EBI External Notifications - Windows

IMPORTANT:

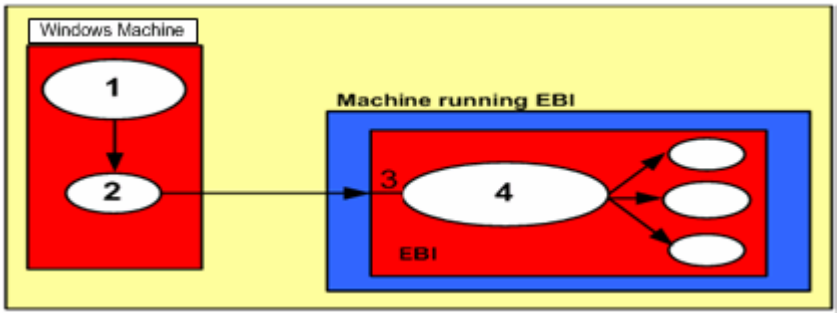
To be able to perform the following process, Java 1.5 **must** be installed on the Windows machine calling the notification program.

This technical memo describes the steps required to allow EBI to receive notifications from a batch file on Windows, as well as making recommendations on its implementation. This document assumes the reader is familiar with Java, Windows, and batch files

Contents

Overview

A batch program is launched, which in turn calls a Java Method. The batch program can be launched manually or using a trigger, this example launches the program manually. The Java Method then uses the EBI Notification API to send a notification to EBI, along with additional properties.



Key
1. Batch program calls java method
2. Java method sends notification to EBI
3. EBI receives notification
4. EBI Event Manager interprets the event and launches associated processes

The remainder of this document outlines the steps required to allow EBI to receive notifications from a batch program on Windows. These steps are:

- **Add Notification Event and Properties**
- **Save and extract the notification.zip file**
- **Create Java Notification Method**
- **SendEbiItmNotif.java**
- **jndi.properties**
- **Create batch program**

Add Notification Event and Properties

The first step is to add a Notification Event and Properties in **EBI Event Manager**.
To add a new event, navigate to the **EBI Event Manager** window. In the **Events** pane, right-click and, when prompted, provide a name for the event. (This example uses the event **notification.api**)
Properties can then be added to the event beyond the default **payload** property. This is done by right-click the event and selecting **Add Property**.

Note: The name of the event (and any properties) must match the name (and properties) provided in the program that sends the notification.

After an event and its associated properties have been defined, a business process can be associated with the event. This business process may contain passed-in variables – these are mapped from the event and then used in the process.
All of the parameters do not need to be used, and more than one business process may be mapped to a single event. The example batch program **EbiNotificationAPI.bat** does not use any event properties. For this example create a business process script that sends an email to yourself. Then attach the business process script to the **notification.api** event.

Save and extract the notification.zip file

First, save the **notification.zip** file sent with this document to c:\ directory. Next, create a directory named **ebi_notification** in the c:\ directory to hold the zip file.
Then, move the **notification.zip** file into the **c:\ebi_notification** directory
Lastly, unzip the contents of the **notification.zip** file into the **c:\ebi_notification** directory.

After extraction, the following objects should exist within the **ebi_notification** directory:

- **c:\ebi_notification\notification\EbiNotificationAPI.bat**
  This is the batch program used in this example.
- **c:\ebi_notification\notification\SendEbiItmNotif.java**
  This is a java program that sends the notification to EBI. Section will follow in this document explaining other changes/modifications that will need to be done to this object.
- **c:\ebi_notification\notification\conf\jndi.properties**
  This is a properties file that points to the server hosting EBI. Section will follow in this document explaining other changes/modifications that will need to be done to this object.
- **c:\ebi_notification\notification\lib**
  This is a directory that contains .jar files needed to support this process. The user should not modify any of the objects in this directory. The following .jar files should exist in this directory:
    - dom4j.jar
    - jboss-j2ee.jar
    - EBI.jar
    - jbossall-client.jar
    - log4j.jar

Create Java Notification Method - SendEbiItmNotif.java

The next step is to create a Java Notification Method for the Notification Event.
This is a java program located in the c:\ebi_notification\notification directory. This program should not have to be modified by the user. The version that we supply should be sufficient for your purpose. However, you will need to compile this program, and can do so by issuing the following command:

***Note:** You will need jdk1.5 or greater installed on the machine running the batch program.
Browse to your c:\program files\java directory and see if you have a JDK folder. If you have a JDK folder then proceed to the next step. If you do not have JDK installed it can be download at http://java.sun.com/javase/downloads/index.jsp

To compile the java program enter the following commands, replacing [jdk install] with your jdk directory installed:

set path=c:\program files\java\[jdk install]\bin

```
javac –classpath .;c:\ebi_notification\notification\lib\EBI.jar SendEbiItmNotif.java
```

Once compiled you should now see a **SendEbiItmNotif.class** file in the **c:\ebi_notification\notification** directory.

Note: If the example that we provide is not sufficient, the customer can use it to create their own java program. If they do so, they can replace the "SendEbiItmNotif.java" name from the `javac` command above with the name of the program that they create. Also if they go this route, any other java archives (.jar files) that their program requires will need to be included in the compilation classpath.

The zip file contains a Java Doc along with an example java program for sending a notification.

The following is a Simple Send Item Notification class example:

```
/* EXTOL International, all rights reserved
* Author: tfrance
* Date: 10/2003
*
* Modifications:
* Developer :
* Date : [YYYY/MM/DD]
* Description:
*
* Developer :
* Date : [YYYY/MM/DD]
* Description:
*
*/
import com.extol.eventmanager.NotificationType;
import com.extol.eventmanager.NotificationService;
import com.extol.eventmanager.Notification;
import com.extol.eventmanager.NotificationException;
import com.extol.eventmanager.NotificationProperty;
/**
* Sample code that demonstrates how a client can use the EBI Notification API to send
and receive notifications.
*/
public class SendEbiItmNotif {
NotificationService service;
NotificationType type;
Notification notification;
public SendEbiItmNotif(String formattedTypeName) {
System.out.println("...constructing the SendEbiItemNotif object...");
```

```
if (service == null) service = new NotificationService();
type = service.createNotificationType(formattedTypeName);
notification = service.createNotification(type);
}
public static void sendItemNotification(String formattedTypeName, String
itemKeyValue, String notificationMessage,
Long sequenceNumber) {
if (formattedTypeName != null) {
System.out.println("formattedTypeLength (prior to trimming): " +
formattedTypeName.length());
formattedTypeName.trim();
formattedTypeName = formattedTypeName.trim();
System.out.println("formattedTypeLength (after to trimming): " +
formattedTypeName.length());
}
SendEbiItmNotif sender = new SendEbiItmNotif(formattedTypeName);
System.out.println("...decorating the notification with the item key value,
notification message," +
" and sequence number...");
if (notificationMessage != null && notificationMessage.length() > 0) {
sender.notification.setMessage(notificationMessage);
} else {
System.out.println("...notification message was either null or the length was
zero...");
}
sender.notification.setSequenceNumber(sequenceNumber.longValue());
if (itemKeyValue != null && itemKeyValue.length() > 0) {
// sender.notification.setUserData(new String[]{itemKeyValue});
NotificationProperty itemNumberProperty = new NotificationProperty("item_number",
"item number of the" +
" database row", 0);
itemNumberProperty.setValue(itemKeyValue);
sender.notification.addProperty(itemNumberProperty);
} else {
System.out.println("...item key value was either null or the lenght was zero...");
}
System.out.println("...sending the Notification object to EBI...");
try {
// Send the notification to EBI:
sender.service.sendNotification(sender.notification);
System.out.println("...sent successfully...");
}
catch (NotificationException ne) {
ne.printStackTrace();
System.out.println("Exception: " + ne.getMessage());
}
}
```

```
public static void testSend() {
SendEbiItmNotif.sendItemNotification("com.extol.test.itmphy01.insert","82131", "Test
notification", new Long(5));
}
public static void main(String[] args) {
// Assume the arguments are in the same order and type as the method
sendItemNotification, except the last arg
// which is a <code>String</code> that needs to be convereted into a Long
// Dump the arguments to System.out
if (args != null) {
System.out.println("Arguments:");
for (int q=0; q < args.length; q++) {
System.out.println(" " + q + ") " + args[q]);
}
}
String localFormattedName = null;
String localItmNumber = null;
String localNotificationMessage = "Triggered Notification on ITMPHY01";
String localSequenceNumber = null;
Long localSequenceNumberL = null;
try {
localFormattedName = args[0];
localItmNumber = args[1];
localSequenceNumber = args[2];
localSequenceNumberL = new Long(localSequenceNumber);
}
catch (Exception e) {
e.printStackTrace(); //To change body of catch statement use Options | File
Templates.
System.err.println("Error while attempting to extract arguments: " +
SendEbiItmNotif.class.getName());
StringBuffer errorBuffer = new StringBuffer();
errorBuffer.append("Expected:\n1)formattedTypeName (e.g.
com.extol.test.itmphy01.insert\n2) itemKeyValue (e.g " +
"12345)\n3) sequenceNumber (e.g. 123)");
}
try {
SendEbiItmNotif.sendItemNotification(localFormattedName, localItmNumber,
localNotificationMessage, localSequenceNumberL);
}
catch (Exception e) {
e.printStackTrace();
}
}
}
```

jndi.properties

jndi.properties contents:

```
java.naming.factory.initial=org.jnp.interfaces.NamingContextFactory
java.naming.factory.url.pkgs=org.jboss.naming:org.jnp.interfaces
java.naming.provider.url=jnp://EBIHOSTNAME:1099
```

jndi.properties must be in the working directory of the notification class.
EBIHOSTNAME must be set to the name or IP address of the machine running EBI.
In the **c:\ebi_notification\notification\conf\jndi.properties** file, change the value of the "java.naming.provider.url" property to reflect the name/IP address of the server that hosts EBI within your system. The default value of this property is "jnp://localhost:1099". You will need to change the "localhost" portion to point to the server that hosts EBI.

Create batch program

The last step is to create a batch program to call the Java Notification method.

Once the program is completed, it needs to be called in some manner. The EbiNotificationAPI.bat program is included in the zip file, you can call it to test the EBI notification API. You will need to modify the batch program to point to your installed version of java.

SET JAVA_HOME=C:\Program Files\Java\jdk1.6.0_02\bin