PROGRESS
SOFTWARE

Artix® ESB

Using the Artix Library

Version 5.6, May 2011

Third Party Acknowledgments — See the Third Party Acknowledgements section on page 13.

Updated: May 26, 2011

Contents

Artix Library Overview

This chapter describes the contents of the Artix Library, how to get additional information, and the documentation conventions used.

In this chapter

This chapter includes the following topics

Artix Documentation Library

Overview

The Artix documentation library is organized into the following sections:

- Getting Started
- Designing Artix Solutions
- Developing Artix Applications
- Deploying and Managing Artix Solutions
- Using Artix Services
- Integrating Artix Solutions
- Reference Material

Getting Started

The books in this section provide you with a background for working with Artix. They describe many of the concepts and technologies used by Artix. They include:

- Release Notes contains release-specific information about Artix.
- Installation Guide describes the prerequisites for installing Artix and the procedures for installing Artix on supported systems.
- Using the Artix Library (this book) introduces the Artix documentation library, explains its conventions, and provides suggested reading paths.
- Getting Started with Artix describes basic Artix and WSDL concepts, and shows a simple example application.
- Artix Technical Use Cases provides a number of step-by-step examples of building common Artix solutions.
- Artix Glossary is a comprehensive reference of Artix terms. It provides quick definitions of the main Artix components and concepts. All terms are defined in the context of the development and deployment of Web services using Artix.

Designing Artix Solutions

The books in this section discuss how to use Artix to solve real-world problems. They describe how to build service-oriented architectures with Artix and how Artix uses WSDL to define services:

- Building SOAs with Artix provides an overview of service-oriented architectures and describes how they can be implemented using Artix.
- Writing Artix Contracts describes the components of an Artix WSDL contract. Special attention is paid to Artix-specific WSDL extensions.
- Artix Bindings and Transports, Java Runtime describes the Artix WSDL extensions used to define payload formats and transports for Artix services written in JAX-WS or JavaScript.

Developing Artix Applications

The books in this section describe how to use the Artix APIs to build new services:

- Developing Artix Applications with JAX-WS explains how to implement services using the Artix Java API for XML-Based Web Services.
- Developing Artix Applications with JavaScript explains how to implement services using the Artix JavaScript API.
- WSDLGen Guide explains how to generate JAX-WS code using the Artix scripting tools.

Deploying and Managing Artix Solutions

Configuring and Deploying Artix Solutions, Java Runtime explains how to set up your Artix environment and how to configure and deploy Artix Java services (for example, written in JAX-WS or JavaScript).

Using Artix Services

The books in this section describe how to use the services provided with Artix:

- Artix Java Router, Getting Started introduces the Artix Java router and describes how to create, build and run a simple example. This guide applies to services written in JAX-WS.
- Artix Java Router, Programmer's Guide provides details of how to program routing processors and describes how to implement custom components.
- Artix Java Router, Deployment Guide explains how to deploy a standalone Artix Java router, and how to deploy into the Spring container. This guide applies to services written in JAX-WS.
- Artix Java Router, Defining Routes provides an introduction to defining routes using the Java-fluent DSL and the Spring XML syntax.
- Implementing Enterprise Integration Patterns describes how you can use Artix Java Router to implement Enterprise Integration Patterns (from the book of the same name by Gregor Hohpe and Bobby Woolf).
- Artix Locator Guide explains how clients can find services using the Artix locator. This guide applies to services written in JAX-WS or JavaScript.
- Artix Security Guide, Java Runtime explains how to configure and develop secure Artix applications. This guide applies to services written in JAX-WS or JavaScript.

Integrating Artix Solutions

The books in this section describe how to integrate Artix solutions with other middleware technologies:

- Artix for CORBA provides information on using Artix in a CORBA environment. This guide applies to services written in JAX-WS or JavaScript.
- Artix for J2EE (JAX-WS) provides information on using Artix to integrate with J2EE applications. This guide applies to services written in JAX-WS.

Reference Material

These books provide detailed reference information about specific Artix APIs, WSDL extensions, configuration variables, and command-line tools. The reference documentation includes:

- Artix Command Line Reference
- Artix Configuration Reference, Java Runtime
- Artix WSDL Extension Reference
- Artix JAX-WS API Reference
- Artix Security Framework Java API Reference
- WSDLGen Java API Reference
- WSDLGen JavaScript API Reference
- Artix Java Router, Java API Reference
- Artix Java Router, Configuration Reference

Getting the Latest Version

The latest updates to the Artix documentation library can be found online.

Compare the version dates on the web page for your product version with the date printed on the copyright page of the PDF edition of the book you are reading.

Searching the Artix Library

You can search the online documentation by using the **Search** box at the top right of the documentation home page.

To search a particular library version, browse to the required index page, and use the **Search** box at the top right of that page.

You can also search within a particular book. To search within a HTML version of a book, use the **Search** box at the top left of the page. To search within a PDF version of a book, in Adobe Acrobat, select **Edit**|**Find**, and enter your search text.

Additional Resources

Additional information can be found on the Progress Web site with Online Support (http://www.progress.com/artix/support/index.ssp). This includes:

- The Knowledge Base that contains helpful articles written by company experts about Artix and other products.
- The Update Center that contains the latest releases and patches for Artix products.

Comments, corrections, and suggestions on Progress documentation can be sent to `docs-support@progress.com`.

Documentation Conventions

Overview

This section shows the typographical and keying conventions used by the Artix documentation library.

Typographical conventions

The Artix library uses the following typographical conventions:

`Fixed width`	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 `IT_Bus::AnyType` class. Constant width paragraphs represent code examples or information a system displays on the screen. For example: `#include <stdio.h>`
`Fixed width italic`	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: `% cd /users/`*`YourUserName`*
Italic	Italic words in normal text represent *emphasis* and introduce *new terms*.
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

The Artix library 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). In graphical user interface descriptions, a vertical bar separates menu commands (for example, select **File**|**Open**).

Third Party Acknowledgements

Suggested Reading Paths

This chapter describes suggested reading paths for different types of Artix users.

In this chapter

This chapter includes the following topics

SOA Architects

Overview

This section describes a suggested reading path for SOA architects, and includes suggestions for background reading.

SOA architect path

SOA architects should start with the following:

1. Building Service Oriented Architectures with Artix presents an overview of SOA and ESBs, of how Artix fits into SOA, and of how Artix works.
2. Installation Guide. You must read the following sections about supported environments:
   i. *Supported Systems and Compilers*
   ii. *Java and Compiler requirements*
3. Writing Artix Contracts includes the following information about basic WSDL concepts and how to write a service interface:
   i. *Introduction*. Overview of WSDL, the structure of a contract, and the steps involved in writing a service contract.
   ii. *Designing Logical Data Units*. How to create data types using XML Schema.
   iii. *Defining Logical Messages Used by a Service*. How to build the data types into the messages that a service will use to implement its operations.
   iv. *Defining Your Logical Interfaces*: How to create a service interface using the logical messages.
4. Artix Bindings and Transports, Java Runtime describes the Artix WSDL extensions used to define payload formats and transports for Artix services written in JAX-WS or JavaScript.
   i. Read the relevant binding chapter that applies to your system (for example, SOAP, CORBA, or XML).
   ii. Read the relevant transport chapter that applies to your system (for example, HTTP, MQ, or JMS).

Background reading

In addition, the following publications provide useful background information on Web services, XML, and WSDL:

- *Understanding Web Services: XML, WSDL, SOAP, and UDDI*, by Eric Newcomer
- *Understanding SOA with Web Services*, by Eric Newcomer and Greg Lomow
- W3Schools online tutorials
  (see http://www.w3schools.com)
  - XML tutorial (http://www.w3schools.com/xml/default.asp)
  - XSD tutorial (http://www.w3schools.com/schema/default.asp)
  - XSLT tutorial (http://www.w3schools.com/xsl/default.asp)
- The W3C XML schema page
  (see www.w3.org/XML/Schema)
- The W3C WSDL specification
  (see www.w3.org/TR/wsdl)

Administrators

Overview

This section describes a suggested reading path for Artix administrators in Java runtime environments.

All Artix administrators

Administrators can approach the Artix library as follows:

1. Installation Guide describes all the prerequisites and procedures for installing Artix on supported systems. You must read the following:
   - *i. Supported Systems and Compilers*
   - *ii. Java and Compiler requirements*
   - *iii. Installing Artix*

Java runtime administrators

Administrators working with applications written in JAX-WS or JavaScript should read the following:

1. Configuring and Deploying Artix Solutions, Java Runtime explains how to configure and deploy Artix Java services (for example, JAX-WS or JavaScript). You should start with the following chapters:
   - i. *Getting Started* explains how to set your Artix Java runtime system environment.
   - ii. *Artix Java Configuration* introduces the main concepts and components in the Artix Java runtime configuration.It also explains how to use Artix Java configuration files to manage your applications.
2. Artix Configuration Reference, Java Runtime provides a comprehensive reference for Artix configuration settings in Spring XML configuration files.

Security and management

Security administrators and administrators using management consoles should read the Artix Security Guide, Java Runtime provides detailed information on Artix security configuration and management. This guide applies to services written in JAX-WS or JavaScript.

Background reading

For background information on Web services, XML, and WSDL, see "Background reading" on page 17.

All Service Developers

Overview

This section describes an initial reading path for all types of service development use case. You should follow this path before writing any code.

All developers

All service developers should read the following path:

1. Building Service Oriented Architectures with Artix presents an overview of SOA and ESBs, of how Artix fits into SOA, and of how Artix works.
2. Artix Installation Guide. You must read the following sections about supported environments:
   - *i. Supported Systems and Compilers*
   - *ii. Java and Compiler requirements*
3. Writing Artix Contracts includes information about basic WSDL concepts and how to write a service interface.
   1. *Introduction*. Overview of WSDL, the structure of a contract, and the steps involved in writing a service contract.
   2. *Designing Logical Data Units*. How to create data types using XML Schema
   3. *Defining Logical Messages Used by a Service*. How to build the data types into the messages that a service will use to implement its operations.
   4. *Defining Your Logical Interfaces*: How to create a service interface using the logical messages.
4. Configuring and Deploying Artix Solutions, Java Runtime explains how to configure and deploy Artix Java services (for example, written in JAX-WS or JavaScript). You must read the following chapters:
   1. *Getting Started* explains how to set your Artix Java runtime system environment.
   2. *Artix Java Configuration* introduces the main concepts and components in the Artix Java runtime configuration.It also explains how to use Artix Java configuration files to manage your applications.

Integration Use Case

Overview

This section describes the reading path for developing a service as a front-end for existing functionality.

Service integration

Service integrators should read the following books:

1. Artix Technical Use Cases. Read the following chapter:
   - i. *Web Service Enabling Backend Services*. Walks through the steps for the integration use case.
2. Artix Java Router, Getting Started. Read the following information about the Java router service.
   - i. *Introduction*. Overview of the Java router and how it is used.
   - ii. *Tutorial*. Describes the Java router in more detail, explains the code for a sample application, and how to build and run the application.
3. Artix Bindings and Transports, Java Runtime includes information about creating bindings and endpoints for Artix services written in JAX-WS or JavaScript.
   - i. Read the relevant binding chapter that applies to your system (for example, SOAP, CORBA, or XML).
   - ii. Read the relevant transport chapter that applies to your system (for example, HTTP, MQ, or JMS).

4. Artix Java Router, Defining Routes. Read the following information about how to define routes between endpoints:
   i. *Defining Routes in Java DSL*. Explains how to define routing rules in Java in a domain specific language (DSL). This is the most flexible way to define rules.
   ii. *Defining Routes in XML*. Explains how to define routing rules in XML. This is not as flexible as Java DSL, but is easy to reconfigure at runtime.
   iii. *Basic Principles of Route Building*. Explains the principles of building a route using the provided building blocks
5. Artix Java Router, Deployment Guide. Read the following information about how to deploy an Artix Java Router:
   i. *Deploying a Standalone Router*. Explains how to deploy the Java router in standalone mode. This means you can deploy the router independent of any container, but some extra programming steps are required.
   ii. *Components*. Provides a reference of components available with the Artix Java router. These are plug-ins that can be used to enable integration with different kinds of protocol, containers, databases, and so on.

Advanced integration topics

In addition, you may wish to read the following:

- Artix Java Router, Programmer's Guide provides details of how to program routing processors and describes how to implement custom components.
- Implementing Enterprise Integration Patterns describes how you can use Artix Java Router to implement Enterprise Integration Patterns (from the book of the same name by Gregor Hohpe and Bobby Woolf).
- Artix for CORBA contains detailed information about using Artix to integrate with CORBA applications.
- Artix for J2EE (JAX-WS) contains detailed information about using Artix with J2EE applications. This guide applies to services written in JAX-WS.

New Development Use Cases

Overview

This section describes reading paths for the following new development use cases:

- "Service consumer"
- "JAX-WS development"
- "JavaScript development"

Service consumer

Read the following if you are developing a new service consumer:

1. Artix Technical Use Cases. Read the following chapter:
   i. *Building a Client for a Web Service*. Provides a walk through of the service consumer use case.
2. Artix Bindings and Transports, Java Runtime includes information about creating bindings and endpoints for Artix services written in JAX-WS or JavaScript.
   i. Read the relevant binding chapter that applies to your system (for example, SOAP, CORBA, or XML).
   ii. Read the relevant transport chapter that applies to your system (for example, HTTP, MQ, or JMS).

JAX-WS development

For detailed information on developing a new JAX-WS service provider or consumer, read the following:

1. Developing Artix Applications with JAX-WS:
   i. *Starting from Java Code.* Describes how to write a JAX-WS application without WSDL.
   ii. *Service Enabling a Java Class*. Describes how to annotate a Java class for use as a service provider. This includes creating the SEI, annotating the code, and generating WSDL.
   iii. *Developing a Consumer without a WSDL Contract*. This includes creating a service object, adding a port to a service, getting a proxy for an endpoint, and implementing the consumer's business logic.

iv. *Starting from a WSDL Contract*. Describes how to write a JAX-WS service starting from WSDL. This includes developing a service and a consumer starting from WSDL.

v. *Publishing a Service*. Describes how to publish a service provider as a standalone Java application.

vi. *Developing RESTful Services*. Describes what it means for a service to be RESTful, and how to build RESTful services using Java classes and annotations.

vii. *Developing Asynchronous Applications*. Describes how to use the JAX-WS asynchronous APIs to develop asynchronous service consumers.

viii. *Generating the Stub Code*. Describes how to create a customization file to alter the code generated to use the asynchronous APIs.

ix. *Implementing an Asynchronous Client with the Polling Approach*. Describes how to use the APIs that provide a mechanism for using the generated response object to pole for an asynchronous response.

x. *Implementing an Asynchronous Client with the Callback Approach*. Describes how to use a callback object to process asynchronous responses.

xi. *Using Raw XML Messages*. Describes how to obtain direct access to raw XML message data on the wire. The JAX-WS client-side interface is `Dispatch`, and the server-side interface is `Provider`.

xii. *Working with Contexts*. Describes how to access the metadata passed by contexts along the messaging chain. This metadata can be accessed by implementation code and by JAX-WS handlers that operate on the message below the implementation level.

JavaScript development

For information on developing a new JavaScript service provider or consumer, read the following:

1. Developing Artix Applications with JavaScript:
   i. *Implementing a service provider using JavaScript*. This includes defining the metadata, which describes how to provide the information provided by the JAX-WS annotations. It also includes implementing the application logic, which is a brief overview of implementing the service using the invoke property
   ii. *Implementing a service provider using E4X*. Describes the minor differences between the JavaScript interface and the ECMAScript for XML (E4X) interface.
   iii. *Deploying a JavaScript service*. Describes how to deploy scripted services using the `ServerApp` Java application provided.

Background reading

For background information on Web services, XML, and WSDL, see "Background reading" on page 17.