Artix 5.6.3

Using the Artix Library: C++

Micro Focus
The Lawn
22-30 Old Bath Road
Newbury, Berkshire RG14 1QN
UK

http://www.microfocus.com

2015-02-10

Contents

Artix Library Overview

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

Artix Documentation Library

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 the Latest Version
- Additional Resources

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.

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, C++ Runtime*** describes the WSDL extensions used to define payload formats and transports for Artix services written in C++.

Developing Artix Applications

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

- ***Developing Artix Applications in C++*** explains how to implement services using the Artix C++ API.
- ***Developing Advanced Artix Plug-ins in C++*** explains how to implement advanced Artix plug-ins (for example, interceptors) using the Artix C++ API.
- ***WSDLGen Guide*** explains how to generate C++ code using the Artix scripting tools.

Deploying and Managing Artix Solutions

The books in this section describe how to configure, deploy, and manage Artix applications and services in your environment:

- ***Configuring and Deploying Artix Solutions, C++ Runtime*** explains how to set up your Artix environment and how to configure and deploy Artix services written in C++.
- ***Artix Management Guide, C++ Runtime*** explains how to monitor and manage Artix services using Java Management Extensions. It also describes how to integrate Artix solutions with a range of third-party enterprise and SOA management systems.

Using Artix Services

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

- ***Artix Router Guide, C++ Runtime*** explains how to integrate and manage services using the Artix C++ router.
- ***Artix Locator Guide*** explains how clients can find services using the Artix locator.

- ***Artix Session Manager Guide*** explains how to manage client sessions using the Artix session manager.
- ***Artix C++ Transactions Guide*** explains how to enable Artix applications written in C++ to participate in transacted operations.
- ***Artix Security Guide, C++ Runtime*** explains how to configure and develop secure Artix applications.

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.

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, C++ Runtime***
- ***Artix WSDL Extension 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.

Additional Resources

Enter http://www.microfocus.com in your browser to go to the Micro Focus home page.

The Micro Focus Community site contains helpful articles written by company experts about Artix and other products.

If you need help with this or any other Artix product, go to Micro Focus SupportLine.

Documentation Conventions

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**).

Suggested Reading Paths

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

SOA Architects

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 Platforms, C++ Compilers, JREs, and JDKs*
    ii. *C++ 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, C++ Runtime*** describes the Artix WSDL extensions used to define payload formats and transports for Artix services written in C++:
    i. Read the relevant binding chapter that applies to your system (for example, SOAP, Fixed, or XML).
    ii. Read the relevant transport chapter that applies to your system (for example, HTTP, Tuxedo, 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

This section describes a suggested reading path for Artix administrators in the C++ runtime environment.

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, Compiler, and Artix Designer Requirements
   iii. Installing Artix

C++ runtime administrators

Administrators working with applications written in C++ should read the following:

1. ***Configuring and Deploying Artix Solutions, C++ Runtime*** explains how to configure and deploy Artix services written in C++. It explains Artix configuration, how to find contracts that control your Artix services, and how to deploy Artix applications. You should start with the following chapters:
   i. *Getting Started* explains how to set up an Artix environment using the `artix_env` script, and Artix system environment variables.
   ii. *Artix Configuration* explains concepts such as Artix `.cfg` configuration files, scopes, namespaces, and variables.
   iii. *Artix Logging* explains how to configure Artix logging for services and Artix subsystems.
   iv. *Deploying Services in an Artix Container* explains how to deploy and manage C++ services using an Artix container.

v. *Deploying High Availability* explains how to configure and deploy high availability in Artix, which is based on Berkeley DB.

vi. *Deploying Reliable Messaging* explains how to configure and deploy WS-RM and WS-Addressing for services in an Artix runtime environment.

vii. *Publishing WSDL Contracts* explains how to publish WSDL files that correspond to specific Web services, and enable clients to access the WSDL file and invoke on the service.

viii. *Accessing Contracts and References* shows how to specify the location of WSDL contracts and references in a configuration file and on the command line.

2. ***Artix Configuration Reference, C++ Runtime*** provides a comprehensive reference for the all configuration variables in an Artix configuration domain.

Security and management

Security administrators and administrators using management consoles should read the following:

1. ***Artix Security Guide, C++ Runtime*** provides detailed information on Artix security configuration and management.
2. ***Artix Management Guide, C++ Runtime*** explains how to monitor and manage Artix services using Java Management Extensions. It also provides information on how to integrate Artix with various third-party enterprise and SOA management systems, such as Aurea Actional® Application Performance Monitoring, AmberPoint, and BMC Patrol.

Background reading

For background information on Web services, XML, and WSDL, see "Background reading" on page 3.

All Service Developers

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, Compiler, and Artix Designer Requirements*

3. ***Writing Artix Contracts*** includes 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. ***Configuring and Deploying Artix Solutions, C++ Runtime*** explains how to configure and deploy Artix services written in C++. You must read the following chapters:
   i. *Getting Started* explains how to set up an Artix environment using the `artix_env` script, and system environment variables.
   ii. *Artix Configuration* explains concepts such as Artix configuration files, scopes, namespaces, and variables.
   iii. *Artix Logging* explains how to configure Artix logging for services and Artix subsystems.

Integration Use Case

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 Router Guide, C++ Runtime***. Read the following information about the C++ router service and how to make routes.
   i. *Introduction*. Overview of the C++ router and how it is used.
   ii. *Compatibility of Ports and Operations*. Explains the requirements for routing between interfaces.
   iii. *Creating Routes Using Artix Tools*. Introduces the GUI and command line tools that can be used to create routes.
2. ***Artix Bindings and Transports, C++ Runtime*** includes information about creating bindings and endpoints for Artix services written in C++:
   i. Read the relevant binding chapter that applies to your system (for example, SOAP, Fixed, or XML).
   ii. Read the relevant transport chapter that applies to your system (for example, HTTP, Tuxedo, or JMS).

3. ***Artix Router Guide, C++ Runtime***. Read the following information about how to define routes between endpoints:
   i. *Creating a Basic Route*. This is required reading. It describes the minimum needed to create a route in Artix.
   ii. *Adding Operation-Based Rules to a Route*: This is optional. It expands on basic route design.
   iii. *Adding Attribute-Based Rules to a Route*. This is optional. It expands on basic route design.
   iv. *Adding Content-Based Rules to a Route*. This is optional. It describes how to create content based routes.
   v. *Linking Routes*. This is optional. It expands on previous chapters.
   vi. *Using Advanced Routing Features*. This is optional. It describes how to create routes for various advanced use cases such as load balancing and service fail-over.
   vii. *Deploying an Artix Router*. This is required reading. It describes how to deploy the router in an Artix runtime environment.

Advanced integration topics

In addition, you may wish to read the following:

- ***Artix for CORBA*** contains detailed information about using Artix to integrate with CORBA applications.

New Development Use Cases

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

- "Service consumer"
- "C++ development"

Service consumer

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

- ***Artix Bindings and Transports, C++ Runtime*** includes information about creating bindings and endpoints for Artix services written in C++:
  - Read the relevant binding chapter that applies to your system (for example, SOAP, Fixed, or XML).
  - Read the relevant transport chapter that applies to your system (for example, HTTP, Tuxedo, or JMS).

C++ development

For detailed information on developing a new C++ service provider or consumer, read the following:

1. ***Developing Artix Applications in C++***:
   i. *Getting Started with Artix Programming*. An overview of how a developer works in the Artix C++ development environment.
   ii. *Artix Programming Considerations*: *Operations and Parameters* section. An overview of how WSDL is mapped into C++.
   iii. *Server Programming*. The basics of developing an Artix C++ service.
   iv. *Client Programming*. The basics of developing an Artix C++ consumer.
   v. *Artix Programming Considerations*: *Compiling and Linking an Artix Application* section. What is needed to build Artix C++ applications.
   vi. *Artix Programming Considerations*: *Building Artix Stub Libraries on Windows* section. How to build Artix stub code into a Windows DLL.
   vii. *Artix Data Types*. Overview of how WSDL types are mapped into C++.
   viii. *Artix Programming Considerations*: *Exceptions* section. Overview of how to create and handle exceptions in Artix C++.
   ix. *Artix Programming Considerations: Locating Services with UDDI* section. How to use the UDDI interface as an alternate method of finding services.
   x. *Endpoint References and Callbacks*. Overview of EPRs and how to use them in implementing callbacks.
   xi. *Artix Contexts*. How to get information from the binding and transport layers of the runtime.
   xii. *Working with Transport Attributes*. How to extract a default set of transport information from the runtime.
   xiii. *Persistent Maps*. How to use persistent data for high availability.
   xiv. *Reflection*. How to determine the structure of an Artix data type without advance knowledge.
   xv. *Default Servants*. How to write a scalable factory pattern.
   xvi. *Artix Programming Considerations: Multi-Threading* section. Describes issues for multi-threaded Artix clients and servers.

Advanced C++ development

For detailed information on developing new advanced C++ plug-ins, read the following:

2. ***Developing Advanced Artix Plug-Ins with C++***. How to write custom interceptors and transport plug-ins.

Background reading

For background information on Web services, XML, and WSDL, see “Background reading” on page 3.