishing the message.
offset	number	The sequence ID number assigned to each record within a partition.
Partition	number	The sequence ID of the partition to which a record is sent within a topic.
SendFailed	complex	Information about the content of the failed message.
errorCode	string	Displays the error code.
errorMessage	string	Displays the error message.

Fault

The **Fault** tab has the following exceptions:

- KafkaCreatedProducerException
- KafkaPluginException

Each exception has the following fields:

Field	Type	Description
msg	string	The error message description returned by the plug-in.
msgCode	string	The error code returned by the plug-in.

Kafka Receive Message Activity

Kafka Receive Message activity is an event source activity which can be configured as a process starter in any TIBCO BusinessWorks process. It starts the process execution on receiving a Kafka message event.

General

On the **General** tab, specify the required parameters before using this activity. The **General** tab has the following fields:

Field	Literal Value/ Process Property/ Module Property?	Description
Name	None	The name to be displayed as the label for the activity in the process.
Kafka Connection	Yes	The Kafka connection resource for communicating with a Kafka server instance.
Group ID	Yes	The group ID for the consumer group.
Topic Names	Yes	The topic name where Kafka cluster stores streams of record. Multiple topic names are supported and these can be separated by a semi colon (;)
Assign Custom Partition	None	This is a check box to select if **Partition ID** needs to be entered. You can select the check box to override Kafka's default partition assignment behavior.
Partition IDs	Yes	Sequence ID or range of the partition to which the message is sent. Default is 0. Multiple partition ID's are supported and these can be separated using a comma (,) For example, **Single Topic and Single Partition:** Partition IDs = 0 **Single Topic and Multiple Partitions:** Partition IDs = 0,1,2 or for specifying range Partition IDs = 0-2 **Multiple Topics and Multiple Partitions:** For two topics - Partition IDs = 0,1,2;0,4 or Partition IDs = 0-2;0,1,2 or Partition IDs = 1-2;0-3 • This field is enabled only if **Assign Custom Partition** field is checked. • For Projects created on Kafka Plug-in version 6.1.0 or earlier, if the **PartitionID** field in Kafka Receive Message Activity is configured by using module or process property, the data type of the property must be changed from Integer to String.

Field	Literal Value/ Process Property/ Module Property?	Description
Consumer Count	Yes	The Consumer Count specifies the number of KafkaConsumer instances started by the activity. The default value is 0 which will create and start KafkaConsumer instances equal to the number of partitions in the topic. The maximum value allowed for Consumer Count is equal to the number of partitions in the topic.
Use Registry	No	This check box is available when **Avro Schema** is selected in the **Key Deserializer** or **Value Deserializer** field. This check box enables you to use the Avro Schema with the Schema Registry.
Avro Encoding Type	Yes	This field is available when the Avro Schema is selected in the **Key Deserializer** or **Value Deserializer** field and the **Use Registry** check box is not enabled. Three types of encoding are available in the dropdown: • JSON Encoding • Binary Encoding • Single Object Encoding For more information on the Avro Encoding types, please refer to Avro documentation.
Key Deserializer	No	Class for the key that implements the serializer interface.
Key Avro Schema File	Yes	This field is available when **Avro Schema** is selected in the **Key Deserializer** field and the **Use Registry** check box is not enabled. Specifies the path to the `.avsc` file which contains the avro schema to be used. Select the file through the resource picker.
Value Deserializer	No	Value for the serializer interface.
Value Avro Schema File	Yes	This field is available when **Avro Schema** is selected in the **Value Deserializer** field and the **Use Registry** check box is not enabled. Specifies the path to the `.avsc` file which contains the avro schema to be used. Select the file through the resource picker provided.
Fetch Timeout	Yes	Specifies the maximum time in milliseconds to get the metadata about the topic before a timeout occurs. The default value is 1000.

Field	Literal Value/ Process Property/ Module Property?	Description
Fetch Min Bytes	Yes	The minimum amount of data that the server would send on receiving a fetch request. The default setting of 1 byte means that fetch requests are answered as soon as a single byte of data is available or the fetch request times out waiting for data to arrive.
Fetch Max Wait	Yes	The maximum amount of time that the server would block before answering a fetch request if there is not sufficient data to immediately satisfy the requirement given by `fetch.min.bytes`. The default value is 500.
Heartbeat Interval	Yes	Time in milliseconds between heartbeats to the consumer. Heartbeats are used to ensure that the consumer's session stays active and to facilitate rebalancing and information when consumers join or leave a group. The default value is 3000.
Session Timeout	Yes	The consumer sends periodic heartbeats to server indicating about its liveness to the broker. If no heartbeats are received by a broker before the expiration of this session timeout, the broker removes this consumer from the group and initiates a rebalance. The default value is 30000.

Description

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

Field	Literal Value/ Process Property/ Module Property?	Description
Description	None	A description of the activity.

Advanced

The following table describes the **Advanced** tab of the Receive Message activity.

Field	Literal Value/ Process Property/ Module Property?	Description
Sequence Key	None	XPath expression that specifies which processes must run in sequence. Process instances with sequencing keys that evaluate to the same value are executed sequentially in the sequence the process instance was created.

Field	Literal Value/ Process Property/ Module Property?	Description
Custom Job Id	None	This field can contain an XPath expression that specifies a custom ID for the process instance.
Enable Auto Commit	None	• Select this check box to auto commit the message record. • Deselect the check box to enable Manual Commit for the message record. Use the Confirm activity from the **General Activities** palette for manual commit.
AutoCommit Interval	Yes	Interval in milliseconds at which the consumer offsets are auto-committed to Kafka when the auto-commit mode is enabled. The default value is 5000.
AutoOffset Reset	Yes	This is required and selected when there is no initial offset or the offset is out of range. • Earliest: resets the offset to the earliest offset. • Latest: resets the offset to the latest offset. • None: throws exception to the consumer. The default value is latest.
Isolation Level	No	This field lets you control how to consume messages written transactionally. The Isolation Levels are of two types: • read_uncommitted • read_committted
Properties	Yes	The properties name and value. • All additional consumer properties can be configured here. For example, to set max poll records value, set property name as max.poll.records and its value. • Module properties of only type string is supported.

Field	Literal Value/ Process Property/ Module Property?	Description
Interceptors	None	The Interceptor class intercepts the received messages that are published to a topic or channel before being consumed. You can add the interceptor classes using ✚ icon and delete the classes using ✖ icon. The interceptor classes are executed in the order in which you specify them. The order of execution can be managed using ⇧ and ⇩ icons. To add user-defined properties to the interceptor class, you must specify those property in the **Properties** field.

Conversation

You can initiate the conversation here. Click the **Add New Conversation** button to initiate multiple conversations.

Output

The following table describes the fields on the **Output** tab.

Field	Type	Description
KafkaReceiverMessageOutput	complex	The complete output for the Receive Message activity.
topic	string	The topic name.
Partition	number	The sequence ID of the partition.
offset	number	The sequence ID assigned to each record within the partition.
Key	String	Specified key of the incoming record.
Message	String	Message received through Kafka.
Headers • Key • Value	String	Headers received through Kafka • Key received through Kafka • Value received through Kafka

Kafka Get Messages Activity

The Kafka Get Messages activity consumes messages from Kafka Topics. It returns available messages based on specified poll iterations and max poll records configuration. The activity also allows you to seek partition offset to beginning, end or custom offset value.

General

On the **General** tab, specify the required parameters before using this activity. The **General** tab has the following fields:

Field	Literal Value/ Process Property/ Module Property?	Description
Name	None	The name to be displayed as the label for the activity in the process.
Kafka Connection	Yes	The Kafka connection resource for communicating with a Kafka server instance.
Group ID	Yes	The group ID for the consumer group.
Topic Names	Yes	The topic name where Kafka cluster stores streams of record. Multiple topic names are supported and these topic names can be separated using a semi colon (;). For example, topicName1;topicName2
Assign Custom Partition	None	This is a check box to select if **Partition ID** needs to be entered. You can select the check box to override Kafka's default partition assignment behavior.

Field	Literal Value/ Process Property/ Module Property?	Description
Partition IDs	Yes	Sequence ID or range of the partition to which the message is received. Default is 0. Multiple partition ID's are supported using a comma separator (,). For example, **Single Topic and Single Partition** Partition IDs = 0 **Single Topic and Multiple Partitions** • Partition IDs = 0,1,2 or for specifying range • Partition IDs = 0-2 **Multiple Topics and Multiple Partitions** For two topics: • Partition IDs = 0,1,2;0,4 or • Partition IDs = 0-2;0,1,2 or • Partition IDs = 1-2;0-3 This field is enabled only if **Assign Custom Partition** field is checked.
Use Registry	No	This check box is available when **Avro Schema** is selected in the **Key Deserializer** or **Value Deserializer** field. This check box enables you to use the Avro Schema with the Schema Registry.
Avro Encoding Type	Yes	This field is available when the Avro Schema is selected in the **Key Deserializer** or **Value Deserializer** field and the **Use Registry** check box is not enabled. Three types of encoding are available in the dropdown: • JSON Encoding • Binary Encoding • Single Object Encoding For more information on the Avro Encoding types, please refer to Avro documentation.
Key Deserializer	No	Class for the key that implements the serializer interface.

Field	Literal Value/ Process Property/ Module Property?	Description
Key Avro Schema File	Yes	This field is available when **Avro Schema** is selected in the **Key Deserializer** field and the **Use Registry** check box is not enabled. Specifies the path to the `.avsc` file which contains the avro schema to be used. Select the file through the resource picker.
Value Deserializer	No	Value for the serializer interface.
Value Avro Schema File	Yes	This field is available when **Avro Schema** is selected in the **Value Deserializer** field and the **Use Registry** check box is not enabled. Specifies the path to the `.avsc` file which contains the avro schema to be used. Select the file through the resource picker provided.
Poll Iterations	Yes	The number of times Kafka Get Message activity executes poll() method. The default value is 3.
Fetch Timeout (msec)	Yes	The time in milliseconds spent waiting in poll if data is not available in the buffer. Used as an input to the poll() method. The default value is 10000.
Max Poll Records	Yes	Maximum number of poll records returned in a single poll() operation. The default value is 500.

Description

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

Field	Literal Value/ Process Property/ Module Property?	Description
Description	None	A description of the activity.

Advanced

The following table describes the **Advanced** tab of the Get Messages activity.

Field	Literal Value/ Process Property/ Module Property?	Description
Manual Commit	None	When selected, the offsets of the consumed messages are committed during confirm activity execution using commitSync(). When unselected, the offset of the consumed messages are committed using commitSync() post execution of all poll iterations. Since number of poll iterations and **`auto.commit.interval.ms`** are major factors for auto commit to work, **`enable.auto.commit`** is set to false by default.
AutoOffset Reset	Yes	This is required and selected when there is no initial offset or the offset is out of range. • Earliest: resets the offset to the earliest offset. • Latest: resets the offset to the latest offset. • None: throws exception to the consumer. The default value is earliest.
Isolation Level	No	This field lets you control how to consume messages written transactionally. The Isolation Levels are of two types: • read_uncommitted • read_committted
Properties	Yes	The properties name and value. • All additional consumer properties can be configured here. For example, to set max poll records value, set property name as max.poll.records and its value. • Module properties of only type string is supported.

Field	Literal Value/ Process Property/ Module Property?	Description
Interceptors	None	The Interceptor class intercepts the received messages that are published to a topic or channel before being consumed. You can add the interceptor classes using ✚ icon and delete the classes using ✖ icon. The interceptor classes are executed in the order in which you specify them. The order of execution can be managed using ⇧ and ⇩ icons. To add user-defined properties to the interceptor class, you must specify those property in the **Properties** field.

Input

The following table describes the fields on the **Input** tab of the Get Messages activity.

Field	Data Type	Description
ConsumerConfig		
TopicNames	String	The topic name where Kafka cluster stores streams of record. Multiple topic names are supported and these topic names can be separated using a semi colon (;). For example, topicName1;topicName2

Field	Data Type	Description
Partition IDs	Number	Sequence ID or range of the partition to which the message is received. Default is 0. Multiple partition ID's are supported using a comma separator (,). For example, **Single Topic and Single Partition** Partition IDs = 0 **Single Topic and Multiple Partitions** • Partition IDs = 0,1,2 or for specifying range • Partition IDs = 0-2 **Multiple Topics and Multiple Partitions** For two topics: • Partition IDs = 0,1,2;0,4 or • Partition IDs = 0-2;0,1,2 or • Partition IDs = 1-2;0-3 This field is enabled only if **Assign Custom Partition** field is checked.
SeekOffset		
Topic	String	Provide name of a topic. Topic can be empty when SeekTo is set to Beginning or End and seek operation is applied on all assigned (including multiple topics) partitions
Partition ID	String	Provide a partition ID. Partition ID is mandatory when **SeekTo** is set to **Offset**. Partition ID can be empty when **SeekTo** is set to **Beginning** or **End** and seek operation is applied on all **assigned** partitions, otherwise seek is applied to specified partition.
seekTo	String	Provide seek operation to perform. Valid values are - Beginning, Offset, and End.
offset	Number	Provide an offset number. Used when **SeekTo** is set to **Offset**.
Additional properties		
• Key • Value	• String • String	• Provide key for additional properties. • Provide value for specified key in additional properties.

Output

The following table describes the fields on the **Output** tab.

Field	Type	Description
KafkaGetMessageOutput	complex	The complete output for the GetMessage activity.
Records	complex	Set of Kafka Messages returned by Kafka Get Messages activity.
topic	string	The topic name.
Partition	number	The partition ID.
offset	number	The sequence ID assigned to each record within the partition.
Key	String	Specified key of the incoming record.
Message	String	Message received through Kafka.
Headers • Key • Value	String	Headers received through Kafka • Key received through Kafka • Value received through Kafka

Fault

The **Fault** tab has the following exceptions:

- KafkaPluginException

Each exception has the following fields:

Field	Type	Description
msg	string	The error message description returned by the plug-in.
msgCode	string	The error code returned by the plug-in.

Sample Project Overview

The sample projects help to understand how TIBCO ActiveMatrix BusinessWorks™ Plug-in for Apache Kafka operates.

TIBCO ActiveMatrix BusinessWorks™ Plug-in for Apache Kafka is packaged with sample projects. After installing the plug-in, the following sample projects can be found in the `TIBCO_HOME\bw\palettes\kafka\`*`version_number`*`\samples` directory.

- Kafka_GeneralSample
- Kafka_CloudSample
- Kafka_InterceptorSample
- Kafka_ErrorHandlingSample
- Kafka_CustomSerializerSample
- Kafka_TransactionSample

Working with Kafka_GeneralSample Project

Prerequisites

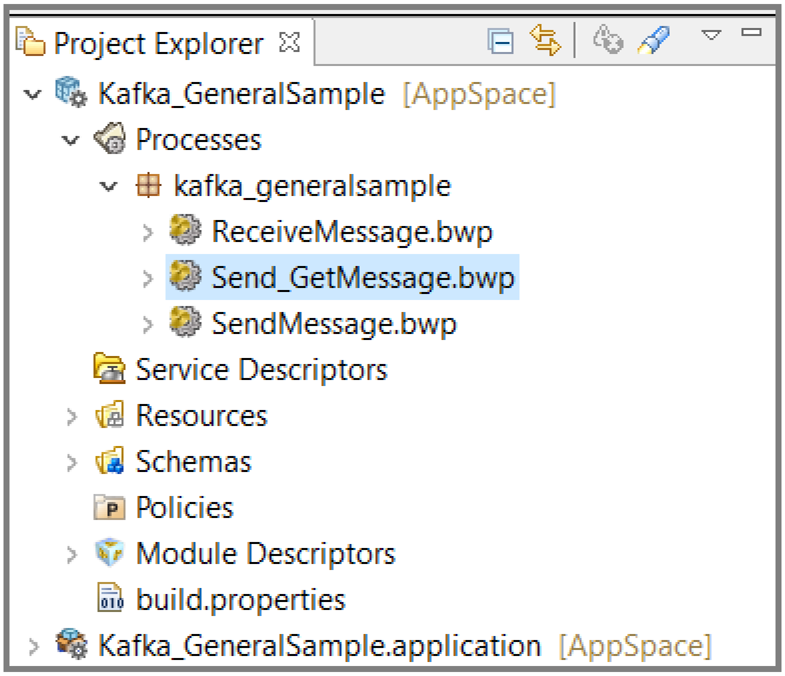
Before running the project, you must import the sample project to TIBCO Business Studio.

Procedure

1. Start TIBCO Business Studio using one of the following ways:
   - For Microsoft Windows, click **Start** > **All Programs** > **TIBCO** > **TIBCO_HOME** > **TIBCO Business Studio version_number** > **Studio for Designers**.
   - For Linux, run the TIBCO Business Studio executable file located in the `TIBCO_HOME\studio\version_number\eclipse directory`.
2. From the menu, click **File** > **Import**.
3. In the Import window, expand the **General** folder and select the **Existing Studio Projects into Workspace** item. Click **Next**.
4. Click **Browse** next to the **Select archive file** field to select the `Kafka_GeneralSample.zip` file. Click **Finish**. The `Kafka_GeneralSample.zip` file is in the `TIBCO_HOME\bw\palettes\kafka\`*`version`*`\samples` directory.

Result

The sample project is imported to TIBCO Business Studio.



Project Explorer

- Kafka_GeneralSample [AppSpace]
  - Processes
    - kafka_generalsample
      - ReceiveMessage.bwp
      - Send_GetMessage.bwp
      - SendMessage.bwp
  - Service Descriptors
  - Resources
  - Schemas
  - Policies
  - Module Descriptors
  - build.properties
- Kafka_GeneralSample.application [AppSpace]

Configuring Kafka Connection for Kafka_GeneralSample

Configuring Kafka shared connection resource is required to configure a connection with the Kafka server.

Procedure

1. In the Project Explorer view, expand **Kafka_GeneralSample**.
2. In the Resources folder, double-click **KafkaConnectionResource.kafkaconnectionResource**.
3. In the KafkaConnectionResource Editor, configure each field accordingly.
4. On the toolbar, click the **Save** icon to save your changes.

For more information, refer Creating a Kafka Connection.

Configuration of the Kafka_GeneralSample Processes

The sample project contains two processes. After Importing the Sample Project, expand the Processes resource to display the processes.

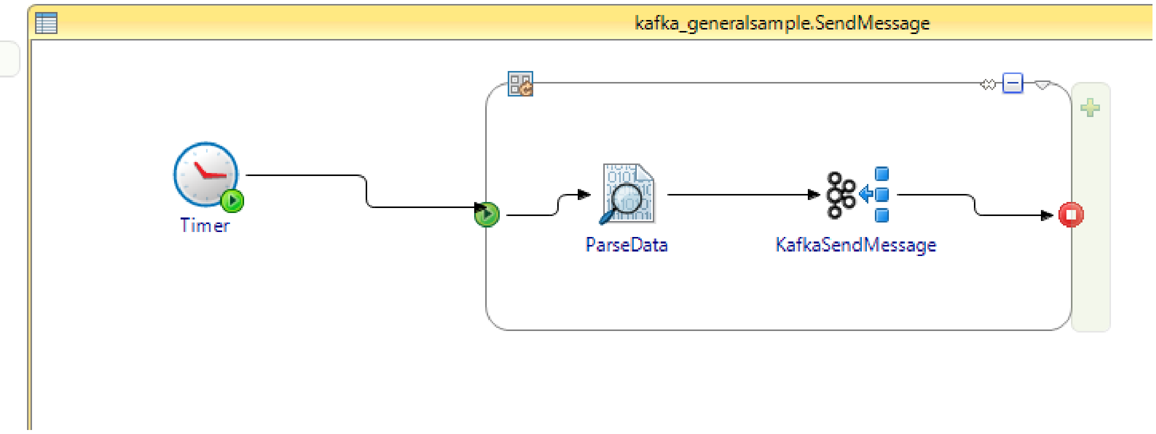
Kafka_GeneralSample

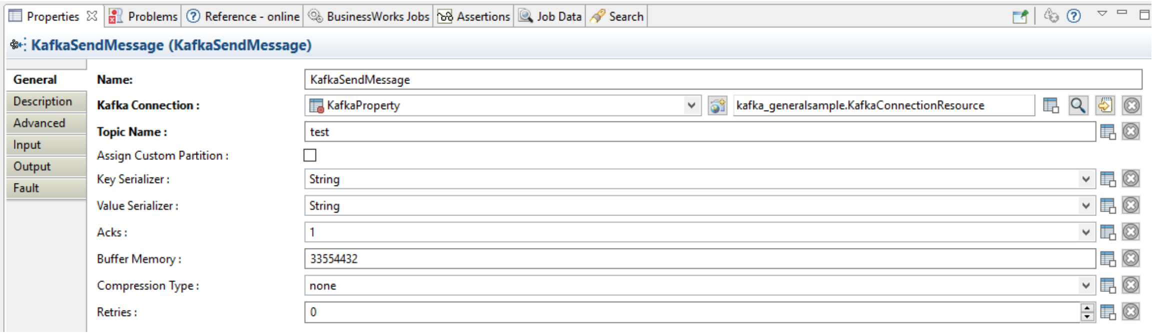
This sample project contains the following processes:

- SendMessage.bwp
- ReceiveMessage.bwp
- Send_GetMessages.bwp

SendMessage.bwp

This process demonstrates how to use the plug-ins to send messages through producer using Kafka server. ParseData activity is used to read messages one at a time from the text file so that each line is sent as a separate message.





KafkaSendMessage (KafkaSendMessage)

Name: KafkaSendMessage

Kafka Connection : KafkaProperty kafka_generalsample.KafkaConnectionResource

Topic Name : test

Assign Custom Partition : □

Key Serializer : String

Value Serializer : String

Acks : 1

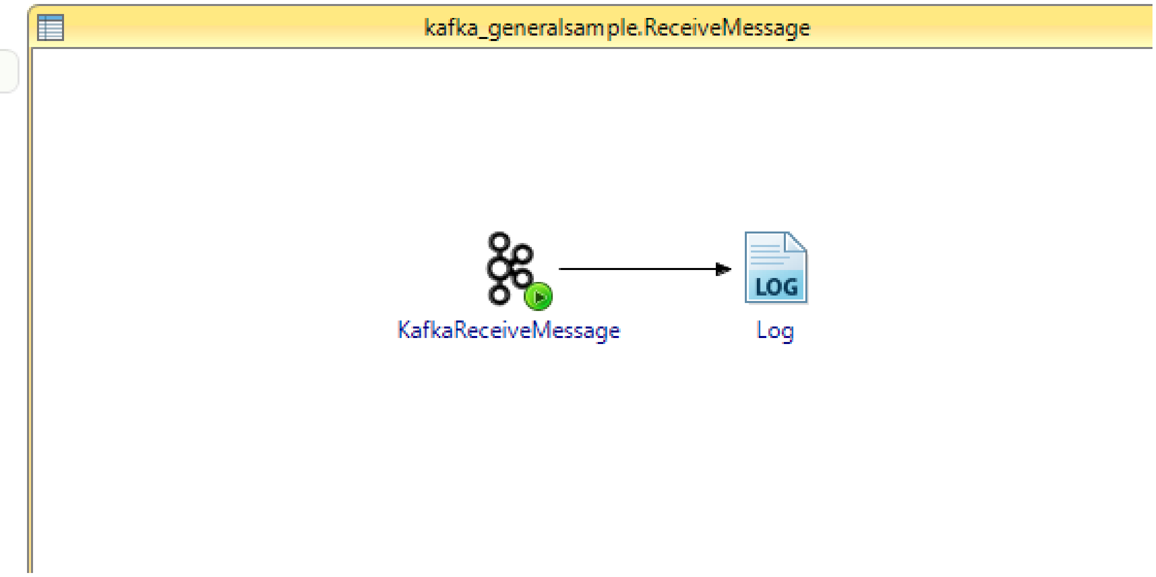
Buffer Memory : 33554432

Compression Type : none

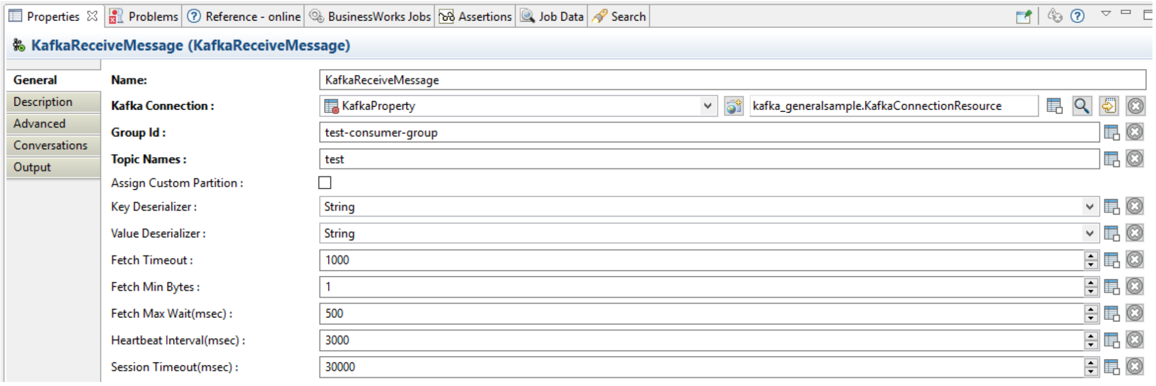
Retries : 0

ReceiveMessage.bwp

This process contains a ReceiveMessage process starter. The ReceiveMessage activity listens for an incoming message event and starts the job execution on receiving the incoming records and writes them to a log.



Activity configuration for the ReceiveMessage.bwp:

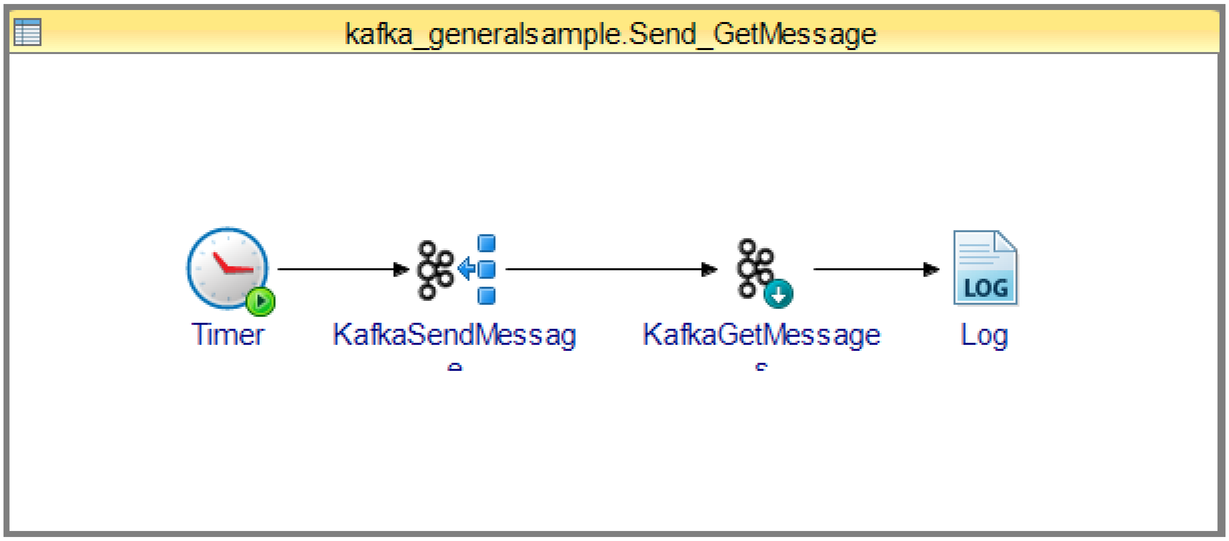


KafkaReceiveMessage (KafkaReceiveMessage)

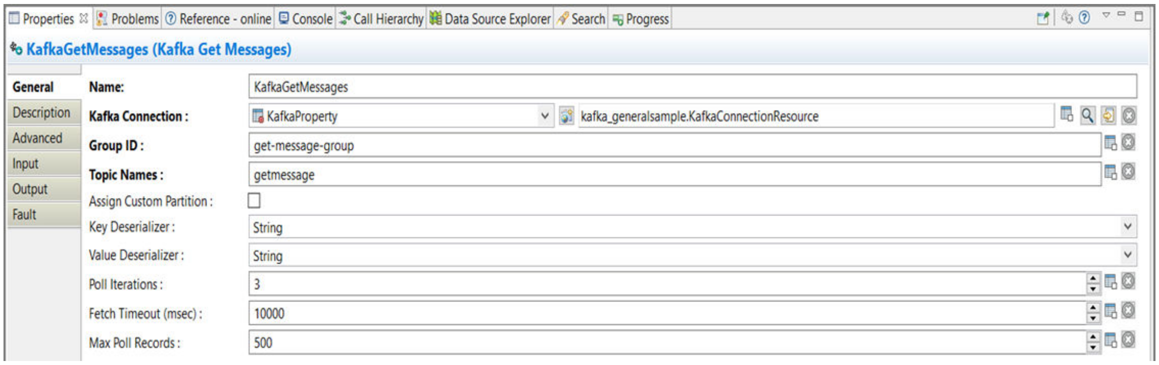
Name:	KafkaReceiveMessage
Kafka Connection :	KafkaProperty kafka_generalsample.KafkaConnectionResource
Group Id :	test-consumer-group
Topic Names :	test
Assign Custom Partition :	□
Key Deserializer :	String
Value Deserializer :	String
Fetch Timeout :	1000
Fetch Min Bytes :	1
Fetch Max Wait(msec) :	500
Heartbeat Interval(msec) :	3000
Session Timeout(msec) :	30000

Send_GetMessages.bwp

This process demonstrates how to use the Kafka Get Messages activity. It consumes published messages from a Kafka topic.



Activity configuration for the Send_GetMessages.bwp:



KafkaGetMessages (Kafka Get Messages)

Name:	KafkaGetMessages
Kafka Connection :	KafkaProperty kafka_generalsample.KafkaConnectionResource
Group ID :	get-message-group
Topic Names :	getmessage
Assign Custom Partition :	□
Key Deserializer :	String
Value Deserializer :	String
Poll Iterations :	3
Fetch Timeout (msec) :	10000
Max Poll Records :	500

Running the Kafka_GeneralSample Project

The sample project shows how to use the plug-in to send and receive messages using Apache Kafka.

Prerequisites

Import the sample project to TIBCO Business Studio, as described in Working with Kafka_GeneralSample Project, and configure Kafka shared resource connection, as described in Creating a Kafka Connection.

Procedure

1. In the Project Explorer view, expand the **Module Descriptors** resource and double-click **Components**.
2. By default, all the processes are listed in the Components editor.
3. On the toolbar, click the **Save** 💾 icon to save your changes.
4. From the menu, click **Run** > **Run Configurations** to run the selected process.
5. In the Run Configuration dialog box, expand **BusinessWorks Application**, and then click **BWApplication.**
6. In the right panel, click the **Applications** tab, and select the check box next to **Kafka_GeneralSample.application**.
7. Click **Run** to run the selected process.
8. Click the **Terminate** 🟥 icon to stop the process.

Working with Kafka_CloudSample Project

Prerequisites

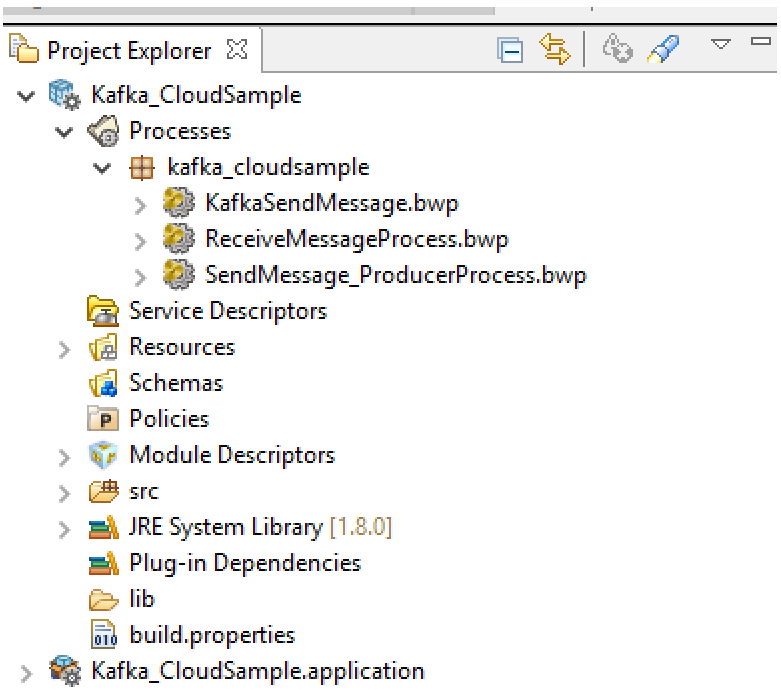
Before running the project, you must import the sample project to TIBCO Business Studio.

Procedure

1. Start TIBCO Business Studio using one of the following ways:
   - For Microsoft Windows, click **Start** > **All Programs** > **TIBCO** > **TIBCO_HOME** > **TIBCO Business Studio version_number** > **Studio for Designers**.
   - For Linux, run the TIBCO Business Studio executable file located in the `TIBCO_HOME\studio\version_number\eclipse directory`.
2. From the menu, click **File** > **Import**.
3. In the Import window, expand the **General** folder and select the **Existing Studio Projects into Workspace** item. Click **Next**.
4. Click **Browse** next to the **Select archive file** field to select the `Kafka_CloudSample.zip` file. Click **Finish**. The `Kafka_CloudSample.zip` file is in the `TIBCO_HOME\bw\palettes\kafka\`*`version`*`\samples` directory.

Result

The sample project is imported to TIBCO Business Studio.



- Kafka_CloudSample
  - Processes
    - kafka_cloudsample
      - KafkaSendMessage.bwp
      - ReceiveMessageProcess.bwp
      - SendMessage_ProducerProcess.bwp
  - Service Descriptors
  - Resources
  - Schemas
  - Policies
  - Module Descriptors
  - src
  - JRE System Library [1.8.0]
  - Plug-in Dependencies
  - lib
  - build.properties
- Kafka_CloudSample.application

Configuring Kafka Connection for Kafka_CloudSample

Configuring Kafka shared connection resource is required to configure a connection with the Kafka server.

Procedure

1. In the Project Explorer view, expand **Kafka_CloudSample**.
2. In the Resources folder, double-click **KafkaConnectionResource.kafkaconnectionResource**.
3. In the KafkaConnectionResource Editor, configure each field accordingly.
4. On the toolbar, click the **Save** 💾 icon to save your changes.

For more information, refer Creating a Kafka Connection.

Configuration of the Kafka_CloudSample Processes

The sample project contains three processes. Each process has a different function. After Importing the Sample Project, expand the Processes resource to display the processes.

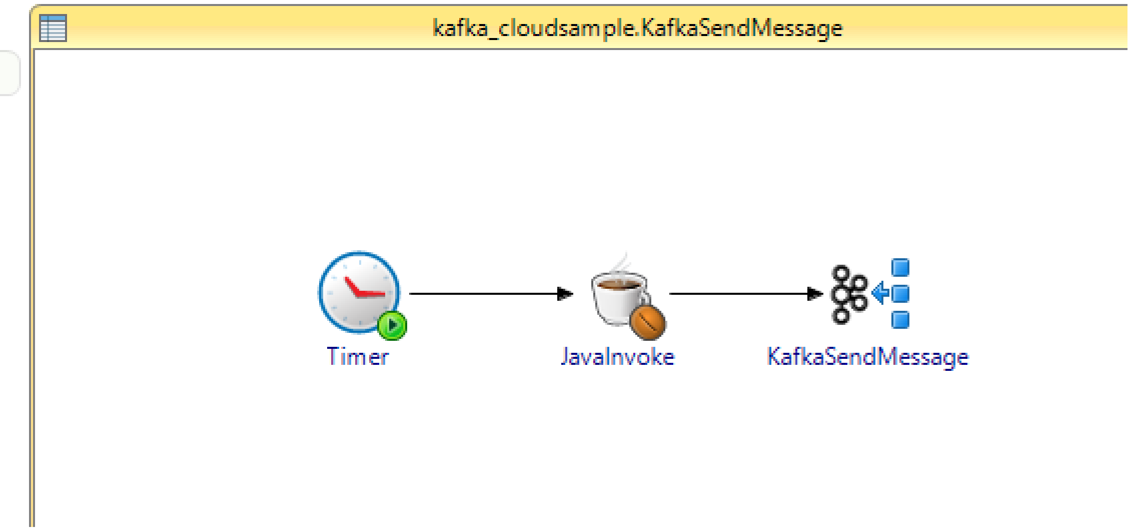
Kafka_CloudSample

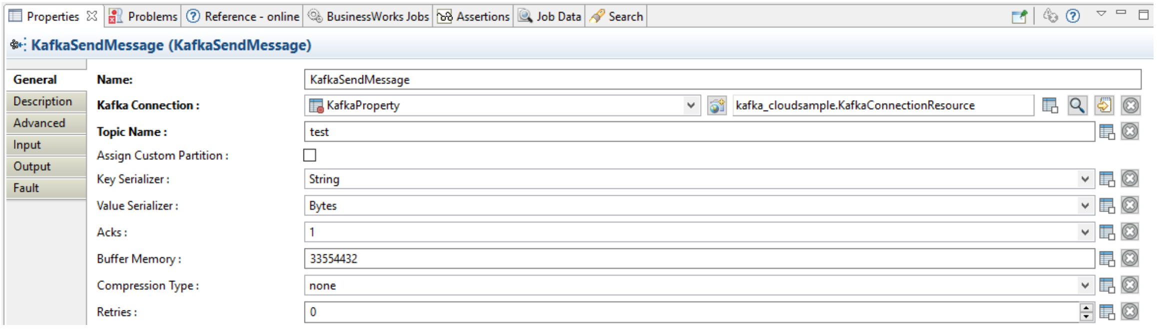
This sample project contains the following three processes:

- KafkaSendMessage.bwp
- SendMessage_ProducerProcess.bwp
- ReceiveMessageProcess.bwp

KafkaSendMessage.bwp

This process demonstrates how to use the plug-ins to send messages through producer using Kafka server. JavaInvoke activity is used to invoke Java class method. It returns the message, which is read by SendMessage activity.





KafkaSendMessage (KafkaSendMessage)

Name: KafkaSendMessage

Kafka Connection : KafkaProperty kafka_cloudsample.KafkaConnectionResource

Topic Name : test

Assign Custom Partition : □

Key Serializer : String

Value Serializer : Bytes

Acks : 1

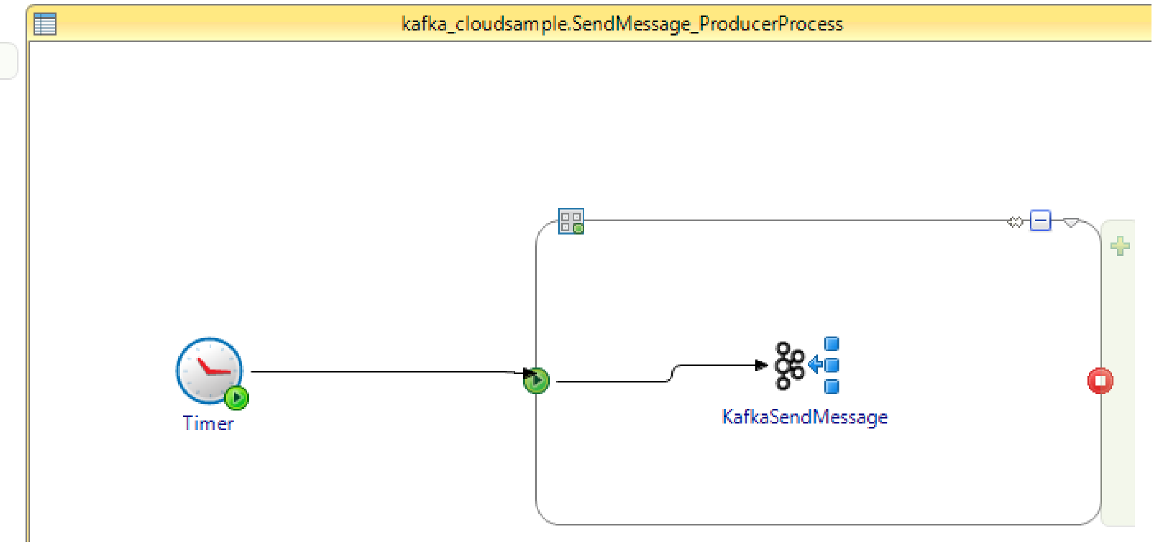
Buffer Memory : 33554432

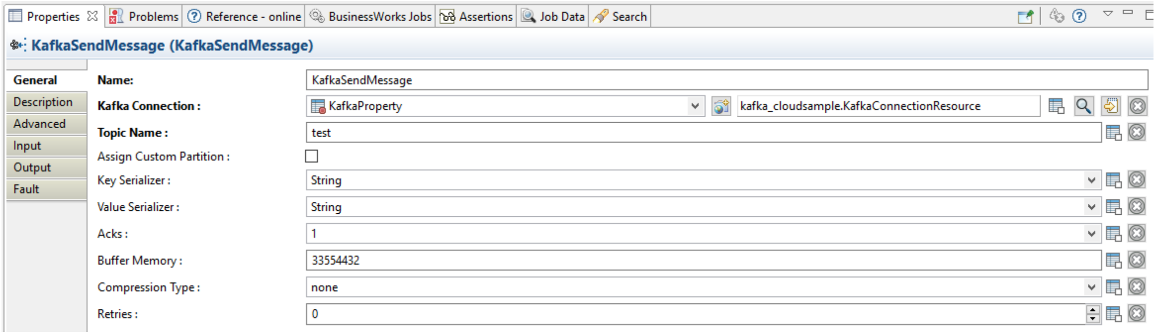
Compression Type : none

Retries : 0

SendMessage_ProducerProcess.bwp

This process demonstrates how to use the plug-ins to send messages through producer using Kafka server. This process uses "While" loop to send message multiple times.



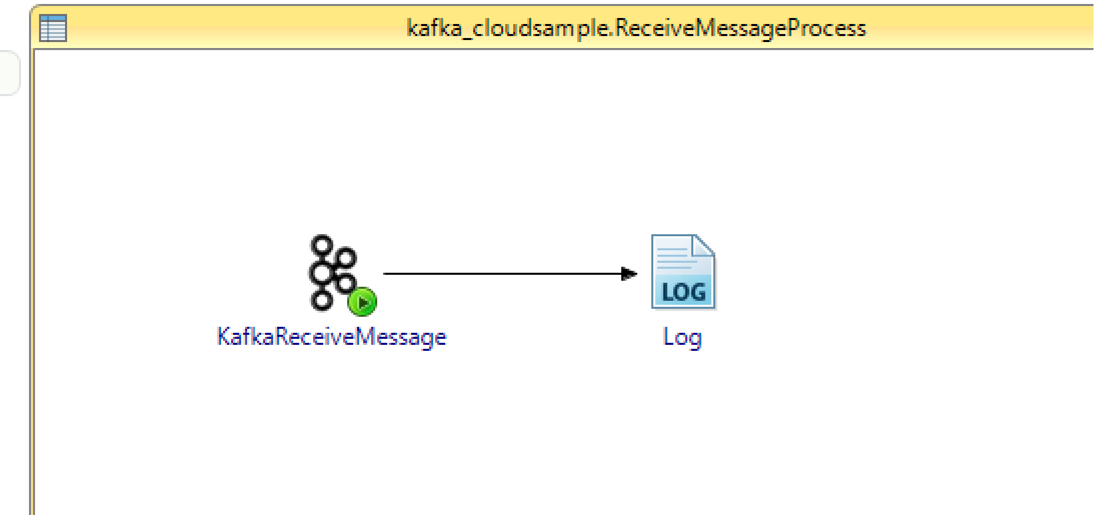


KafkaSendMessage (KafkaSendMessage)

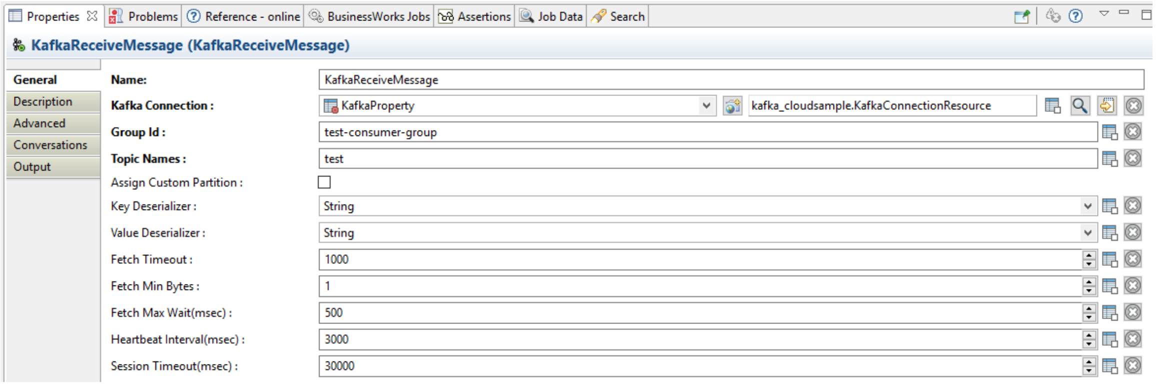
Name:	KafkaSendMessage
Kafka Connection :	KafkaProperty kafka_cloudsample.KafkaConnectionResource
Topic Name :	test
Assign Custom Partition :	☐
Key Serializer :	String
Value Serializer :	String
Acks :	1
Buffer Memory :	33554432
Compression Type :	none
Retries :	0

ReceiveMessageProcess.bwp

This process contains a ReceiveMessage process starter. ReceiveMessage activity listens for incoming message event and starts the job execution on receiving the incoming records and write them to a log.



Activity configuration for the ReceiveMessage.bwp:



KafkaReceiveMessage (KafkaReceiveMessage)

Name:	KafkaReceiveMessage
Kafka Connection :	KafkaProperty kafka_cloudsample.KafkaConnectionResource
Group Id :	test-consumer-group
Topic Names :	test
Assign Custom Partition :	☐
Key Deserializer :	String
Value Deserializer :	String
Fetch Timeout :	1000
Fetch Min Bytes :	1
Fetch Max Wait(msec) :	500
Heartbeat Interval(msec) :	3000
Session Timeout(msec) :	30000

Running the Kafka_CloudSample Project

The sample project shows how to use the plug-in to send and receive messages using Apache Kafka.

Prerequisites

Import the sample project to TIBCO Business Studio, as described in Working with Kafka_CloudSample Project, and configure Kafka shared resource connection, as described in Creating a Kafka Connection.

Procedure

1. In the Project Explorer view, expand the **Module Descriptors** resource and double-click **Components**.
2. By default, all the processes are listed in the Components editor.
3. On the toolbar, click the **Save** 💾 icon to save your changes.
4. From the menu, click **Run** > **Run Configurations** to run the selected process.
5. In the Run Configuration dialog box, expand **BusinessWorks Application**, and then click **BWApplication.**
6. In the right panel, click the **Applications** tab, and select the check box next to **Kafka_CloudSample.application**.
7. Click **Run** to run the selected process.
8. Click the Terminate 🟥 icon to stop the process.

Working with Kafka_InterceptorSample Project

This sample demonstrates how to configure and use the ConsumerInterceptor and ProducerInterceptor in the kafka plugin to intercept (and possibly mutate) records received by the consumer and producer respectively. The java code for the interceptor resides in the Java shared module provided in the sample.

Prerequisites

Before running the project, you must import the sample project to TIBCO Business Studio.

Procedure

1. Start TIBCO Business Studio in one of the following ways:
   - For Microsoft Windows, click **Start** > **All Programs** > **TIBCO** > **TIBCO_HOME** > **TIBCO Business Studio version_number** > **Studio for Designers**.
   - For Linux, run the TIBCO Business Studio executable file located in the `TIBCO_HOME\studio\version_number\eclipse directory`.
2. From the menu, click **File** > **Import**.
3. In the Import window, expand the **General** folder and select the **Existing Studio Projects into Workspace** item. Click **Next**.
4. Click **Browse** next to the **Select archive file** field to select the `Kafka_InterceptorSample.zip` file. Click **Finish**. The `Kafka_InterceptorSample.zip` file is in the `TIBCO_HOME\bw\palettes\kafka\`*`version`*`\samples` directory.

Result

The sample project is imported to TIBCO Business Studio.

Configuration of the Kafka_InterceptorSample Processes

The sample project contains two processes and a Java Shared Module. After Importing the Sample Project, expand the Processes resource to display the processes.

Kafka_InterceptorSample

This sample project contains the following:

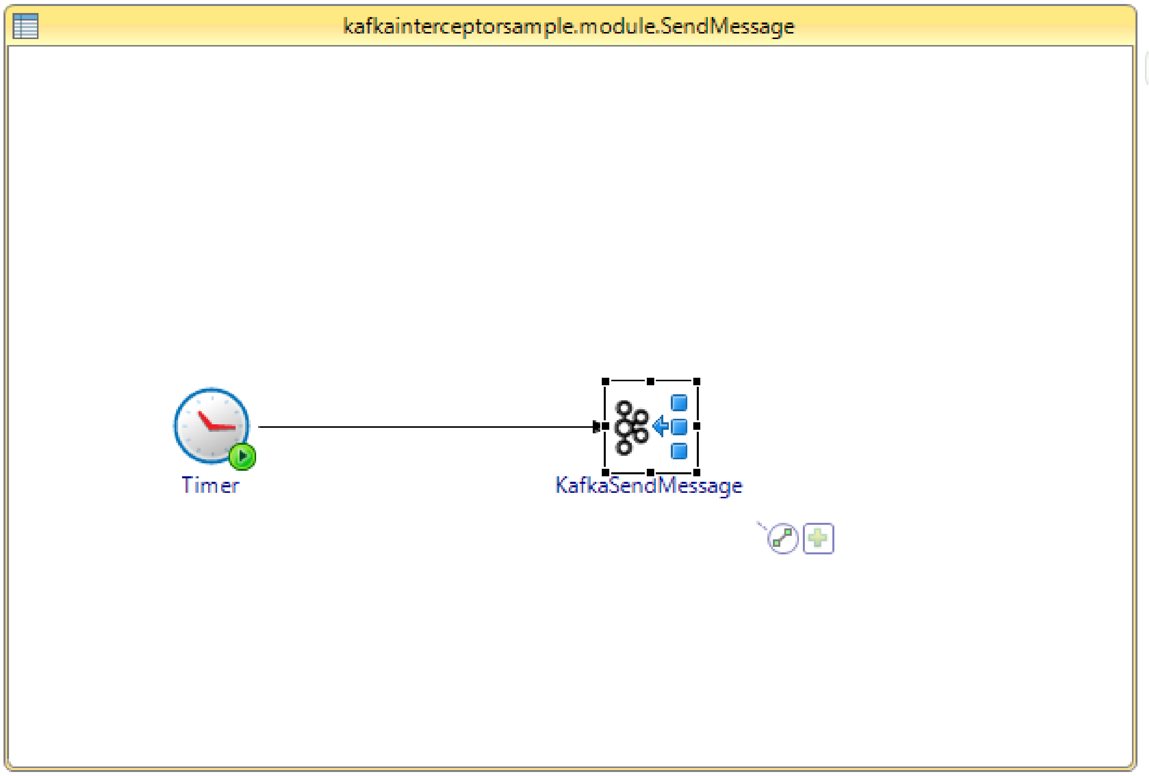
- Java Shared Module
- KafkaSendMessage.bwp
- ReceiveMessageProcess.bwp

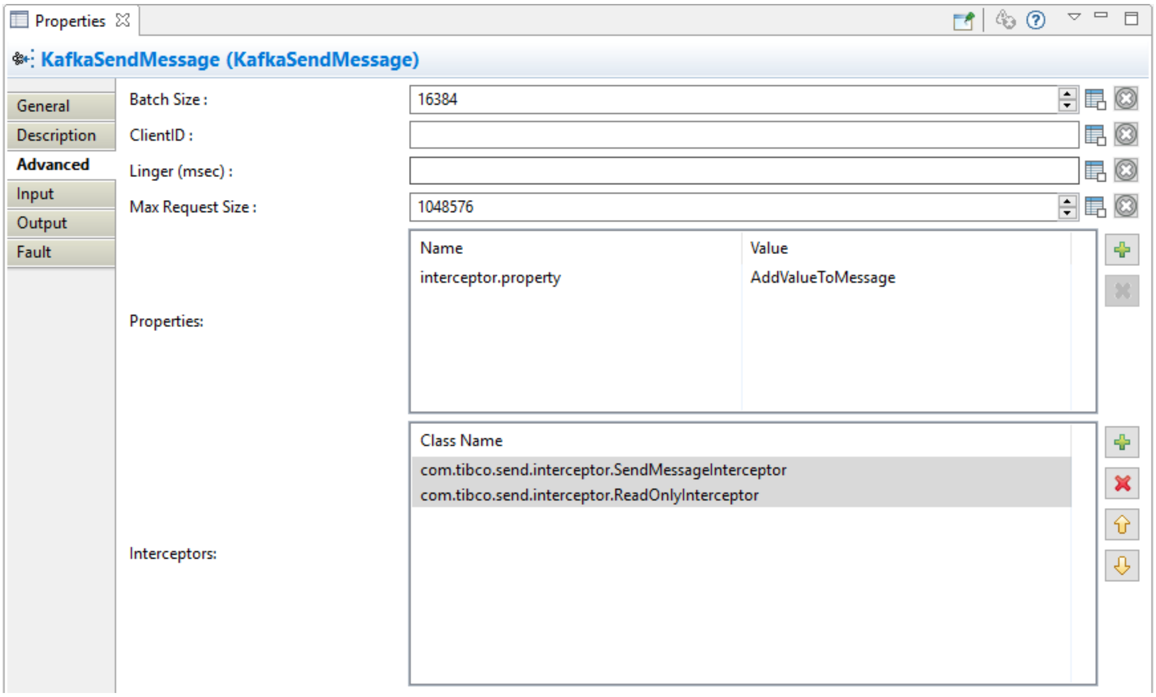
Java Shared Module

Java shared module contains the Java interceptor classes for the SendMessage and ReceiveMessage activities. In the SendMessage activity, the sent message is intercepted by the Java classes in the `com.tibco.send.interceptor` package. In the ReceiveMessage activity, the received message is intercepted by the `ReceiverInterceptor` Java class.

KafkaSendMessage.bwp

The KafkaSendMessage.bwp process demonstrates how interceptors can be added to the SendMessage activity to intercept the message before it is published on the server. On the **Advanced** tab, in the **Properties** field, an `interceptor.property` is configured as an additional property. This property is used by the `SendMessageInterceptor` that is configured in the **Interceptors** field.





KafkaSendMessage (KafkaSendMessage)

Advanced

Batch Size : 16384

ClientID :

Linger (msec) :

Max Request Size : 1048576

Properties:

Name	Value
interceptor.property	AddValueToMessage

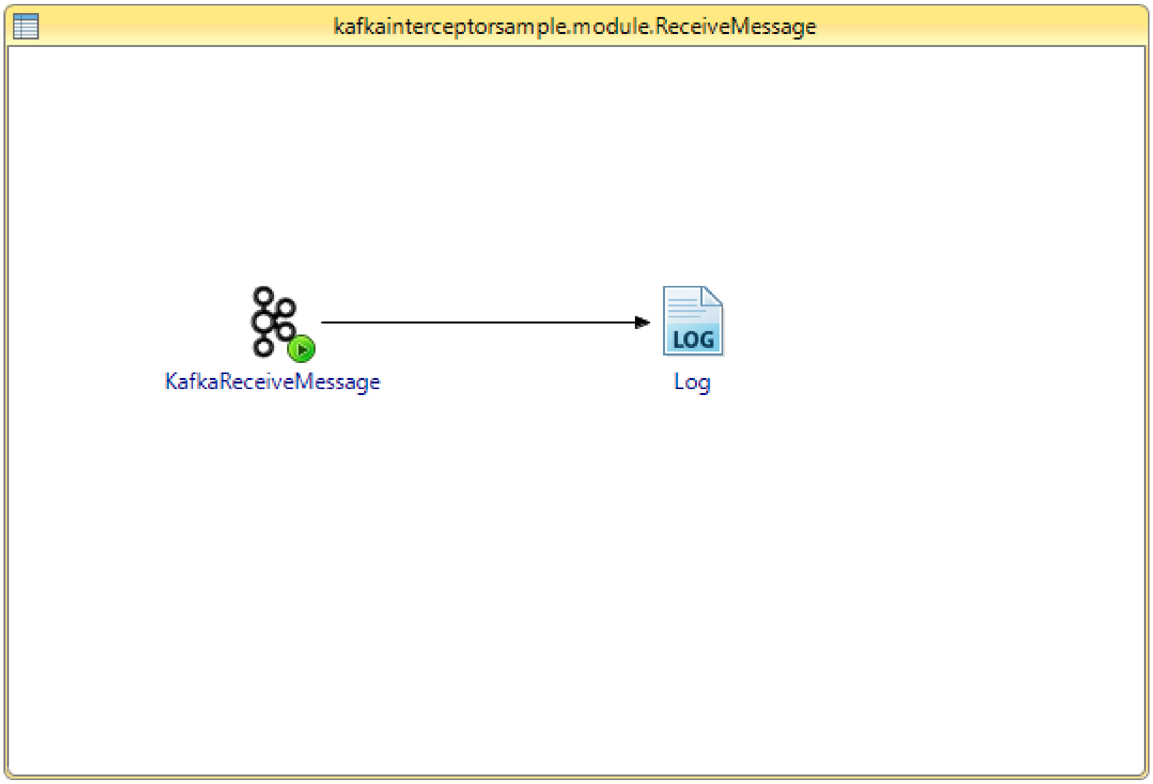
Interceptors:

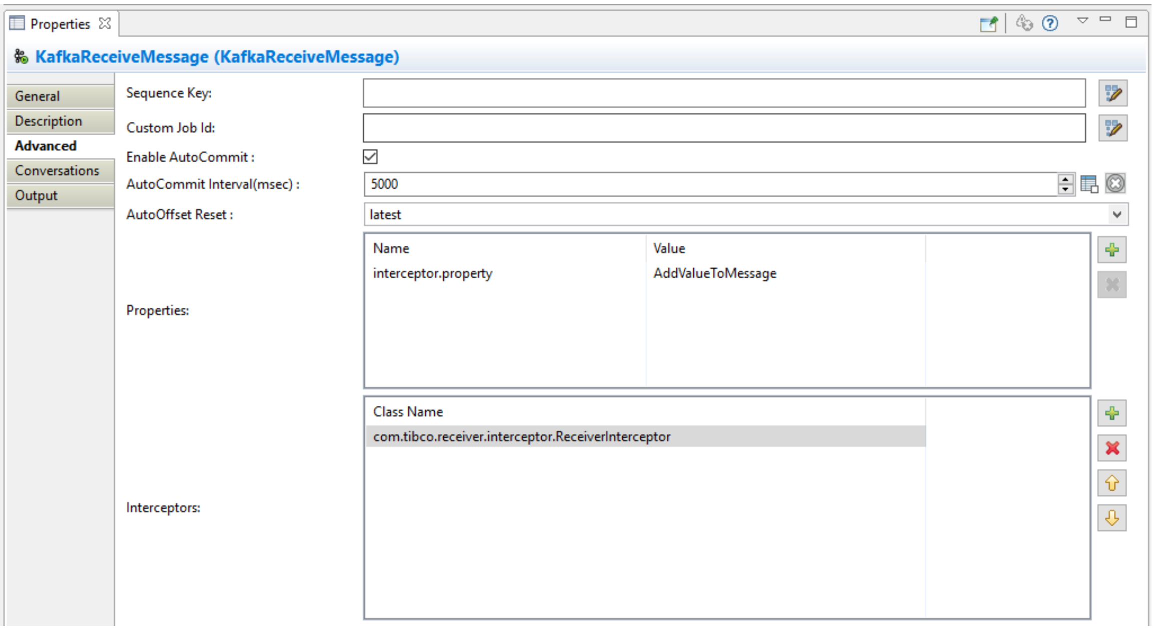
Class Name
com.tibco.send.interceptor.SendMessageInterceptor
com.tibco.send.interceptor.ReadOnlyInterceptor

ReceiveMessage.bwp

The ReceiveMessage.bwp process demonstrates how interceptors can be added to the ReceiveMessage activity to intercept the message before it is received by the activity. On the **Advanced** tab, in the **Properties** field, an `interceptor.property` is configured as an additional property. This property is used by the `ReceiveInterceptor` that is configured in the **Interceptors** field.

The configure method assigns the value received from `interceptor.property` to the `interceptorProperty` variable in the ReceiverInterceptor.java class. The onConsume method prints the received records to check if message value contains the value that is specified in the `interceptorProperty`.





KafkaReceiveMessage (KafkaReceiveMessage)

Sequence Key:

Custom Job Id:

Enable AutoCommit : ☑

AutoCommit Interval(msec) : 5000

AutoOffset Reset : latest

Properties:

Name	Value
interceptor.property	AddValueToMessage

Interceptors:

Class Name
com.tibco.receiver.interceptor.ReceiverInterceptor

Running the Kafka_InterceptorSample Project

Prerequisites

Import the sample project to TIBCO Business Studio, as described in Working with Kafka_InterceptorSample Project, and configure Kafka shared resource connection, as described in Creating a Kafka Connection.

Procedure

1. In the Project Explorer view, expand the **Module Descriptors** resource and double-click **Module Properties**.
2. Configure the **bootstrapServers**, **topicName**, and **consumerGroup** module properties.
3. On the toolbar, click the **Save** 💾 icon to save your changes.
4. From the menu, click **Run** > **Run Configurations** to run the selected process.
5. In the Run Configuration dialog box, expand **BusinessWorks Application**, and then click **BWApplication** .
6. In the right panel, click the **Applications** tab, and select the **Kafka_InterceptorSample** check box.
7. Click **Run** to run the selected process.

   You can stop the process at any time by clicking the terminate icon 🟥.

Working with Kafka_ErrorHandlingSample Project

Prerequisites

Before running the project, you must import the sample project to TIBCO Business Studio.

Procedure

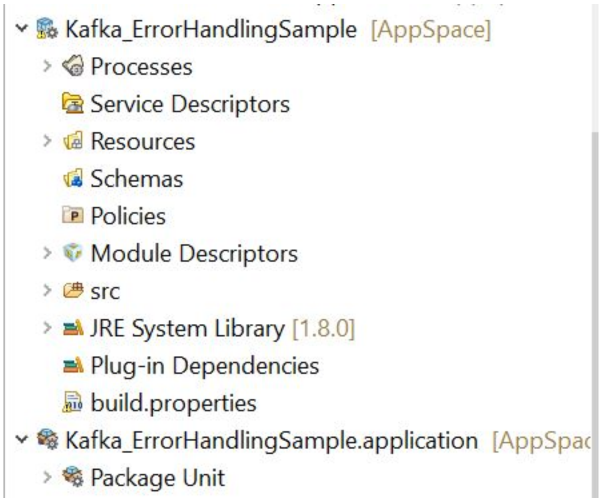
1. Start TIBCO Business Studio in one of the following ways:

   - For Microsoft Windows, click **Start** > **All Programs** > **TIBCO** > **TIBCO_HOME** > **TIBCO Business Studio version_number** > **Studio for Designers**.
   - For Linux, run the TIBCO Business Studio executable file located in the `TIBCO_HOME\studio\version_number\eclipse directory`.

2. From the menu, click **File** > **Import**.
3. In the Import window, expand the **General** folder and select the **Existing Studio Projects into Workspace** item. Click **Next**.
4. Click **Browse** next to the **Select archive file** field to select the `Kafka_ErrorHandlingSample.zip` file. Click **Finish**. The `Kafka_ErrorHandlingSample.zip` file is in the `TIBCO_HOME\bw\palettes\kafka\`*`version`*`\samples` directory.

Result

The sample project is imported to TIBCO Business Studio.



- Kafka_ErrorHandlingSample [AppSpace]
  - Processes
  - Service Descriptors
  - Resources
  - Schemas
  - Policies
  - Module Descriptors
  - src
  - JRE System Library [1.8.0]
  - Plug-in Dependencies
  - build.properties
- Kafka_ErrorHandlingSample.application [AppSpac
  - Package Unit

Configuring Kafka Connection for Kafka_ErrorHandlingSample

You must configure the Kafka shared connection resource to set up a connection with the Kafka server.

Procedure

1. In the Project Explorer view, expand **Kafka_ErrorHandlingSample**.
2. In the Resources folder, double-click **KafkaConnectionResource.kafkaconnectionResource**.
3. In the KafkaConnectionResource Editor, configure each field accordingly.
4. On the toolbar, click the **Save** icon to save your changes.

For more information, see Creating a Kafka Connection.

Configuration of the Kafka_ErrorHandlingSample Processes

The sample project contains two processes. After Importing the Sample Project, expand the Processes resource to display the processes.

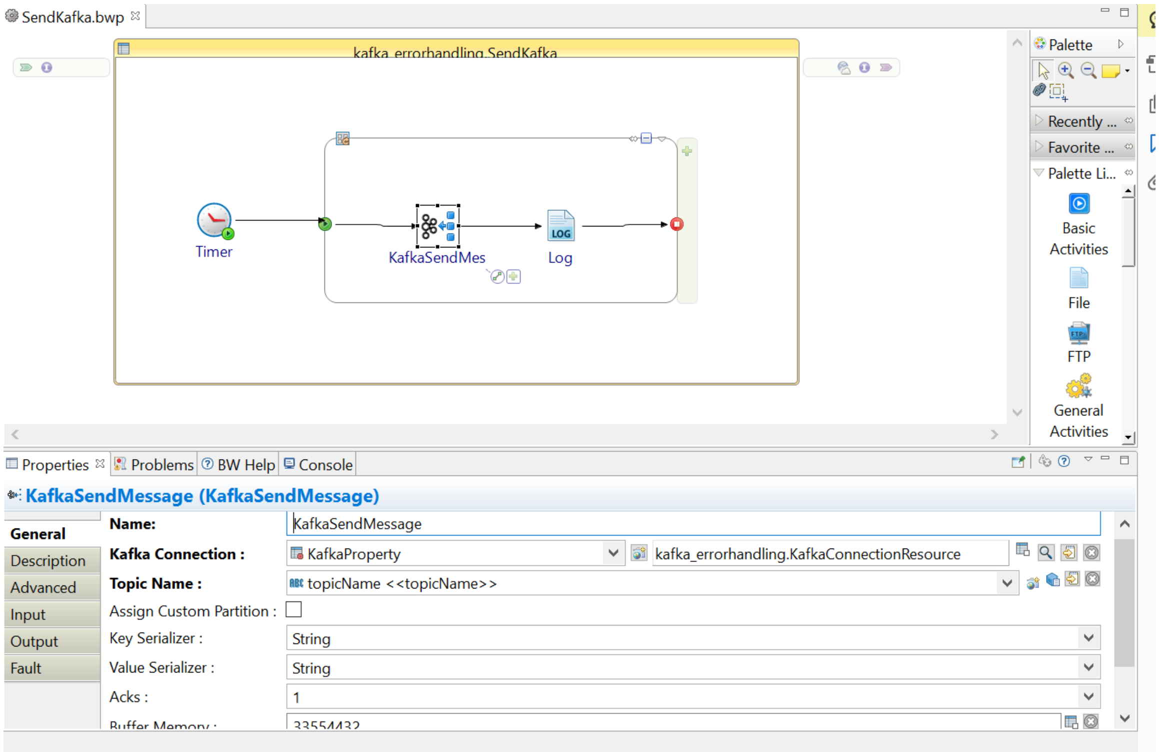
Kafka_ErrorHandlingSample

This sample project contains the following processes:

- SendKafka.bwp
- KafkaErrorHandling.bwp

SendKafka.bwp

The SendKafka.bwp process demonstrates how to use the plug-in to send messages to the Kafka server, which are then consumed by the KafkaErrorHandling.bwp process. KafkaSendMessage activity is used here to send 50 messages to a kafka topic specified in the activity configuration.



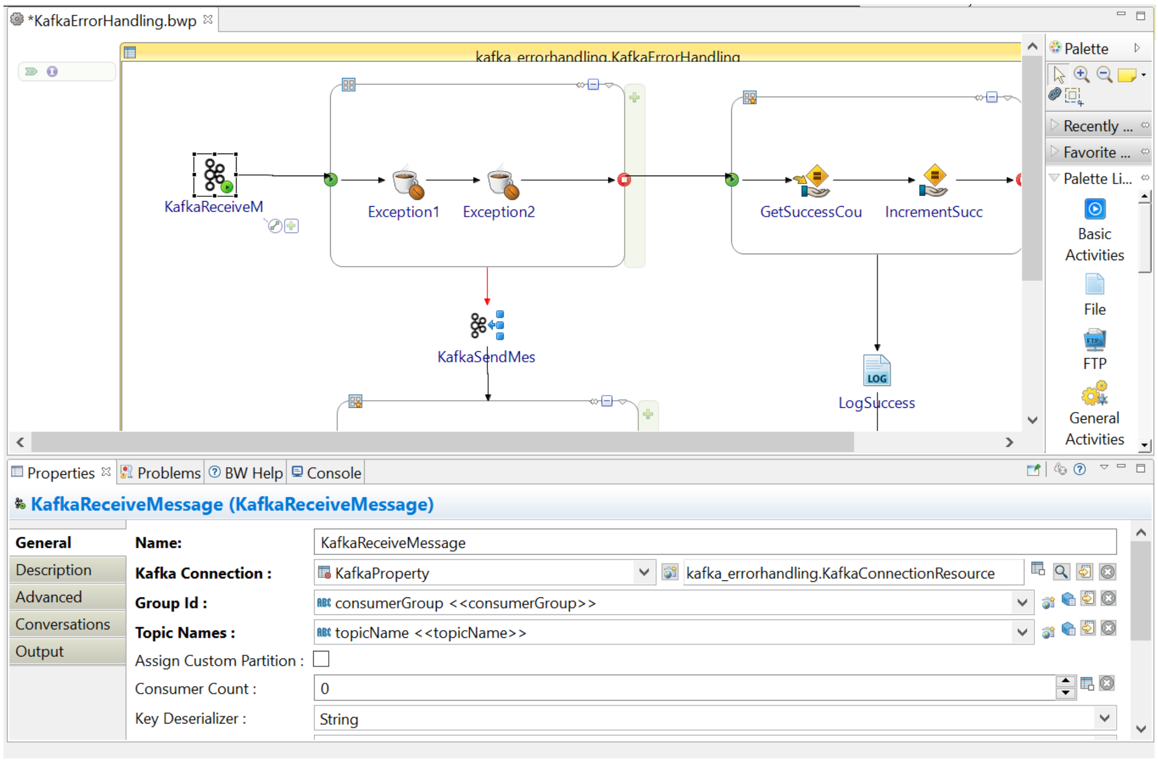
KafkaSendMessage (KafkaSendMessage)

General | Description | Advanced | Input | Output | Fault

Name: KafkaSendMessage
Kafka Connection : KafkaProperty kafka_errorhandling.KafkaConnectionResource
Topic Name : topicName <<topicName>>
Assign Custom Partition : □
Key Serializer : String
Value Serializer : String
Acks : 1
Buffer Memory : 33554432

KafkaErrorHandling.bwp

The KafkaErrorHandling.bwp process demonstrates how to use the plug-in for handling errors during Kafka receive. It contains a ReceiveMessage process starter that listens for incoming message event and receives the 50 messages sent using SendKafka.bwp. The first Java Invoke error occurs on every 5th offset and the second Java Invoke error occurs on every 7th offset. The error messages are then sent to KafkaSendMessage activity, which in turn sends them to the specified topic. 16 messages error out and remaining 34 go through the success transition.



KafkaReceiveMessage (KafkaReceiveMessage)

General | Description | Advanced | Conversations | Output

Name: KafkaReceiveMessage
Kafka Connection : KafkaProperty kafka_errorhandling.KafkaConnectionResource
Group Id : consumerGroup <<consumerGroup>>
Topic Names : topicName <<topicName>>
Assign Custom Partition : □
Consumer Count : 0
Key Deserializer : String

Running the Kafka_ErrorHandlingSample Project

The sample project shows how to use the plug-in to send and receive messages using Apache Kafka.

Prerequisites

Import the sample project to TIBCO Business Studio, as described in Working with Kafka_ErrorHandlingSample Project , and configure Kafka shared resource connection, as described in Creating a Kafka Connection.

Procedure

1. In the Project Explorer view, expand the **Module Descriptors** resource and double-click **Components**.
2. Configure the **bootstrapServers**, **topicName**, and **consumerGroup**, **exceptionTopic** module properties.
3. On the toolbar, click the **Save** 💾 icon to save your changes.
4. From the menu, click **Run** > **Run Configurations** to run the selected process.
5. In the Run Configuration dialog box, expand **BusinessWorks Application**, and then click **BWApplication**.
6. In the right panel, click the **Applications** tab, and select the check box next to **Kafka_ErrorHandlingSample.application**.
7. Click **Run** to run the selected process.
8. Click the **Terminate** icon 🟥 to stop the process.

Working with Kafka_CustomSerializerSample Project

This sample demonstrates how to configure and use the Custom Serializer and Deserializer in the **Send** and **Receive** activities. The java code for the serializer or deserializer is in the `src` folder. The example uses Google protocol buffer to demonstrate the custom serializer ability.

Prerequisites

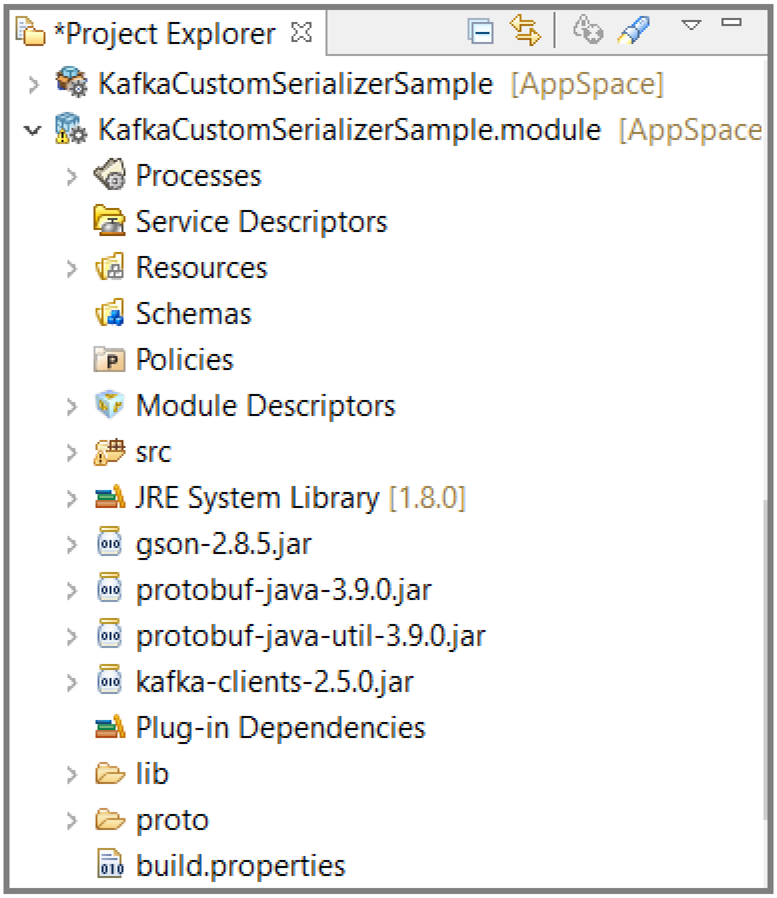
Before running the project, you must import the sample project to TIBCO Business Studio.

Procedure

1. Start TIBCO Business Studio in one of the following ways:
   - For Microsoft Windows, click **Start** > **All Programs** > **TIBCO** > **TIBCO_HOME** > **TIBCO Business Studio version_number** > **Studio for Designers**.
   - For Linux, run the TIBCO Business Studio executable file located in the `TIBCO_HOME\studio\version_number\eclipse directory`.
2. From the menu, click **File** > **Import**.
3. In the Import window, expand the **General** folder and select the **Existing Studio Projects into Workspace** item. Click **Next**.
4. Click **Browse** next to the **Select archive file** field to select the `Kafka_CustomSerializerSample.zip` file. Click **Finish**. The `Kafka_CustomSerializerSample.zip` file is in the `TIBCO_HOME\bw\palettes\kafka\`*`version`*`\samples` directory.

Result

The sample project is imported to TIBCO Business Studio.



Configuring Kafka Connection for Kafka_CustomSerializerSample

Configuring Kafka shared connection resource is required to configure a connection with the Kafka server.

Procedure

1. In the Project Explorer view, expand **Kafka_CustomSerializerSample**.
2. In the Resources folder, double-click **KafkaConnectionResource.kafkaconnectionResource**.
3. In the KafkaConnectionResource Editor, configure each field accordingly.
4. On the toolbar, click the **Save** 💾 icon to save your changes.

For more information, refer Creating a Kafka Connection.

Configuration of the Kafka_CustomSerializer Processes

The sample project contains two processes. After Importing the Sample Project, expand the Processes resource to display the processes.

The `src` folder contains the `PersonProtos.java` file which is a google protobuf defined java representation of the `.proto` file in a proto folder.

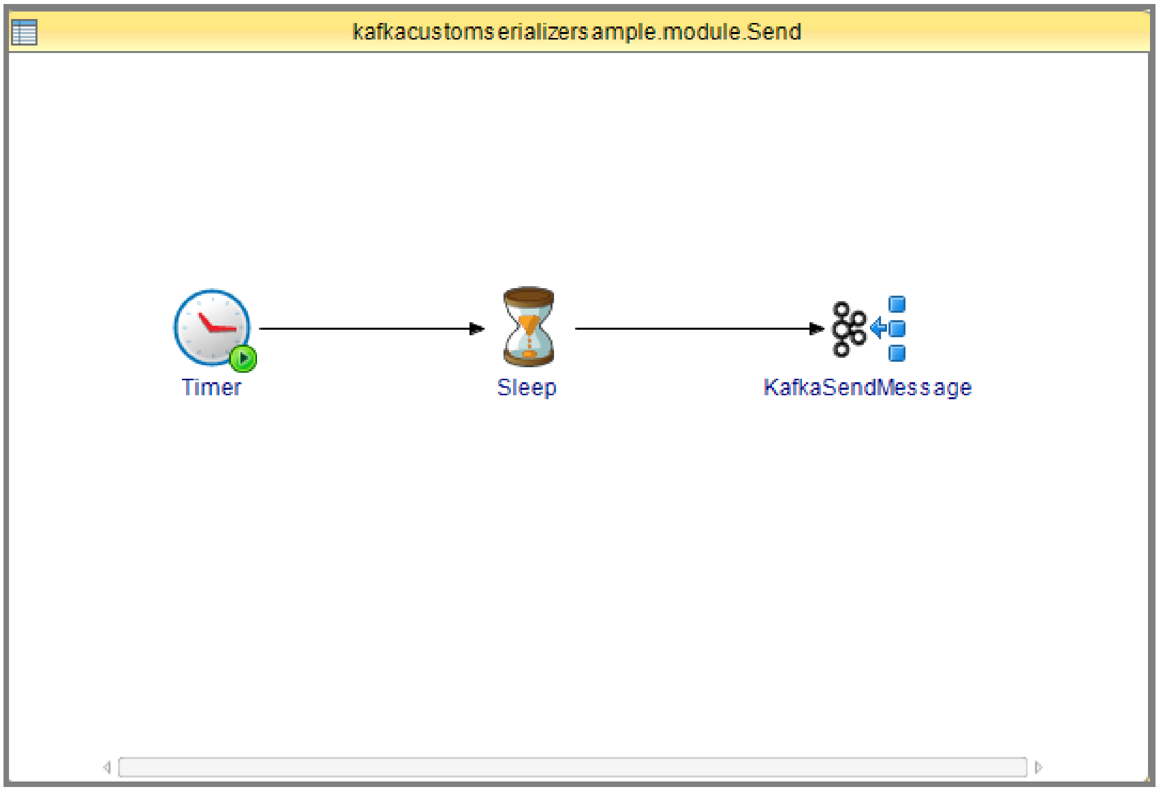
Kafka_CustomSerializerSample

This sample project contains the following two processes:

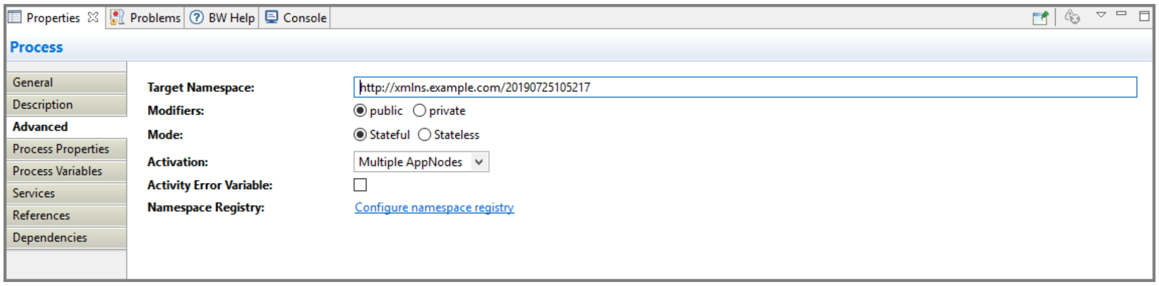
- Send.bwp
- Receive.bwp

Send.bwp

The Send.bwp process demonstrates how custom serializer is added to the **Send** activity to serialize the message. On the **Advanced** tab in the **Properties** field, a `value.serializer` property is configured as an additional property. This property lets the **Send** activity use the class for serializing the message. For example, Google Protocol Buffer is used to serialize the message from a Person objects (defined in the `PersonProtos.java` file in the src folder) JSON representation to a byte data.



Activity configuration for the Send.bwp:



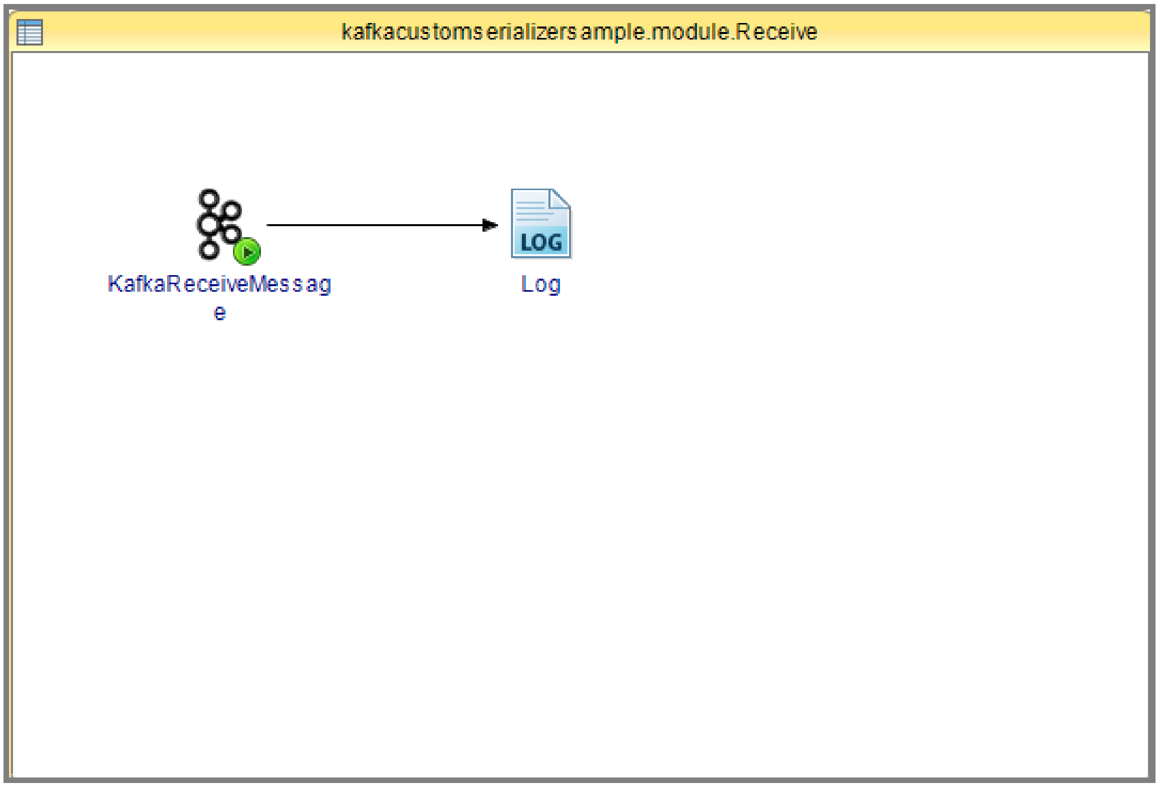
Process

General
Description
Advanced
Process Properties
Process Variables
Services
References
Dependencies

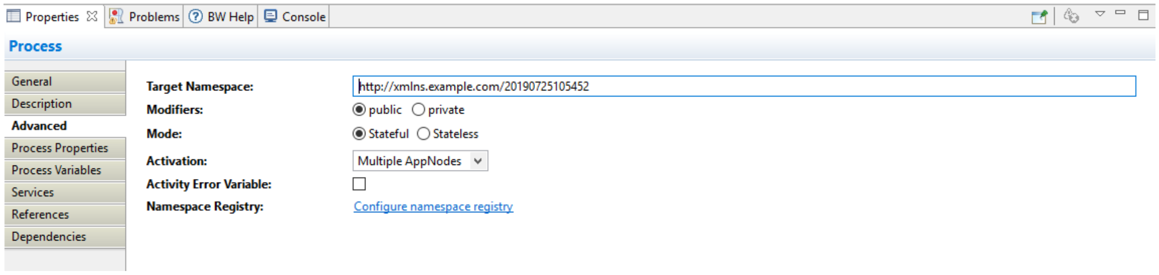
Target Namespace:	http://xmlns.example.com/20190725105217
Modifiers:	● public ○ private
Mode:	● Stateful ○ Stateless
Activation:	Multiple AppNodes
Activity Error Variable:	□
Namespace Registry:	Configure namespace registry

Receive.bwp

The Receive.bwp process demonstrates how custom deserializer is added to the **Receive** activity to deserialize the message. On the **Advanced** tab in the **Properties** field a `value.deserializer` property is configured as an additional property. This property lets the **Receive** activity use a class for deserializing the message. For example, Google Protocol Buffer is used to deserialize the message from a byte data to a Person objects (defined in the `PersonProtos.java` file in the `src` folder) JSON representation.



Activity configuration for the Receive.bwp:



Target Namespace: http://xmlns.example.com/20190725105452

Modifiers: ● public ○ private

Mode: ● Stateful ○ Stateless

Activation: Multiple AppNodes

Activity Error Variable: □

Namespace Registry: Configure namespace registry

Running the Kafka_CustomSerializerSample Project

The sample project shows how to use the plug-in to use Custom Serializer or Deserializer to send and receive messages using Apache Kafka.

Prerequisites

Import the sample project to TIBCO Business Studio, as described in Working with Kafka_CustomSerializerSample Project, and configure Kafka shared resource connection, as described in Creating a Kafka Connection.

Procedure

1. In the Project Explorer view, expand the **Module Descriptors** resource and double-click **Components**.
2. By default, all the processes are listed in the Components editor.
3. On the toolbar, click the **Save** 💾 icon to save your changes.
4. From the menu, click **Run** > **Run Configurations** to run the selected process.

5. In the Run Configuration dialog box, expand **BusinessWorks Application**, and then click **BWApplication.**
6. In the right panel, click the **Applications** tab, and select the check box next to **Kafka_GeneralSample.application**.
7. Click **Run** to run the selected process.
8. Click the **Terminate** 🟥 icon to stop the process.

Working with Kafka_TransactionSample Project

Prerequisites

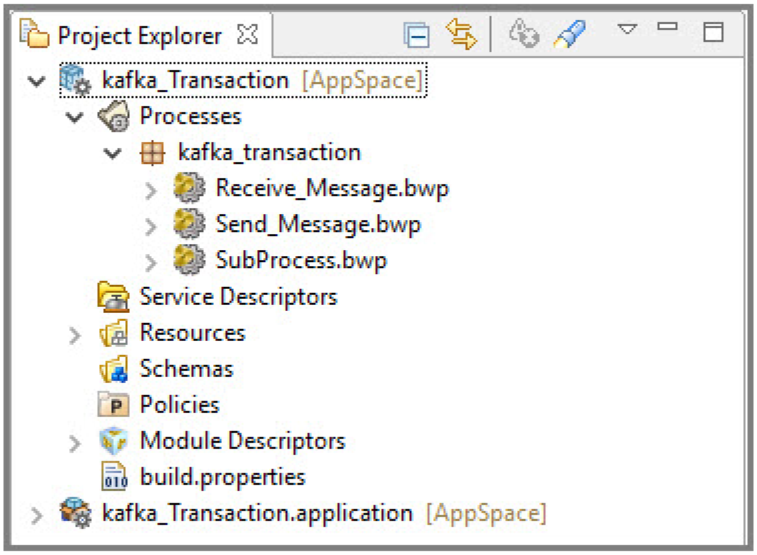
Before running the project, you must import the sample project to TIBCO Business Studio.

Procedure

1. Start TIBCO Business Studio using one of the following ways:
    - For Microsoft Windows, click **Start** > **All Programs** > **TIBCO** > **TIBCO_HOME** > **TIBCO Business Studio version_number** > **Studio for Designers**.
    - For Linux, run the TIBCO Business Studio executable file located in the `TIBCO_HOME\studio\version_number\eclipse directory`.
2. From the menu, click **File** > **Import**.
3. In the Import window, expand the **General** folder and select the **Existing Studio Projects into Workspace** item. Click **Next**.
4. Click **Browse** next to the **Select archive file** field to select the `Kafka_TransactionSample.zip` file. Click **Finish**. The `Kafka_TransactionSample.zip` file is in the `TIBCO_HOME\bw\palettes\kafka\`*`version`*`\samples` directory.

Result

The sample project is imported to TIBCO Business Studio.



Configuring Kafka Connection for Kafka_TransactionSample

Configuring Kafka shared connection resource is required to configure a connection with the Kafka server.

Procedure

1. In the Project Explorer view, expand **Kafka_TransactionSample**.
2. In the Resources folder, double-click **KafkaConnectionResource.kafkaconnectionResource**.
3. In the KafkaConnectionResource Editor, configure each field accordingly.
4. On the toolbar, click the **Save** 💾 icon to save your changes.

For more information, refer Creating a Kafka Connection.

Configuration of the Kafka_TransactionSample Processes

The sample project contains three processes. Each process has a different function. After Importing the Sample Project, expand the Processes resource to display the processes.

Kafka_TransactionSample

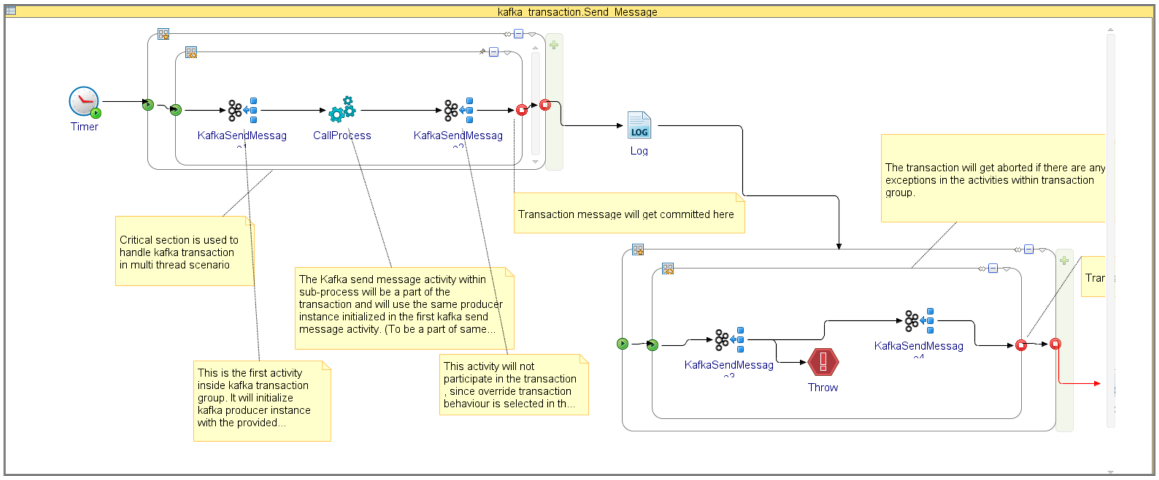
This sample project contains the following three processes:

- Receive_Message.bwp
- Send_Message.bwp
- SubProcess.bwp

Send_Message.bwp

This process demonstrates how to use the plug-in to implement Kafka transactions. The Kafka SendMessage activity which participates in a transaction are added to the Local Transaction Group. The Local Transaction Group is configured with transport type Kafka. All the Kafka SendMessage activities which belongs to a transaction should share the same **Transaction ID** and **Kafka connection shared resource**. A critical section group is used to handle Kafka transactions when multiple threads are executing in parallel. The process demonstrates two scenarios:

1. In the first Local Transaction Group, one or more Kafka SendMessage activity participates in a transaction and the transaction gets committed successfully.
2. In the second Local Transaction Group, one or more Kafka SendMessage activity participates in a transaction but the transaction gets aborted due to an exception thrown from within the transaction group.



Critical section is used to handle kafka transaction in multi thread scenario

This is the first activity inside kafka transaction group. It will initialize kafka producer instance with the provided...

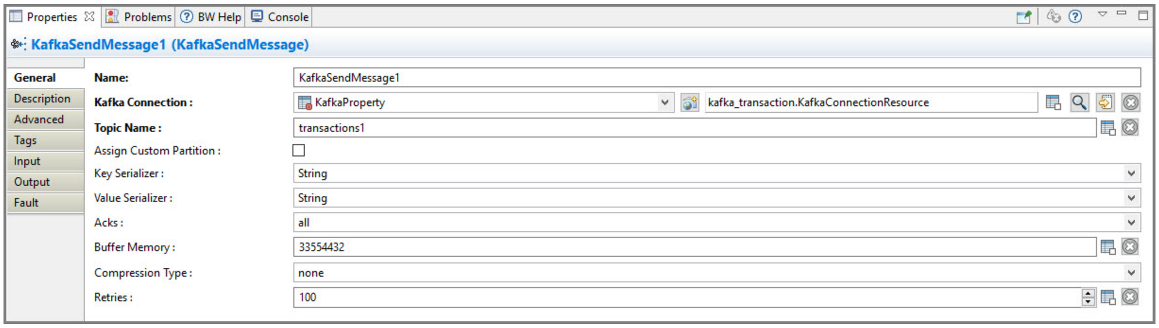
The Kafka send message activity within sub-process will be a part of the transaction and will use the same producer instance initialized in the first kafka send message activity. (To be a part of same...

This activity will not participate in the transaction , since override transaction behaviour is selected in th...

Transaction message will get committed here

The transaction will get aborted if there are any exceptions in the activities within transaction group.

Activity configuration for the Send_Message.bwp:

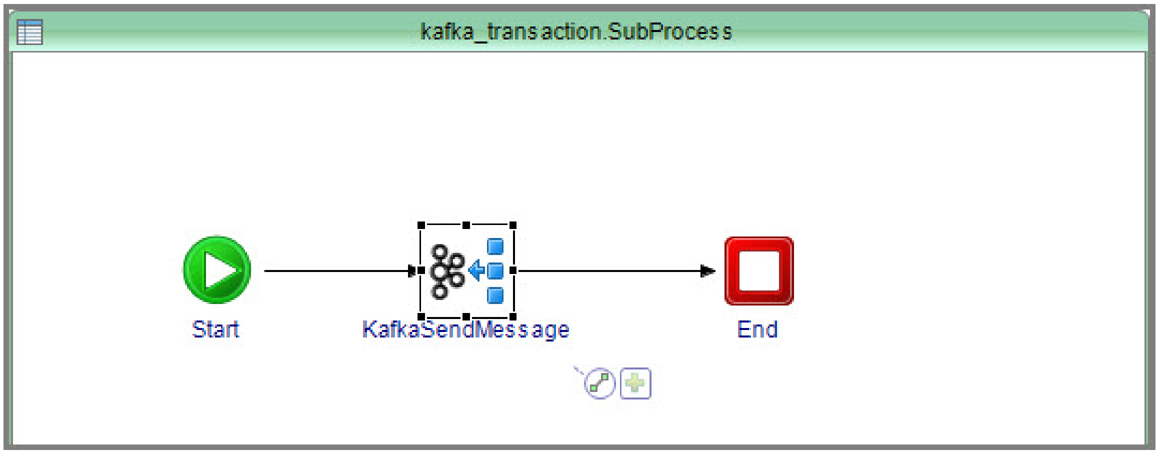


KafkaSendMessage1 (KafkaSendMessage)

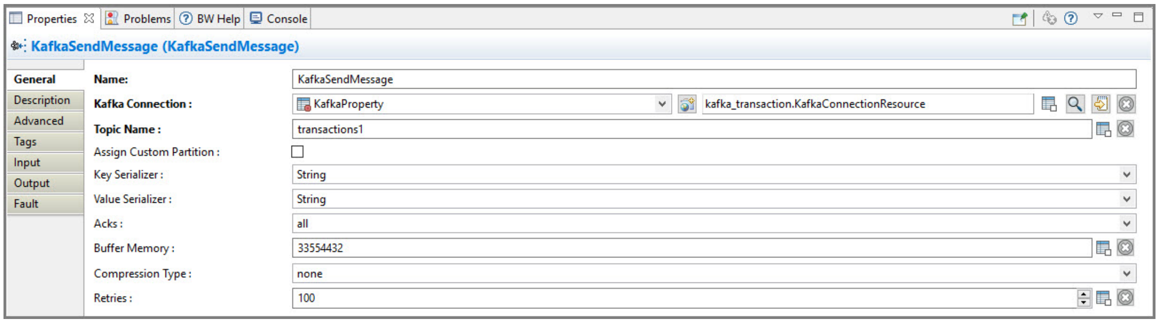
Name:	KafkaSendMessage1
Kafka Connection :	KafkaProperty — kafka_transaction.KafkaConnectionResource
Topic Name :	transactions1
Assign Custom Partition :	□
Key Serializer :	String
Value Serializer :	String
Acks :	all
Buffer Memory :	33554432
Compression Type :	none
Retries :	100

SubProcess.bwp

The subprocess demonstrates Kafka SendMessage activity participating in the transaction initiated by parent process if it is configured with identical Transaction ID and the Kafka connection shared resource. Spawned subprocess will not participate in the parent initiated transaction.



Activity configuration for the SubProcess.bwp:

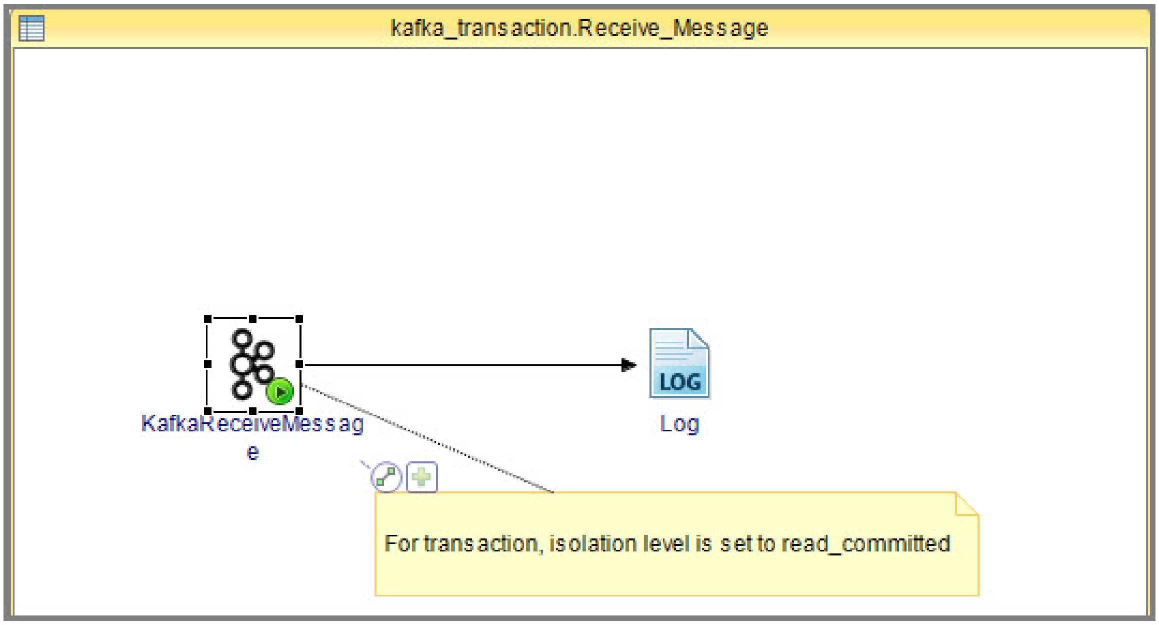


KafkaSendMessage (KafkaSendMessage)

Name: KafkaSendMessage
Kafka Connection : KafkaProperty kafka_transaction.KafkaConnectionResource
Topic Name : transactions1
Assign Custom Partition : □
Key Serializer : String
Value Serializer : String
Acks : all
Buffer Memory : 33554432
Compression Type : none
Retries : 100

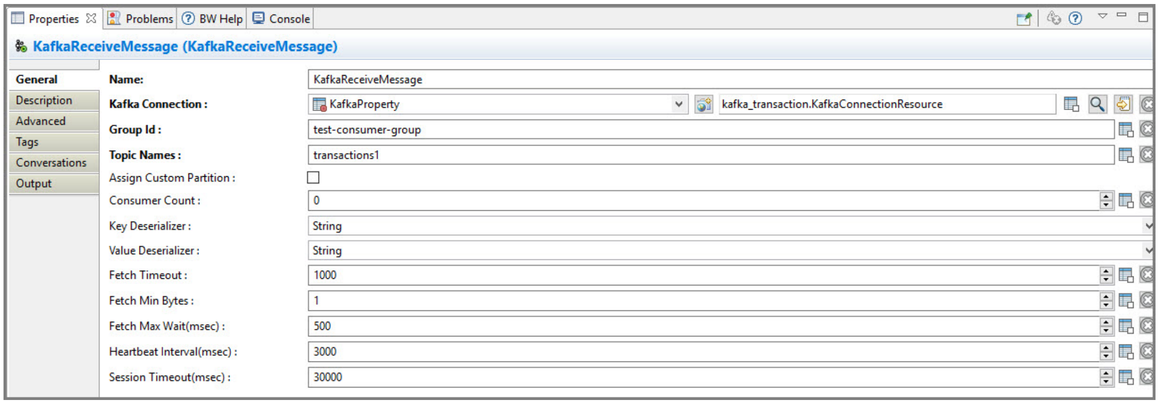
Receive_Message.bwp

This process demonstrates how to use the Kafka ReceiveMessage activity to receive transactional messages. In the **Advanced** tab, the isolation level is configured to read_committed which ensures that the activity will receive only those messages which are successfully committed during the transaction.



For transaction, isolation level is set to read_committed

Activity configuration for the Receive_Message.bwp:



KafkaReceiveMessage (KafkaReceiveMessage)

Name: KafkaReceiveMessage
Kafka Connection : KafkaProperty kafka_transaction.KafkaConnectionResource
Group Id : test-consumer-group
Topic Names : transactions1
Assign Custom Partition : □
Consumer Count : 0
Key Deserializer : String
Value Deserializer : String
Fetch Timeout : 1000
Fetch Min Bytes : 1
Fetch Max Wait(msec) : 500
Heartbeat Interval(msec) : 3000
Session Timeout(msec) : 30000

Running the Kafka_TransactionSample Project

The sample project shows how to use the plug-in to send and receive messages using Apache Kafka.

Prerequisites

Import the sample project to TIBCO Business Studio, as described in Working with Kafka_TransactionSample Project, and configure Kafka shared resource connection, as described in Creating a Kafka Connection.

Procedure

1. In the Project Explorer view, expand the **Module Descriptors** resource and double-click **Components**.
2. By default, all the processes are listed in the Components editor.
3. On the toolbar, click the **Save** 💾 icon to save your changes.
4. From the menu, click **Run** > **Run Configurations** to run the selected process.
5. In the Run Configuration dialog box, expand **BusinessWorks Application**, and then click **BWApplication.**
6. In the right panel, click the **Applications** tab, and select the check box next to **Kafka_GeneralSample.application**.
7. Click **Run** to run the selected process.
8. Click the **Terminate** 🟥 icon to stop the process.

Log Management

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

By default, error logs are displayed in the Console view when you run a process in the debug mode. 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.

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.
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 different log levels 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 logger area to specify a log level for the plug-in:

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

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

   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. Optional. Add one of the following nodes in the BusinessWorks Palette and Activity loggers' area to control a log level for the activity.

   - For example, to control the debug log level for the Kafka Send Message activity, set the following parameters:

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

- For example, to control the debug log level for the Kafka Receive Message activity, set the following parameters:

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

- For example, to control the debug log level for the Kafka Get Messages activity, set the following parameters:

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

- For example, to control the debug log level for the Kafka Shared resource, set the following parameters:

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

  > 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. In Windows system, navigate to `TIBCO_HOME\bw\version_number\config\design\logback` directory and open the `logback.xml` file.

   > 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-kafka.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 end of the `logback.xml` file:

   ```
   <root level="DEBUG">
   <appender-ref ref="STDOUT"/>
   <appender-ref ref="FILE"/>
   </root>
   ```

4. Save the file.

Error Codes

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

Error Code and Error Message	Role	Category	Description	Solution
BW-KAFKA-100001 `PARAMETER_NOT_SPECIFIED.`	errorRole	BW-Plug-in	Occurs when a parameter is not specified while running activity. Message will also contain parameter name.	Provide required parameter.
BW-KAFKA-100002 `SPECIAL_CHARACTER_NOT_ALLOWED.`	errorRole	BW-Plug-in	Occurs when special character is used for group ID.	Remove special character from group ID.

ERROR_OCCURED_RETRIEVE_RESULT.errorCode=500002
`IOException occurred while retrieving XML Output for activity [{0}].`
	errorRole	BW-Plug-in	Occurs when an activity retrieves an XML output.	None.

ERROR_OCCURED_INVOKE_EXECUTE_METHOD.errorCode=500003
`Exception occurred while invoking execute method for activity [{0}].`
	errorRole	BW-Plug-in	Occurs when an activity invokes an execute method.	None.

SEND_PRODUCER_CREATED_EXCEPTION.errorCode=500004
`Exception occurred while create producer.{0}`
	errorRole	BW-Plug-in	Occurs when producer creation fails.	Check if Kafka configuration is correct and it is running.

SEND_MESSAGE_EXCEPTION.errorCode=500005
`Exception occurred while send message to broker. {0}.`
	errorRole	BW-Plug-in	Occurs when sending message to broker fails.	Check if broker is running.

RECEIVED_CONSUMER_CREATED_EXCEPTION.errorCode=500007
`Exception occurred while create consumer.{0}`
	errorRole	BW-Plug-in	Occurs when Consumer creation fails.	Check if Kafka configuration is correct and it is running.

Error Code and Error Message	Role	Category	Description	Solution
ERROR_OCCURED_RECEIVE_MESSAGE.errorCode=500008 Exception occurred while receive message from broker. {0}.	errorRole	BW-Plug-in	Occurs when receiving message from broker fails.	Check if broker is running.
ERROR_KAFKA_CONNECTION_ESTABLISH.errorCode=500009 Exception occurred while establishing the Kafka connection.Please verify the Kafka connection configuration [{0}].	errorRole	BW-Plug-in	Occurs when kafka topic metadata is incorrect.	Check Kafka connection configuration for metadata.
SEND_MESSAGE_BUFFER_MEMORY_EXCEPTION.errorCode=500010 Exception occurred while sending message to broker because message is larger than the total memory buffer you have configured with the buffer.memory configuration.	errorRole	BW-Plug-in	Occurs when memory buffer configuration is incorrect.	Change buffer memory settings.
SEND_TOPICNAME_EXCEPTION.errorCode=500011 Exception occurred because topic name is illegal, contains a character other than ASCII alphanumerics, '.', '_' and '-'.	errorRole	BW-Plug-in	Occurs when topic name contains special character other than ., -, _.	Update topic name in BW.
ERROR_SEND_CUSTOMPARTITION.errorCode=500012 Exception occurred because provided customer partition does not exist on Kafka broker. {0}.	errorRole	BW-Plug-in	Occurs when specified partition does not exist.	Check partition for the broker.
ERROR_OCCURED_RECEIVE_MESSAGE_CUSTOMPARTITION.errorCode=500013 Exception occurred because given partition does not exist on Kafka broker. {0}.	errorRole	BW-Plug-in	Occurs when specified partition does not exist.	Check partition for the broker.

Error Code and Error Message	Role	Category	Description	Solution

ERROR_SEND_MAX_REQUEST_SIZE.errorCode=500014
`Exception occurred because message is too large. {0}.`
	errorRole	BW-Plug-in	Occurs when message is larger than allowed size.	Archive message or increase allowed size in server settings.

ERROR_AVRO_SCHEMA.errorCode=500015
`Exception occurred when configuring Kafka producer or consumer with Avro Schema. {0}`
	errorRole	BW-Plug-in	Occurs when the avro schema is not given or is incorrect.	Check if correct avro schema has been provided. If use Registry is checked check if schema registry is correctly provided.

KAFKA_ACTIVITY_TRANSACTION_ERROR.errorCode=500016
`Exception occurred during Kafka transaction {0}`
	errorRole	BW-Plug-in	Occurs when the activity is configured incorrectly for kafka transaction.	Check if the activity is inside the transaction group and a valid transaction id is provided. Ensure that Enable Idempotence is selected.

KAFKA_ACTIVITY_NEW_PRODUCER_TRANSACTION_ERROR.errorCode=500017
`The new producer system property configuration and transactions are not allowed together.`
	errorRole	BW-Plug-in	Occurs when `Dcom.tibco.plugin.kafka.new.producer=true` system property is configured along with kafka transactions	Both Kafka transaction and new producer system property together is not allowed. Please remove system property to use Kafka transactions.

ERROR_OCCURED_DURING_GET_MESSAGE.errorCode=500018
`Exception occurred in get message activity. {0}`
	errorRole	BW-Plug-in	Occurs when receiving message from broker fails.	Check if broker is running.

Error Code and Error Message	Role	Category	Description	Solution

GET_MESSAGES_CONSUMER_CREATED_EXCEPTION.errorCode=500019
`Exception occurred while creating consumer in get messages`
	errorRole	BW-Plug-in	Occurs when Consumer creation fails	Check if Kafka configuration is correct and broker is running.

Legal and Third-Party Notices

SOME TIBCO SOFTWARE EMBEDS OR BUNDLES OTHER TIBCO SOFTWARE. USE OF SUCH EMBEDDED OR BUNDLED TIBCO SOFTWARE IS SOLELY TO ENABLE THE FUNCTIONALITY (OR PROVIDE LIMITED ADD-ON FUNCTIONALITY) OF THE LICENSED TIBCO SOFTWARE. THE EMBEDDED OR BUNDLED SOFTWARE IS NOT LICENSED TO BE USED OR ACCESSED BY ANY OTHER TIBCO SOFTWARE OR FOR ANY OTHER PURPOSE.

USE OF TIBCO SOFTWARE AND THIS DOCUMENT IS SUBJECT TO THE TERMS AND CONDITIONS OF A LICENSE AGREEMENT FOUND IN EITHER A SEPARATELY EXECUTED SOFTWARE LICENSE AGREEMENT, OR, IF THERE IS NO SUCH SEPARATE AGREEMENT, THE CLICKWRAP END USER LICENSE AGREEMENT WHICH IS DISPLAYED DURING DOWNLOAD OR INSTALLATION OF THE SOFTWARE (AND WHICH IS DUPLICATED IN THE LICENSE FILE) OR IF THERE IS NO SUCH SOFTWARE LICENSE AGREEMENT OR CLICKWRAP END USER LICENSE AGREEMENT, THE LICENSE(S) LOCATED IN THE "LICENSE" FILE(S) OF THE SOFTWARE. USE OF THIS DOCUMENT IS SUBJECT TO THOSE 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 TIBCO Software Inc.

TIBCO, the TIBCO logo, the TIBCO O logo, TIBCO ActiveMatrix BusinessWorks™, TIBCO Administrator™, TIBCO Business Studio, Hawk®, Rendezvous®, TIBCO Runtime Agent™, TIBCO Designer™, TIBCO BusinessEvents®, TIBCO BusinessConnect™, and TIBCO ActiveMatrix BusinessWorks™ Plug-in for Apache Kafka are either registered trademarks or trademarks of TIBCO Software Inc. in the United States and/or other countries.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle Corporation in the U.S. and 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.

This 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. Please see the readme.txt file for the availability of this software version 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. TIBCO SOFTWARE INC. MAY MAKE IMPROVEMENTS AND/OR CHANGES IN THE PRODUCT(S) AND/OR THE PROGRAM(S) DESCRIBED IN THIS DOCUMENT AT ANY TIME.

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 "READ ME" FILES.

This and other products of TIBCO Software Inc. may be covered by registered patents. Please refer to TIBCO's Virtual Patent Marking document (https://www.tibco.com/patents) for details.