GT4: JAVA WS CORE

Contact: gawor@mcs.anl.gov
Web: http://dev.globus.org/wiki/Java_WS_Core

The Globus Toolkit's Java Web Services Core (Java WS Core) is a Java development kit for building stateful web services based on the WS-Resource framework. It also provides a lightweight hosting environment for the web services.

Java WS Core manages the lifecycle of services and their resources, provides persistence support and offers advanced security features. It also provides facilities for starting periodic and background tasks and has a unified way to store and retrieve service configuration data. At runtime Java WS Core provides an internal JNDI-based registry with service-specific and other configuration information. Web services can use this registry to lookup their configuration information, communicate with other services, or discover container-provided facilities.

The security features of Java WS Core include a pluggable authorization engine, declarative and programmatic security, and multiple authentication mechanisms. For example, the declarative security lets the user configure the security properties of a service using a configuration file without changing any of the service code. The transport-level (HTTPS) and message-level authentication methods are supported. Username-password, secure message and secure conversation authentication mechanisms of message-level security are also supported.

The programming model of Java WS Core is very flexible as it lets the service developer pick and choose small service building blocks and combine them together to create a service with the exact functionality the developer wants to provide. It also allows the developer to customize the built-in functionality or substitute it with alternative implementations.

The programming model of Java WS Core decouples the web service (business logic) functionality from the resource (state). The web service implementation is usually a plain, stateless Java object. A service can be composed from several independent web service operation implementations called "operation providers." These operation providers enable easy reuse of common web service operations among different services. The resource implementation is a stateful Java object that implements a set of 'callback' interfaces depending on what functionality the resource wants to provide. Resource objects are managed by a *ResourceHome* which is responsible for discovery, destruction, and/or creation of resources. Each web service usually has a corresponding *ResourceHome* implementation.

Java WS Core comes with a basic container for hosting the web services. However, Java WS Core can also run in other containers such as Tomcat, JBoss or OracleAS, etc. The basic container provided by Java WS Core offers less features than the other advanced containers but has much less overhead and therefore can be easily embedded in client applications. The Java WS Core container, for example, can be embedded in client applications to enable them to receive notifications.

Java WS Core is created with 'standard' Apache software components such as Apache Axis 1 (SOAP processor), Apache Addressing (WS-Addressing support), Apache WSS4J (message security support), and a few others. Java WS Core is an open source project licensed under an open license. Community participation is encouraged and always welcomed. For more information please see: http://dev.globus.org/wiki/Java_WS_Core

GT4: JAVA WS CORE

Contact: gawor@mcs.anl.gov
Web: http://dev.globus.org/wiki/Java_WS_Core

New features coming in Java WS Core in GT 4.2:

- **HTTP/S connection persistence** – By default Java WS Core will reuse all HTTP/S connections between the calls. This feature will improve performance especially for HTTPS connections as the SSL authentication will only be performed once and the authenticated connection will be reused for all subsequent calls.
- **Dynamic deployment** (standalone container only) – The dynamic deployment feature will enable the user to deploy or undeploy a service without restarting the container. It will also allow the user to direct the container to refresh itself and its services after a configuration change. An administrator could use this feature to install or uninstall services from a remote container.
- **WS-Enumeration support** – Java WS Core will include basic WS-Enumeration support and provide a service API for adding WS-Enumeration capabilities to any web service. With the WS-Enumeration support large XML data sets can be returned a chunk at a time instead of all at once in one huge message. This feature will improve scalability and robustness as clients and servers will not need to handle large data objects in memory.
- **TargetedXPath query dialect** – Java WS Core will provide a new XPath-based query dialect that enables reliable use of namespace prefixes within the query expression (since explicit prefix-namespace mappings will be passed along with the query). The new query dialect will also support querying of a particular resource property instead of the entire resource property document and have ability to return the query results as a WS-Enumeration.
- **SOAP with Attachments** – SOAP messages with attachments will be supported by the standalone and other containers. Attachments in DIME, MIME, and MTOM formats will be supported.
- **Advanced authorization engine** – Java WS Core will contain a new attribute-based authorization engine. Please see the *Globus Toolkit Security Infrastructure* brochure for more information on the new engine and other security improvements.
- Many other improvements and optimizations are included such as better application server support, automatic validation of web service and security deployment descriptors, and updated 3rd party libraries including Apache Axis.