Red Hat JBoss Fuse 6.3

Glossary

A reference to the terms used when talking about Red Hat JBoss Fuse

Red Hat JBoss Fuse 6.3 Glossary

A reference to the terms used when talking about Red Hat JBoss Fuse

JBoss A-MQ Docs Team
Content Services
fuse-docs-support@redhat.com

Legal Notice

Abstract

This guide defines a number of terms that are specific to the integration space.

Table of Contents

CHAPTER 1. GENERAL COMPUTER TERMS

Abstract

This chapter defines a number of general computing terms and abbreviations.

DEFINITIONS

dependency injection

A form of inversion of control, where an object's external dependencies are given to it, either programmatically or through a framework that is driven by configuration information. The result is to decouple dependent objects and allow the dependencies to be resolved at run time.

i18n

An abbreviation for internationalization, used in the context of preparing products, especially software and documentation, for use in more than one national locale and language.

Java Management eXtensions, JMX

A Java technology that supplies tools for managing and monitoring applications, system objects, devices, and service-oriented networks.

Java Database Connectivity, JDBC

An API specified in Java technology that provides Java applications with access to databases and other data sources.

Java Naming and Directory Interface, JNDI

A set of APIs specified in Java technology that assists Java applications with interfacing to multiple naming and directory services.

Java Architecture for XML Binding, JAXB

An API that provides a way to bind an XML Schema to a representation in Java code.

Java Authentication and Authorization Service, JAAS

A Java security framework for user-centric security to augment the Java code-based security.

l10n

An abbreviation for localization, used in the context of preparing products, especially software and documentation, for use in more than one national locale and language. Localization is the process of translating the elements of a product for a particular locale and language.

marshalling

The process of taking in-memory objects and converting them to a binary or textual format for transmission over a transport.

See also: unmarshalling

OASIS

An international consortium that drives the development, convergence, and adoption of Web services standards. See http://www.oasis-open.org.

Spring framework

A comprehensive programming and configuration model for modern Java-based enterprise applications the uses dependency injection.

See also: dependency injection

Uniform Resource Identifier,URI

A string of characters used to identify or name a resource on the Internet.

unmarshalling

The process of taking a binary or textual format payload and converting that into objects.

See also: marshalling

CHAPTER 2. GENERAL INTEGRATION TERMS

DEFINITIONS

application server

A software platform that provides the services and infrastructure required to develop and deploy middle-tier applications. Middle-tier applications implement the business logic necessary to provide web clients with access to enterprise information systems. In a multi-tier architecture, an application server sits beside a web server or between a web server and enterprise information systems. Application servers provide the middleware for enterprise systems. JBoss, WebLogic and WebSphere are J2EE application servers.

client

An application or process that requests services from other applications known as servers. The server processes may be running on the same or a different machine. In the context of a SOA network, a client process is called a consumer or service consumer.

Enterprise Application Integration, EAI

Enterprise Application Integration (EAI), the use of software and architectural principles to integrate disparate enterprise applications.

Enterprise Integration Patterns, EIP

A collection of patterns describing common EAI problems. For more information see http://www.enterpriseintegrationpatterns.com/.

Enterprise Service Bus, ESB

The infrastructure that allows service providers and service consumers to interact in a distributed environment. The bus handles the delivery of messages between different middleware systems, and provides management, monitoring, and mediation services such as routing, service discovery, or transaction processing.

Java Platform, Enterprise Edition 5, JEE

A specification and toolkit from Oracle for the development and deployment of enterprise applications. JEE is the Java 5 version of J2EE.

middleware

A software communications layer that manages the interaction of disparate applications across heterogeneous hardware and network environments.

message exchange pattern, MEP

The pattern of messages used by an application. There are two major message exchange patterns:

- request-response—one client sends a message and expects a message to be returned

- one-way—a client sends a message without expecting a response

The WSDL specification defines a number of more detailed MEPs that are all variations of the two basic patterns.

See also: request-reply pattern

transport mediation

The capability to move a message from one transport to another. This includes transforming message data between the formats required by each protocol and managing the metadata differences between the transports. It also means managing the differences in how the protocols operate. For example, when mediating between HTTP and JMS the bridge must manage the differences between the HTTP transports synchronous, request/reply style and the JMS transports asynchronous style.

CHAPTER 3. OSGI TERMS

DEFINITIONS

OSGi

OSGi is set of open specifications aimed at making it easier to build and deploy complex software applications. The key piece of OSGi technology is the OSGi Framework. It defines standardized mechanism for packaging and managing application bundles. It can dynamically resolve dependencies between bundles and can handle having multiple versions of a bundle deployed simultaneously.

The OSGi specifications are maintained by the OSGi Alliance. See http://www.osgi.org.

Apache Karaf

An open source project that provides the OSGi runtime container used by Red Hat JBoss A-MQ.

See http://karaf.apache.org

bundle

The primary deployment format used in Red Hat JBoss A-MQ. They are either ZIP or JAR files that contain resources and classes for providing a set of functionality to other bundles or to the end user. Bundles differ from standard JAR files in that they must contain metadata describing the bundle and its dependencies.

See also: Fuse Application Bundle

Blueprint

A dependency injection framework designed for use in an OSGi container. It is governed by the Blueprint Container Specification in the OSGi Service Platform Release 4 Version 4.2 Enterprise Specification.

See also: dependency injection

child container

A container that is created by a container on the same host. Child containers are run on the same host as their parent container, but each child runs in a separate JVM.

When created using the console's **admin:create-container**, a child container inherits the features, feature repositories, and configuration from its parent. When a child container is created using the **fabric:container-create** command, the **fabric:contaier-create-child**, command, or the management console, it does not inherit any configuration from its parent.

Regardless of how they are created, child containers can be started and shutdown from their parent container's console without using SSH.

feature

A unit of OSGi deployment that enables you to deploy multiple bundles in a single step.

feature repository

An XML file that defines one or more features.

feature URL

A URL that points to a feature repository file.

persistent identifier, PID

A registration property used by the OSGi Configuration Admin Service to identify a group of configuration attributes.

CHAPTER 4. FUSE FABRIC TERMS

DEFINITIONS

agent

See Fabric Agent

clustered service

A service that can be discovered via Fuse Fabric and has master/slave support.

ensemble

See Fabric Ensemble

Fuse Application Bundle, FAB

A bundle that uses a POM file to specify its dependencies.

Fuse Fabric

An open source project that provides a distributed runtime registry that provides configuration, deployment, and discovery services to a collection of distributed containers.

See also: fabric

Fabric Agent

The service running inside a Fabric Container that is responsible for configuring and provisioning the container according to the profiles assigned to the container . It is also responsible for updating the registry with runtime information about the services in container.

fabric

A group of containers that are connected to a common Fabric Ensemble. The ensemble makes it possible for all of the containers to share runtime information about the services deployed in each container and allows them to share common configuration profiles.

Fabric Container

An Apache Karaf-based container that is managed by a Fabric Agent.

See also: Fabric Agent

Fabric Registry

A ZooKeeper-based distributed registry that stores runtime and configuration information about the services in a fabric.

Fabric Server

A server that, as part of a Fabric Ensemble, provides a number of services that bind a fabric. These services include the Fabric Registry, dynamic load balancing, and location transparency.

See also: Fabric Registry

Fabric Ensemble

A group of one or more Fabric Servers that provide a number of services that bind a fabric. These services include the Fabric Registry, dynamic load balancing, and location transparency.

See also: Fabric Registry, Fabric Server

managed container

See Fabric Container

non-managed container

An Apache Karaf-based container that is registered with a fabric, but is *not* managed by a Fabric Agent.

profile

A set of data that defines runtime artifacts and configuration settings for provisioning a Fabric Container.

registry

See Fabric Registry

standalone container

A container that is not part of a fabric and does not have a Fabric Agent installed.

standalone broker

See standalone container

version

A collection of configuration profiles in a Fabric Registry.

See also: profile

CHAPTER 5. COMMON MESSAGING TERMS

DEFINITIONS

Java Message Service, JMS

A Java API implementing a messaging standard that allows application components based on J2EE to create, send, receive, and read messages. It enables distributed communication that is loosely coupled, reliable, and asynchronous.

client

An application that uses the message broker to communicate with other applications. These applications use one of the broker's client API to connect to and interact with the broker.

consumer

An application that consumes messages from a messaging destination.

connection factory

An object that a client uses to create a connection to a broker. A factory supports attributes that configure the quality of service for the connections it creates.

destination

A logical holding area for messages in a message broker. Clients publish messages to and consume messages from destinations.

See also: queue, topic

durable subscriber

A message consumer that receives all messages published on a topic, including those published while the subscriber is inactive.

message

An atomic unit of data that is passed between two or more clients. A message consists of three components:

- headers—contain a predefined set of metadata that is used to communicate information about a message between the different parties that handle the message

- properties—contain application defined metadata about a message to the different parties that handle the message

- body—contains the messages payload

message selector

A string containing a boolean SQL statement using SQL 92 syntax that is used to select messages based on JMS message header properties.

message group

A collection of JMS messages that are assigned the same JMSXGroupID.

When used in conjunction with the JMSXGroupSeq message groups can be used to ensure that messages are processed in the proper sequence.

master/slave

A topology in which a single instance, the master, is active and one or more instances, the slaves, are ready to resume when the active instance stops.

producer

An application that creates messages and posts them to a messaging destination.

point-to-point messaging

A messaging style where messages are sent between two known endpoints. This messaging style is typically implemented using queues.

publish and subscribe messaging,pub/sub

A messaging style where message producers send(publish) messages to a destination and interested consumers can register(subscribe) to receive messages from the destination. This style of messaging is implemented using topics.

queue

A destination that uses first in/first out semantics.

See also: destination

request-reply pattern

A messaging pattern in which a message producer receives a message and returns a correlated message.

Session

A JMS object that provides a single-threaded context for producing and consuming messages. JMS clients use the **Session** object to create producers, consumers, messages, and other artifacts used to work with messages.

Streaming Text Orientated Messaging Protocol,STOMP

A language agnostic, simple text-based protocol that allows clients to talk with any message broker supporting the protocol.

transport

A standards-based network protocol, such as HTTP or STOMP, that defines how objects communicate over a network.

topic

A destination that uses publish and subscribe semantics.

See also: destination

CHAPTER 6. ROUTING TERMS

DEFINITIONS

Apache Camel

An open source project that provides the EIP-based routing technology used by Red Hat JBoss Fuse.

See http://camel.apache.org

consumer

The source of messages in a route.

component

A factory that creates a routing endpoint that connects to a particular message source or message sink.

CamelContext

A single routing rule base that defines the context for configuring routes, and specifies which policies to use during message exchanges between endpoints.

dead letter channel

An EIP processor that handles messages that cannot be delivered to the intended recipient.

endpoint

The sources and sinks of messages in a route.

from

The DSL command that creates a message source for a route.

message

routing—The data passed processed by a route. A message consists of three components:

- headers—contain metadata that is used to communicate information about a message between the different processors that handle the message

- attachments—contain binary data that is associated with the message

- body—contains the messages payload

out message

A temporary holder for messages as they are processed.

processor

A Java object that performs work on a message as it passes along a route. Processors typical performs tasks like modifying the contents of a message or determining its path through a route.

route

A chain of processors through which a message travels.

wiretap

An EIP in which an messages are directed to an additional message channel in addition to the primary channel.

CHAPTER 7. SOA TERMS

DEFINITIONS

abstract contract

See logical contract

abstract head element

An XML Schema element that cannot appear in an instance document. When a substitution group's head element is declared as abstract with abstract="true", a member of that element's substitution group must be used instead.

anyType

The root type for all XML Schema type definitions hierarchies. All primitive types are derivatives of this type, as are all user-defined complex types.

Apache CXF

An open source project that provides the Web services framework used by Red Hat JBoss Fuse.

See http://cxf.apache.org

binding

A description of the message format and protocol details for a set of operations and messages. Bindings are created based on the information specified in a WSDL **binding** element.

consumer

The end user of a service, also called a client for that service. The more exact term in the context of a service-oriented network is service consumer.

choice complex type

An XML Schema construct defined using the **choice** element to constrain the possible elements in a complex data type. When using a choice complex type, only one of the elements defined in the complex type can be present at a time.

concrete contract

See physical contract

contract

A description of the messages and formats accepted and generated by a service. A service's contract is specified in a WSDL document that defines the interface and all connection-related information for that interface. A WSDL contract contains two sets of components: logical (or abstract) and physical (or concrete).

The logical components of the contract are those that describe the data types, message formats, operations, and interfaces for the services defined in the contract. Logical components are specified with the WSDL elements **types**, **message**, **portType**, and **operation**.

endpoint

The point of contact that a service provides for its consumers.

endpoint reference, EPR

A self-contained object that describes the network contact and policy information for an endpoint, as defined in the WS-Addressing standard.

facet

A rule in an XML Schema definition used in the derivation of user-defined simple types. Common facets include length, pattern, totalDigits, and fractionDigits.

fault message

A message containing error or exception information passed between a service and its consumers. Fault messages are defined using the **fault** element in a WSDL document.

See also: request-response operation , solicit-response operation

in message

The message being processed by the processors in the route.

input message

A message passed from a service consumer to a service. When mapped into Java, the parts of an input message are mapped into a method's parameter list. Input messages are defined using the **input** element in a WSDL document.

See also, request-response operation , solicit-response operation, one-way operation

interface

The external touch point between applications to collaborate or share functional behavior. Interfaces are completely described by the combination of logical and physical portions of a WSDL document.

Once defined in a contract, an interface is the abstract boundary that a service exposes. A service's interface is the set of message types and message exchange patterns through which service consumers can interact with that service. In a WSDL 1.0 document, interfaces are defined using the **portType** element.

intermediary

A service whose main role is to process all received messages in a value-added way, such as converting them from one data format to another, or routing them to another service. An intermediary has characteristics of both a service provider and a service consumer. Most intermediaries have an intermediary contract, which is similar in form to a service contract, except that it includes rules for processing messages.

Java API for XML Web Services, JAX-WS

A document centric API for Web services. It was designed to take the place of JAX-RPC in Web services and Web applications.

Java API for RESTful Services, JAX-RS

A standardized set of APIs and annotations designed to simplify the creation of Web applications using REST architectural principles.

list type

A data type defined in an XML Schema definition as a space-separated list of primitive type elements, defined using the **xsd:list** element.

logical contract

The abstract portion of a WSDL document that defines the data types, message types, and the interfaces for the services defined in the contract. The logical contract answers questions such as:

- What kinds of data will this service work with?

- What kinds of data are grouped together for processing?

- What operations are related and what are their interfaces?

WSDL elements used in the logical contract include: **portType** element, **operation** element, **message** element, and **types** element. Compare with physical contract.

message

services—Any data passed between a service provider and a service consumer, or between two endpoints. Messages are defined in using the WSDL **message** element.

See also: fault message, input message, output message

nilable

In an XML Schema definition, an attribute of an element that specifies that the element is optional within a complex type.

notification operation

One type of WSDL-defined abstract operation, in which the service endpoint sends a message, but does not expect a return message.

one-way operation

One type of WSDL-defined abstract operation, in which the service endpoint receives a message, but does not provide a return message. One-way operations specify only input message types.

operation

A message interaction between a service and a service consumer. The WSDL specification provides for four types of operations:

- one-way operation

- request-response operation

- solicit-response operation

- notification operation

output message

A message passed from a service provider to a service consumer. When mapped into Java, the parts of an output message are mapped to a method's output parameter list, including any return value. Output messages are defined using the **output** element in a WSDL contract.

See also: request-response operation, solicit-response operation, notification operation

port

> The address used to access a service. Ports are created based on the information specified in a WSDL **port** element.

participant

> A member of a SOA network, whether service provider, service consumer, or intermediary.

payload format

> The on-the-wire structure of messages over a given transport specified using a WSDL **binding** element.

physical contract

> The concrete portion of a WSDL contract that defines the bindings and transport details used by the services defined by that contract. The physical contract answers questions such as:
>
> - How is message traffic formatted on the wire?
>
> - How and where does message traffic travel?
>
> - Is there more than one option for transmitting a request?
>
> WSDL elements used in the physical contract include: **binding** element, **service** element, **operation** element, and **port** element.

reply

> A message returned by a service to a service consumer in response to a request from that consumer.
>
> See also: output message

request

> A message sent from a service consumer to a service provider asking for the service to perform an action.
>
> See also: input message

request-response operation

> One type of WSDL-defined abstract operation, in which the service endpoint receives a message and returns a correlated message. Request-response operations specify input message, output message, and fault message types.

response

> See reply

Representational State Transfer, REST

> An architectural style for services based on Roy Fielding's doctoral dissertation. REST takes the view that services can be fully implemented using the concepts encapsulated in the design of the Web. A service's operations are handled as if they were resources addressed by a URI. Requests are made using one of the four simple HTTP verbs: **GET**, **PUT**, **POST**, and **DELETE**.

resource

On the Web a resource is anything that can be identified using a URI. When developing RESTful services, a resource is a class or method that implements a piece of the application's functionality.

RESTful service

A service provider implemented using RESTful principles.

See also: Representational State Transfer, Java API for RESTful Services

Service Component Architecture, SCA

A set of specifications that describe a model for building applications and systems using a Service-Oriented Architecture. SCA extends and complements prior approaches to implementing services, and SCA builds on open standards such as Web services. SCA is developed by a consortium of companies. Compare with JBI.

Service-Oriented Architecture, SOA

A loosely-coupled distributed architecture in which service providers make resources available to service consumers in a standardized way. SOA is language and protocol independent.

service consumer

See consumer

solicit-response operation

One type of WSDL-defined abstract operation, in which the service endpoint sends a message and receives a correlated message.

substitution group

A feature of XML Schema that allows you to define groups of elements that may be used interchangeably in instance documents. For example, a **vehicle** head element might be defined with **automobile**, **boat**, and **airplane** substitution elements, any of which could be used wherever the **vehicle** element might be used. A substitution group is defined using the **substitutionGroup** attribute of the XML Schema element.

See also: abstract head element

service provider

A process or application that can respond to requests from a service consumer.

Web Services Addressing, WS-A, WS-Addressing

A specification that provides transport-neutral mechanisms to address Web services and messages. See the WS-Addressing specification.

Web Services Description Language, WSDL

An XML grammar for describing network services as a set of endpoints operating on messages containing either document-oriented or procedure-oriented information. WSDL is the language used to express service contracts. For further information see the WSDL specification.

Web Services Reliable Messaging, WS-RM

A specification that describes a protocol that allows messages to be delivered reliably between distributed applications in the presence of software component, system, or network failures.

Web Services Security, WSS

An OASIS specification that describes enhancements to SOAP messaging to provide a means for applying security to Web services. For further details, see the WSS specification.

Web service

An open set of standards for how systems connect to each other and communicate information. The standards are based on a distributed computing framework and provide a facility for applications or systems to collaborate regardless of location, hardware, or other implementation details.

CHAPTER 8. JAVA BUSINESS INTEGRATION TERMS

DEFINITIONS

Java Business Integration, JBI

A specification for a standards-based, vendor-neutral architecture, based on SOA principles, for the integration of disparate applications, service providers, and service consumers. JBI-compliant components are expected to plug in and interoperate with other JBI-compliant components. This frees vendors to concentrate on supplying components that implement their particular area of expertise without worrying about implementing the other necessary portions of a complete solution. JBI also frees end-users to pick and choose among many JBI-compliant components to assemble a SOA network sized to their needs, without locking in to one vendor's approach. The JBI specification was developed by the Java Community Process. Compare with SCA.

binding component

A JBI component that provides connectivity to services external to the JBI environment.

normalized message router, NMR

Part of the JBI architecture responsible for receiving message exchanges from JBI components and routing them to the appropriate component for processing.

service assembly

A collection of service units.

See also: service unit

service engine

A JBI component that provides business logic and transformation services and also consumes such services.

service unit

Artifacts deployed to a JBI component. A service unit configures the component to provide a piece of functionality such as expose an endpoint or route messages.