FUSE™ Services Framework

Migrating to 2.2.x

Version 2.2.x
April 2009

Migrating to 2.2.x

Version 2.2.x

Publication date 17 Jul 2009

Legal Notices

While the information in this publication is believed to be accurate Progress Software Corporation makes no warranty of any kind to this material including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. Progress Software Corporation shall not be liable for errors contained herein, or for incidental or consequential damages in connection with the furnishing, performance or use of this material.

Table of Contents

List of Examples

Part I. Migrating from version 2.0 to version 2.1

Chapter 1. Code Changes

FUSE Services Framework 2.1 supports version 2.1 of the JAX-WS specification and version 2.1 of the JAXB specification. The code used by the newer specifications are incompatible with previous versions of FUSE Services Framework.

JAXB 2.1 annotations

The code generated by the 2.1 code generators adds some JAXB 2.1 specific annotations. These annotations are not compatible with previous versions of FUSE Services Framework.

WS-Addressing

JAX-WS 2.1 supports WS-Addressing directly in the APIs. WSDLs that use the EnpointReferenceType will now generate the JAX-WS 2.1 `EndpointReference` instead of the FUSE Services Framework proprietary type that was generated in FUSE Services Framework 2.0.

You can use the **wsdl2java** command's `-noAddressBinding` option to force the generation of the old-style endpoint reference code.

Chapter 2. Tooling Changes

FUSE Services Framework 2.1 has a new tool for generating service support files. It also includes a new JavaScript code generation tool.

WSDL generation tooling

The tool has been removed from FUSE Services Framework. To generate WSDL from an SEI, you can use the **java2ws** command's `-wsdl` option.

Web service support file generation

FUSE Services Framework has a new **java2ws** command. The new command generates a number of support files from an SEI. These files include:

- WSDL
- the server code needed to deploy the service as a POJO
- client code for accessing the service
- wrapper and fault beans

JavaScript generation

FUSE Services Framework has a new **wsdl2js** command that generates all of the needed JavaScript classes needed to implement a service in Javascript.

Maven plug-ins

The Maven plug-in has been removed. It has been replaced by a **java2ws** plug-in.

Chapter 3. Runtime Changes

FUSE Services Framework 2.1 now support Java 6. It also introduces some new runtime dependencies.

Java 6

FUSE Services Framework 2.1 supports Java 6 update 4 and later. If you need to use Java 6 update 3 or earlier, you will need to add the `jaxws-api.jar` and `jaxb.jar` to the `jre/lib/endorsed` directories.

ASM

The JAX-WS frontend now requires asm 2.x or 3.x to be able to process some of the JAXB annotations on the SEI. There can be conflicts with other applications, such as Hibernate, that use asm.

If you don't use those annotations on the SEI or if you have generated wrapper classes, you can remove `asm jar` from your installation.

If you use the annotations, the workaround for Hibernate is to remove Hibernate's asm 1.x jar used by it and replace Hibernate's `cglib jar` with the `cglib-nodeps jar` that includes a special internal version of asm that would not conflict with the 2.x/3.x version used by FUSE Services Framework.

Part II. Migrating from version 2.1 to version 2.2.x

Chapter 4. Code Changes

FUSE Services Framework 2.2.x updates the performance and spec compliance of some of its front-ends.

JAX-RS 1.0

Version 2.2.x of FUSE Services Framework supports the 1.0 release of the JAX-RS specification. Applications written using the JAX-RS APIs available previous releases will need to be updated to work with this release.

Aegis front-end

The building of the schema in the Aegis type system has changed from using JDom to using XmlSchema directly. This avoids a 3 hop JDOM -> DOM -> XmlSchema performance penalty, but requires changes to any custom Aegis type objects.

The long deprecated `org.codehaus.xfire.aegis.type.java5` annotations have been removed. When migrating applications from XFire, you will need to update to use the JAXB annotations or their equivalents in `org.apache.cxf.aegis.type.java5`.

Chapter 5. Tooling Changes

FUSE Services Framework 2.2.x has updated its Maven tooling to follow Maven best practices.

Maven wsdl2java plug-in

The wsdl2java plug-in now defaults to placing generated code into `${project.build.directory}/generated-sources/cxf`. This follows the best practices for using Maven.

You can change the default behavior using the `sourceRoot` variable.

Chapter 6. Runtime Changes

FUSE Services Framework 2.2.x has changed two key runtime behaviors.

Policy engine

In 2.2.x the policy engine is turned on by default. This means that WS-Policy assertions contained in WSDL documents are evaluated. If the runtime supports a given policy assertion, it will act accordingly.

To disable the policy engine you can add the configuration snipit in Example 6.1 on page 21 to your application's configuration.

Example 6.1. Turning Off the Policy Engine

```
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:cxf="http://cxf.apache.org/core"
       xmlns:p="http://cxf.apache.org/policy"
       xsi:schemaLocation="
http://cxf.apache.org/core http://cxf.apache.org/schemas/core.xsd
http://cxf.apache.org/policy http://cxf.apache.org/schemas/policy.xsd
http://www.springframework.org/schema/beans http://www.springframe
work.org/schema/beans/spring-beans-2.0.xsd">
  ...
  <cxf:bus>
    <cxf:features>
      <p:policies enabled="false" />
    </cxf:features>
  </cxf:bus>
  ...
</beans>
```

JMS

Starting in version 2.2.x, the JMS transport defaults to using the JMS 1.1 API. If you are using a JMS 1.0 provider you can change this behavior by setting the `useJms11` configuration attribute to `false`.