PEtALS-SE-XSLT

This document explain how to install and configure the petals-se-xslt JBI component.

PEtALS Team
Marie Sauvage <>
- June 2007 -

OW2 Consortium
# Table of Contents

PEtALS-SE-XSLT ................................................................................................................................. 5  
1. Component Configuration .................................................................................................................. 6  
2. Service Configuration .......................................................................................................................... 7  
   2.1. Transform XML messages ............................................................................................................... 7  
      2.1.1. Service Unit descriptor ........................................................................................................... 7  
      2.1.2. Service Unit content ............................................................................................................. 8  
      2.1.3. Usage ........................................................................................................................................ 8  
3. Samples ............................................................................................................................................... 9
List of Figures

2.1. The XSLT Service Engine ........................................................................................................... 7
List of Tables

1.1. component installation configuration attributes ................................................................. 6
1.2. Advanced configuration of the component ........................................................................... 6
1.3. Interceptors configuration in the component ....................................................................... 6
2.1. Service Unit attributes to provide services ........................................................................ 8
2.2. Advanced configuration of Service Unit (provides elements) ............................................. 8
2.3. Interceptors configuration in the Service Unit .................................................................... 8
PEtALS-SE-XSLT

This component allows to process xml transformations based on xsl style sheet. It creates an XML output from a given XML source content and an XSLT Stylesheet defined in the JBI description of a Service Unit.

If you want to have more details about XSLT specification, you can consult it at this url: http://www.w3.org/TR/xslt
Chapter 1. Component Configuration

The following attributes can be set during the installation phase to configure the component, using the params element of the jbi-install-component ANT task:

*no configuration for this component*

Table 1.1. component installation configuration attributes

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
<th>Default</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 1.2. Advanced configuration of the component

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Default</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>pool-size</td>
<td>Number of threads listening to messages coming from the JBI container (JBIListeners).</td>
<td>0</td>
<td>No</td>
</tr>
<tr>
<td>ignored-status</td>
<td>Status of messages exchanges that component must ignore.</td>
<td>DONE_AND_ERROR_IGNORED</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>Accepted values : DONE_AND_ERROR_IGNORED, DONE_IGNORED, ERROR_IGNORED or NOTHING_IGNORED</td>
<td></td>
<td></td>
</tr>
<tr>
<td>jbi-listener-class-name</td>
<td>Fully qualified name of the class extending AbstractJBIListener</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>external-listener-class-name</td>
<td>Fully qualified name of the class extending AbstractExternalListener</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>properties-file</td>
<td>Name of the file containing values of keys used as reference by other parameters. To be able to configure a service-unit, you will use a key that has its value hosted by the component (ie. CDK documentation). The value of this parameter is :</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• whether an URL,</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• or a file relative to the directory defined by the environment variable PETALS_HOME.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 1.3. Interceptors configuration in the component

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Default</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>class</td>
<td>Name of the interceptor class. This class must extend the abstract class org.objectweb.petals.component.common.interceptor.Interceptor. This class have to be present in the classloader, in component or CF or in a shared library.</td>
<td></td>
<td>Yes</td>
</tr>
<tr>
<td>name</td>
<td>Name of the interceptor. This name will be used for additional configuration in the SU.</td>
<td>class name</td>
<td>No</td>
</tr>
<tr>
<td>active</td>
<td>Interceptor is active for all SU.</td>
<td>true</td>
<td>No</td>
</tr>
</tbody>
</table>
Chapter 2. Service Configuration

2.1. Transform XML messages

PROVIDE SERVICE : Expose an external service in the JBI environment

2.1.1. Service Unit descriptor

For each xsl style sheet one endpoint is activated. Then, a client can use this service sending a request to this endpoint.

Figure 2.1. The XSLT Service Engine

To activate a new endpoint, you must deploy a service unit that contains an endpoint definition in a provides node and an xsl stylesheet. Here is a sample of a su descriptor that activates a new Endpoint (XsltEndpoint) linked to a test.xsl stylesheet.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<jbi:jbi xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:petals="http://petals.ow2.org/extensions"
    xmlns:tns="http://petals.ow2.org/"
    xmlns:jbi="http://java.sun.com/xml/ns/jbi" version="1.0">
  <jbi:services binding-component="false">
    <jbi:provides interface-name="tns:BottleInformationAsHTMLInterface"
        service-name="tns:BottleInformationAsHTMLService"
        endpoint-name="BottleInformationAsHTMLEndpoint">
      <petals:wsdl>
        <!-- URL of specific contract service if exists -->
        </petals:wsdl>
      <!-- Specific interceptors for this service unit -->
      <petals:interceptors/>
      <!-- Specific parameters for this service unit -->
      <petals:params>
        <petals:param name="xsl.output-attachment-name">output.html</petals:param>
        <petals:param name="xsl.file">test.xsl</petals:param>
      </petals:params>
    </jbi:provides>
  </jbi:services>
</jbi:jbi>
```
Table 2.1. Service Unit attributes to provide services

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
<th>Default Value</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>provides</td>
<td>Name of the JBI service that will be activated to expose the XSLT Stylesheet endpoint into the JBI environment. interface (qname), service (qname) and endpoint (string) name are required.</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>xsl.file</td>
<td>Location of the xsl stylesheet. This path can be a relative path, starting at the SU root or a classpath resource from the SU point of view (the xsl stylesheet can be embedded in a JAR file inside the SU).</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>xsl.output-attachment-name</td>
<td>If defined, the result of the transformation is set as an attached file, with the specified name.</td>
<td>No</td>
<td></td>
</tr>
</tbody>
</table>

Table 2.2. Advanced configuration of Service Unit (provides elements)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Default</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>wsdl</td>
<td>path to a wsdl file describing services and operations offered by an endpoint activated by the SU. This extension is only usable with provides fields.</td>
<td>No</td>
<td></td>
</tr>
</tbody>
</table>

The path can be a url "http" or "file" or relative to the root directory of the SU archive. Ex : "file://user/ofabre/test.wsdl" or "/WSDL/test.wsdl"

If no wsdl path is specified, a simplified description will automatically be written by the CF.

Table 2.3. Interceptors configuration in the Service Unit

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Default</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the interceptor to use. That's the name defined in the component.</td>
<td>Yes</td>
<td></td>
</tr>
</tbody>
</table>

2.1.2. Service Unit content

The Service Unit has to contain the following elements, packaged in an archive:

- The META-INF/jbi.xml descriptor file, has described above,
- An optional wsdl file describing the related service

```
service-unit.zip
 + META-INF
    - jbi.xml (as defined above)
    - service.wsdl (optional)
    - test.xsl (required)
```

2.1.3. Usage

When this endpoint is reached, the content of the incoming normalized message (an xml content) of the message exchange is processed with the linked xsl style sheet. The processing result is returned in the outgoing normalized message content of the message exchange.

**Caution**

For the moment, only InOut and InOptionalOut message exchange pattern are allowed.
Chapter 3. Samples

See the following Service Assembly sample that illustrates the configuration of this component: