IBM WEBSPHERE BUSINESS MONITOR 6.0.2 – LAB EXERCISE

Monitoring using the sample file emitter

What this exercise is about

This exercise shows you how to use IBM® WebSphere® Integration Developer to update the File Emitter source code to customize it to emit specific user defined events using the CEI SDK that will be monitored using WebSphere Business Monitor. You will customize the supplied File emitter source code, export the EAR, and deploy the EAR to WebSphere Process Server 6.0.2. You will configure WebSphere Process Server 6.0.2 scheduler. Finally you will conduct a simple test to see if the events are being emitted.

Lab requirements

List of system and software required for the student to complete the lab.

- WebSphere Integration Developer V6.0.2
- WebSphere Process Server V6.0.2 or WebSphere Application Server 6.0.2 with CEI API SDK

What you should be able to do

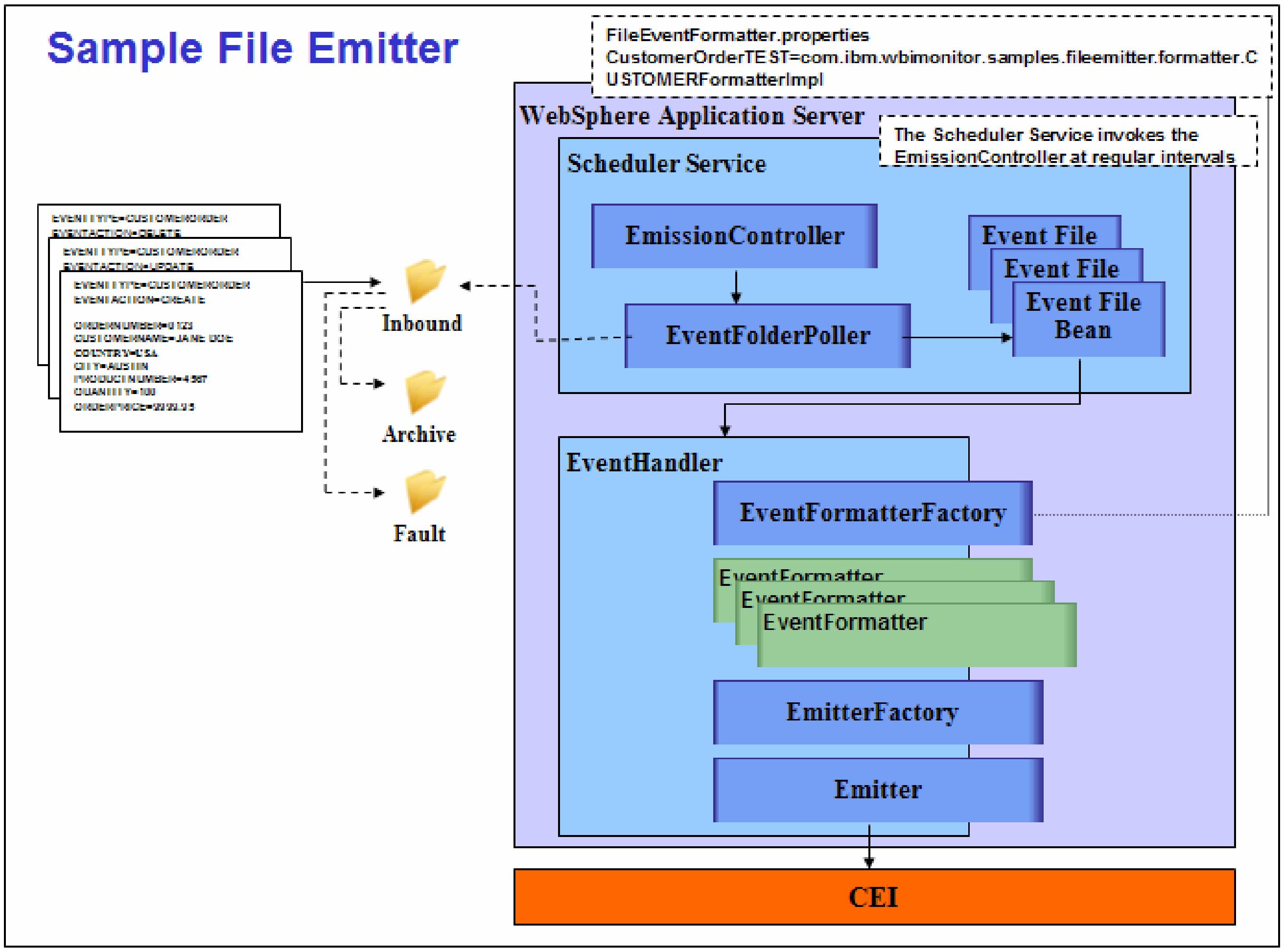
At the end of this lab you should be able to:

- Use WebSphere Integration Developer to import the DB2 Emitter source with CEI SDK emitter transport and perform necessary modifications to it, build the File Emitter project and finally export an EAR file for deployment
- Use the Cview tool to create a database for the scheduler database and configure a data source for the same using the Process Server's administrative console
- Use the WebSphere Process Server administrative console to create a scheduler service
- Use the WebSphere Process Server administrative console to install the EAR
- Run a simple test to view successful events emitted
- View the events emitted using the common base event browser (CBE Browser)

Introduction

The Sample File Event Emitter is a sample program written in Java™ that demonstrates how an enterprise information system (EIS) resource which stores data pertaining to the state of a business can be instrumented to contribute to the overall monitoring of the activities of a business.

The main goal of the Sample File Event Emitter is to introduce the use of the libraries and APIs provided by the Common Event Infrastructure (CEI) to generate and emit business events in the form of Common Base Events (CBEs). Common Base Events are the data packaging and format used by the WebSphere Business Monitor (WBM) Server to propagate business events.



Exercise instructions

Some instructions in this lab may be Windows® operating-system specific. If you plan on running the lab on an operating-system other than Windows, you will need to run the appropriate commands, and use appropriate files (.sh vs. .bat) for your operating system. The directory locations are specified in the lab instructions using symbolic references, as follows:

Reference variable	Windows location	AIX®/UNIX® location
<WID_HOME>	C:\WID602	
<WPS_HOME>	C:\IBM\WebSphere\ProcServer	
<WPS_PROFILE_DIR>	<WPS_HOME>\profiles\<profile_name>	
<LABFILES>	C:\Labfiles602	/tmp/Labfiles602

Important: This LAB does not cover the Monitor Model deployment to the WebSphere Business Monitor Server. You should have first hand experience deploying and running a Monitor Model on the Monitor Server.

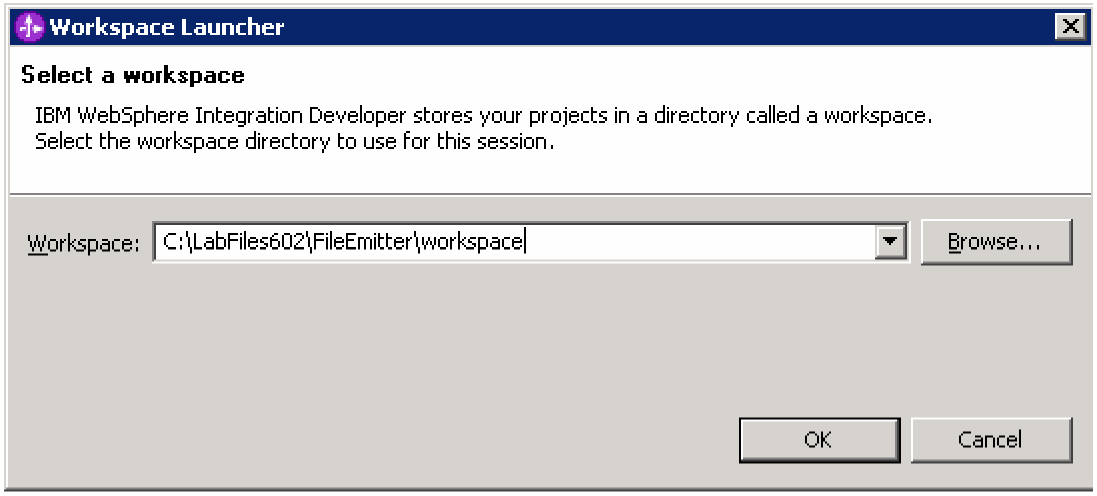
Important: You can either use WebSphere Process Server 6.0.2 (at times designated as Process Server) or a WebSphere Application Server 6.1 with CEI API SDK installed. This LAB uses WebSphere Process Server 6.0.2.

Important: The source code modification is a demonstration on how the events emitted will be correlated with the Monitor Model. Each event to be consumed by WebSphere Business Monitor needs to be matched to the Monitor Model.

Part 1: Modify Code to handle CustomerOrder event using the CEI API SDK and export the FileEmitter application (EAR)

____ 1. Start WebSphere Integration Developer, creating a new workspace in the folder **C:\ C:\LabFiles602\FileEmitter\workspace**, and turn off the auto-build feature

__ a. The workspace Launcher window will be displayed. Click the **Browse…** button and select your workspace directory.



Workspace Launcher

Select a workspace

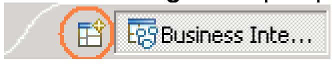
IBM WebSphere Integration Developer stores your projects in a directory called a workspace.
Select the workspace directory to use for this session.

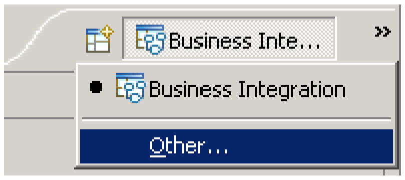
Workspace: C:\LabFiles602\FileEmitter\workspace

__ b. Click **OK**

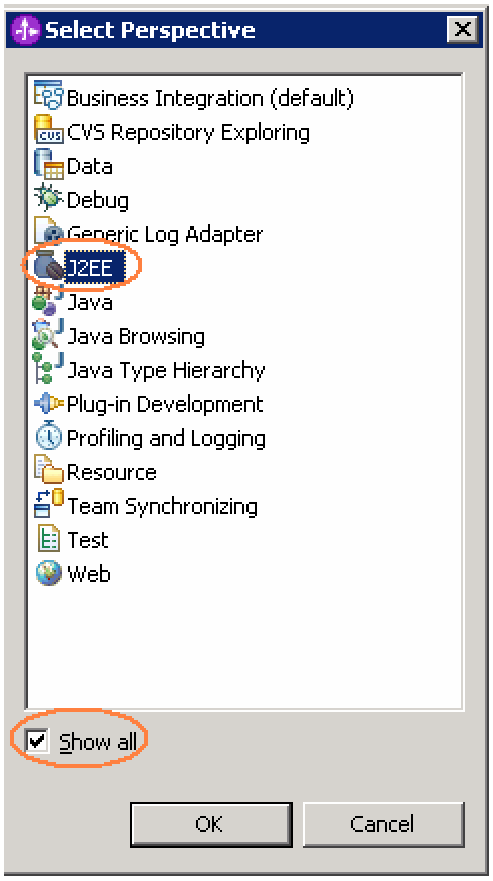
__ c. Close the welcome window by clicking the arrow in the top right corner of the welcome window

____ 2. Switch to the J2EE perspective and turn off automatic builds

__ a. By default WebSphere Integration Developer opens in **Business Integration** perspective. You need to change it to **J2EE** perspective. To do this click  on the top right corner of the WebSphere Integration Developer and choose **Other** if **J2EE** is not listed here.

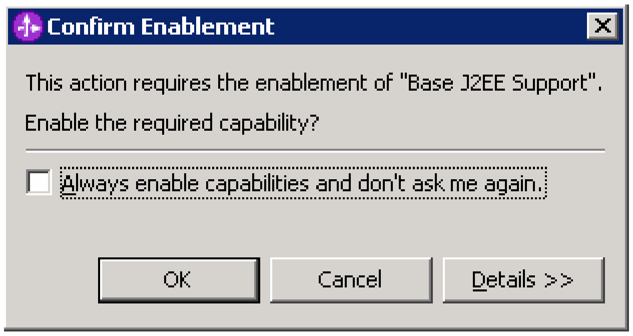


__ b. From the **Select Perspective** dialog select **J2EE** and click **OK**



Note: Select the **Show all** check box if **J2EE** perspective does not show up.

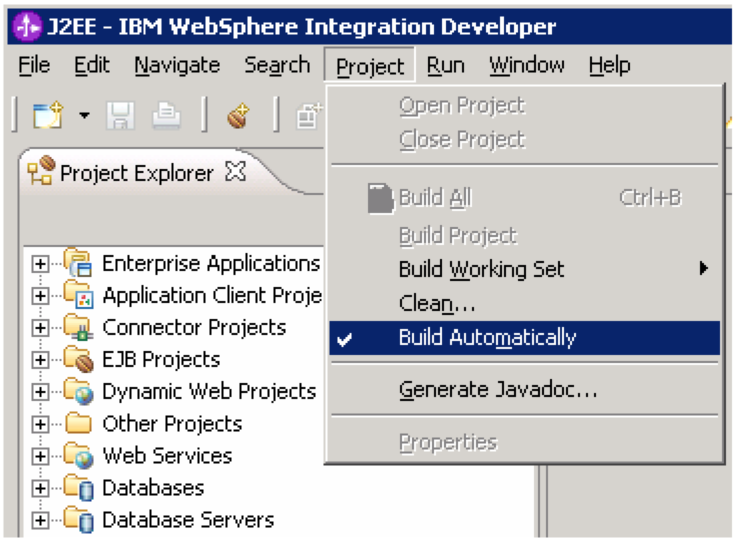
__ c. Click **OK** on the confirm enablement warning.



This action requires the enablement of "Base J2EE Support".

Enable the required capability?

__ d. Select **Project > Build Automatically**, to turn off automatic builds.



____ 3. Import the **FileEmitter** Projects

In this step, import all File Emitter projects; make necessary build configurations based on the required emitter transport (CEI API). Also import the necessary jars to support CEI API. Following is a list of the projects that will be imported (part of the Project Interchange **FileEmitter.zip**):

- **FileEmitterEAR:** Project to package artifacts into EAR
- **EmitterFW:** Common emitter framework source
- **CEIEmitter :** CEI emitter source
- **FileEmitterEJB:** File Emitter specific source
- **FileEmitterImpl:** File Emitter specific source. Implementation classes specific to retrieving and formatting events

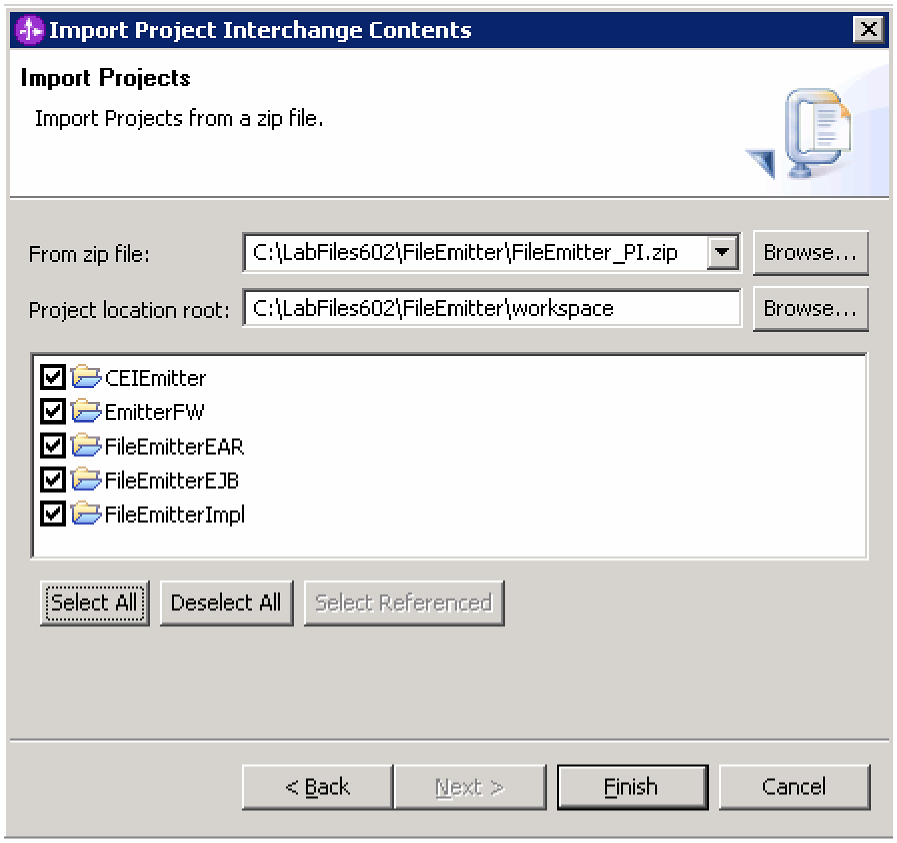
__ a. Select **File > Import** from the main menu.

__ b. From the **Import** window, select **Project Interchange** as the source and click **Next**

__ c. The **Import Project Interchange Contents** window will be opened.

__ d. Now click the first **Browse…** button and select **<LABFILES>\FileEmitter\FileEmitter_PI.zip** as the source archive file

Ex: C:\LabFiles602\FileEmitter\FileEmitter.zip



Import Project Interchange Contents

Import Projects

Import Projects from a zip file.

From zip file: C:\LabFiles602\FileEmitter\FileEmitter_PI.zip

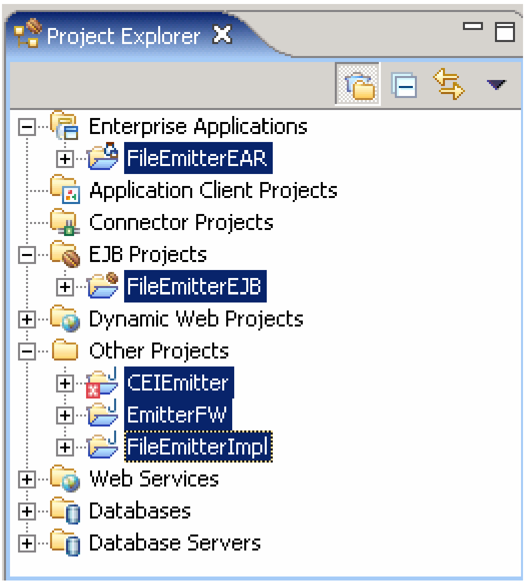
Project location root: C:\LabFiles602\FileEmitter\workspace

- [x] CEIEmitter
- [x] EmitterFW
- [x] FileEmitterEAR
- [x] FileEmitterEJB
- [x] FileEmitterImpl

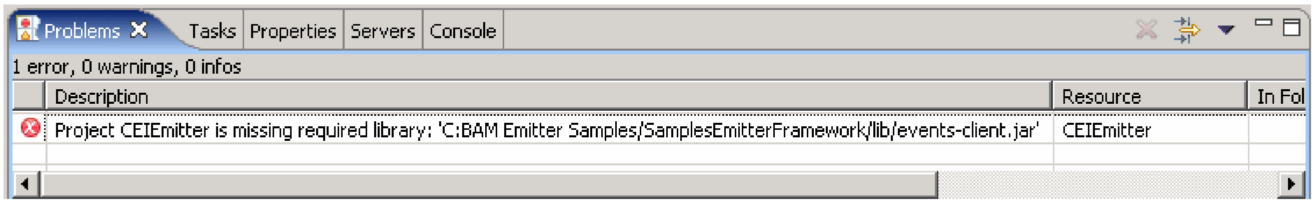
__ e. Select all the projects and then click **Finish**

____ 4. Review the imported artifacts

__ a. In the Project Explorer, expand **Enterprise Applications, EJB Projects** and **Other Projects** to view the projects as shown below



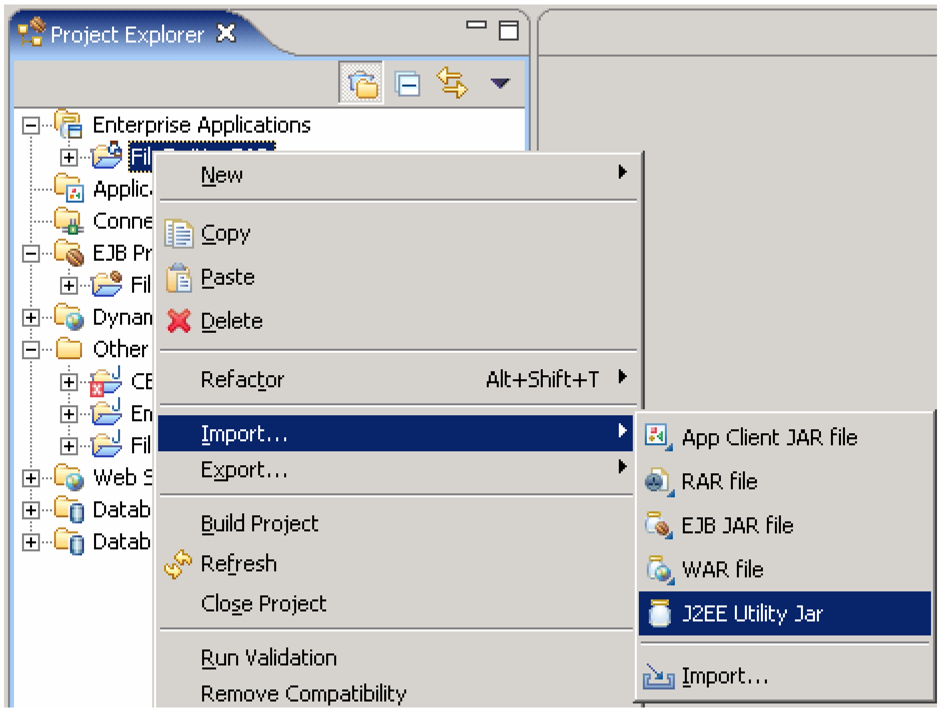
__ b. At this time there will be some errors noticed. Click on the **Problems** tab to view the problems as shown below


1 error, 0 warnings, 0 infos

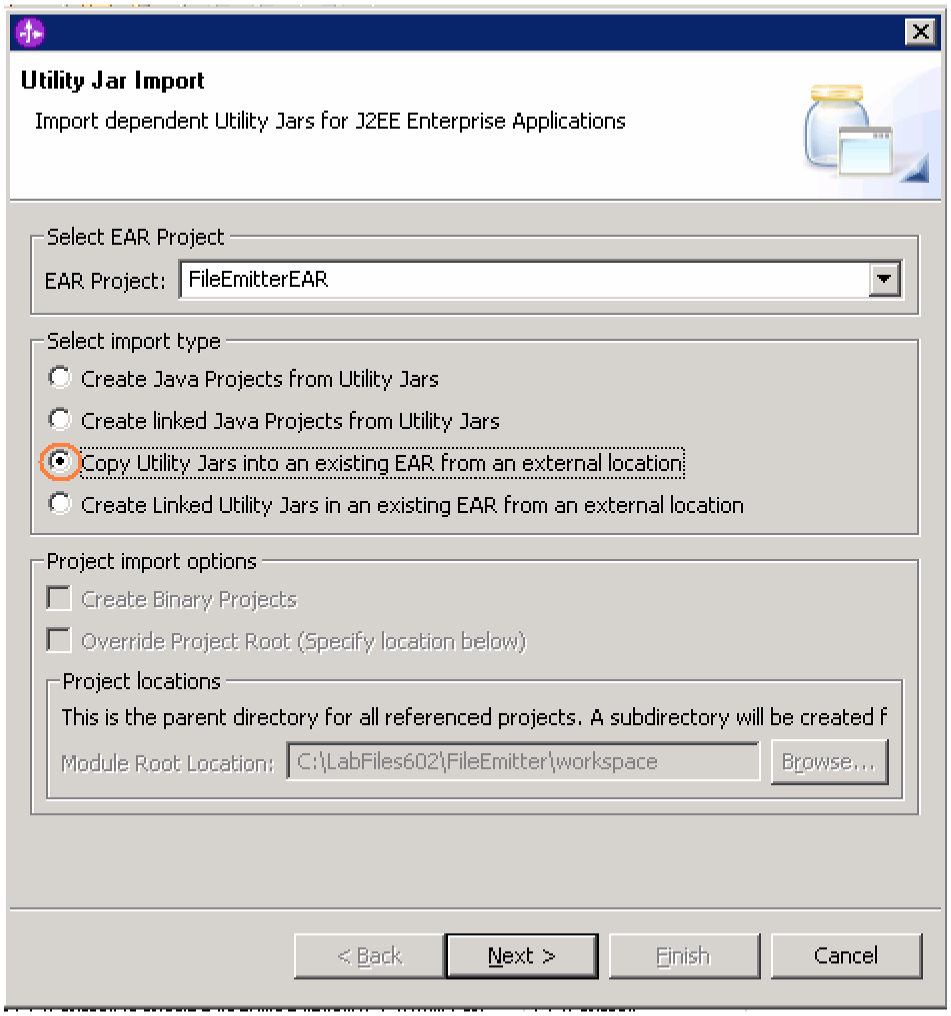
Description	Resource	In Fol
Project CEIEmitter is missing required library: 'C:BAM Emitter Samples/SamplesEmitterFramework/lib/events-client.jar'	CEIEmitter	

____ 5. Working towards avoiding the build path errors

__ a. It clearly states that the CEIEmitter project is missing a required library **events-client.jar**. Import it as J2EE Utility JAR by right-clicking on the **FileEmitterEAR** project from the project explorer as shown below



__ b. In the **Utility Jar Import** dialog select the type of Import as **Copy Utility Jars into an existing EAR from an external location** as shown below



Utility Jar Import

Import dependent Utility Jars for J2EE Enterprise Applications

Select EAR Project

EAR Project: FileEmitterEAR

Select import type

- ○ Create Java Projects from Utility Jars
- ○ Create linked Java Projects from Utility Jars
- ● Copy Utility Jars into an existing EAR from an external location
- ○ Create Linked Utility Jars in an existing EAR from an external location

Project import options

- ☐ Create Binary Projects
- ☐ Override Project Root (Specify location below)

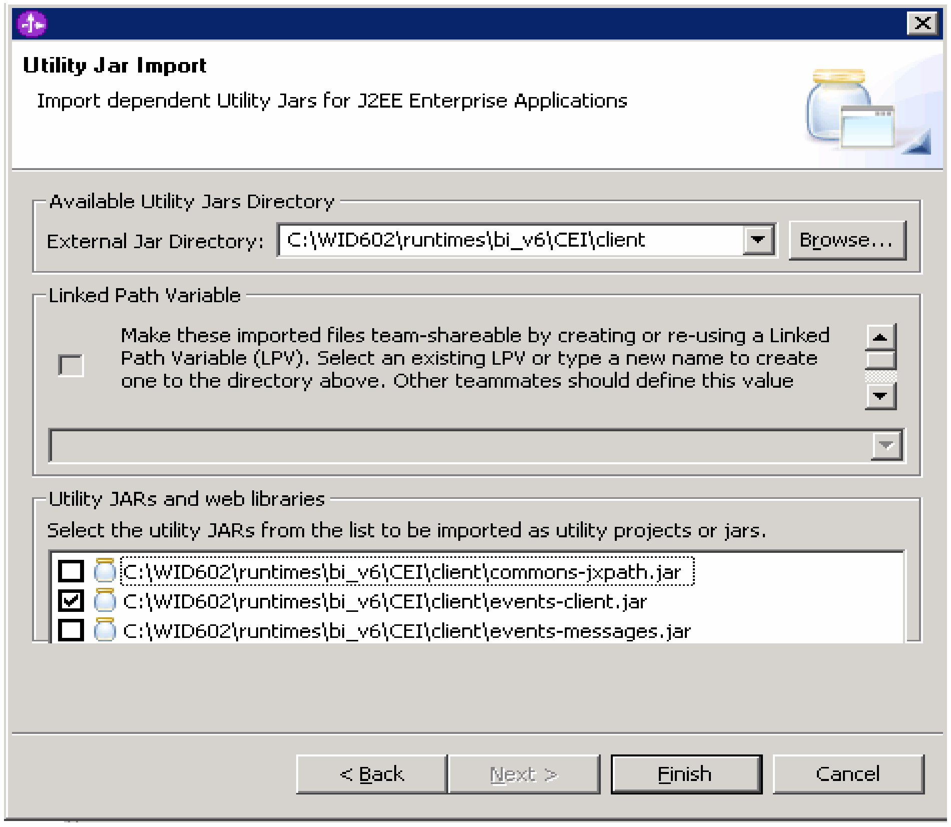
Project locations

This is the parent directory for all referenced projects. A subdirectory will be created f

Module Root Location: C:\LabFiles602\FileEmitter\workspace

__ c. Click **Next**

__ d. In the next dialog click **Browse** to locate the **events-client.jar** and make sure you select the jar as shown below



Utility Jar Import

Import dependent Utility Jars for J2EE Enterprise Applications

Available Utility Jars Directory

External Jar Directory: C:\WID602\runtimes\bi_v6\CEI\client

Linked Path Variable

☐ Make these imported files team-shareable by creating or re-using a Linked Path Variable (LPV). Select an existing LPV or type a new name to create one to the directory above. Other teammates should define this value

Utility JARs and web libraries

Select the utility JARs from the list to be imported as utility projects or jars.

☐ C:\WID602\runtimes\bi_v6\CEI\client\commons-jxpath.jar
☑ C:\WID602\runtimes\bi_v6\CEI\client\events-client.jar
☐ C:\WID602\runtimes\bi_v6\CEI\client\events-messages.jar

Note: You can find these libraries in WebSphere Process Server or WebSphere Integration Developer as follows:

<WID_HOME>\runtimes\bi_v6\CEI\client or

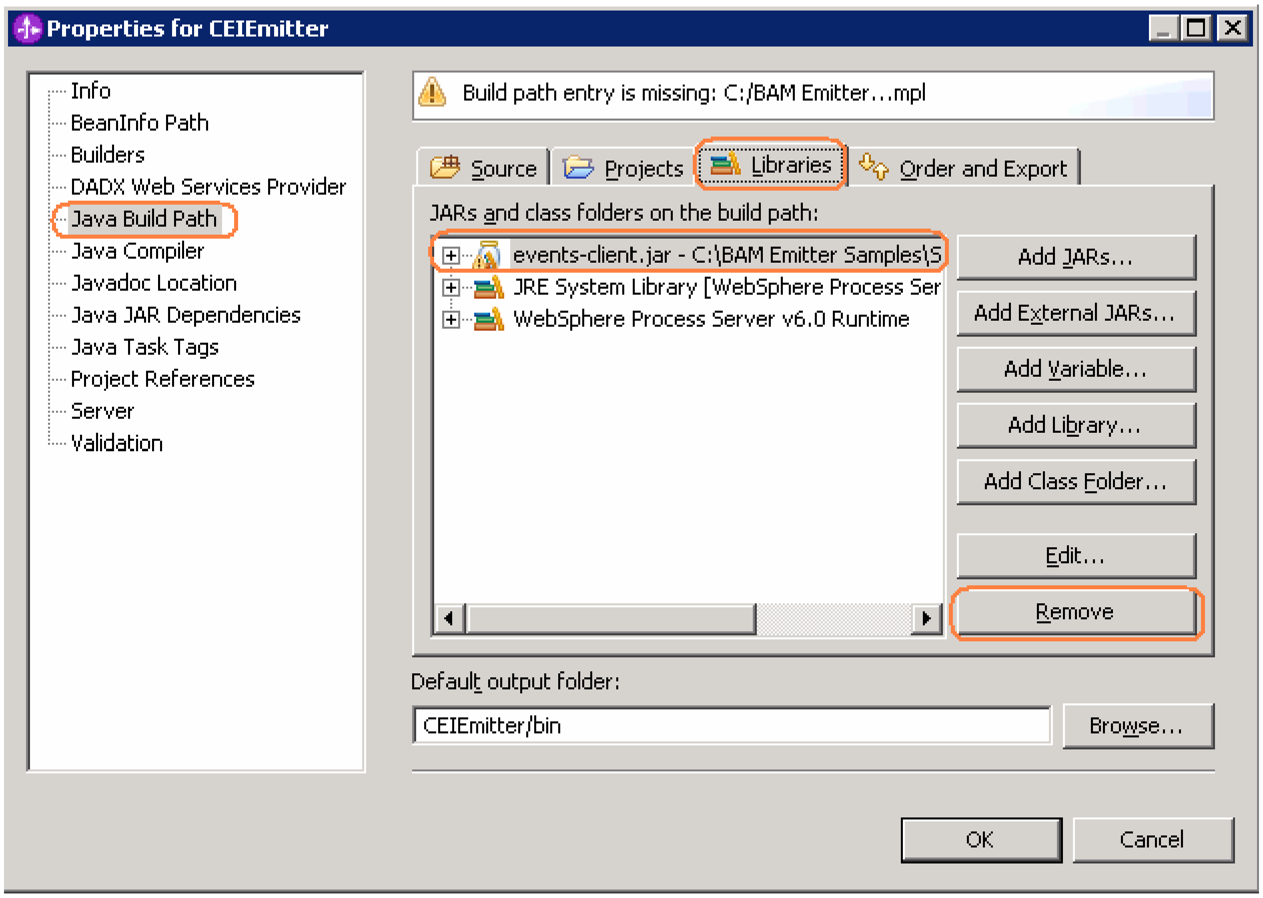
<WPS_HOME>\CEI\client

If CEI SDK is not installed on your machine local to the WebSphere Integration Developer, copy the necessary libraries from a remote WebSphere Process Server installation.

__ e. Click **Finish**

__ f. In the Project Explorer, right-click on the CEIEmitter project under **Other Projects** and select properties

__ g. In the **Properties for CEIEmitter** window, select Java Build Path in the left frame, select Libraries tab in the right frame


Properties for CEIEmitter

Build path entry is missing: C:/BAM Emitter…mpl

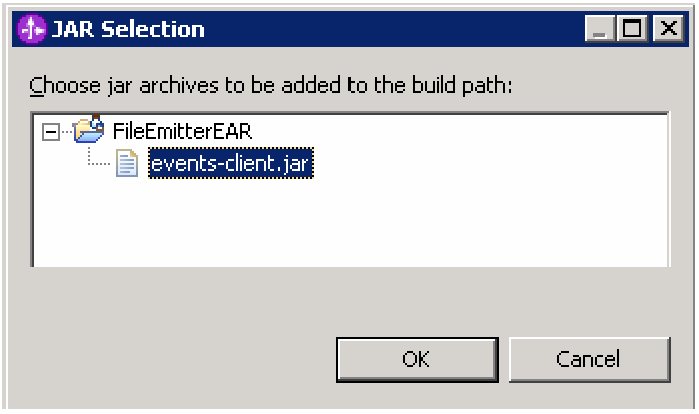
JARs and class folders on the build path:

- events-client.jar - C:\BAM Emitter Samples\S
- JRE System Library [WebSphere Process Ser
- WebSphere Process Server v6.0 Runtime

Default output folder: CEIEmitter/bin

__ h. Select the **events-client.jar** from the **JARs** listed and click the **Remove** button to remove build path.

__ i. Now click the **Add JARs** button and select the **events-client.jar** (FileEmitterEAR → events-client.jar)


JAR Selection

Choose jar archives to be added to the build path:

- FileEmitterEAR
  - events-client.jar

__ j. Click **OK** over the JAR Selection pop-up window

__ k. Now click **OK** over the **Properties for CEIEmitter** window

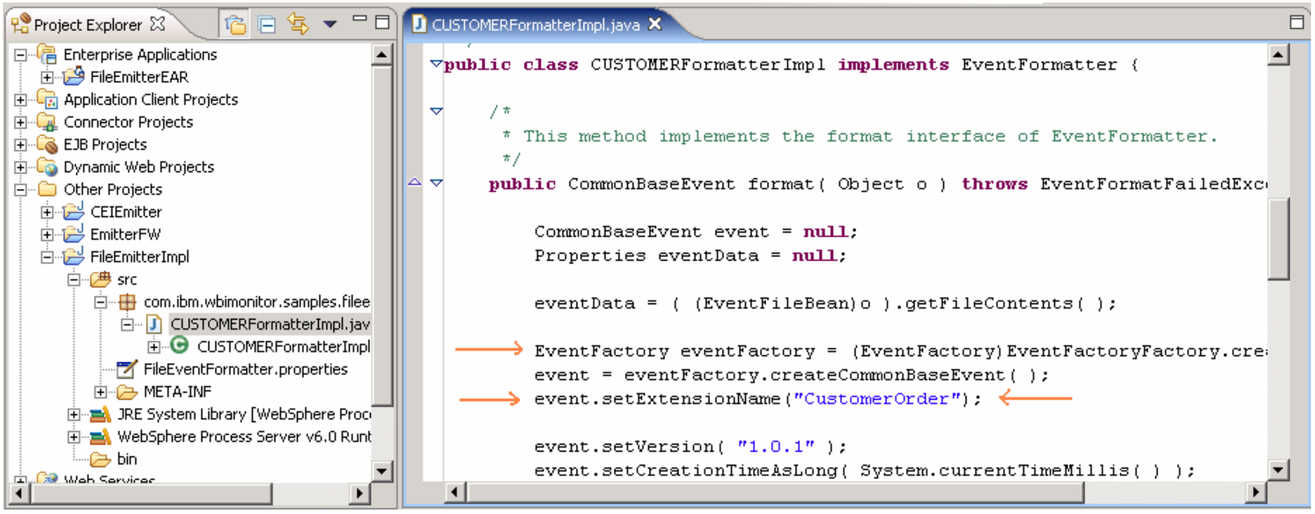
__ l. There must not be any errors reflected in the Problems tab at this time

____ 6. Now you will modify the File Emitter sample code **CUSTOMERFormatterImpl.java** which implements the EventFormatter class. The sample code that is provided is based on a CUSTOMER event, but you should also see how to modify it to handle a different event type, CustomerOrder.

Note: A text file named **CUSTOMERFormatterImpl.txt** with lines of code that are to be modified is under <LABFILES>\FileEmitter\src. It is easy to copy these lines of code and paste at the appropriate location.

__ a. In the Project Explorer expand **Other Projects > FileEmitterImpl > src > com.ibm.wbimonitor.samples.fileemitter.formatter** and double click on **CUSTOMERFormatterImpl.java**

__ b. Set an extension name for the CommonBaseEvent that is being returned. Add the line **event.setExtensionName("CustomerOrder");** after the Event factory is created


Project Explorer
Enterprise Applications
FileEmitterEAR
Application Client Projects
Connector Projects
EJB Projects
Dynamic Web Projects
Other Projects
CEIEmitter
EmitterFW
FileEmitterImpl
src
com.ibm.wbimonitor.samples.filee
CUSTOMERFormatterImpl.jav
CUSTOMERFormatterImpl
FileEventFormatter.properties
META-INF
JRE System Library [WebSphere Proc
WebSphere Process Server v6.0 Runt
bin
Web Services
CUSTOMERFormatterImpl.java

```
public class CUSTOMERFormatterImpl implements EventFormatter {

    /*
     * This method implements the format interface of EventFormatter.
     */
    public CommonBaseEvent format( Object o ) throws EventFormatFailedExc

        CommonBaseEvent event = null;
        Properties eventData = null;

        eventData = ( (EventFileBean)o ).getFileContents( );

        EventFactory eventFactory = (EventFactory)EventFactoryFactory.cre
        event = eventFactory.createCommonBaseEvent( );
        event.setExtensionName("CustomerOrder");

        event.setVersion( "1.0.1" );
        event.setCreationTimeAsLong( System.currentTimeMillis( ) );
```

__ c. Search for the following code segment

ExtendedDataElement id = event.addExtendedDataElement("ID");

id.setValuesAsInt(Integer.parseInt((String)eventData.getProperty("ID")));

And replace them with the following

ExtendedDataElement ordernumber = event.addExtendedDataElement("ORDERNUMBER");

ordernumber.setValuesAsInt(Integer.parseInt((String)eventData.getProperty("ORDERNUMBER")));

__ d. Search for the following code segment

ExtendedDataElement name = event.addExtendedDataElement("NAME");

name.setValuesAsString((String)eventData.getProperty("NAME"));

ExtendedDataElement address = event.addExtendedDataElement("ADDRESS");

address.setValuesAsString((String)eventData.getProperty("ADDRESS"));

ExtendedDataElement tel = event.addExtendedDataElement("TEL");

tel.setValuesAsString((String)eventData.getProperty("TEL"));

ExtendedDataElement email = event.addExtendedDataElement("EMAIL");

email.setValuesAsString((String)eventData.getProperty("EMAIL"));

And replace them with the following

ExtendedDataElement customername = event.addExtendedDataElement("CUSTOMERNAME");

customername.setValuesAsString((String)eventData.getProperty("CUSTOMERNAME"));

ExtendedDataElement country = event.addExtendedDataElement("COUNTRY");

country.setValuesAsString((String)eventData.getProperty("COUNTRY"));

ExtendedDataElement city = event.addExtendedDataElement("CITY");

city.setValuesAsString((String)eventData.getProperty("CITY"));

ExtendedDataElement productnumber = event.addExtendedDataElement("PRODUCTNUMBER");

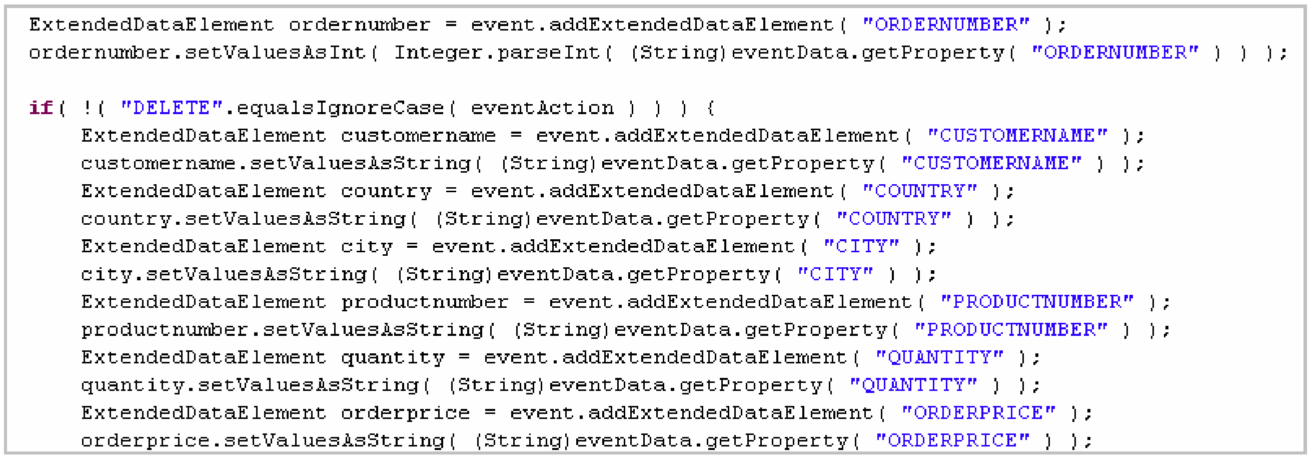
productnumber.setValuesAsString((String)eventData.getProperty("PRODUCTNUMBER"));

ExtendedDataElement quantity = event.addExtendedDataElement("QUANTITY");

quantity.setValuesAsString((String)eventData.getProperty("QUANTITY"));

ExtendedDataElement orderprice = event.addExtendedDataElement("ORDERPRICE");

orderprice.setValuesAsString((String)eventData.getProperty("ORDERPRICE"));



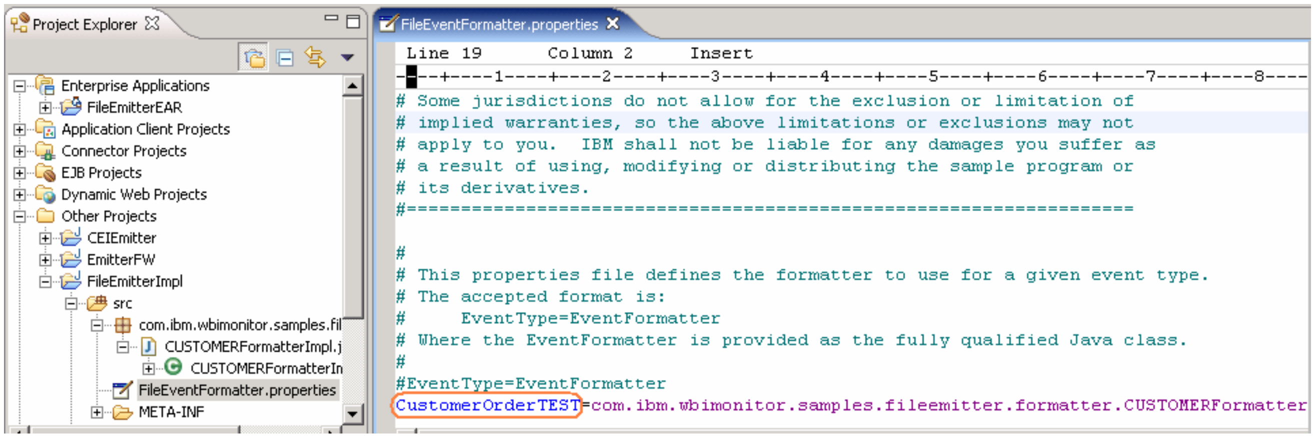
```
ExtendedDataElement ordernumber = event.addExtendedDataElement( "ORDERNUMBER" );
ordernumber.setValuesAsInt( Integer.parseInt( (String)eventData.getProperty( "ORDERNUMBER" ) ) );

if( !( "DELETE".equalsIgnoreCase( eventAction ) ) ) {
    ExtendedDataElement customername = event.addExtendedDataElement( "CUSTOMERNAME" );
    customername.setValuesAsString( (String)eventData.getProperty( "CUSTOMERNAME" ) );
    ExtendedDataElement country = event.addExtendedDataElement( "COUNTRY" );
    country.setValuesAsString( (String)eventData.getProperty( "COUNTRY" ) );
    ExtendedDataElement city = event.addExtendedDataElement( "CITY" );
    city.setValuesAsString( (String)eventData.getProperty( "CITY" ) );
    ExtendedDataElement productnumber = event.addExtendedDataElement( "PRODUCTNUMBER" );
    productnumber.setValuesAsString( (String)eventData.getProperty( "PRODUCTNUMBER" ) );
    ExtendedDataElement quantity = event.addExtendedDataElement( "QUANTITY" );
    quantity.setValuesAsString( (String)eventData.getProperty( "QUANTITY" ) );
    ExtendedDataElement orderprice = event.addExtendedDataElement( "ORDERPRICE" );
    orderprice.setValuesAsString( (String)eventData.getProperty( "ORDERPRICE" ) );
```

__ e. Save **(Ctrl+S)** and close the editor

__ f. Modify **FileEventFormatter.properties.** In the Project Explorer expand **Other Projects > FileEmitterImpl > src** and double click to open **FileEventFormatter.properties**

__ g. The relationship between an event type and the EventFormatter class for the type is specified in this property file; that is, Event Formatter registers to the Event Type **CustomerOrderTEST**. For example, specify the event type as **CustomerOrderTEST** to be more specific to the scenario.



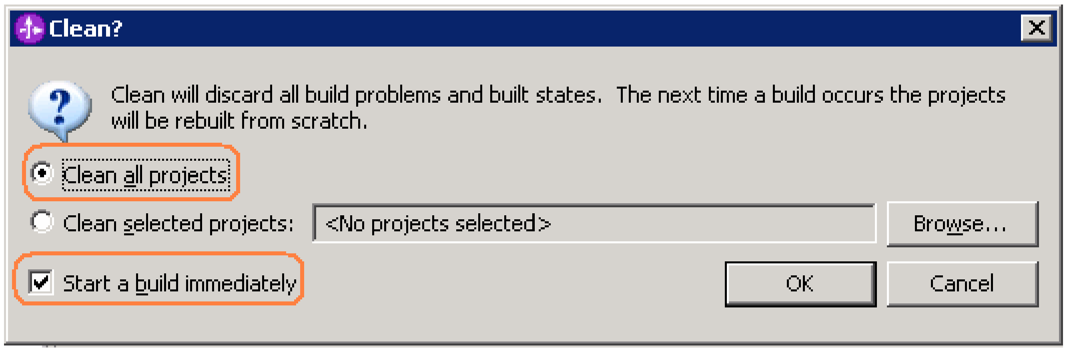
```
# Some jurisdictions do not allow for the exclusion or limitation of
# implied warranties, so the above limitations or exclusions may not
# apply to you.  IBM shall not be liable for any damages you suffer as
# a result of using, modifying or distributing the sample program or
# its derivatives.
#=================================================================

#
# This properties file defines the formatter to use for a given event type.
# The accepted format is:
#     EventType=EventFormatter
# Where the EventFormatter is provided as the fully qualified Java class.
#
#EventType=EventFormatter
CustomerOrderTEST=com.ibm.wbimonitor.samples.fileemitter.formatter.CUSTOMERFormatter
```

__ h. Modify the event type from **CUSTOMER** to **CustomerOrderTEST** and leave the EventFormatter class `com.ibm.wbimonitor.samples.fileemitter.formatter.CUSTOMERFormatterImpl` as it is

__ i. Save **(Ctrl+S)** and close the file

__ j. If workspace errors are not removed, it may be necessary to Refresh the projects and clean by clicking on the **Project** > **Clean** and then choose **"Clean All Projects"** in the dialog



Clean will discard all build problems and built states. The next time a build occurs the projects will be rebuilt from scratch.

[✓ Clean all projects] Clean selected projects: <No projects selected>

[✓ Start a build immediately]

__ k. Ensure that there are no errors

____ 7. Review the EJB Deployment Descriptor

__ a. In the Project Explorer, expand **EJB Projects > FileEmitterEJB** and double click on the **Deployment Descriptor : FileEmitterEJB** to open it in an editor

__ b. In the EJB Deployment Descriptor Editor, ensure that the **Source** tab is selected

__ c. The following are the default values specified in the EJB deployment descriptor. These can be customized by editing the source projects and exporting a new EAR archive.

`<env-entry-name>emitterFactoryJNDI</env-entry-name>`
<env-entry-value>iiop://localhost:2809/com/ibm/events/configuration/emitter/Default</env-entry-value>

Note: The default emitter factory specified for the "**emitterFactoryJNDI**" environment entry uses port 2809. Check your server's BOOTSTRAP_ADDRESS to ensure that is the correct port to use. Otherwise, import the source projects into WebSphere Integration Developer and make the necessary modifications to the EJB deployment descriptor.

```
<env-entry-name>schedulerJNDI</env-entry-name>
<env-entry-value>sched/FolderPoller</env-entry-value>
```

Note: The JNDI name for the scheduler - the "**schedulerJNDI**" environment entry - is **sched/FolderPoller**. Configure the Scheduler with this JNDI name.

```
<env-entry-name>inboundEventsDirectory</env-entry-name>
<env-entry-value>C:\LabFiles602\FileEmitter\inbound</env-entry-value>
```

Note: The "**inboundEventsDirectory**" environment entry **C:\LabFiles602\FileEmitter\inbound** is the file system directory where the events are expected to arrive.

```
<env-entry-name>archivedEventsDirectory</env-entry-name>
<env-entry-value>C:\LabFiles602\FileEmitter\archive</env-entry-value>
```

Note: The "**archivedEventsDirectory**" environment entry **C:\LabFiles602\FileEmitter\archive** is the file system directory where the successfully processed events are moved.

```
<env-entry-name>faultDirectory</env-entry-name>
<env-entry-value>C:\LabFiles602\FileEmitter\fault</env-entry-value>
```

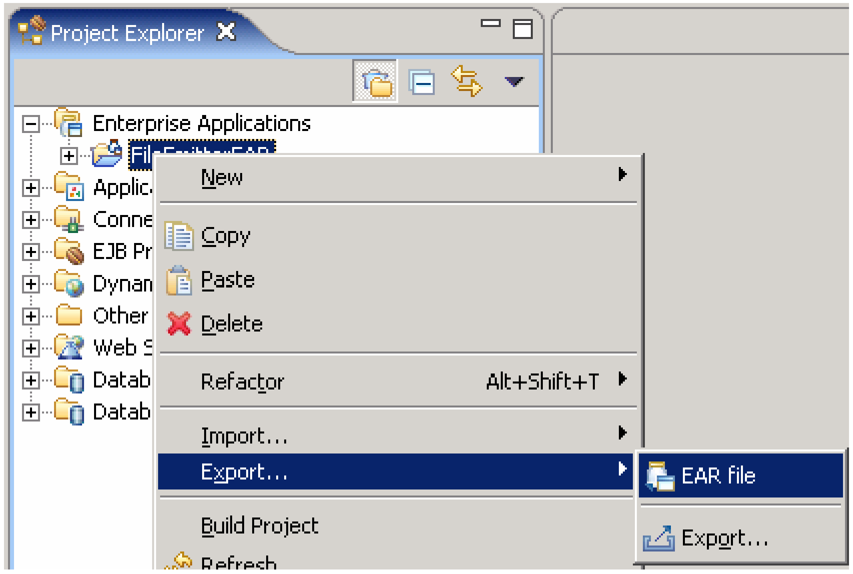
Note: The "**faultDirectory**" environment entry **C:\LabFiles602\FileEmitter\fault** is the file system directory where the events that encountered processing problems are moved.

__ d. Close the EJB Deployment Descriptor Editor.

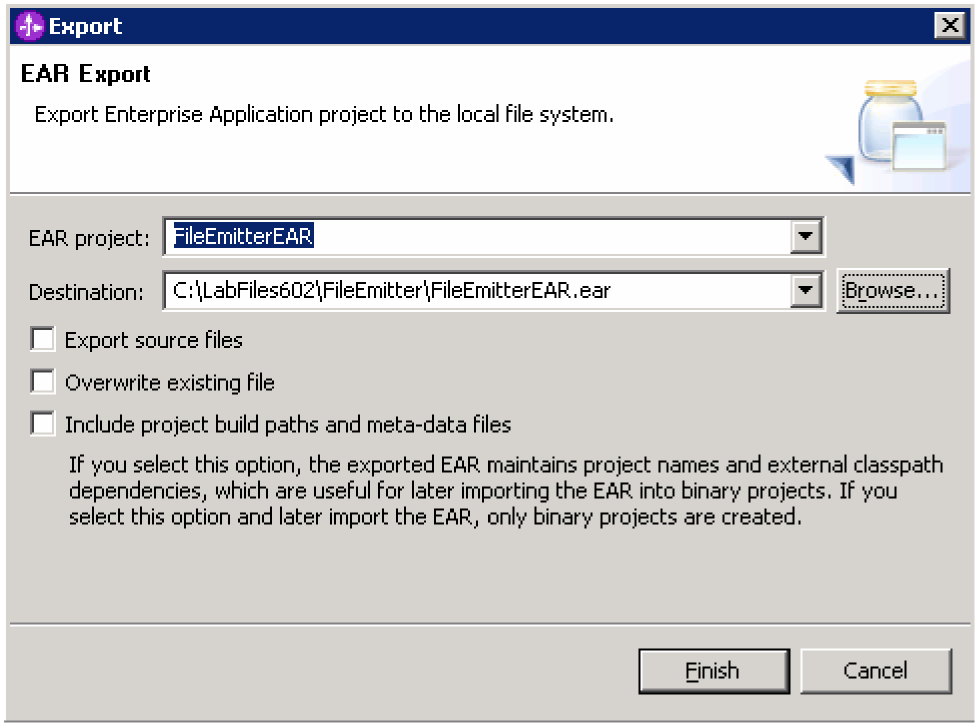
____ 8. Export the **FileEmitterEAR** ear file.

__ a. Select **FileEmitterEAR** project in the Enterprise Applications.

__ b. Right-click on the **FileEmitterEAR** project to open the context menu. Select **Export > EAR file**



__ c. In the **Export** dialog, select **FileEmitterEAR** for EAR Project and Browse to choose a Destination for the EAR



EAR Export

Export Enterprise Application project to the local file system.

EAR project: FileEmitterEAR

Destination: C:\LabFiles602\FileEmitter\FileEmitterEAR.ear

☐ Export source files

☐ Overwrite existing file

☐ Include project build paths and meta-data files

If you select this option, the exported EAR maintains project names and external classpath dependencies, which are useful for later importing the EAR into binary projects. If you select this option and later import the EAR, only binary projects are created.

__ d. Press **Finish** button

Part 2: Create file system for file emitter

In this part, a file system is created for the File Emitter. The file system includes an **inbound** directory where the events are expected to arrive, an **archive** directory where the successfully processed events are moved and a **fault** directory where the events that encountered processing problems are moved.

____ 1. Creating an **inbound** directory

Note: This inbound Events Directory structure must match the environment entry in the FileEmitterEJB deployment descriptor.

- In the local file system, explore to **C:\Labfiles602\FileEmitter** and create a directory named **inbound**

____ 2. Creating an **archive** directory

Note: This archived Events Directory structure must match the environment entry in the FileEmitterEJB deployment descriptor.

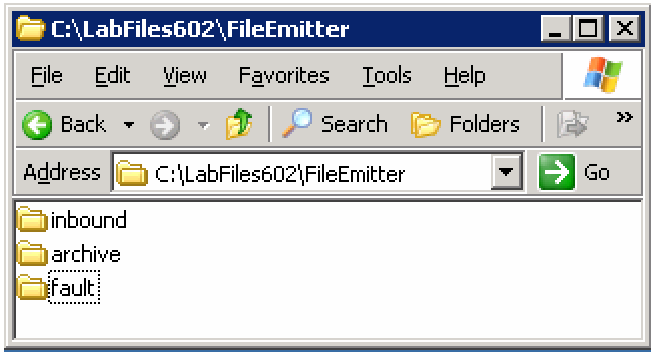
- In the local file system, explore to **C:\Labfiles602\FileEmitter** and create a directory named **archive**

____ 3. Creating a **fault** directory

Note: This fault Directory structure must match the environment entry in the FileEmitterEJB deployment descriptor.

- In the local file system, explore to **C:\Labfiles602\FileEmitter** and create a directory named **fault**

The file system created for the File Emitter must look as shown below:



Review the Event files:

The event files are located at **<LABFILES>\FileEmitter\FILEEVENTS** directory.

The first two lines are a mandatory

```
Ex:  EVENTTYPE=CustomerOrderTEST

     EVENTACTION=CREATE or EVENTACTION=UPDATE or EVENTACTION=DELETE
```

Part 3: Configure a scheduler service and deploy the FileEmitter EAR

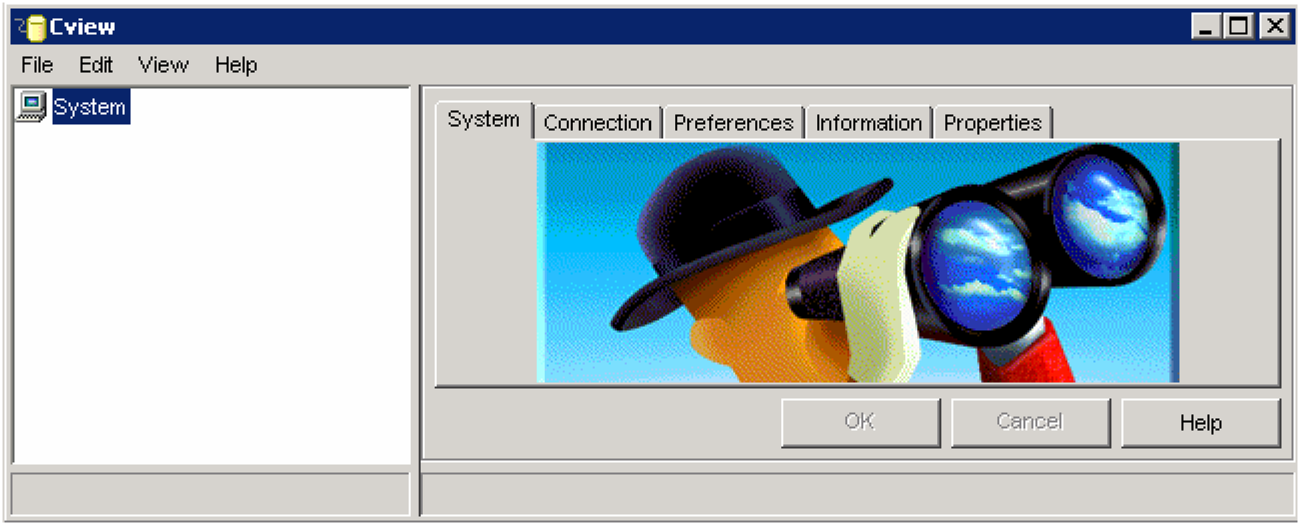
In this part, a Scheduler Service is created for the FileEmitter to invoke the EmissonController at a specified interval.

Follow the steps below to create a Cloudscape™ database for the scheduler service; configure a Data source for the same and finally a scheduler for the FileEmitter

____ 1. Create a Cloudscape database for the Scheduler Service

__ a. Launch cview.bat

__ b. Explore to <WPS_HOME>/cloudscape/bin/embedded and double-click **cview.bat** to launch the Cview window

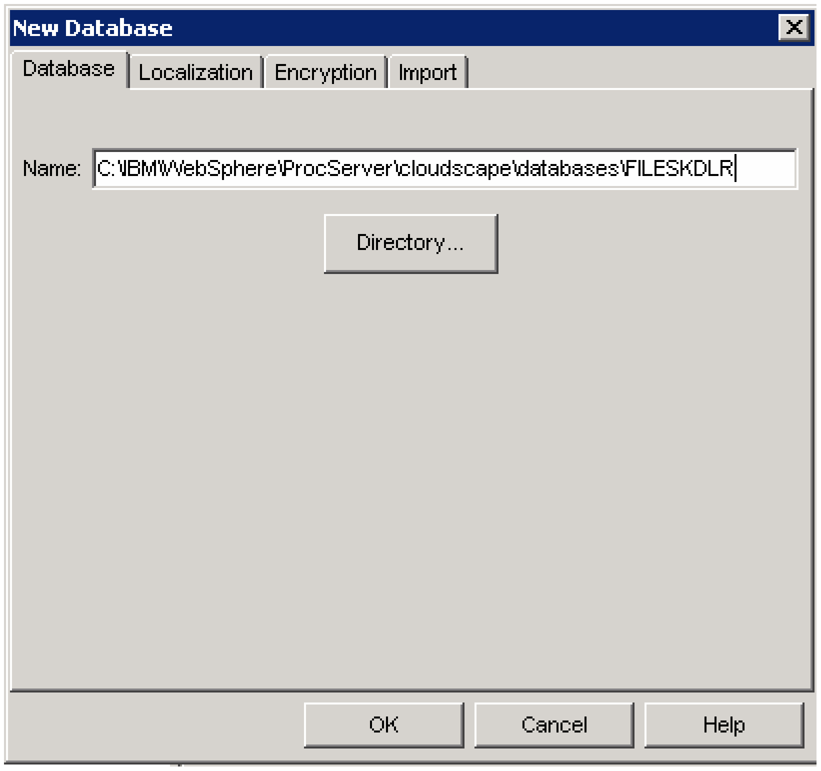


__ c. From the main menu, select **File → New → Database**

__ d. In the name field, enter <WPS_HOME>/cloudscape/databases/FILESKDLR

Where as <WPS_HOME> is C:\IBM\WebSphere\ProcServer

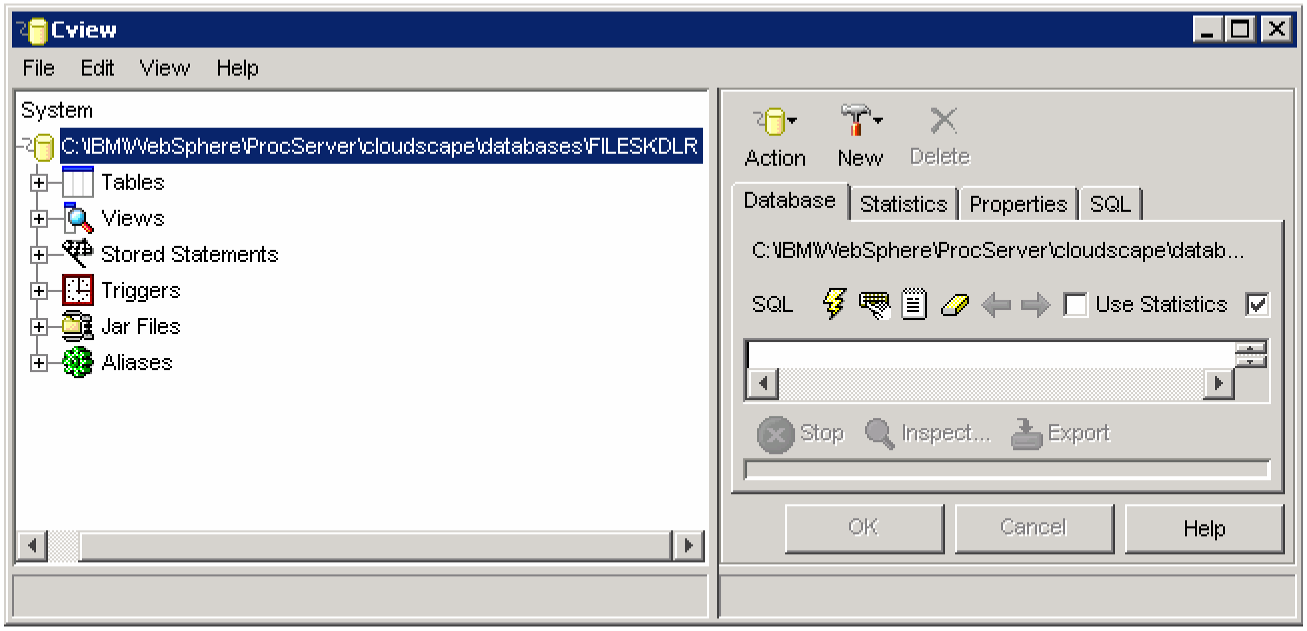
Ex: - C:\IBM\WebSphere\ProcServer\cloudscape\databases\FILESKDLR



Name: C:\IBM\WebSphere\ProcServer\cloudscape\databases\FILESKDLR

__ e. Click **OK**

__ f. Verify that the database is created and is listed on the left pane of the Cview window



C:\IBM\WebSphere\ProcServer\cloudscape\databases\FILESKDLR

C:\IBM\WebSphere\ProcServer\cloudscape\datab...

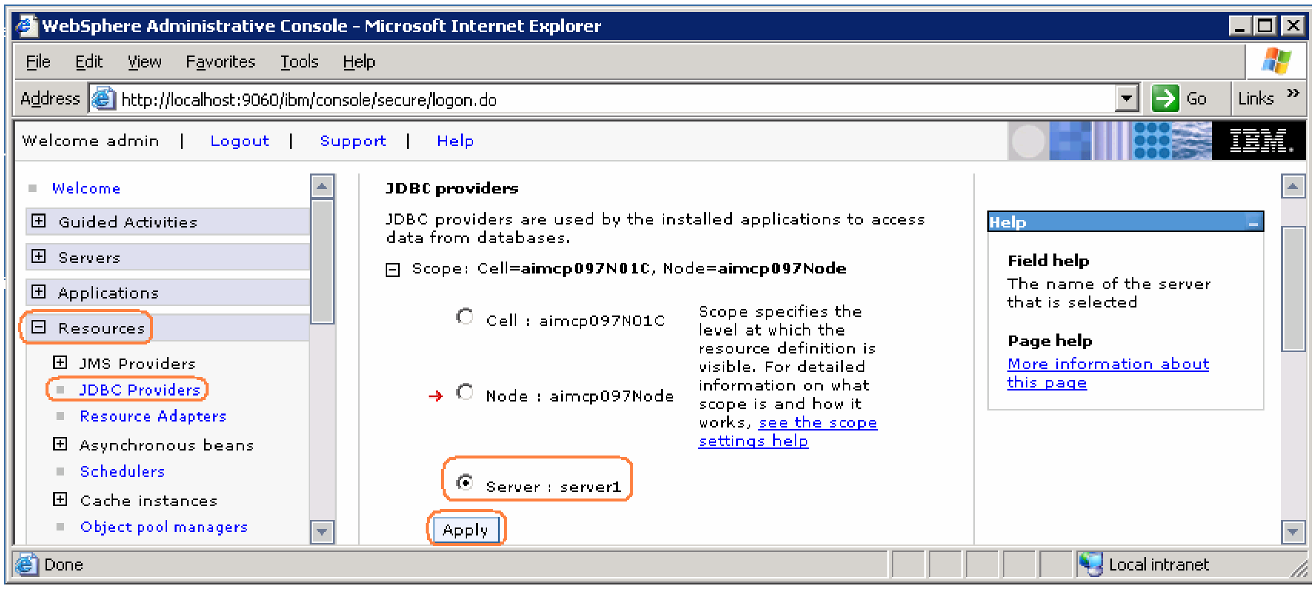
__ g. Close the Cview window (**File → Exit**)

____ 2. Create a Data source for the Scheduler Service

__ a. Launch the WebSphere Process Server Administrative Console and logon to it

Ex: **- http://localhost:9060/ibm/console/**

__ b. In the left pane, locate Resources → JDBC Providers, set the scope to **Servers** and click **Apply**


WebSphere Administrative Console - Microsoft Internet Explorer

Address http://localhost:9060/ibm/console/secure/logon.do

Welcome admin | Logout | Support | Help

Welcome
Guided Activities
Servers
Applications
Resources
JMS Providers
JDBC Providers
Resource Adapters
Asynchronous beans
Schedulers
Cache instances
Object pool managers

JDBC providers

JDBC providers are used by the installed applications to access data from databases.

Scope: Cell=**aimcp097N01C**, Node=**aimcp097Node**

Cell : aimcp097N01C

Node : aimcp097Node

[✓ Server : server1]

Scope specifies the level at which the resource definition is visible. For detailed information on what scope is and how it works, see the scope settings help

Apply

Help

Field help
The name of the server that is selected

Page help
More information about this page

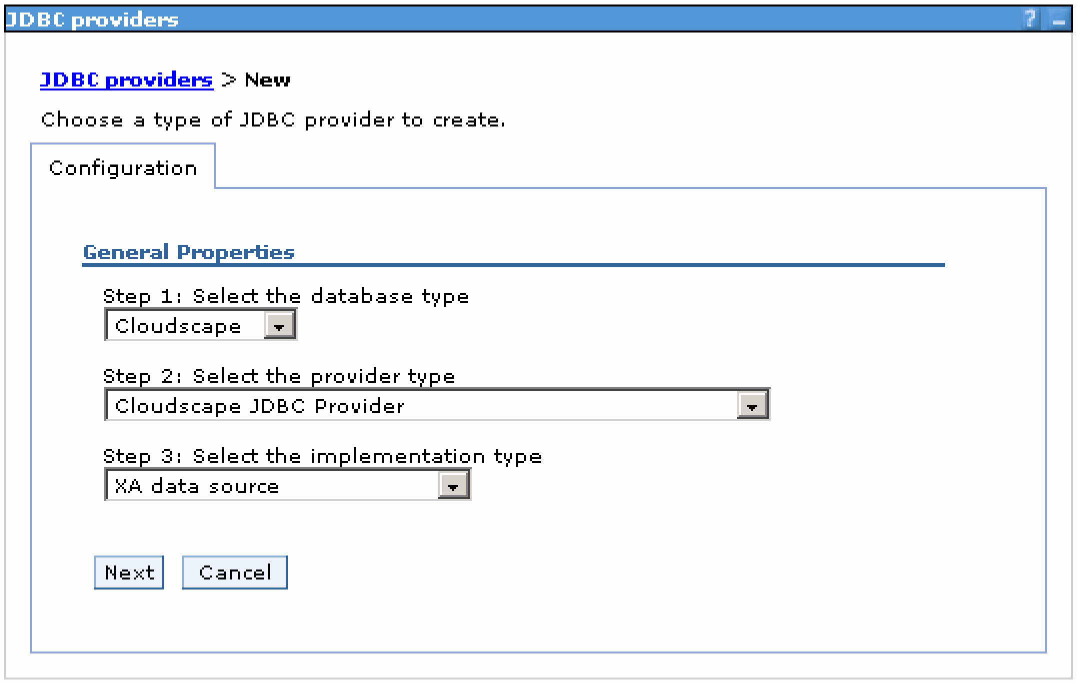
Done | Local intranet

__ c. Click the **New** button to create a new JDBC Provider

1) Set **Cloudscape** in database type

2) Set **Cloudscape JDBC Provider** in provider type

3) Set **XA data source** in implementation type


JDBC providers

JDBC providers > New

Choose a type of JDBC provider to create.

Configuration

General Properties

Step 1: Select the database type
Cloudscape

Step 2: Select the provider type
Cloudscape JDBC Provider

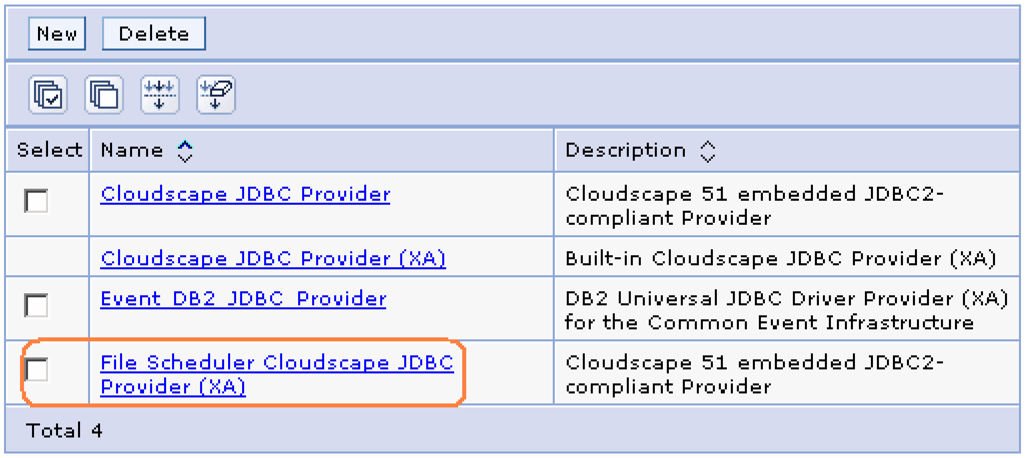
Step 3: Select the implementation type
XA data source

Next Cancel

__ d. Click **Next**

__ e. In the following screen, enter the **Name** filed to **File Scheduler Cloudscape JDBC Provider (XA)**, Click **OK** and **Save** to the master configuration

__ f. You must the new the Cloudscape JDBC Provider listed as shown below:

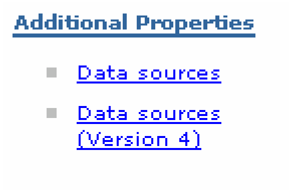


Select	Name	Description
☐	Cloudscape JDBC Provider	Cloudscape 51 embedded JDBC2-compliant Provider
	Cloudscape JDBC Provider (XA)	Built-in Cloudscape JDBC Provider (XA)
☐	Event DB2 JDBC Provider	DB2 Universal JDBC Driver Provider (XA) for the Common Event Infrastructure
☐	File Scheduler Cloudscape JDBC Provider (XA)	Cloudscape 51 embedded JDBC2-compliant Provider

Total 4

__ g. Now that a JDBC provider had been created, create a data source

__ h. Locate **Resources > JDBC Providers > File Scheduler Cloudscape JDBC Provider (XA)** and click on **Data sources** under **Additional properties**



__ i. In the following screen, click the **New** button to create a new data source

__ j. Under general properties, enter these values:

1) Name : **File SKDLR DataSource**

2) JNDI name: **jdbc/fileskdlr**

3) Disable "Use this Data Source in container managed persistence (CMP)"

4) Description: **JDBC DataSource for FILESKDLR Database**

5) Database name: **C:\IBM\WebSphere\ProcServer\cloudscape\databases\FILESKDLR**

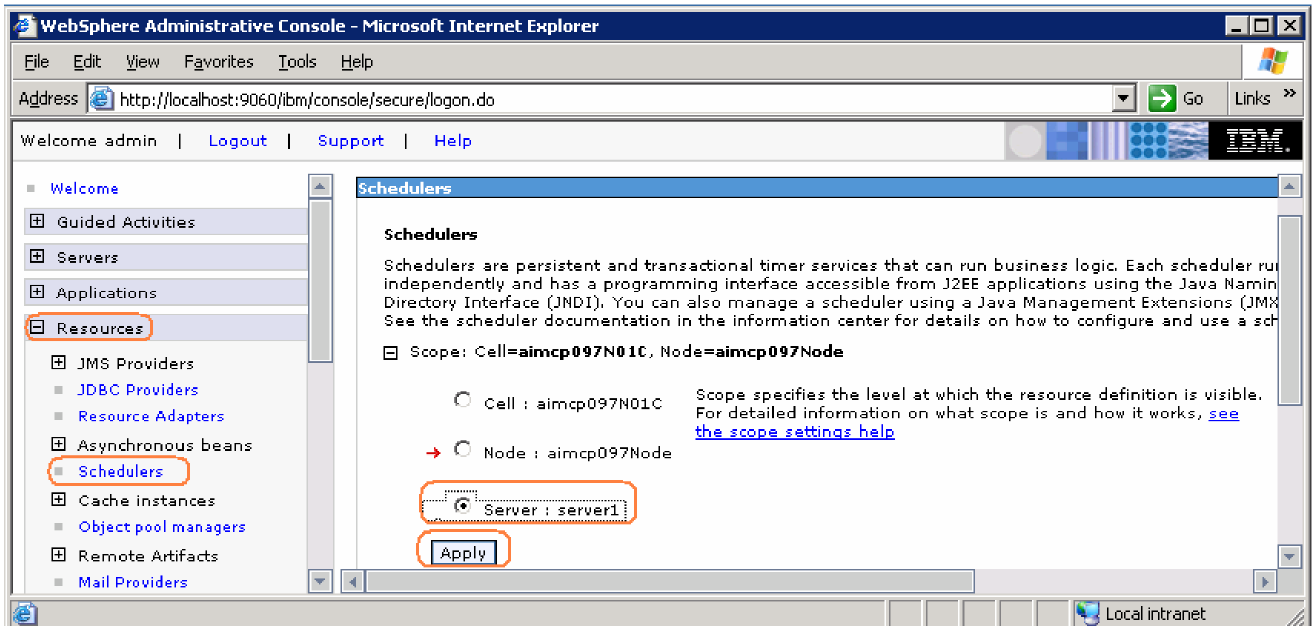
(<WPS_HOME>\cloudscape\databases\FILESKDLR)

__ k. Click **OK** and **Save** to the master configuration

__ l. Test the connection to the scheduler database and ensure that a successful message is resulted

____ 3. Create a scheduler for the File Emitter:

__ a. Locate **Resources > Schedulers** and the set the scope to **Servers**



Schedulers

Schedulers are persistent and transactional timer services that can run business logic. Each scheduler ru
independently and has a programming interface accessible from J2EE applications using the Java Namin
Directory Interface (JNDI). You can also manage a scheduler using a Java Management Extensions (JMX
See the scheduler documentation in the information center for details on how to configure and use a sch

Scope: Cell=**aimcp097N01C**, Node=**aimcp097Node**

Cell : aimcp097N01C

Node : aimcp097Node

[✓ Server : server1]

Scope specifies the level at which the resource definition is visible. For detailed information on what scope is and how it works, see the scope settings help

__ b. Click **New** to create a new Scheduler configuration

__ c. In the Scheduler > New screen, enter the following information:

1) Name: **FolderPoller**

2) JNDI name: **sched/FolderPoller** (this is the default specified as an environment entry in the EJB deployment descriptor)

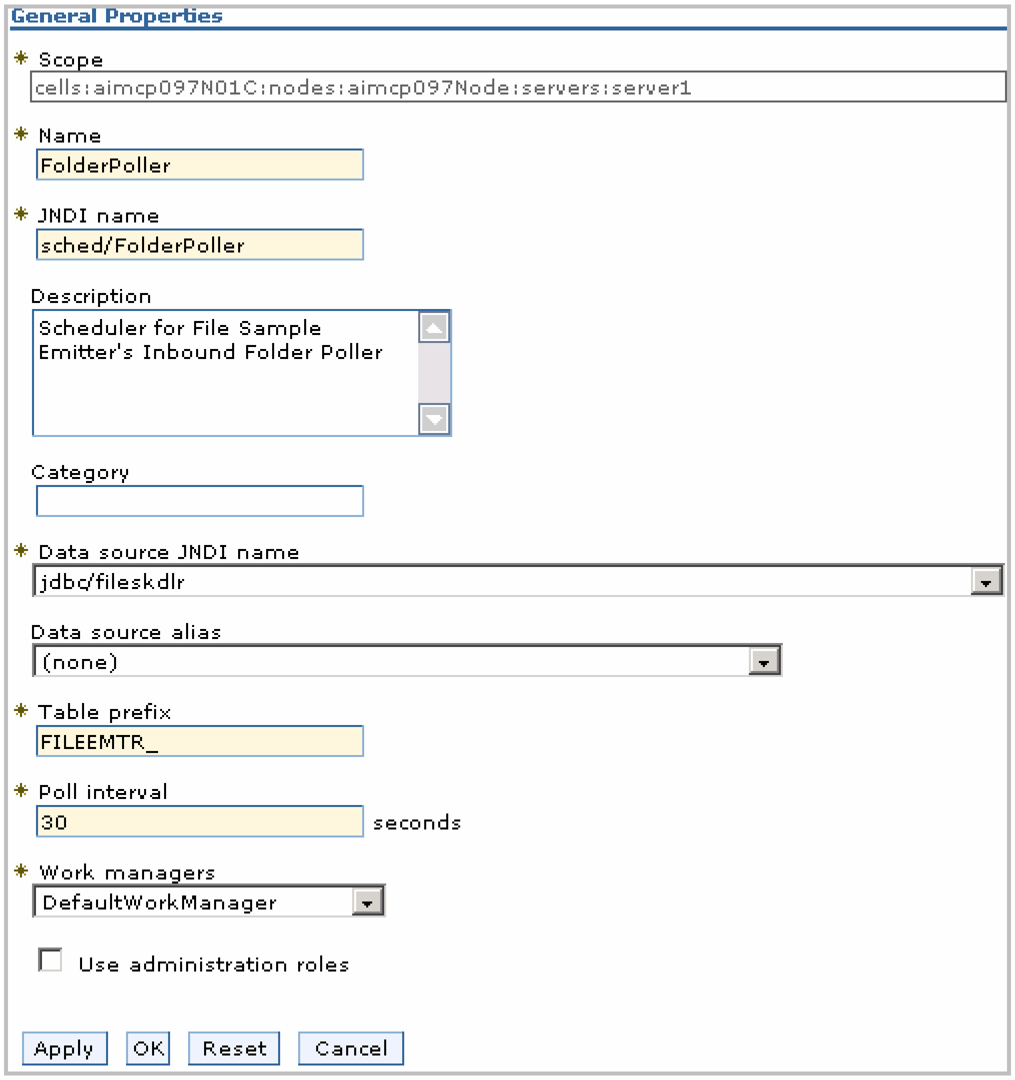
3) Description: **Scheduler for File Sample Emitter's Inbound Folder Poller**

4) Data source JNDI name: **jdbc/fileskdlr** (select from the drop down list)

5) Table prefix: **FILEEMTR_**

6) Poll interval: **30**

7) Work managers: **DefaultWorkManager**



General Properties

* Scope
cells:aimcp097N01C:nodes:aimcp097Node:servers:server1

* Name
FolderPoller

* JNDI name
sched/FolderPoller

Description
Scheduler for File Sample Emitter's Inbound Folder Poller

Category

* Data source JNDI name
jdbc/fileskdlr

Data source alias
(none)

* Table prefix
FILEEMTR_

* Poll interval
30 seconds

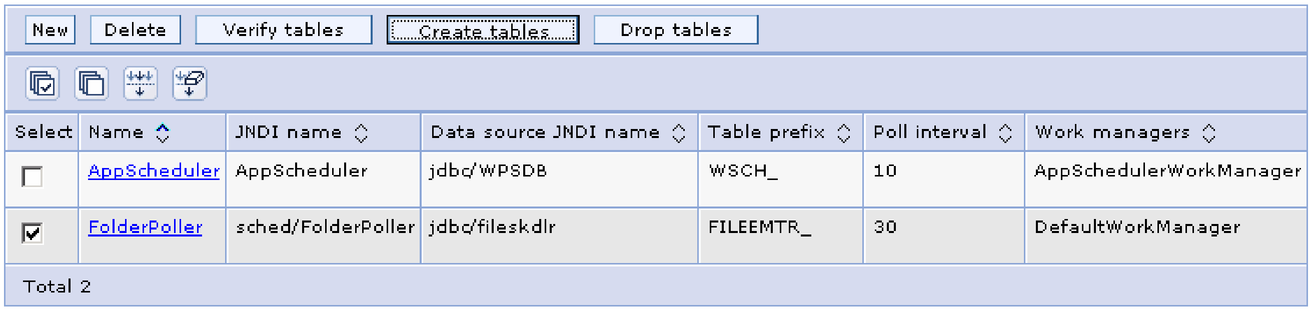
* Work managers
DefaultWorkManager

☐ Use administration roles

Apply OK Reset Cancel

__ d. Click **OK** and **Save** to the master configuration

__ e. Select check box next to **FolderPoller** and click the **Create Tables** button



New Delete Verify tables Create tables Drop tables

Select	Name	JNDI name	Data source JNDI name	Table prefix	Poll interval	Work managers
☐	AppScheduler	AppScheduler	jdbc/WPSDB	WSCH_	10	AppSchedulerWorkManager
✓	FolderPoller	sched/FolderPoller	jdbc/fileskdlr	FILEEMTR_	30	DefaultWorkManager

Total 2

__ f. Ensure that the tables are created successfully

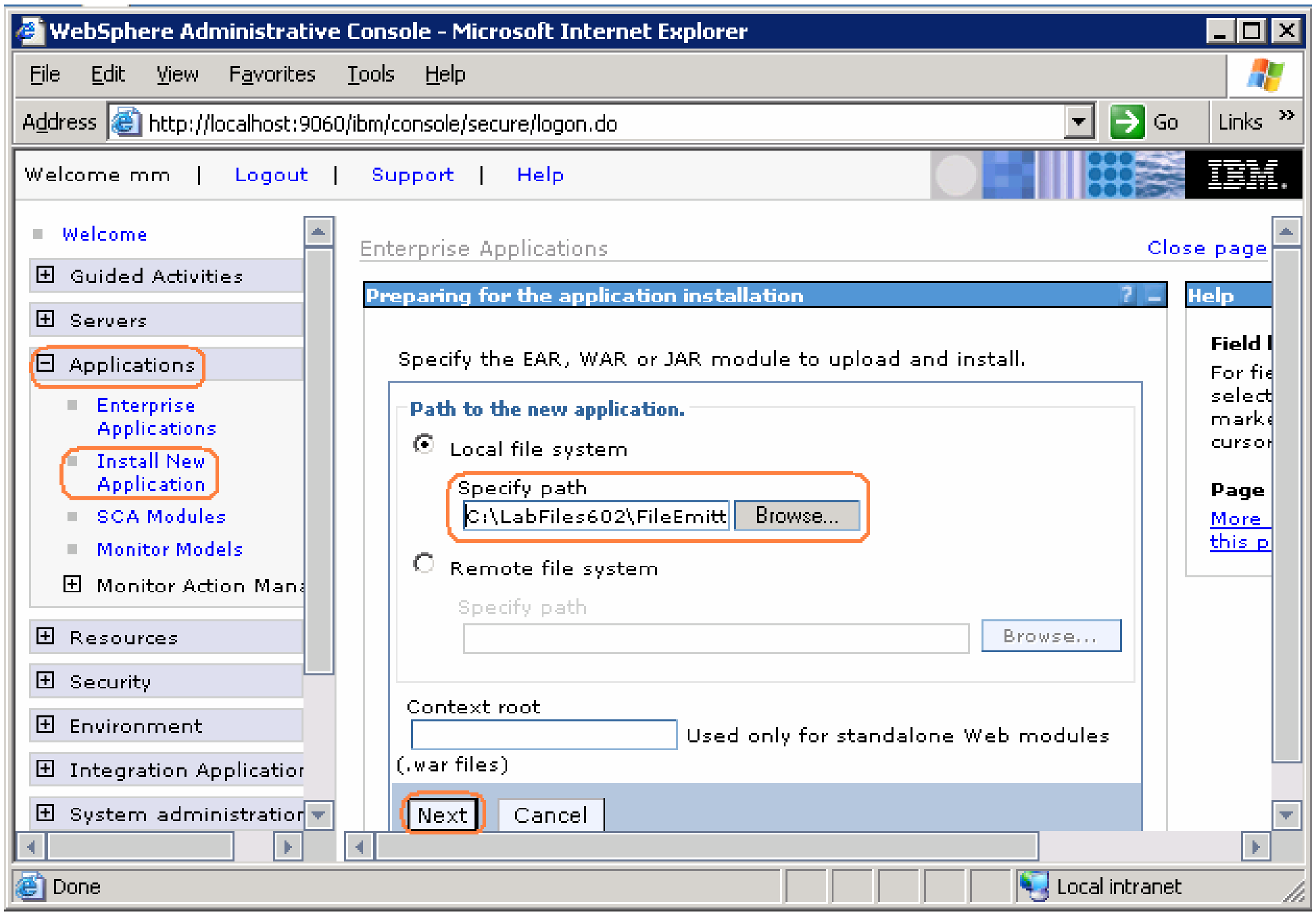
____ 4. The Scheduler configuration is complete

____ 5. Deploy the File Emitter EAR

Note: Ensure that the EAR is copied to the file system local to the application server where you would like to deploy

__ a. Logon to the Administration console. In the left pane, select **Applications >Install New Application**

__ b. Click **Browse** to **Specify Path** on the Local file system in the right panel and click **Next**



Specify the EAR, WAR or JAR module to upload and install.

Path to the new application.

[✓ Local file system]

Specify path: C:\LabFiles602\FileEmitt [Browse...]

Remote file system

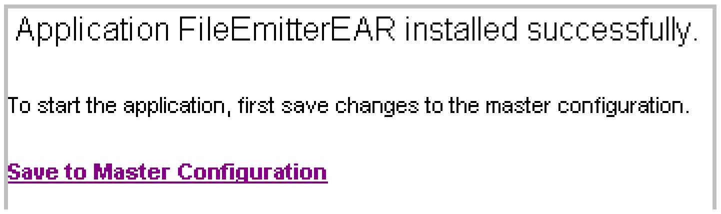
Specify path: [Browse...]

Context root: Used only for standalone Web modules (.war files)

__ c. Progress through all the 6 steps of “Install New Application” screens, accepting the defaults

__ d. In Step 6, the **Summary** screen, click the **Finish** button. The install will take several moments

__ e. After the installation is complete, ensure that the message “Application FileEmitterEAR installed successfully” is displayed.

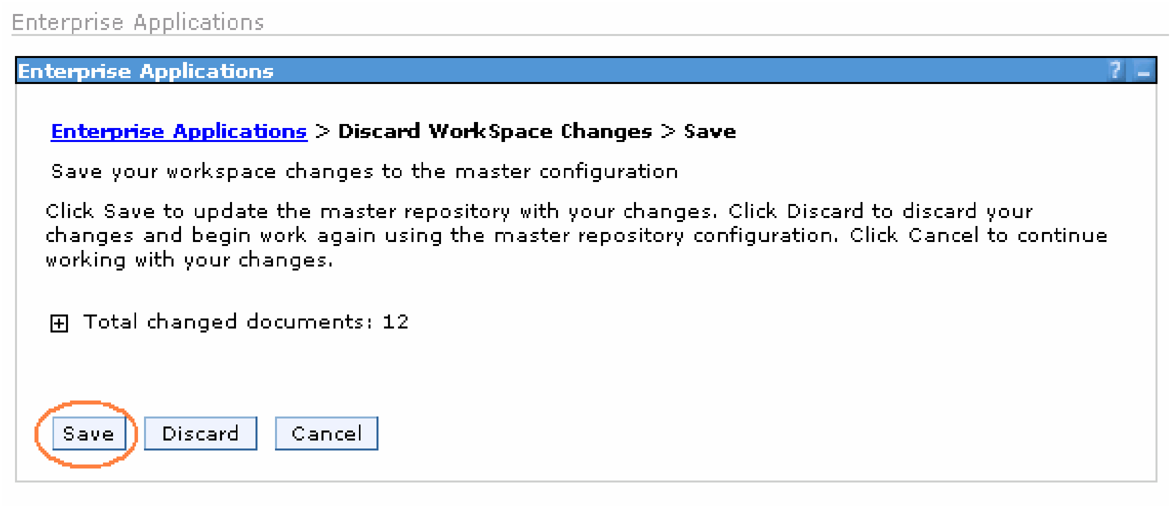


Application FileEmitterEAR installed successfully.

To start the application, first save changes to the master configuration.

Save to Master Configuration

__ f. Click “**Save to Master Configuration**”, and finally click **Save** on the “Save” screen.


Enterprise Applications

Enterprise Applications

Enterprise Applications > Discard WorkSpace Changes > Save

Save your workspace changes to the master configuration

Click Save to update the master repository with your changes. Click Discard to discard your changes and begin work again using the master repository configuration. Click Cancel to continue working with your changes.

Total changed documents: 12

____ 6. **Restart** the Process Server

Part 4: Testing file emitter

____ 1. Now that the Process Server configuration is complete and FileEmitter application is deployed, it is time to do a simple test for CEI event emissions.

__ a. Ensure that the Process Server and the FileEmitter application are running.

__ b. In the local file system, explore to **C:\Labfiles602\FileEmitter\inbound** and copy (occur an event) the **CUSTOMER_CREATE.txt** here

Note: The event file (**CUSTOMER_CREATE.txt**) is located at **<LABFILES>\FileEmitter\FILEEVENTS** directory.

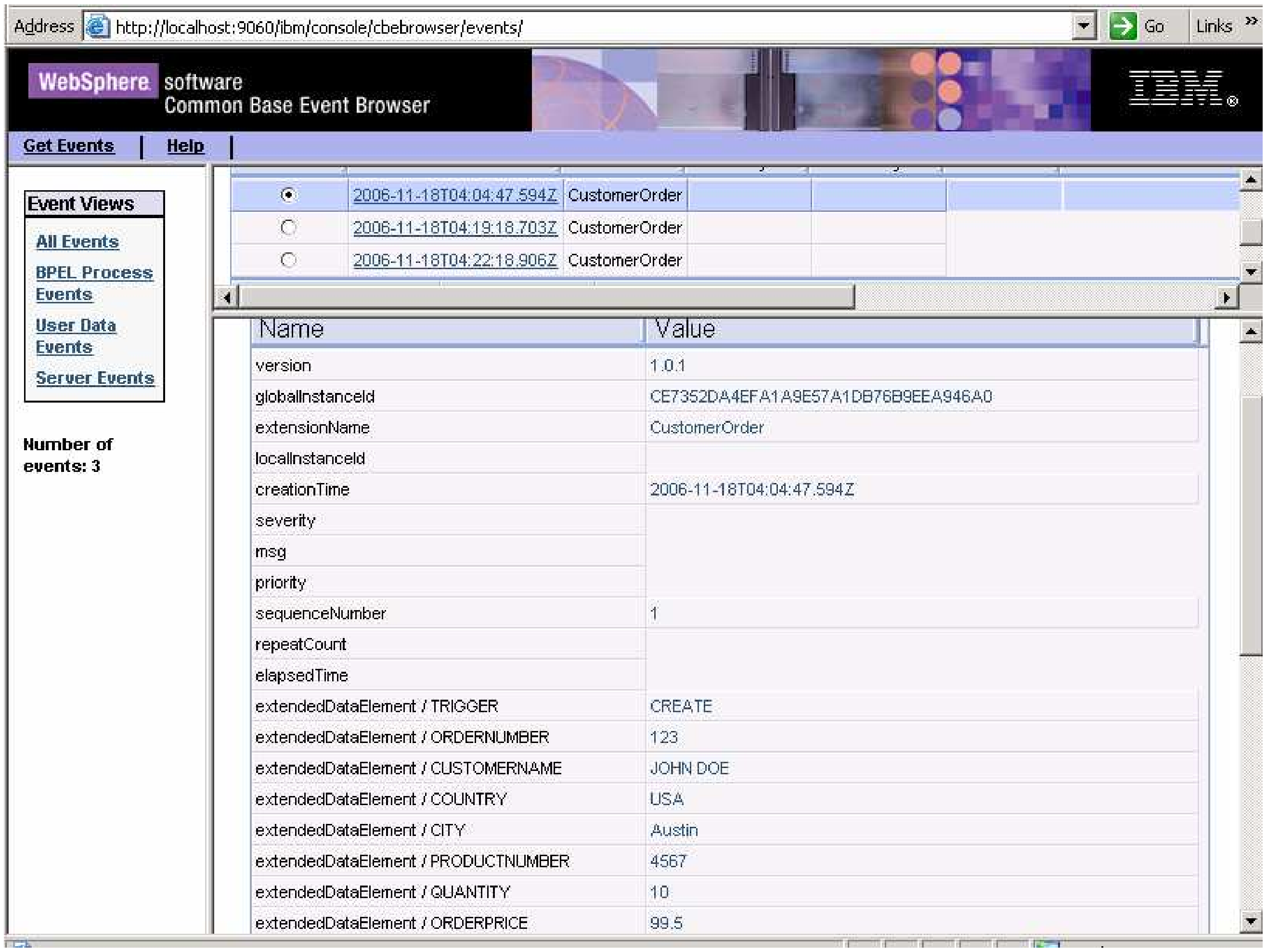
__ c. Once the event file is copied (event occurred) to the **inbound** directory the CREATE event trigger is fired and on a successful event emission, the event file is archived to the **archive** directory

__ d. Repeat the step b to copy the **CUSTOMER_UPDATE.txt** and **CUSTOMER_DELETE.txt** event files in to the inbound directory

__ e. Once these event files are copied (event occurred) to the **inbound** directory the UPDATE and DELETE event triggers are fired and on the successful event emissions, the event files are archived to the **archive** directory

Note: On a failed event emission the failed event occurred (files) are moved to the **fault** directory.

____ 2. To view the events emitted in a CBE browser, logon to the Process Server's Administrative console, Locate **Integration Applications > Common Base Event Browser** click on the **Get Events** and then **All Events**



Number of events: 3

●	2006-11-18T04:04:47.594Z	CustomerOrder
○	2006-11-18T04:19:18.703Z	CustomerOrder
○	2006-11-18T04:22:18.906Z	CustomerOrder

Name	Value
version	1.0.1
globalInstanceId	CE7352DA4EFA1A9E57A1DB76B9EEA946A0
extensionName	CustomerOrder
localInstanceId	
creationTime	2006-11-18T04:04:47.594Z
severity	
msg	
priority	
sequenceNumber	1
repeatCount	
elapsedTime	
extendedDataElement / TRIGGER	CREATE
extendedDataElement / ORDERNUMBER	123
extendedDataElement / CUSTOMERNAME	JOHN DOE
extendedDataElement / COUNTRY	USA
extendedDataElement / CITY	Austin
extendedDataElement / PRODUCTNUMBER	4567
extendedDataElement / QUANTITY	10
extendedDataElement / ORDERPRICE	99.5

Part 5: Solution

If you do not to perform the source modification task, then follow the instructions in this section to import a project interchange file which has the updated source. Import the project interchange that is provided, into WebSphere Integration Developer, Build All projects and export the EAR.

If using the solution, skip Part 1, complete this section, and then continue with Part 2.

____ 1. Open WebSphere Integration Developer

__ a. Select **Start > Programs > IBM WebSphere > Integration Developer V6.0.2 > WebSphere Integration Developer 6.0.2**

__ b. Workspace Launcher window will be displayed. Click the **Browse…** button and select your workspace directory. (**Ex:** C:\LabFiles602\FileEmitter\workspace)

__ c. Click **OK**.

__ d. Close the welcome window by clicking the arrow in the top right corner of the welcome window

__ e. By default WebSphere Integration Developer opens in **Business Integration** perspective. You need to change it to **J2EE** perspective. To do this click on the top right corner of the WebSphere Integration Developer and choose **Other** if **J2EE** is not listed here.

__ f. Select **Project > Build Automatically**, to turn off automatic builds

____ 2. Import the solution FileEmitter project interchange into WebSphere Integration Developer

__ a. Select **File > Import** from the main menu.

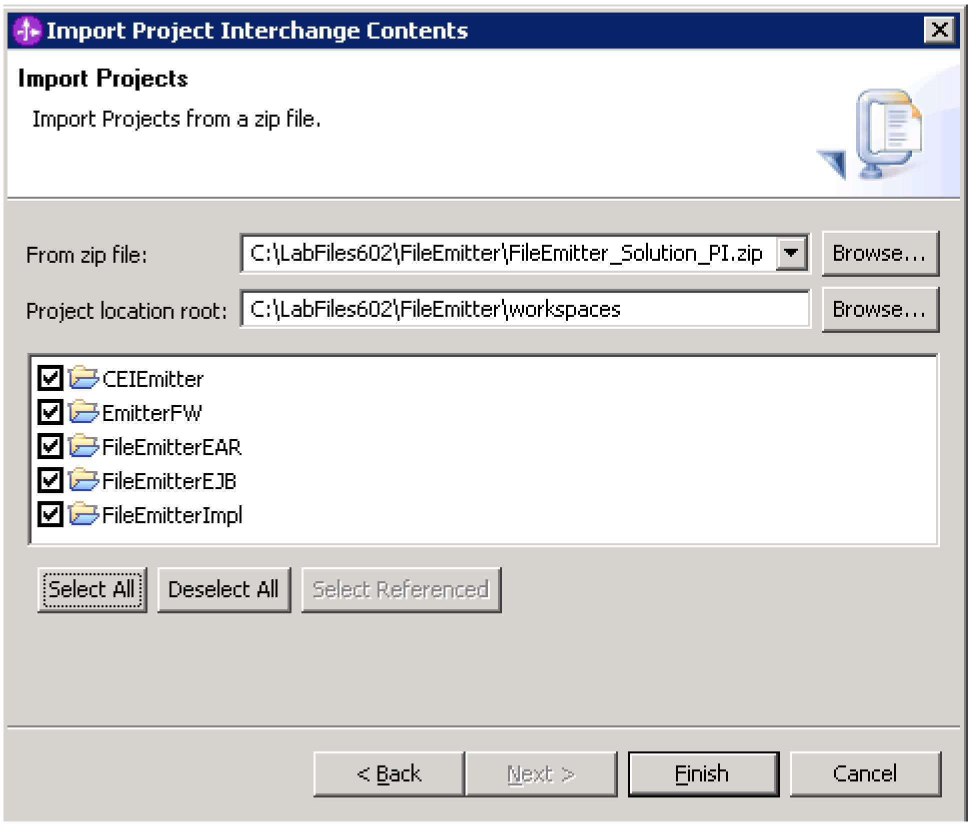
__ b. From the **Import** window select **Project Interchange** and click **Next**

__ c. The **Import Project Interchange Contents** window will be opened.

__ d. Click **Browse…** and select **<LABFILES>\FileEmitter\FileEmitter_Solution_PI.zip** as the source zip file.

Ex: C:\LabFiles602\FileEmitter\FileEmitter_Solution_PI.zip

__ e. Click **Select All** then click **Finish**


Import Project Interchange Contents

Import Projects

Import Projects from a zip file.

From zip file: C:\LabFiles602\FileEmitter\FileEmitter_Solution_PI.zip

Project location root: C:\LabFiles602\FileEmitter\workspaces

- [✓] CEIEmitter
- [✓] EmitterFW
- [✓] FileEmitterEAR
- [✓] FileEmitterEJB
- [✓] FileEmitterImpl

__ f. Refresh the projects and clean by clicking on **Project** > **Clean** and then choose **"Clean All Projects"** in the dialog

__ g. Click **Project > Build All** to build all the projects

__ h. Finally in the Project Explorer, right click on **FileEmitterEAR** under Enterprise Applications to choose **Export > EAR file**.

__ i. Provide the Destination you would like to save the EAR.

__ j. Close WebSphere Integration Developer.

__ k. Continue the lab with **Part 2** of this document.

What you did in this exercise

You imported the sample File Emitter source into WebSphere Integration Developer and modified it to match the File Emitter's CUSTOMERORDER data. Then you exported the FileEmitter EAR to WebSphere Process Server 6.0.2 that has a CEI SDK installed.

You Deployed the Application to WebSphere Process Server 6.0.2

You created a local file system where the event occur

You created a cloudscape database for the Scheduler, created a Data source for it and then created a Scheduler service.

You copied (to occur events) event files to the inbound directory to test for the Event Emissions to the CEI server

Finally you used the Process Server's Common Event Browser to view the emitted events.

This page is left intentionally blank.