|  |
| --- |
| Tibco Software Inc |
| BusinessEvents Channel API |
| Functional Specifications Document |

|  |  |  |
| --- | --- | --- |
| Author | Date | Summary of Change |
| Vandit Sharma | 16th August 2016 | Initial Document |

Contents

[1 Requirement 1](#_Toc459121438)

[2 Functional Specification 1](#_Toc459121439)

[2.1 Java API 1](#_Toc459121440)

[2.1.1 class com.tibco.be.custom.channel.BaseDriver 1](#_Toc459121441)

[2.1.2 class com.tibco.be.custom.channel.BaseChannel 1](#_Toc459121442)

[2.1.3 class com.tibco.be.custom.channel.BaseDestination 1](#_Toc459121443)

[2.1.4 interface com.tibco.be.custom.channel.Event 2](#_Toc459121444)

[2.1.5 interface com.tibco.be.custom.channel.EventWithId 2](#_Toc459121445)

[2.1.6 2](#_Toc459121446)

[2.1.7 class com.tibco.be.custom.channel.BaseEventSerializer 2](#_Toc459121447)

[2.2 Define a Channel Descriptor file called drivers.xml 2](#_Toc459121448)

[2.3 Studio Support 3](#_Toc459121449)

[2.4 Runtime Support 3](#_Toc459121450)

# Requirement

To provide a Channel Extensibility API that would allow users to build their own BE Channels.

# Functional Specification

This section lists the functional specifications for Custom Channels in BE

## Java API

A Java API is to be provided so that customers can develop their own custom channels. This section highlights the main components of the API. Detailed Javadocs for the API would be provided as part of the product.

The main classes/interfaces are:

### class com.tibco.be.custom.channel.BaseDriver

Users needs to implement the following methods exposed by the BaseDriver class.These methods will be called by BE runtime during startup for initializing Channel and Destination

* getChannel
* getDestination

### class com.tibco.be.custom.channel.BaseChannel

Users needs to implement the following methods exposed by the BaseChannel class.These methods will be called by BE runtime during startup, connecting and closing of channels.

* init

The following methods have default implementations , but still user can override these methods and provide channel specific implementation logic.

* connect
* close
* start

### class com.tibco.be.custom.channel.BaseDestination

Users needs to implement the following methods exposed by the BaseDestination class.These methods will be called by BE runtime during binding,starting,suspending ,resuming the destination .

* init
* connect
* start
* close
* send
* requestEvent

### interface com.tibco.be.custom.channel.Event

The Event interface is required to be implemented by the user and will signify the communication unit which will hold the event related data.User will have to provide implementation for the getters and setters for the following members :

* extId - An Id to uniquely identify the event.
* destinationUri - The URI of the destination
* eventUri - the BE event URI
* payload - a byte array to hold event payload
* eventProps - the event properties

### interface com.tibco.be.custom.channel.EventWithId

This interface extends the Event interface.This is same as an Event with the exception that it exposes the BE’s internal event Id.If implemented , in addition to Event’s methods, user must provide implementation ***getId***which would return the BE’s internal id.

**Note** : A default implementation of an Event is provided in the form of the class : *com.tibco.be.custom.channel.DefaultEventImpl* .User can use/extend this class as needed.

### 

### class com.tibco.be.custom.channel.BaseEventSerializer

The user defined implementation for a serializer should be given by extending the BaseEventSerializer class.User must provide implementation for the following methods which will used to serialize and deserialize the UserEvent.

* initUserEventSerializer
* serializeUserEvent
* deserializeUserEvent

-The qualified path to the serializer must be specified in the *drivers.xml*

### interface com.tibco.be.custom.channel.EventContext

The EventContext interface enables the user to provide implementation for acknowledging or replying to a message.It represents the context in which the message was received which led to the creation of the Event. User will have to provide implementation for the following methods:

* reply
* acknowledge
* rollback
* getDestination
* getMessage

### class com.tibco.be.custom.channel.DefaultEventContext

When user does not provide implementation for a custom EventContext this will be used as a default context.This is just a hollow class without any implementation.

## Define a Channel Descriptor file called drivers.xml

Create a drivers.xml channel descriptor file as shown below and add it to the custom channel JAR file.

<?xml version="1.0" encoding="UTF-8"?>

<drivers>

<driver>

<type>CHANNEL\_TYPE</type>

<label>CHANNEL\_LABEL</label>

<class>DRIVER\_CLASS </class>

<description>CHANNEL\_DESCRIPTION</description>

<version>CHANNEL\_VERSION</version>

<properties>

<!--Channel Properties -->

</properties>

<destinations>

<!--Destination Properties -->

</destinations>

<serializers userdefined="true">

<serializer type="SERIALIZER\_TYPE" class="SERIALIZER\_CLASS"/>

<!--Additional Serializers -->

</serializers>

</driver>

</drivers>

Where,

DRIVER\_CLASS : User implemented class URI for BaseDriver

CHANNEL\_TYPE: User defined channel type (must be unique).

CHANNEL\_LABEL: The display label for the channel.

CHANNEL\_DESCRIPTION: Description for the channel.

CHANNEL\_VERSION: Version for the channel. Maintained by the user.

SERIALIZER\_TYPE: User defined type of the serializer.There can be multiple serializers.

SERIALIZER\_CLASS: The implementation class for the user defined serializer.Users must implement at least one serializer.

## Studio Support

After placing the custom channel JAR file in BE classpath, it should be possible to configure the channel via Studio. This includes Channel and Destination configuration as you would for any other channel using a Properties configurator.

\*Configuration via “Shared Resources” will not be supported for Custom Channel.

Global Variable and BE Properties support would be provided for Channel and Destination configurations.

There would be no Global Variable toggle. A value of the form %%val%% would have to be specified where val is defined as a Global Variable.

The preferred location for custom channel JAR files would be *$BE-HOME/lib/ext/tpcl/*

## Runtime Support

As an example that demonstrates the use of the Channel API, a KafkaChannel example will be shipped along with instructions to run the example in readme.html of the example channel.