



FUSE Services Framework

Using the FUSE Services Framework Library

Version 2.0
December 2007

Making Software Work Together™

Using the FUSE Services Framework Library

IONA Technologies

Version 2.0

Published 28 Apr 2008

Copyright © 2001-2008 IONA Technologies PLC

Trademark and Disclaimer Notice

IONA Technologies PLC and/or its subsidiaries may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this publication. Except as expressly provided in any written license agreement from IONA Technologies PLC, the furnishing of this publication does not give you any license to these patents, trademarks, copyrights, or other intellectual property. Any rights not expressly granted herein are reserved.

IONA, IONA Technologies, the IONA logo, Orbix, High Performance Integration, Artix, FUSE, and Making Software Work Together are trademarks or registered trademarks of IONA Technologies PLC and/or its subsidiaries.

Java and J2EE are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries. CORBA is a trademark or registered trademark of the Object Management Group, Inc. in the United States and other countries. All other trademarks that appear herein are the property of their respective owners.

While the information in this publication is believed to be accurate, IONA Technologies PLC 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. IONA 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.

Copyright Notice

No part of this publication may be reproduced, stored in a retrieval system or transmitted, in any form or by any means, photocopying, recording or otherwise, without prior written consent of IONA Technologies PLC. No third-party intellectual property right liability is assumed with respect to the use of the information contained herein. IONA Technologies PLC assumes no responsibility for errors or omissions contained in this publication. This publication and features described herein are subject to change without notice. Portions of this document may include Apache Foundation documentation, all rights reserved.

Table of Contents

Library Overview	5
Document Conventions	6
The FUSE Services Framework Library	8
Open Source Project Resources	10
Reading Paths	11
Service Consumer Developers	12
Java-First Service Developers	14
WSDL-First Service Developers	16
RESTful Service Developers	18
JavaScript Service Developers	20

Library Overview

Table of Contents

Document Conventions	6
The FUSE Services Framework Library	8
Open Source Project Resources	10

Document Conventions

Typographical conventions

This book uses the following typographical conventions:

<code>fixed width</code>	<p>Fixed width (Courier font) in normal text represents portions of code and literal names of items such as classes, functions, variables, and data structures. For example, text might refer to the <code>javax.xml.ws.Endpoint</code> class.</p> <p>Constant width paragraphs represent code examples or information a system displays on the screen. For example:</p> <pre>import java.util.logging.Logger;</pre>
<i>Fixed width italic</i>	<p>Fixed width italic words or characters in code and commands represent variable values you must supply, such as arguments to commands or path names for your particular system. For example:</p> <pre>% cd /users/YourUserName</pre>
<i>Italic</i>	Italic words in normal text represent <i>emphasis</i> and introduce <i>new terms</i> .
Bold	Bold words in normal text represent graphical user interface components such as menu commands and dialog boxes. For example: the User Preferences dialog.

Keying conventions






This book uses the following keying conventions:

No prompt	When a command's format is the same for multiple platforms, the command prompt is not shown.
%	A percent sign represents the UNIX command shell prompt for a command that does not require root privileges.
#	A number sign represents the UNIX command shell prompt for a command that requires root privileges.
>	The notation > represents the MS-DOS or Windows command prompt.
...	Horizontal or vertical ellipses in format and syntax descriptions indicate that material has been eliminated to simplify a discussion.
[]	Brackets enclose optional items in format and syntax descriptions.
{ }	Braces enclose a list from which you must choose an item in format and syntax descriptions.

	In format and syntax descriptions, a vertical bar separates items in a list of choices enclosed in {} (braces).
--	---

Admonition conventions

This book uses the following conventions for admonitions:

	Notes display information that may be useful, but not critical.
	Tips provide hints about completing a task or using a tool. They may also provide information about workarounds to possible problems.
	Important notes display information that is critical to the task at hand.
	Cautions display information about likely errors that can be encountered. These errors are unlikely to cause damage to your data or your systems.
	Warnings display information about errors that may cause damage to your systems. Possible damage from these errors include system failures and loss of data.

The FUSE Services Framework Library

The FUSE Services Framework documentation library consists of the following books:

- The Glossary [<http://open.iona.com/docs/glossary/index.html>] provides definitions for commonly used terms relating to Web services and using FUSE Services Framework.
- Writing WSDL Contracts provides an overview to writing logical interfaces using WSDL.
- Developing Applications Using JAX-WS provides detailed information on developing services using the JAX-WS APIs.
- Developing RESTful Services describes how to develop and deploy RESTful applications using FUSE Services Framework.
- Developing Applications with Dynamic Languages describes how to develop and deploy distributed applications using dynamic languages such as JavaScript.
- Using the SOAP Binding provides detailed information on using the FUSE Services Framework's SOAP binding.
- Using the XML Binding provides detailed information on using the FUSE Services Framework's XML binding.
- Using the HTTP Transport provides detailed information on using the FUSE Services Framework's HTTP transport.
- Using the JMS Transport provides detailed information on using the FUSE Services Framework's JMS transport.
- FUSE Services Framework Deployment Guide provides detailed information on using the FUSE Services Framework's JMS transport.
- The Security Guide provides details on configuring the FUSE Services Framework to work in applications requiring security.
- FUSE Services Framework Command Reference is a quick reference to FUSE Services Framework's command-line tools.

In addition to the above books, you may also want to read the documentation for each of the components that FUSE Services Framework bundles. This

documentation is available from the projects responsible for developing the component.

Open Source Project Resources

Apache Incubator CXF

Web site: <http://incubator.apache.org/cxf/>

User's list: <cxf-user@incubator.apache.org>

Apache Tomcat

Web site: <http://tomcat.apache.org/>

User's list: <users@tomcat.apache.org>

Apache ActiveMQ

Web site: <http://activemq.apache.org/>

User's list: <users@@activemq.apache.org>

Apache Camel

Web site:
<http://activemq.apache.org/camel/enterprise-integration-patterns.html>

User's list: <camel-user@activemq.apache.org>

Apache ServiceMix

Web site: <http://servicemix.org/site/home.html>

User's list: <servicemix-users@geronimo.apache.org>

Reading Paths

Table of Contents

Service Consumer Developers	12
Java-First Service Developers	14
WSDL-First Service Developers	16
RESTful Service Developers	18
JavaScript Service Developers	20

Service Consumer Developers

Overview

Developing a service consumer is, perhaps, the most basic task a user of FUSE Services Framework would perform. In general, consumer developers will be given a WSDL contract and need to generate the stubs needed to make requests on the remote service. This reading path provides a quick introduction to WSDL, a description of generating the JAX-WS stubs, and a discussion of how to configure your consumer for deployment.

Path

For a quick introduction to developing service consumers, you can read the following:

1. *Introducing WSDL Contracts* in *Writing WSDL Contracts* provides a quick overview of the contents of a WSDL document.
 2. *Developing a Consumer From a WSDL Contract* in *Developing Applications Using JAX-WS* describes how to develop a service consumer using a provided WSDL document. It discusses how to use the command-line tools to generate the stubs and what additional code is required.
 3. *Developing a Consumer Without a WSDL Contract* in *Developing Applications Using JAX-WS* describes how to develop a service consumer if you are provided a Java interface instead of a WSDL document.
 4. *Getting Started* in *FUSE Services Framework Deployment Guide* describes how to set up your environment to run a FUSE Services Framework application.
 5. *FUSE Services Framework Configuration* in *FUSE Services Framework Deployment Guide* provides details about the files used to configure the FUSE Services Framework runtime.
-

Next steps

The following provide additional information about consumer development:

- *Developing Asynchronous Applications* in *Developing Applications Using JAX-WS* describes how to develop consumers that make asynchronous invocations on remote services.

- Using XML in a Consumer with the `Dispatch` Interface in *Developing Applications Using JAX-WS* describes how to use the `Dispatch` interface to develop consumers that work with raw XML messages.
- *Working with Contexts* in *Developing Applications Using JAX-WS* describes how to use the JAX-WS context mechanism.
- Using the SOAP Binding provides detailed information on using the FUSE Services Framework's SOAP binding.
- *HTTP Consumer Endpoint Configuration* in *Using the HTTP Transport* describes how to configure consumer-side HTTP endpoints.
- *Using the Decoupled HTTP Transport* in *Using the HTTP Transport* describes how to configure a consumer to use a decoupled HTTP conduit.
- Using the JMS Transport provides detailed information on using the FUSE Services Framework's JMS transport.
- Using the XML Binding provides detailed information on using the FUSE Services Framework's XML binding.

Java-First Service Developers

Overview

One of the strengths of the JAX-WS specification is that it provides a way to develop services without using WSDL. In Java first development an annotated interface acts as the contract. This reading path walks you through documentation concerning the creation of a service endpoint interface, the code needed to deploy the service as a standalone application, and configuring the runtime.

Path

For a quick introduction to developing a service without using WSDL, you can read the following:

1. *Creating the SEI in Developing Applications Using JAX-WS* describes how to develop a SEI for your service.
 2. *Annotating the Code in Developing Applications Using JAX-WS* describes the annotations used by the JAX-WS framework.
 3. *Publishing a Service in Developing Applications Using JAX-WS* describes the code needed to publish a service as a standalone application.
 4. *Getting Started in FUSE Services Framework Deployment Guide* describes how to set up your environment to run a FUSE Services Framework application.
 5. *FUSE Services Framework Configuration in FUSE Services Framework Deployment Guide* provides details about the files used to configure the FUSE Services Framework runtime.
-

Next steps

The following provide additional information about consumer development:

- *Deploying to the Spring Container in FUSE Services Framework Deployment Guide* describes how to deploy a service into the FUSE Services Framework Spring container.
- *Deploying to a Servlet Container in FUSE Services Framework Deployment Guide* describes how to deploy a service into a servlet container.
- *Using XML in a Service Provider with the `Provider` Interface in Developing Applications Using JAX-WS* describes how to use the `Provider` interface to develop services that work with raw XML messages.

- *Working with Contexts* in *Developing Applications Using JAX-WS* describes how to use the JAX-WS context mechanism.
- Using the SOAP Binding provides detailed information on using the FUSE Services Framework's SOAP binding.
- *Sending Binary Data with SOAP MTOM* in *Using the SOAP Binding* provides detailed information on sending binary data using the FUSE Services Framework's SOAP binding.
- *HTTP Service Provider Configuration* in *Using the HTTP Transport* describes how to configure server-side HTTP endpoints.
- Using the JMS Transport provides detailed information on using the FUSE Services Framework's JMS transport.
- Using the XML Binding provides detailed information on using the FUSE Services Framework's XML binding.

WSDL-First Service Developers

Overview

WSDL is an XML grammar used to describe services. In many instances, the first step in developing a service is describing it in WSDL. The Java code that implements the actual service is generated from the WSDL document. This reading path walks you through the process of creating a WSDL document describing a service, generating the required skeleton code, implementing the service, and deploying the service as a standalone application.

Path

For a quick introduction to developing a service using WSDL, you can read the following:

1. *Introducing WSDL Contracts* in *Writing WSDL Contracts* describes the contents of a WSDL document.
2. *Defining Logical Data Units* in *Writing WSDL Contracts* describes how to define complex data types in XML Schema.
3. *Defining Logical Messages Used by a Service* in *Writing WSDL Contracts* describes how to define the messages exchanged by your service.
4. *Defining Your Logical Interfaces* in *Writing WSDL Contracts* describes how to define your service's interface.
5. *Using SOAP 1.2 Messages* in *Using the SOAP Binding* describes how to add a SOAP 1.2 binding to a WSDL document.
6. *Basic Endpoint Addressing* in *Using the HTTP Transport* describes how to add an HTTP port to a WSDL document.
7. *Developing a Service Provider From a WSDL Contract* in *Developing Applications Using JAX-WS* describes how to implement your service.
8. *Publishing a Service* in *Developing Applications Using JAX-WS* describes the code needed to publish a service as a standalone application.
9. *Getting Started* in *FUSE Services Framework Deployment Guide* describes how to set up your environment to run a FUSE Services Framework application.

10. *FUSE Services Framework Configuration* in *FUSE Services Framework Deployment Guide* provides details about the files used to configure the FUSE Services Framework runtime.

Next steps

The following provide additional information about consumer development:

- *Deploying to the Spring Container* in *FUSE Services Framework Deployment Guide* describes how to deploy a service into the FUSE Services Framework Spring container.
- *Deploying to a Servlet Container* in *FUSE Services Framework Deployment Guide* describes how to deploy a service into a servlet container.
- *Working with Contexts* in *Developing Applications Using JAX-WS* describes how to use the JAX-WS context mechanism.
- *Sending Binary Data with SOAP MTOM* in *Using the SOAP Binding* provides detailed information on sending binary data using the FUSE Services Framework's SOAP binding.
- *HTTP Service Provider Configuration* in *Using the HTTP Transport* describes how to configure server-side HTTP endpoints.
- *Using the JMS Transport* provides detailed information on using the FUSE Services Framework's JMS transport.
- *Using the XML Binding* provides detailed information on using the FUSE Services Framework's XML binding.

RESTful Service Developers

Overview

Using FUSE Services Framework you can develop applications that conform to the principles of REST-based design. FUSE Services Framework allows you to develop RESTful services in three ways:

- Using the automatically generated URLs.
- Using FUSE Services Framework specific annotations to generate the URLs.
- Using the JAX-WS `Provider` interface.

Using automatic URL generation

For a quick introduction to developing RESTful services with automatically generated URLs, you can read the following:

1. *Introduction to RESTful Services* in *Developing RESTful Services* provides an overview of RESTful design principles.
2. *Using Automatic REST Mappings* in *Developing RESTful Services* describes how to develop a service that uses the FUSE Services Framework to automatically generate your URIs.
3. *Publishing a RESTful Service* in *Developing RESTful Services* describes how publish your service as a standalone application.
4. *Getting Started* in *FUSE Services Framework Deployment Guide* describes how to set up your environment to run a FUSE Services Framework application.
5. *FUSE Services Framework Configuration* in *FUSE Services Framework Deployment Guide* provides details about the files used to configure the FUSE Services Framework runtime.

Using annotations

For a quick introduction to developing RESTful services with FUSE Services Framework annotations, you can read the following:

1. *Introduction to RESTful Services* in *Developing RESTful Services* provides an overview of RESTful design principles.
2. *Using JAVA REST Annotations* in *Developing RESTful Services* describes how to use the FUSE Services Framework annotations.

3. *Publishing a RESTful Service* in *Developing RESTful Services* describes how publish your service as a standalone application.
4. *Getting Started* in *FUSE Services Framework Deployment Guide* describes how to set up your environment to run a FUSE Services Framework application.
5. *FUSE Services Framework Configuration* in *FUSE Services Framework Deployment Guide* provides details about the files used to configure the FUSE Services Framework runtime.

Using annotations

For a quick introduction to developing RESTful services with FUSE Services Framework annotations, you can read the following:

1. *Introduction to RESTful Services* in *Developing RESTful Services* provides an overview of RESTful design principles.
2. Using XML in a Service Provider with the `Provider` Interface in *Developing Applications Using JAX-WS* describes how to develop a service using the JAX-WS `Provider` interface.
3. *Publishing a Service* in *Developing Applications Using JAX-WS* describes the code needed to publish a service as a standalone application.
4. *Getting Started* in *FUSE Services Framework Deployment Guide* describes how to set up your environment to run a FUSE Services Framework application.
5. *FUSE Services Framework Configuration* in *FUSE Services Framework Deployment Guide* provides details about the files used to configure the FUSE Services Framework runtime.

JavaScript Service Developers

Overview

FUSE Services Framework includes a mechanism for developing services using JavaScript. The JavaScript mechanism mimics the use of the JAX-WS `Provider` interface. This path walks you through the process of developing and deploying a service using the JavaScript interface.

Path

For a quick introduction to developing a service in JavaScript, you can read the following:

1. Defining the Metadata in *Developing Applications with Dynamic Languages* describes how to add provide the metadata needed to publish a service to your application.
 2. Implementing the Service Logic in *Developing Applications with Dynamic Languages* describes how to implement your service.
 3. *Publishing Services Developed in a Dynamic Language* in *Developing Applications with Dynamic Languages* describes how to publish your service.
 4. *Getting Started* in *FUSE Services Framework Deployment Guide* describes how to set up your environment to run a FUSE Services Framework application.
 5. *FUSE Services Framework Configuration* in *FUSE Services Framework Deployment Guide* provides details about the files used to configure the FUSE Services Framework runtime.
-

Next steps

The following provide additional information about consumer development:

- Using the SOAP Binding provides detailed information on using the FUSE Services Framework's SOAP binding.
- *HTTP Service Provider Configuration* in *Using the HTTP Transport* describes how to configure server-side HTTP endpoints.
- Using the JMS Transport provides detailed information on using the FUSE Services Framework's JMS transport.