TIBCO Foresight® Transaction Insight®

Web Services at Foresight

Software Release 5.2
September 2017

Two-second advantage®

Important Information

General Contact Information
TIBCO Software Inc.
3303 Hillview Avenue
Palo Alto, CA 94304 USA
Tel: +1 650 846 1000
Fax: +1 650 846 1005

Technical Support
E-mail: support@tibco.com
Web: https://support.tibco.com
(Note: Entry to this site requires a username and password. If you do not have one, you can request one. You must have a valid maintenance or support contract to use this site.)

Contents

Introduction

Overview

TIBCO Foresight provides web services for the following TIBCO Foresight® Transaction Insight® portal applications:

- TIBCO Foresight® Archive and Retrieval System
- TIBCO Foresight® Operational Monitor

Additionally the following web services are provided for use with Transaction Insight®:

- Document External Note

About Payara (GlassFish)

Web services are hosted by the application server Payara (formerly GlassFish).

You can obtain Payara and detailed installation instructions from the following link:

http://www.payara.fish/

This document explains how to use Payara with TIBCO Foresight web services. It does not explain how to set up and use Payara itself.

For complete directions on using Payara, please see the official Payara documentation at http://www.payara.fish/documentation.

Foresight Archive and Retrieval System Web Service

Overview

The Foresight® Archive and Retrieval System Web Service provides some of the TIBCO Foresight Archive and Retrieval System functionality to the Transaction Insight Portal and other applications. In order to utilize the Foresight Archive and Retrieval System Repository with Transaction Insight Portal, this web service is required. For further information on Foresight Archive and Retrieval System and its functionality, please refer to **TIB_fsp_archive_*n.n*_archiveadmin.pdf**.

The Foresight Archive and Retrieval System web services are Java-based. During installation, the WAR files are copied to Foresight Archive and Retrieval System's webService folder and then are deployed to Payara (GlassFish) by the installer. You can also do this from the Payara console, if needed.

It is sometimes necessary to re-start Payara when an external property changes.

Preconditions

The Foresight Archive and Retrieval System application must be installed either on the same machine or on another machine where the Foresight Archive and Retrieval System Web Service can access its Bin folder, database, and repository.

Web Service Operations

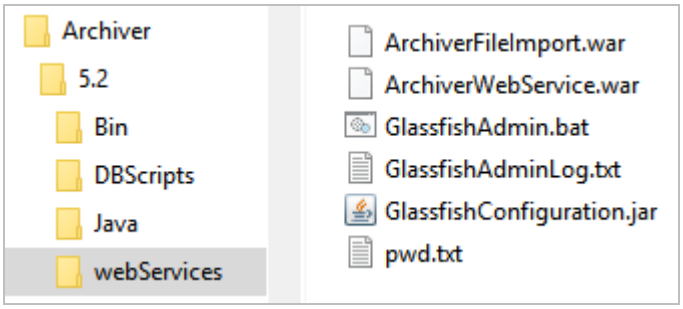
Foresight Archive and Retrieval System offers the following non-Transaction Insight applications web service operations:

- getFiltersByStandardVersionTransSet
- getAppDocFSUIDbyFilters
- dearchiveAppDocByFSUID

Web Services Directory

Foresight Archive and Retrieval System's **webServices** directory contains the files used to deploy the web services to Payara (GlassFish) and to configure it at installation.

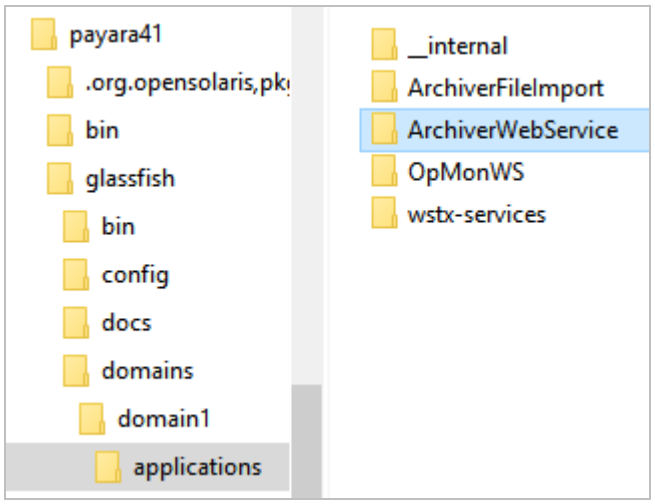
The installation program places these files in Foresight Archive and Retrieval System's webServices directory for use during installation and when you want to create your own web service client:



Name	Description
ArchiverFileImport.war	Contains the File import services. These are used by external clients to "import" files into Foresight Archive and Retrieval System outside of the normal Automator workflow model. Deploy this to your application server (Payara). It will handle requests sent to it by your web service client.
ArchiverWebService.war	Contains the Foresight Archive and Retrieval System Web Service. This is used by the Foresight Archive and Retrieval System portal to create views of transmissions and documents.
endorsed directory	AIX only This contains JAR files that you need to copy to the endorsed directory under Java during installation.
GlassfishAdmin.bat	Windows only Used to configure Payara (GlassFish) during installation.
GlassfishAdminLog.txt	Windows only Log file created by Payara (GlassFish).

Name	Description
GlassfishConfiguration.jar	Windows only Used during installation to set up Payara (GlassFish) properties. Users do not use it.
pwd.txt	Used to configure Payara during installation.

When the web service is deployed, the WAR files are unzipped into folders beneath the Payara (GlassFish) installation, where they can be used when requested by a client.



Web Services Installed with Foresight Archive and Retrieval System

You have these web services created by the installation program:

ArchiverImporterService - used by external HTTP clients to import files to Foresight Archive and Retrieval System. It is invoked via a URL with this format (the port may be different):

http://*serverName*:8080/ArchiverImporterService/ArchiverImporterService?wsdl

Its available operations:

importFile	Given a SOAP attachment and an XML document describing the relationship of the attachment to other files in the Repository, this will import the attachment into the Repository and associate it with a fileset.

ArchiverWebService - used internally by the Archive Portal, and invoked via the URL with this format (the port may be different):

http://*serverName*:8080/ArchiverWebService/ArchiverWebService?wsdl

Its available operations:

submitFileToWorkflow	Copy a local file to a given folder.
submitFilesToWorkflow	Copy a set of files from the repository to the inbound folder for a workflow.
submitFilesToArchiverImporterWorkflow	Copy a set of files from the repository to the inbound folder of the Archiver_UI_Importer workflow, where it can be displayed on the portal in a Transaction Insight-type transmissions view and document view.

Installation requirements

System property com.foresightcorp.archiver.config, needs to be assigned the full path to the Archiver Properties file, for example: C:\TIBCO\Archiver\5.2\Bin.

NOTE: Refer to Setting up a System Property in Payara (GlassFish) on page 41 for information on setting up a system property.

Testing

The Available operation allows you to test not only the installation of the web service, but the web service's ability to access the resources it needs to operate. By submitting the value of "diagnostics:pingall" the web service will report on all available resources. Here is an example of the SOAP call to the web service and a sample response.

SOAP Call

```
<?xml version="1.0" encoding="UTF-8"?><S:Envelope
xmlns:S="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-
ENV="http://schemas.xmlsoap.org/soap/envelope/">

    <SOAP-ENV:Header/>

    <S:Body>

        <ns2:available
xmlns:ns2="http://archiverws.foresightcorp.com/">

            <inString>diagnostics:pingall</inString>

        </ns2:available>

    </S:Body>
```

SOAP Response


```
<?xml version="1.0" encoding="UTF-8"?><S:Envelope
xmlns:S="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-
ENV="http://schemas.xmlsoap.org/soap/envelope/">
    <SOAP-ENV:Header/>
    <S:Body>
      <ns2:availableResponse
xmlns:ns2="http://archiverws.foresightcorp.com/">
          <return>
              <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
              <WebServicePing
xmlns="http://com.foresightcorp.archiver/schema/WebServicePing">
                  <root>
                      <processingContext>
                          <property>
                              <name>Glassfish Version</name>
                              <value>Payara Server  4.1.1.171.1 #fish
(build 139)</value>
                          </property>
                          <property>
                              <name>Java Endorsed Directories</name>

<value>C:\payara41\glassfish/modules/endorsed;C:\payara41\glassfish/li
b/endorsed</value>
                          </property>
                          <property>
                              <name>Java Home</name>
                              <value>C:\Program
Files\Java\jdk1.8.0_131\jre</value>
                          </property>
                          <property>
                              <name>Java Temp Folder</name>

<value>C:\Users\ADMINI~1\AppData\Local\Temp\1\</value>
                          </property>
                          <property>
                              <name>Java Version</name>
                              <value>1.8.0_131</value>
                          </property>
                          <property>
                              <name>OS Name</name>
                              <value>Windows Server 2016</value>
                          </property>
```

```
                        <property>
                            <name>OS Version</name>
                            <value>10.0</value>
                        </property>
                        <property>
                            <name>Test Name</name>
                            <value>Ping All</value>
                        </property>
                        <property>
                            <name>User Name</name>
                            <value>Administrator</value>
                        </property>
                        <property>
                            <name>Web Service Name</name>
                            <value>Archiver Web Service</value>
                        </property>
                        <property>
                            <name>Archiver Version:</name>
                            <value>5.2.0, Build Number:005,
BuildDate:2017-06-12 17:01:29</value>
                        </property>
                        <property>
                            <name>Host Name</name>
                            <value>10.97.194.120</value>
                        </property>
                        <property>
                            <name>Report Date</name>
                            <value>Tue Jun 13 15:30:37 EDT
2017</value>
                        </property>
                    </processingContext>
                    <diagnostics>
                        <test>
                            <name>properties</name>
                            <description>Checks that the Archiver
properties file can be found and contains the necessary properties for
database connections.</description>
                            <status>Pass</status>
                            <diagnosticMessage>Archiver Properties
file is
C:/TIBCO/Archiver/5.2/Bin\archiver.properties</diagnosticMessage>
                        </test>
```

```
                        <test>
                            <name>database</name>
                            <description>Checks that a connection
can be made to the database.</description>
                            <status>Pass</status>
                            <diagnosticMessage>Connection String is
jdbc:sqlserver://na-dub-
sql2008;databaseName=DEV_BIL_ARS520A;user=sa;password=**********;;Data
base type is SQLSERVER</diagnosticMessage>
                        </test>
                        <test>
                            <name>repository</name>
                            <description>Checks that the current
repository root folder exists and that the web service has read/write
permissions.</description>
                            <status>Pass</status>
                            <diagnosticMessage>Repository root
folder is C:\TIBCO\Archiver\Data\Repository</diagnosticMessage>
                        </test>
                        <test>
                            <name>actions</name>
                            <description>Checks that each action
folder exists and that the web service has write permissions for
them.</description>
                            <status>Pass</status>

<diagnosticMessage></diagnosticMessage>
                        </test>
                    </diagnostics>
                </root>
            </WebServicePing>
        </return>
    </ns2:availableResponse>
  </S:Body>
</S:Envelope>
```

Foresight Operational Monitor System Web Service

Overview

The Foresight Operational Monitor Web Service allows you create your own web service client that will track events and put them in the TIBCO Foresight® Operational Monitor database for viewing through the portal

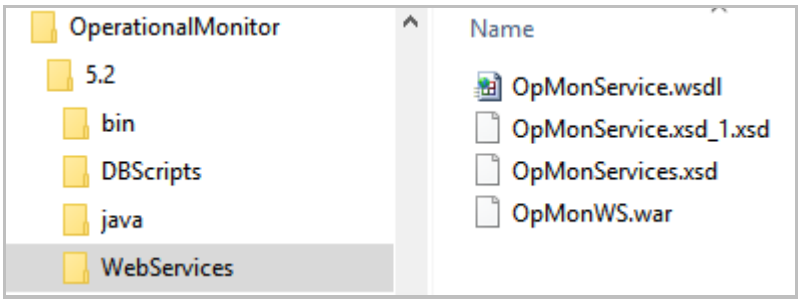
For further information on Foresight Operational Monitor and its functionality, please refer to **TIB_operationalmonitor_*n.n*_adminguide.pdf**.

The Foresight Operational Monitor web services are Java-based. During installation, the WAR files are copied to Foresight Operational Monitor's webService folder and then are deployed to Payara (GlassFish) by the installer. You can also do this from the Payara console, if needed.

It is sometimes necessary to re-start Payara when an external property changes.

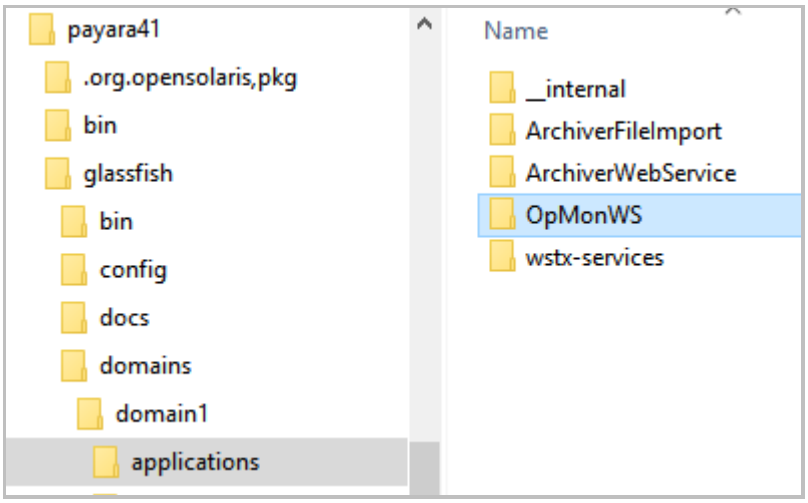
Foresight Operational Monitor Web Services Directory

The installation program places these files in Foresight Operational Monitor's webServices directory for use during installation and when you want to create your own web service client:



Name	Description
GlassfishAdmin.bat	Windows only Used to configure Payara (GlassFish) during installation.
GlassfishAdminLog.txt	Windows only Log file created by Payara (GlassFish).
GlassfishConfiguration.jar	Windows only Used during installation to set up Payara (GlassFish) properties. Users do not use it.
OpMonService.wsdl OpMonService.xsd_1.xsd	Definition of the web service, used to create your own Foresight Operational Monitor client for non-Automator workflows such as those made with TIBCO ActiveMatrix™ BusinessWorks Designer.
OpMonServices.xsd	Deprecated.
OpMonWS.war	Contains the Foresight Operational Monitor web services that handle requests from the Foresight Operational Monitor client. These are used by external clients to track files into the Foresight Operational Monitor portal outside of the normal Automator workflow model. Deploy this to your application server (Payara). It will handle requests sent to it by your web service client.
pwd.txt	Used by the Payara installation program.

When the web service is deployed, the files are unzipped into folders beneath the Payara installation, where they can be used when requested by a client.



Web Services Installed with Foresight Operational Monitor

You have these web services created by the installation program:

OpMonService

- Used by external HTTP clients to record data operations in the Foresight Operational Monitor database for viewing in the Foresight Operational Monitor portal.
- Invoked via a URL with this format (the port may be different – Payara [Glassfish] uses 8080):

  http://*serverName*:8080/OpMonService/OpMonService?wsdl

Its available operations:

(Date fields containing 0 default to the system's timestamp in milliseconds.)

cpFileIn	Issue an event that will log a file entering a collection point. This will usually be a file that is to be acted upon at this point in the process or workflow. Parameters: jobID The JobID associated with this event collectionPointID GUID associated with this collection point in RegisterCollectionPoint processID GUID of the specific process or workflow issuing the event filename Name of the file entering the collection point realTime If True, this event is counted as a real time event; Otherwise False Returns: True if the event was successful logged; otherwise False

cpFileOut	Issue an event that will log a file exiting a collection point. This can be a file that entered the collection point or one created within the collection point. Parameters: jobID The JobID associated with this event collectionPointID GUID associated with this collection point in RegisterCollectionPoint processID GUID of the specific process or workflow issuing the event filename Name of the file exiting the collection point realTime If True, this event is counted as a real time event If False, this is considered a normal event Returns: True if the event was successful logged; otherwise False
cpStart	Creates and sends an event signifying that we are starting to log from a given collection point. Parameters: jobID The JobID associated with this event collectionPointID GUID associated with this collection point in RegisterCollectionPoint processID GUID of the specific process or workflow issuing the event realTime If True, this event is treated as a real time event Otherwise False Returns: True if the event was successful logged; otherwise False
cpStop	Creates and sends an event signifying that we are no longer logging from a given collection point. Parameters: jobID The JobID associated with this event collectionPointID GUID associated with this collection point in RegisterCollectionPoint processID GUID of the specific process or workflow issuing the event conditionCode A value indicating a return code to be logged realTime If True, this event is treated as a real time event Otherwise False Returns: True if the event was successful logged; otherwise False
Register CollectionPoint	Registers a specified collection point, associating a part of a process or workflow with a name and GUID. Parameters: cpName Name associated with the collection point cpID GUID associated with the collection point Returns: True if the collection point was successful registered; otherwise False

registerFile	Log a file created in, or passed to, the process or workflow being logged. This is to assign a JobID and FileID pair to a file. This will indicate that the event logging should now start checking this file. Parameters: filename Full file path of the file to be registered jobID The JobID string (usually a GUID) assigned to the file. Afterwards, this should remain constant throughout the entire process or workflow. Multiple files may have the same JobID value fileID The FileID string (usually a GUID) assigned to the file Afterwards, this should remain constant throughout the entire process or workflow. Multiple files should have different FileID values fileSize Size of the file in bytes initialFile If True, this file was created in this part of the process or workflow If False, this file was created before this part of the process or workflow was called Returns: True if the collection point was successful registered; otherwise False
writeAlert	Log an Alert message to the event log. Parameters: jobID The JobID associated with this event subject The subject of the alert message message The message Returns: True if the alert was successful logged; otherwise False
writeEvent	Writes an event message. While there are constant values that can be used for event information, such as severity or status, you can use any integer values for your own types of events. Parameters: eventType The type of event to register eventDate The time that this event occurred Pass as a String in CCYYMMDDHHMMSSTTT format jobID The JobID associated with this event collectionPointID GUID of the collection point issuing the event processID GUID of the specific process or workflow issuing the event message Optional message attached to the event severity Severity associated with the event status Status associated with the event realTime If True, this is counted as a real time event This implies that the event is time-critical If False, this is considered a normal event meta A string array that contains metadata records These can be any string values If not using metadata entries, specify null Returns: True if the event was successful logged; otherwise False

Using the Foresight Operational Monitor Web Service

While the Foresight Operational Monitor Web Service allows you to track events as they occur throughout your business process, it also requires you to keep track of unique identifiers so that your applications can reference the same identifier at each stage. This is accomplished within Automator by creating a track file (.trk file) that is passed with the file being monitored. The .trk file contains the job and file identifiers.

You can use any unique identifier (we strongly recommend Generally or Universally Unique Identifiers [GUIDs or UUIDs]) within any tracking mechanism as long as you are always referring to the same job and file identifier when calling a web service for a process related to a particular file.

You may also want to pass those identifiers into the Automator workflow. To do this, create a file with the same name as the file you are tracking, and add ".trk" at the end. For example, if you have a file named "abc123.edi", the track file should be named "abc123.edi.trk".

For information on the contents of the track file, please refer to the document **Trk_Files.pdf**.

Transaction Insight Web Services

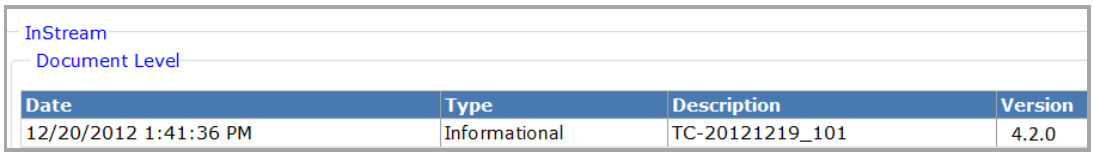
Document External Note

Overview

The Document External Note web service lets you attach one or more notes to a document that is already in the TIBCO Foresight® Transaction Insight® database, using the FSUID as a means of identifying the document to receive the note.

The note displays on Transaction Insight®'s Document Summary page.

In this example, we can see an informational description `TC-20121219_101` on this Document Summary page.

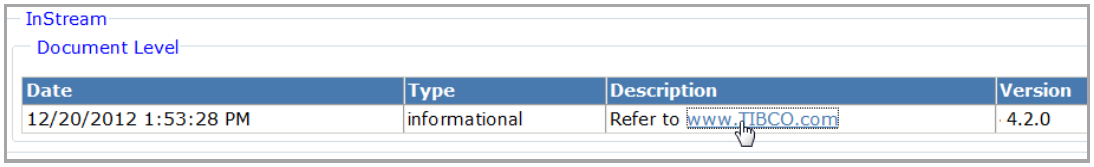


InStream

Document Level

Date	Type	Description	Version
12/20/2012 1:41:36 PM	Informational	TC-20121219_101	4.2.0

Links work in the Description:



InStream

Document Level

Date	Type	Description	Version
12/20/2012 1:53:28 PM	informational	Refer to www.TIBCO.com	4.2.0

Preconditions

- Transaction Insight Database
- TIBCO Foresight® Instream® - creating detail results files with FSUIDs in IDENT records. Please see **FSUID_and_AppDocs.pdf**.

Installation requirements

Installing Document External Note will require the following:

- System property `com.tibco.foresight.documentexternalnote.logfile` which needs to be assigned the full path to the log file, for example: `C:\TIBCO\TransactionInsight\5.1.0\WebServices\logs\DocumentExternalNote.txt`.
  NOTE: Refer to Setting up a System Property in Payara (GlassFish) on page 41 for instructions.
- A Web Service Connection pool

Testing

1. Locate a test document FSUID that is in the Transaction Insight database. These are stored in the DocumentGUID column of the Document table. Copy it. Note the value in its DocumentID column.
2. In Payara (Glassfish), go to **Applications | DocumentExternalNote | View Endpoint** and click the Tester link.
3. Click the top link:

   Application Name: DocumentExternalNote

   Links: [server] http://lbrownst-T420:8080/DocumentExternalNote
   [server] https://lbrownst-T420:8181/DocumentExternalNote

4. In the fields for **addDocumentExternalNoteDocumentFSUID**, add these parameters:

First field:	The FSUID that you copied
Second field:	`1`
Third field:	`informational`
Fourth field :	`5.2.0`
Fifth field:	`Test message 1`

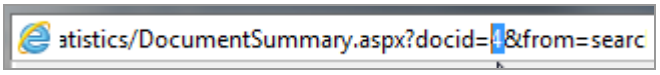
5. Click the addDocumentExternalNoteDocumenFSUID button.

   Under SOAP Response, see if the return code is 0 (success):

   ```
   <return>0</return>
   ```

6. In the Transaction Insight portal, go to Transaction Insight's **Search Documents** page and find any document. Go to its **Document Summary** page.

   In the URL, replace the document ID with the number from the DocumentID column in the database:

Refresh the page.

Look at the bottom to see the note:

InStream

Document Level

Date	Type	Description	Version
1/8/2013 11:28:12 AM	Informational	Test message 1	4.2

Web Service Operations

The AddDocumentExternalNote_documentFSUID method has these parameters:

documentFSUID	The document's FSUID, which is in the IDENT record from the validation detail results file.
externalSystemId	The ExternalSystemID from the Transaction Insight database's **ExternalSystem** table. Normally, this is 1 for Instream®. It controls which heading is used for the note in the Document Summary page. InStream Document Level Date 12/20/2012 3:46:20 PM
externalNoteTypeValue	For your own use. The value is text of your choice. It appears in the Type column on the Document Summary page. InStream Document Level Date | Type 12/20/2012 3:46:20 PM | informational
webServiceVersionValue	For your own use. It populates the Version column on the Document Summary page. Description | Version Test message 1 | 4.2
descriptionValue	The text of the note that you are attaching to the document. If you include a URL, starting with http or www, it will be an actual link on the document summary page. Description Refer to www.TIBCO.com

Other Variations

Two other web service variations exist. To identify the document that will receive the note, these methods use the Document ID from the Document table in the Transaction Insight database. The Document ID may be an integer or a long integer.

AddDocumentExternalNote

```
public int AddDocumentExternalNote(int documentId,
                                   int externalSystemId,
                                   java.lang.String externalNoteTypeValue,
                                   java.lang.String webServiceVersionValue,
                                   java.lang.String descriptionValue)
```

Add a note to a document using a document id to identify the document.

Parameters:

`documentId` - the document id, which is in the TI database.
`externalSystemId` - the Id that TI has assigned to the External System that is attaching the note. Use 1 for Instream.
`externalNoteTypeValue` - the Type of the note that is being attached (informational, etc.). This is user-defined.
`webServiceVersionValue` - the version of the web service client that is inserting the record into the TI database. This is user-defined.
`descriptionValue` - the text of the external note that is being attached to the document. This is user-defined.

Returns:

SUCCESS=0, anything else is an error. The error code and an explanation will be in the Document External Note log if logging is enabled. Otherwise it will be in the log managed by the application server.

AddDocumentExternalNote_Long

```
public int AddDocumentExternalNote_Long(long documentId,
                                        int externalSystemId,
                                        java.lang.String externalNoteTypeValue,
                                        java.lang.String webServiceVersionValue,
                                        java.lang.String descriptionValue)
```

Web service operation added for 4.2 to accommodate values larger than 2^^32

See Also:

`AddDocumentExternalNote`

Using Payara (GlassFish) with Web Services

Overview

This section explains how to use Payara (formerly GlassFish) with TIBCO Foresight products. It does not explain how to set up and use Payara itself. For complete instructions, please see the Payara Documentation Set at http://www.payara.fish/documentation.

Put Payara on any Windows and AIX server where it can access:

- The Archive repository
- Foresight Archive and Retrieval System-related workflows that use TIBCO Foresight web services, such as Archiver_UI_Importer.
- OMProcessor if Foresight Operational Monitor web service is being used.

Payara (GlassFish) Requirements

For current Payara (GlassFish) requirements, refer to **TIB_transactioninsight_*n.n*_installation.pdf**, Appendix F: Payara (GlassFish).

Before you Begin

- Ensure the Java JDK version required for Payara (GlassFish). Refer to **TIB_transactioninsight_*n.n*_installation.pdf**, Appendix F: Payara (GlassFish).
- Note the directory containing the Java JDK installation, as it will be needed during the install.

  Example: C:\Program Files\Java\jre1.8.0_31

For information about installing Java, see Appendix L: Additional System Requirements in **TIB_transactioninsight_*n.n*_installation.pdf**.

Installing Payara (GlassFish)

Installing Payara on Windows

1. Go to http://www.payara.fish/downloads.

   Detailed installation instructions are provide at this link as well.
2. Download the required version of Payara for your platform.
3. Unzip the package and place the newly created folder anywhere on your system.

   Example: C:\payara41
4. A default domain (domain1) is configured ready to run. To start the domain, cd to the bin directory

   cd C:\ payara41\bin
5. Use the **asadmin** tool

   asadmin start-domain domain1

   To install Payara as a Windows service use:

   asadmin create-service --name domain1

   You can rename the service using the Service Control command, for

   Example: sc config domain1 DisplayName= "Payara domain1"

4. Continue through the installation, responding to the dialogs as follows:

Dialog Title	Response
Installation Directory	Accept *<payara>*
Administration Settings	Accept the defaults, but change the password. **Note:** Don't use port 8080 for Payara (GlassFish) if you are using ActiveMatrix™ BusinessWorks.
Update Configuration	Uncheck the boxes to disable this feature.
JDK Selection	Choose **Select a Java SDK** from the list. From the drop down list, select the high-level directory where you installed Java JDK. Example: C:\Program Files\Java\jre1.8.0_31
Ready To Install	Click **Install**.
Progress	Displayed as the installation proceeds.
Summary	Click **Exit**.

Checking Payara after running the installation program

1. Click **Start | All Programs |** < *Payara*> **| Start Application Server**.
2. Wait a minute or two to give the server a chance to start.
3. Open a browser and enter the url `http://server:4848` (or whatever port Payara [GlassFish] uses).
4. You should be able to logon to Payara with the administrative username and password that you chose during installation.

Installing Payara (GlassFish) on AIX

Step	Description
Check Prerequisites	Payara (GlassFish) listens on TCP/IP port 8080 so it must be available for it. For other ports used by Payara see the Payara documentation at http://www.payara.fish/documentation. Install the version of Java JDK required by your version of Payara. Refer to **TIB_transactioninsight_*n.n*_installation.pdf**, Appendix F: Payara (GlassFish). This web site explains how to download, install, and configure Java on AIX: http://www.ibm.com/developerworks/systems/library/es-JavaOnAix_install.html $JAVA_HOME must be defined to point to the high-level Java JDK directory.
Create Folders	Create a root folder where you will install Payara. This is typically something that identifies the version of Payara you are working with, such as **payara41**. In this root folder, create a sub-folder named `tmp`.
Download Payara (GlassFish)	Go to http://www.payara.fish/downloads and download the supported version of for your operating system. Save it to the **tmp** folder under your Payara root directory.
Run the Installer	From the command line in the Payara root folder, enter the following command. `java -Xms1024m -Xmx2048m -jar tmp/glassfish-installer-<version>-aix.jar` The –Xms and –Xmx values can be adjusted up or down as necessary. As part of the installation you will be prompted to accept the terms of the distribution license agreement. Enter *A* at the prompt.

Finish Installation and Start Payara (GlassFish)	After Payara (GlassFish) has been installed, from the command line enter these commands:
	`> cd rootFolder/glassfish`
	`> chmod -R +x lib/ant/bin`
	`> lib/ant/bin/ant -f setup.xml`
	`> ./bin/asadmin start-domain domain1`

Copy files to endorsed folder	Copy **jaxws-api.jar** and **jaxb-api.jar** to the java endorsed folder. This folder is: *java-home*/**lib/endorsed** *java-home* is the directory where the runtime software is installed: the **jre** directory in the JDK. These files should come with the IBM AIX Java installation.
Test Installation	Open a browser and point it to the URL `http://yourServer:4848/` The default logon account will be **admin/adminadmin** (or the password may be the empty string).

Determining whether Payara (GlassFish) is Running

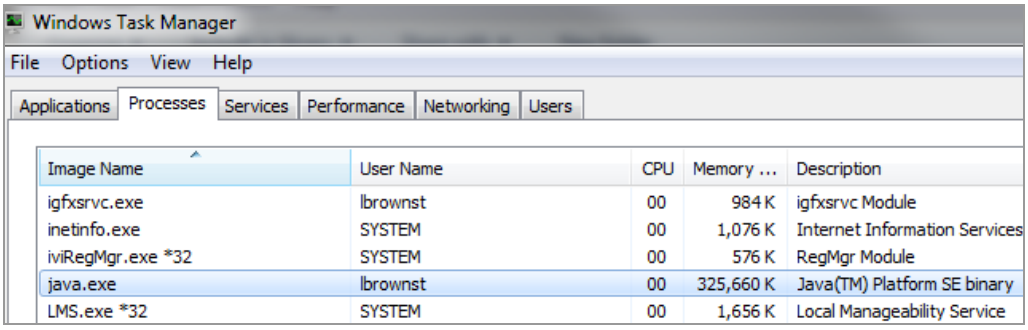
- If Payara (GlassFish) is running, you should see the Payara console when you open a browser and go to **`http://`***`server`***`:4848/`** (use the correct port for the Payara console).

  Example `http://dragon:4848/`

- Go to Payara's bin directory and issue this command:

  AIX `./asadmin list-domains`
  Windows `asadmin list-domains`

- On Windows, use Task Manager, where Payara appears as a large java.exe entry under the Processes tab:



Starting and Stopping Payara (GlassFish)

Payara (GlassFish) is a 3rd party application, so it is managed outside of the TIBCO Foresight suite of applications. Starting and stopping it are up to the system administrators.

Starting Payara (GlassFish) on Windows

- Using the Start menu:

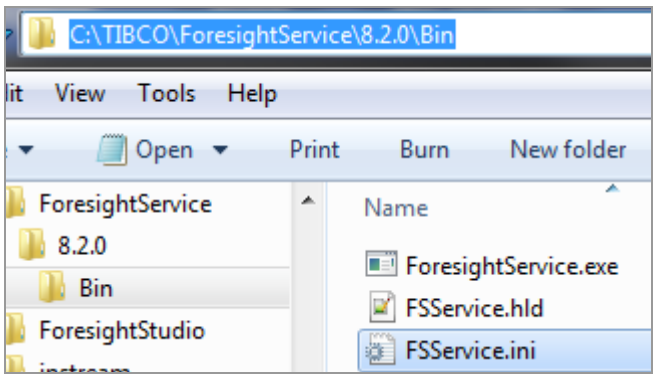
  Start | All Programs | Payara *xxx* **| Start Application Server**

  This will start Payara. However, it will stop running when the person who started it logs off.

- Using ForesightService:

  To get around the problem of Payara stopping when you logoff, run it as a process spawned by Foresight Service.

  Start ForesightService under Control Panel's Services. Be sure Payara is enabled in FSService.ini:



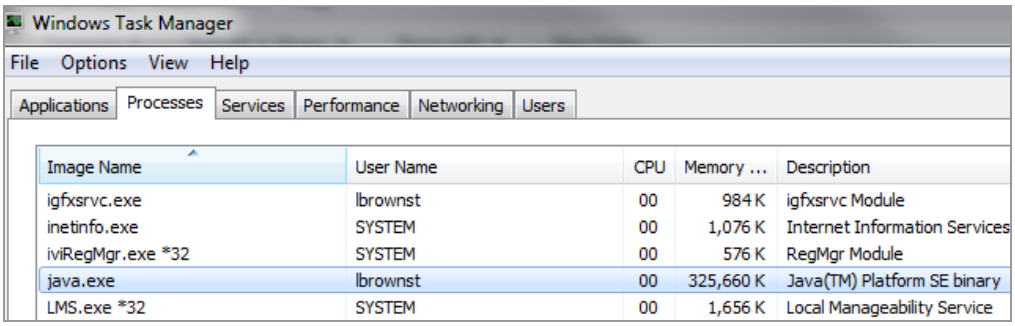
See Setting up Payara (GlassFish) to run as a Service on page 32 and Starting Archive in **TIB_transactioninsight_*n.n*_installation.pdf**.

Stopping Payara (GlassFish) on Windows

- Use the Start menu:

  Start | All Programs | Payara *xxx* **| Stop Application Server**

- Use **End Process** under Task Manager, where Payara appears as a large java.exe entry under the Processes tab.



Starting and Stopping Payara (GlassFish) on AIX or Windows:

Go to Payara **bin** folder and enter these commands

AIX:

```
./asadmin start-domain domain1
./asadmin stop-domain domain1
```

Windows:

```
asadmin.bat start-domain domain1
asadmin.bat stop-domain domain1
```

Opening the Payara (GlassFish) Admin Console

Windows or AIX	In a browser's address line: `http://` *`server`*`:4848/` (or whatever port is used for the Payara console)
Windows	**Start | Programs |** *Payara* **| Administration Console**

Payara (GlassFish) Configuration Problems

The Archive installation program adds some configuration to Payara (GlassFish).

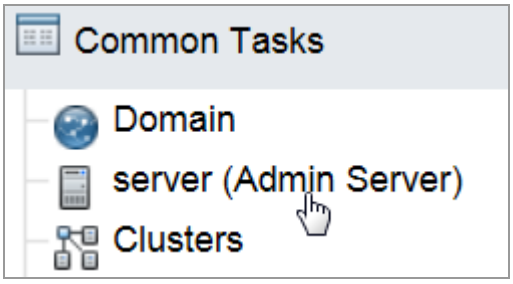
If there was a problem with the Payara configuration during Archive installation, perform the following tasks manually and then re-start Payara.

Payara (GlassFish) Configuration Problems on Windows

Set the com.foresightcorp.archiver.config Property

With Payara (GlassFish) running:

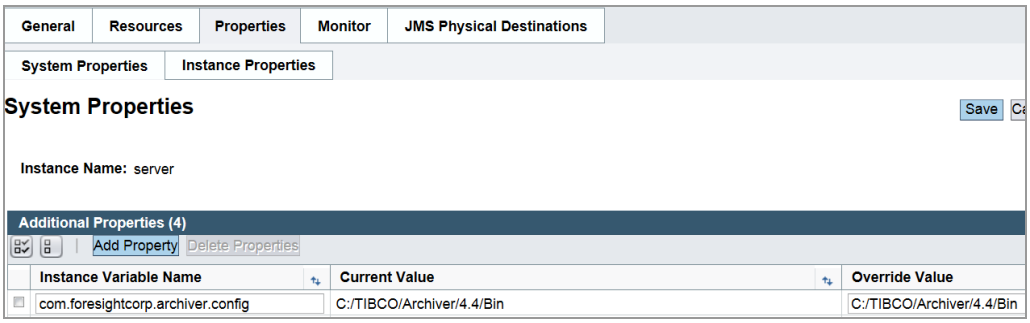
1. Open the Payara (GlassFish) admin console and log on.
2. Choose **Server (Admin Server) | Properties | System Properties | Add Property**.



For **Property Name**, use **com.foresightcorp.archiver.config**

For **Instance Variable Name** and **Override Value**, use the path to Foresight Archive and Retrieval System's Bin directory. (Example: C:/TIBCO/Archiver/5.2/Bin).

For version 3.1.x, use forward slashes in file paths.



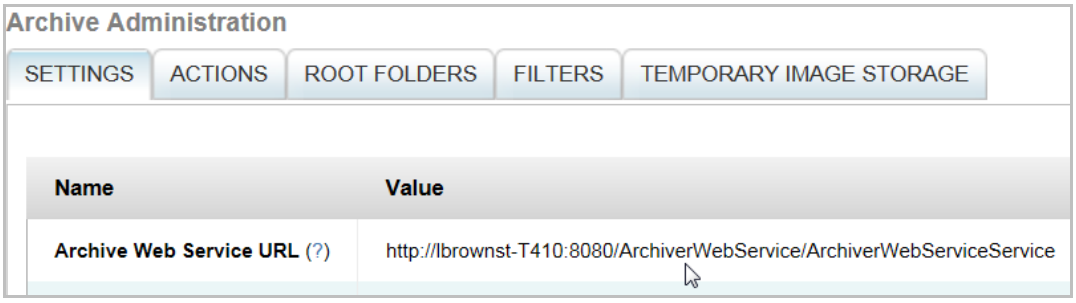
3. Save.

Deploy the web services:

1. On the right side of the Payara (GlassFish) admin console, click **Applications** and then click deploy.
2. Click **Browse,** go to Foresight Archive and Retrieval System's webServices directory, select ArchiverWebService.war, and click OK.
3. Repeat the deployment process for FileImport.war in the webServices folder.

Add the Foresight Archive and Retrieval System web service URL to the database

1. Logon to Transaction Insight and go to the Archive Portal.
2. Choose **Archive | Admin** and check the Foresight Archive and Retrieval System Web Service URL.



3. Test it with the PING button to the right of the URL as described in Testing Payara (GlassFish) Setup on page 30.

Payara (GlassFish) Configuration Problems on AIX

If you get an error while running **configure** (the script that sets up Foresight Archive and Retrieval System), that means that a Payara (GlassFish) property could not be set. You will need to do it manually:

1. With Payara running, log on to the Payara console at **http://<***yourServer***>:4848/** .
2. Click **server (Admin Server) | Properties | Add Property**.

   For Property Name, use **com.foresightcorp.archiver.config**.

   For Property Value, use *archiverRoot***\Bin** (use your own value of *archiverRoot*). Use forward slashes in paths for version 3.1.x.

   For Description use **Folder for Archive configuration files**.
3. Click **Save**.

(xRs is not needed for AIX.)

Testing Payara (GlassFish) Setup

To test Payara (GlassFish) configuration:

1. Start a browser and point it at this (use your own server's name):

   `http://yourServer:8080/ArchiverWebService/ArchiverWebServiceService?Tester`

2. In the parameter box for the **available** method, enter some string and click the button.

The response should show your input string and the contents of archiver.properties. Look for your database connection information.

Using PING:

1. Log on to the Foresight Archive and Retrieval System's Portal.
2. Choose **Archive | Admin** and check the Foresight Archive and Retrieval System's Web Service URL.
3. Test it with the PING button to the right of the URL:

You should see a report like this:

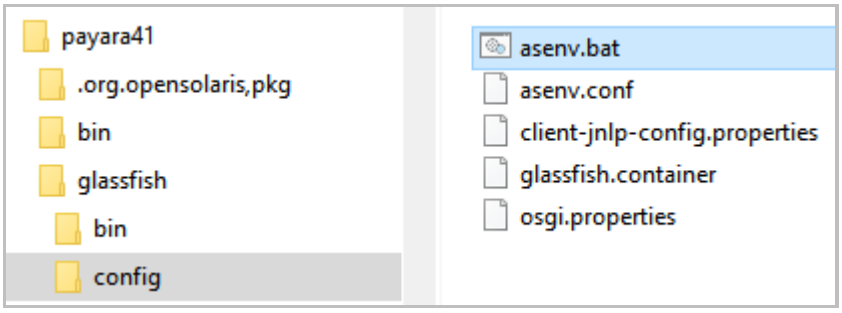
name	value
Glassfish Version	Payara Server 4.1.1.171.1 (build 139)
Java Endorsed Directories	C:\payara41\glassfish/modules/endorsed C:\payara41\glassfish/lib/endorsed
Java Home	C:\Program Files\Java\jdk1.8.0_131\jre
Java Temp Folder	C:\Users\ADMINI~1\AppData\Local\Temp\1\
Java Version	1.8.0_131
OS Name	Windows Server 2016
OS Version	10.0
Test Name	Ping All
User Name	Administrator
Web Service Name	Archiver Web Service
Archiver Version:	5.2.0, Build Number:005, BuildDate:2017-06-12 17:01:29
Host Name	10.97.194.120
Report Date	Tue Jun 13 15:20:44 EDT 2017

Be sure that everything passes. In this example, a directory is not accessible:

name	description	status
properties	Checks that the Archiver properties file can be found and contains the necessary properties for database connections.	Pass
database	Checks that a connection can be made to the database.	Pass
repository	Checks that the current repository root folder exists and that the web service has read/write permissions.	Pass
actions	Checks that each action folder exists and that the web service has write permissions for them.	Warning

Setting the Java Version for Payara (GlassFish)

1. Check here:



2. Look for a line similar to this:

```
set AS_JAVA=C:\Program Files\Java\jdk1.7.0_07
```

3. Update it if necessary.

4. Re-start Payara.

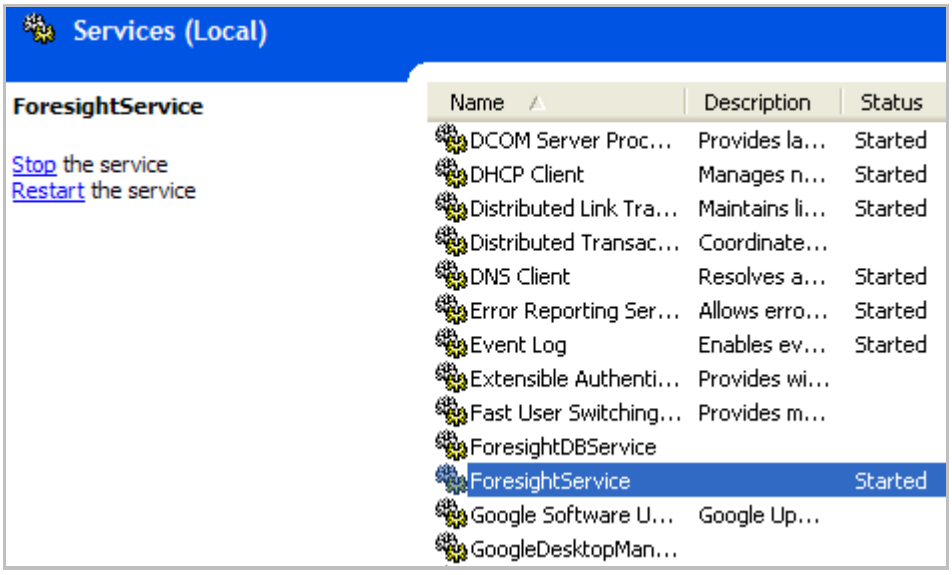
Setting up Payara (GlassFish) to run as a Service

When Payara (GlassFish) is installed on a Windows system, it runs as long as the user who installed it is logged on. When that user logs off, the Payara process stops.

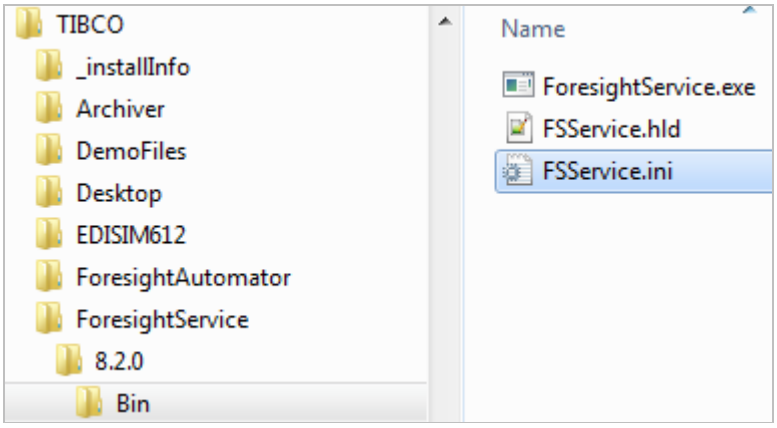
To get around this, use ForesightService. This Windows service will spawn processes that have been defined in its configuration file, FSService.ini. One of these is Payara.

To see if ForesightService has been installed:

1. Go to Control Panel's Services and see if it is there:



2. If it is not on the Services list, find FSService.ini:



Open a command window, go to the folder where **FSService.ini** resides, and enter the command:

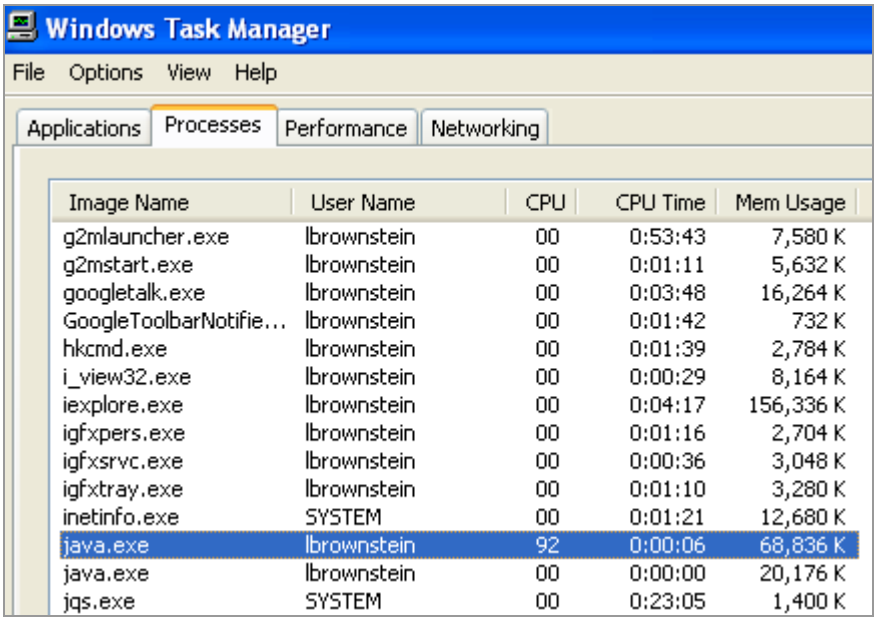
```
ForesightService.exe -Service
```

Once the FSService.ini has been configured to start Payara, and ForesightService starts, it will start Payara.

Setting up ForesightService to run Payara (GlassFish)

1. If ForesightService is running, stop it under Control Panel's Services.
2. If Payara (GlassFish) is running, stop it as described on page 25.

   As an alternative, you can stop it with Task Manager. It will appear as a large Java.exe entry in Task Manager's Processes tab.



3. Open FSService.ini.

4. Remove the leading semi-colon from the line that defines **ArchiverGlassfishServer** (some details may vary depending on your installation).

```
[Servers]
AutomatorStart=C:\Foresight\ForesightAutom

AutomatorStart=C:\Foresight\ForesightAutom
AutomatorStart=C:\Foresight\ForesightAutom
AutomatorStart=C:\Foresight\ForesightAutom
AutomatorStart=C:\Foresight\ForesightAutom
AutomatorStart=C:\Foresight\ForesightAutom
;ArchiverGlassfishServer=C:\Program Files\
;
```

5. Save the file.

6. Open Control Panel's Services and re-start **ForesightService**.

Task Manager now shows ForesightService and a large Java process that represents Payara (GlassFish), and possibly some ForesightAutomator processes that are running workflows.

Image Name	User Name	CPU	Memory ...	Description
EvtEng.exe	SYSTEM	00	5,220 K	Intel(R) PROSet/Wireless Eve
explorer.exe	lbrownst	00	42,660 K	Windows Explorer
ForesightAutomator.exe	SYSTEM	00	32,752 K	ForesightAutomator.exe
ForesightService.exe *32	SYSTEM	00	1,036 K	ForesightService Module
FrameworkService.exe *32	SYSTEM	00	1,436 K	Framework Service
googletalk.exe *32	lbrownst	00	11,152 K	Google Talk
GoogleToolbarNotifier.exe *32	lbrownst	00	1,500 K	GoogleToolbarNotifier
GoogleToolbarUser_32.exe *32	lbrownst	00	1,640 K	Google Toolbar Broker
Greenshot.exe *32	lbrownst	00	21,212 K	Greenshot
hkcmd.exe	lbrownst	00	940 K	hkcmd Module
ibmpmsvc.exe	SYSTEM	00	144 K	ThinkPad Power Management
iexplore.exe	lbrownst	00	11,844 K	Internet Explorer
iexplore.exe	lbrownst	00	66,860 K	Internet Explorer
igfxext.exe	lbrownst	00	620 K	igfxext Module
igfxpers.exe	lbrownst	00	1,252 K	persistence Module
igfxsrvc.exe	lbrownst	00	1,028 K	igfxsrvc Module
inetinfo.exe	SYSTEM	00	904 K	Internet Information Services
iviRegMgr.exe *32	SYSTEM	00	472 K	RegMgr Module
java.exe	lbrownst	00	367,120 K	Java(TM) Platform SE binary
LMS.exe *32	SYSTEM	00	1,636 K	Local Manageability Service

Troubleshooting Payara (GlassFish)

You will probably need the help of a system administrator for this.

1. Is Payara actually installed?

   Ask the system administrator to confirm this.

2. Is Payara running?

   Open a browser on the Payara host system and go to the Payara Administration Console at **http://***server***:4848/** (or whatever the port is for the console.) If you can't connect, ask the system administrator to start Payara (see below for details).

3. Is the URL in Archive's database? (check the Metadata table where MetadataKey == ARCHIVER_WS_URL).

4. Is the URL valid?

   Try this in the browser:

   url`?wsdl`

   where *url* is the value from the Metadata table.

5. Are the ArchiverFileImport, ArchiverWebService, and OpMonWS Web Services deployed?

   Windows
   In the Payara Administration Console, click on **Applications**.

   AIX
   In the Payara Administrative Console, click on **Applications | Web Applications**.

6. Is the System property **com.foresightcorp.archiver.config** defined in Payara and pointing to the directory where Foresight Archive and Retrieval System's properties files are stored?

   Windows
   In the Payara Administration Console, click on **Enterprise Server | System Properties**.

   AIX
   In the Payara Administration Console, click on **Configuration | System Properties**.

7. For Foresight Archive and Retrieval System, is the **Archiver_UI_Importer** workflow running? This workflow and Payara should be running whenever the Foresight Archive and Retrieval System portal is available.

Trying the Available Method

Windows

In the Payara Administration Console, click on **Applications | Archiver File Import | View Endpoint | Tester** link.

In the parameter box for the available method, enter some characters and click the available button. The response should include your input plus the fully expanded paths of the configuration files as well as some properties.

AIX

Open a browser and use this URL:

`http://`*`server`*`:8080/ArchiverFileImport/ArchiverFileImportService?Tester`

8080 is the default Payara (GlassFish) port, although your site might have changed it. In the parameter box for the available method, enter some data and click the available button. The response should include your input plus the directory path to the property files.

AIX Payara (GlassFish) Permissions

Each Foresight Archive and Retrieval System task checks to see if the admin owns the root folder where the work will be done. The ideal scenario is that the admin has installed both Foresight Archive and Retrieval System and Payara. That will ensure the necessary permissions have been set.

Task	Own Foresight Archive and Retrieval System?	Own Payara (GlassFish)?
Set Archiver properties	√	
Set Payara (GlassFish) properties	No permissions required	No permissions required
Deploy Foresight Archive and Retrieval System Web Service	√	√
Deploy File Import Web Service	√	√
Stop/Start Payara (GlassFish)		√

Questions to ask:

- Under which account was Payara started?
- What group does it belong to?
- Does it have the necessary permissions to read/write files to the Repository? This includes creating directories.
- Does it have the necessary permissions to write files to the various directories associated with Transmission/Document viewing in Foresight Archive and Retrieval System portal and any other directories associated with user-defined actions in Foresight Archive and Retrieval System Portal?
- Does it have permissions to read the property files from the configuration directory and to write to the log files for the web services

Deploying a Web Service to Payara (GlassFish)

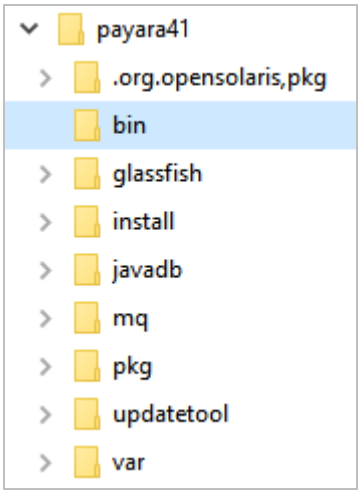
Overview

This section provides instructions for deploying a web service to Payara (GlassFish).

Setting up a new Web Service

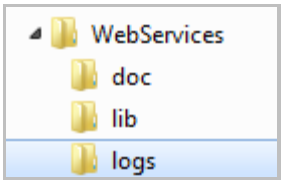
This information is for new installation of the web service. To upgrade, see Upgrading (or Hot Fixing) an Existing Web Service on page 41.

<PAYARA_ROOT> refers to the Payara folder that contains the \bin directory. In this example, <PAYARA_ROOT> is **c:\payara41**:



Steps:

1. If <TransactionInsight_ROOT>\<version>\WebServices does not already exist, create it and create the subfolders doc, lib and logs (refer to the following example):



2. Place the .war file for the web service you are deploying in the WebServices folder.

3. In the doc directory, check **pwd.txt** and update the Payara (GlassFish) password in it. If that file does not exist, create a new file with that name and place the following into it: `AS_ADMIN_PASSWORD=<Payara(GlassFish)AdminPassword>`

   (With <Payara(GlassFish)AdminPassword>replaced by the actual password for the admin account.)

4. Open a command window and enter the following command, updating the paths and Payara (GlassFish) user to match yours. Variable information is underlined:

```
"<PAYARA_ROOT>\glassfish\bin\asadmin.bat" --user "admin"
--passwordfile "<TransactionInsight_ROOT>\5.1.0\WebServices\doc\pwd.txt"
-I false deploy --force=true "<TransactionInsight_ROOT>\5.1.0\WebServices\
<WARfileNameWithExtension>"
```

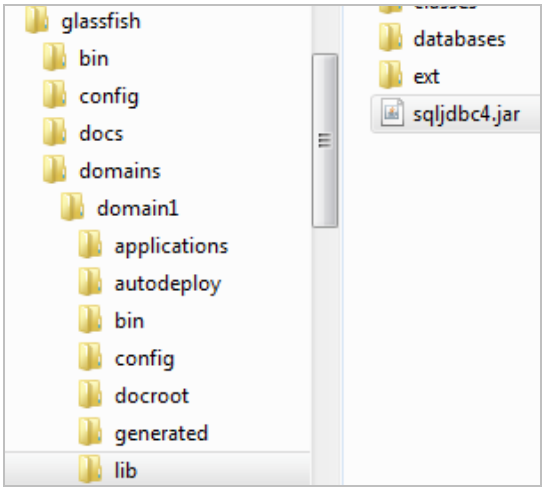
5. Enter one of these commands from the command window. Update the paths for the underlined sections:

   For Oracle Transaction Insight database:

```
copy "path to Oracle Install\ojdbc6.jar"
"<PAYARA_ROOT>\glassfish\ domains\domain1\lib"
```

   For SQL Server Transaction Insight database:

```
copy "sqljdbc4.jar" (available here: http://msdn.microsoft.com/en-
us/sqlserver/aa937724.aspx)to
"<PAYARA_ROOT>\glassfish\domains\domain1\lib"
```



Upgrading (or Hot Fixing) an Existing Web Service

Use the following steps to update a web service or apply a hotfix:

1. Read the instructions that accompany the update. Pay careful attention to the extraction location for the .war file and alter the next step appropriately.
2. Open a command window and enter the following command, updating the paths and Payara (GlassFish) user to match yours. Variable information is underlined:

```
"<PAYARA_ROOT>\glassfish\bin\asadmin.bat" --user "admin"
--passwordfile "<TransactionInsight_ROOT>\5.1.0\WebServices\doc\pwd.txt"
-I false deploy --force=true "<TransactionInsight_ROOT>\5.1.0\WebServices\
<WARfileNameWithExtension>"
```

Setting up a System Property in Payara (GlassFish)

It may be necessary to set up system properties within Payara (GlassFish). The specific web service will detail which system properties need to be set up and what the value of that property should be. You can set up system properties from the command line with the following:

```
"<PAYARA_ROOT>\glassfish\bin\asadmin.bat" create-system-
properties <PropertyName>=<PropertyValue>
```

NOTE: If the PropertyValue is a path, you must add backslashes to the colon and folder separators within windows if you are using backslashes as the folder separator.

Example: To use the path to the Foresight Archive and Retrieval System Bin folder as the Property Value, enter this: `C\:\\TIBCO\\Archiver\\5.2\\Bin`

When Payara parses this value, it removes the extra backslashes. If you use forward slashes, the additional slash is unnecessary.

Setting up a Web Service Connection Pool

1. Create a connection pool.

   a. Refer to your Web.config's **DB:TransactionInsight** property to find the name of the database server, the database name, the port, the user name, and the password.

   b. From the Payara (GlassFish) admin console, click **Resources | JDBC | JDBC Connection Pools | New**.

   c. Under **General Settings**, set these:

      Pool Name One of these:

      `TransactionInsight_Oracle`
      `TransactionInsight_SqlServer`

      Resource Type `javax.sql.DataSource`

      Database Vendor One of these:

      `Oracle`
      `MicrosoftSqlServer`

   d. Click **Next** | **Additional Properties** and set these properties:

      databaseName *Your Transaction Insight database name*

      Description Transaction Insight Database

      Password *The SQL Server or Oracle password*

      ServerName *The Transaction Insight database server*

      User *The Transaction Insight database user*

   e. Click **Add Property** and type URL for Name.

   f. Add the **Value**, using forward slashes in all cases:

      For SQL Server:

      `jdbc:sqlserver://`*serverName*`:`*port*`;databaseName=`*dbName*`;user=`*user*`;password=`*pass*;

      Example:

      `jdbc:sqlserver://localhost;databaseName=TIDemo;user=sa;password=ndune891;`

      For Oracle:

      `jdbc:oracle:thin:`*user*`/`*pass*`@//`*serverName*`:`*port*`/`*sid*

g. Click **Save**.

h. Test by clicking **TransactionInsight_Oracle** or **TransactionInsight_SqlServer** in the Connection Pool column.

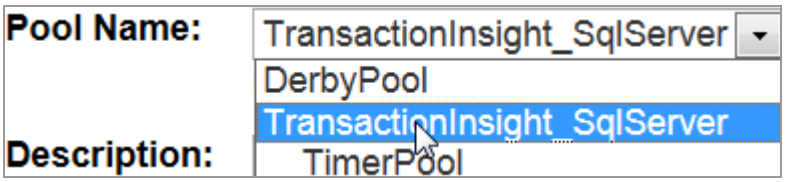
   Click **Ping** and look at the top for the message "Ping Succeeded."

2. Still in the Payara (GlassFish) admin console, click **Resources | JDBC | JDBC Resources | New** and create this resource:

JNDI Name `jdbc/TransactionInsightDB`

Pool Name Select your newly created connection pool.

Example:



Description `JDBC Resource for TI Database`

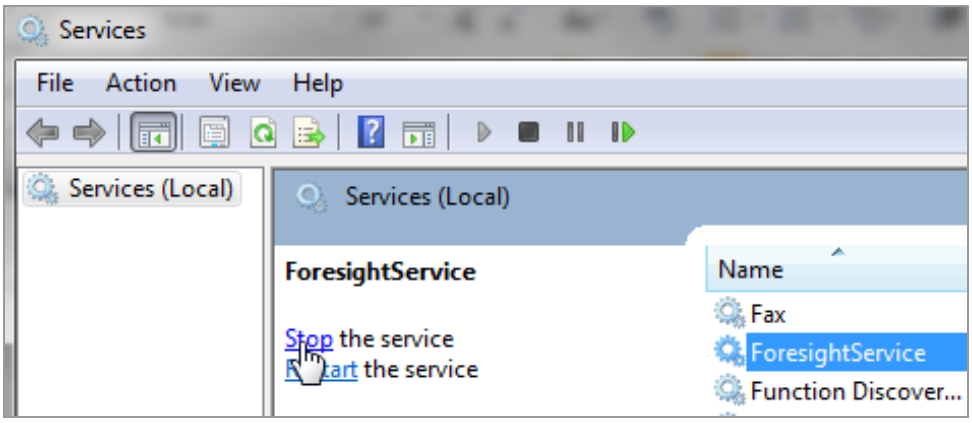
Click **OK**.

Example:

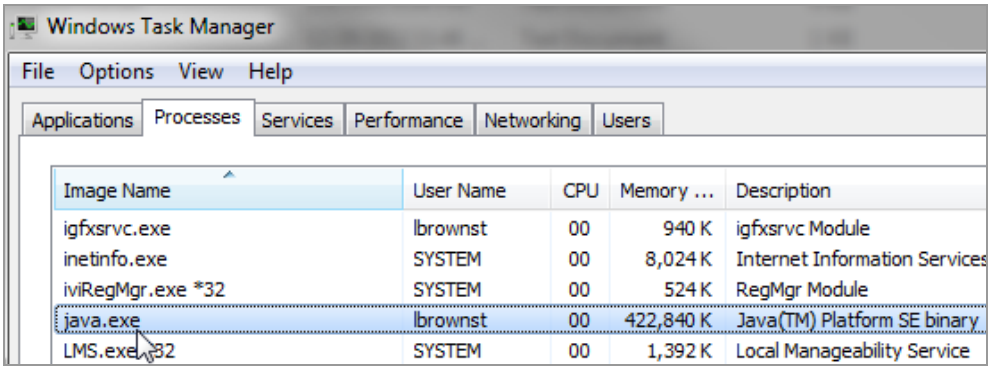
JNDI Name	Enabled	Connection Pool	Description
jdbc/TransactionInsightDB	✔	TransactionInsight_SqlServer	JDBC Resource for TI Database

Restarting Payara (GlassFish)

1. Stop **Foresight Service** from Services:



2. Stop Payara (GlassFish) from Task Manager. This will be a large Java process:



3. Start Foresight Service from Services.

4. Select **Start -> All Programs -> Payara -> Start Application Server**