jUDDI Documentation

1. jUDDI

1.1. Welcome to jUDDI

jUDDI (pronounced "Judy") is an open source Java implementation of the Universal Description, Discovery, and Integration (UDDI) specification for Web Services.

1.1.1. jUDDI Features

- Open Source
- Platform Independent
- Supports for JDK 1.3.1 and later
- UDDI version 2.0 compliant implementation
- Use with any relational database that supports ANSI standard SQL (MySQL, DB2, Sybase, JDataStore, etc.)
- Deployable on any Java application server that supports the Servlet 2.3 specification (Jakarta Tomcat, JOnAS, WebSphere, WebLogic, Borland Enterprise Server, JRun, etc.)
- jUDDI registry supports a clustered deployment configuration.
- Easy integration with existing authentication systems

1.1.2. About UDDI

The Universal Description, Discovery and Integration (UDDI) protocol is one of the major building blocks required for successful Web services. UDDI creates a standard interoperable platform that enables companies and applications to quickly, easily, and dynamically find and use Web services over the Internet. UDDI also allows operational registries to be maintained for different purposes in different contexts. UDDI is a cross-industry effort driven by major platform and software providers, as well as marketplace operators and e-business leaders within the OASIS standards consortium.

Additional information regarding UDDI can be found at UDDI.org.

1.2. jUDDI News

1.2.1. News

March 24, 2004 jUDDI graduates from the Apache Incubator into Apache Web Services Project.

Oct 08, 2003 The Apache Software Foundation's Web Services PMC (http://ws.apache.org/) voted to accept jUDDI into the Apache Incubator. The move is expected to be complete in early December.

July 22, 2003 A proposal to contribute jUDDI to the Apache Software Foundation's Web Services Project is being considered.

July 22, 2003 Version 0.8.0 is now available.

June 27, 2003 Version 0.7.1 is now available.

April 22, 2003 Version 0.7.0 is now available.

1.3. jUDDI CVS Repository

1.3.1. CVS Repository

Most users of the source code probably don't need to have day to day access to the source code as it changes. For these users we provide easy to unpack source code downloads via our download pages.

View the Source Tree

The latest CVS sources can be viewed at http://cvs.apache.org/viewcvs/ws-juddi/

Access the Source Tree (AnonCVS)
So, you've decided that you need access to the source tree to see the latest and greatest code. There's two different forms of CVS access. The first is anonymous and anybody can use it. The second is not and you must have a login to the development server. If you don't know what this means, join the mailing list and find out.

Anyone can checkout source code from our anonymous CVS server. To do so, simply use the following commands (if you are using a GUI CVS client, configure it appropriatly):

```
cvs -d :pserver:anoncvs@cvs.apache.org:/home/cvspublic login
password: anoncvs

cvs -d :pserver:anoncvs@cvs.apache.org:/home/cvspublic checkout ws-juddi
```
Full Remote CVS Access
If you are a Committer and have a login on the Apache development server, this section is for you. If you are not a Committer, but you want to submit patches or even request commit privelages, please see the Jakarta GuideLines page (we follow the same rules) for more information.

To have full access to the CVS server, you need to follow the links depending on the operating system you are using:

- Unix
- Windows

1.4. jUDDI Mailing Lists

We encourage everyone interested to get involved and contribute. These forums are where the jUDDI developer and user communities meet to discuss issues and plans. Please join us!

Before subscribing to any of the mailing lists, please make sure you have read and understand the mailing list guidelines.

1.4.1. jUDDI Users List

Medium Traffic Subscribe Unsubscribe Subscribe Digest Unsubscribe Digest Send mail to list MBOX Archive

The juddi-user@ws.apache.org mailing list is the primary source of support for anyone using jUDDI. Use this list for posting questions (and answers) related to the installation, configuration and use of jUDDI.

Browse juddi-user archives at Apache.org
Browse juddi-user archives at MARC

Search MARC:
Subjects Authors Bodies for list: juddi-user

1.4.2. jUDDI Developers List

Medium Traffic Subscribe Unsubscribe Subscribe Digest Unsubscribe Digest Send mail to list MBOX Archive

The juddi-dev@ws.apache.org mailing list is for discussions about the jUDDI architecture, build process, testing and implementation details.

Browse juddi-dev archives at Apache.org
Browse juddi-dev archives at MARC

Search MARC:
Subjects Authors Bodies for list: juddi-dev

1.4.3. jUDDI Commit List

Low Traffic Subscribe Unsubscribe Subscribe Digest Unsubscribe Digest MBOX Archive

The juddi-cvs@ws.apache.org mailing list is for tracking changes to the jUDDI source as they are committed. This list is informational only so please do not post messages to this list. Questions regarding activity on this list should be posted on the Developer list.

Browse juddi-cvs archives at Apache.org
Browse juddi-cvs archives at MARC

Search MARC:
Subjects Authors Bodies for list: juddi-cvs

1.4.4. Mailing List Guidelines

Every volunteer project obtains its strength from the people involved in it. Mailing lists provide a simple and effective communication mechanism.

You are welcome to join any of our mailing lists (or all of them if you wish). You can choose to lurk, or actively participate. It's up to you.

Before you join these lists, you should make sure that you read and follow the guidelines below.

Do your best to respect the charter of the mailing list
There are three types of lists that you can join:

- The User list is where you should send questions and comments about configuration, setup, usage and other "user" types of questions.
- The Developer list is where you should should send questions and comments about the actual software source code and general "development" types of questions.
- The Commit list is used for tracking committer activity in the project's CVS repository. This list is used for informational purposes only. Please do not post messages to this list. Questions regarding activity on this list should be posted on the Developer list.

Join the lists that are appropriate for your discussion
Please make sure that you are joining the list that is appropriate for the topic or product that you would like to discuss.

Do not abuse resources in order to get help
Asking your configuration or user type of question on the developers list because you think that you will get help more quickly by going directly to the developers instead of to the user base is not very nice. Chances are that doing this will actually prevent people from answering your question because it is clear that you are trying to abuse resources.

Avoid sending HTML or "Stylized" email to the list

Do your best to ensure that you are not sending HTML or "Stylized" email to the list. If you are using Outlook or Outlook Express or Eudora, chances are that you are sending HTML email by default. There is usually a setting that will allow you to send "Plain Text" email. If you are using Microsoft products to send email, there are several bugs in the software that prevent you from turning off the sending of HTML email.

Watch where you are sending email
The majority of our mailing lists have set the Reply-To to go back to the list. That means that when you Reply to a message, it will go to the list and not to the original author directly. The reason is because it helps facilitate discussion on the list for everyone to benefit from. Be careful of this as sometimes you may intend to reply to a message directly to someone instead of the entire list.

Do not crosspost messages
In other words, pick a mailing list and send your messages to that mailing list only. Do not send your messages to multiple mailing lists. The reason is that people may be subscribed to one list and not to the other. Therefore, some people may only see half of the conversation.

1.5. Participation

Every volunteer project obtains its strength from the people involved in it. We invite you to participate as much or as little as you choose. The roles and responsibilities that people can assume in the project are based on merit. Everybody's input matters!

There are a variety of ways to participate. Regardless of how you choose to participate, we suggest you join some or all of our mailing lists.

1.5.1. Use the Products & Give Us Feedback

Using the products,reporting bugs, making feature requests, etc. is by far the most important role. It's your feedback that allows the technology to evolve.
* Join Mailing Lists
* Download Binary Builds
* Report bugs/Request additional features

1.5.2. Contribute Code or Documentation Patches

In this role, you participate in the actual development of the code. If this is the type of role you'd like to play, here are some steps (in addition to the ones above) to get you started:
* Read Mailing List Guidelines
* Review Reference Library
* Download the Source Code

- [Access CVS Repository](#)

Any and all new development or bug fixing uses the latest code base. If you have a patch, it should be submitted against the latest available code.

1.5.3. Getting Support on the Lists

We do all development and bug fixing on the latest available code base. If you've encountered what you think is a bug, please first check the [bug database](#) for known issues. If you can't find your bug, make sure you're using the latest code base. We are not able to fix bugs on earlier code bases.

Please include any relevant version information and the error message text, if there is one. If you have a problem with a specific interface, like find_business, you'll likely be asked to post a log of the SOAP message exchanges between your client and the jUDDI server. You can use the axis tool [TCPMon](#) to capture these messages.

Remember, Apache helps those that help themselves. Please take a look at Eric S. Raymond's and Rick Moen's explanation of ["How to Ask Questions The Smart Way](#).

1.5.4. jUDDI Wiki

Like other Apache projects, jUDDI maintains a wiki, a sort of virtual meeting place for the jUDDI community. This is a resource available to everyone, and you are encouraged to contribute. This is a good resource for 'HowTos' for common configuration issues.
- http://wiki.apache.org/ws/jUDDI

1.6. jUDDI Reference Library

The jUDDI Project lives or fails based on its human resources. Users and contributors alike help the project with ideas and brainpower. A common foundation of knowledge is required to effectively participate in this virtual community. The following is a list of documents that we have found helpful for us and may be helpful to you.

1.6.1. UDDI Specifications

[UDDI.org](#)
[UDDI Version 1.0 Specification](#)
[UDDI Version 2.0 Specification](#)
[UDDI Version 3.0 Specification](#)

1.6.2. Articles, Tutorials and Best Practices

Best practices for Web services versioning
Keep your Web services current with WSDL and UDDI
Kyle Brown and Michael Ellis, IBM developerWorks
January 2004

Registering a Web Service in UDDI using the new WSDL mapping model
Anne Thomas Manes, WebServices Journal
October 2003

Bringing Order to Enterprise Service Proliferation
The foundation of a flexible infrastructure
Toufic Boubez, WebServices Journal
October 2003

A Second Look at UDDI
The stepchild of Web services could be a cornerstone of service-oriented architectures beginning to take shape
Jim Ericson, Line56.com
August 2003

Discovering Web Services: UDDI (a Tutorial)
Rick Hightower, IBM developerWorks
February 2003

Registering and Discovering RSS Feeds in UDDI
Keep your Web services current with WSDL and UDDI
Karsten Januszewski, GotDotNet.com
July 2002

Using WSDL in a UDDI Registry
John Colgrave and Karsten Januszewski, UDDI.org
November 2003

Using UDDI at Runtime Part I
Building the Web Services Infrastructure to Support Client Applications
Karsten Januszewski, Microsoft
December 2001

Using UDDI at Runtime Part II
Dynamically Binding to Multiple Web Service Implementations
Karsten Januszewski, Microsoft
May 2002

Understanding UDDI and JAXR

The why and what of the new approach
Satya Komatineni, O'Reilly ONJava.com
February 2002

Registration and Discovery of Web Services Using JAXR with XML Registries such as UDDI and ebXML
The why and what of the new approach
Qusay H. Mahmoud, Sun Microsystems
June 2002

Java API for XML Registries (a Tutorial)
The why and what of the new approach
Kim Haase, Sun Microsystems

A New Approach to UDDI and WSDL Part 1
The why and what of the new approach
John Colgrave, IBM developerWorks
August 2003

A New Approach to UDDI and WSDL Part 2
Usage scenarios in publishing and finding WSDL service descriptions
John Colgrave, IBM developerWorks
September 2003

Understanding UDDI tModels and Taxonomies
The key integration point for products supporting Web services
Andy Grove, Web Services Journal (Volume 1, Issue 2)
January 2001

Understanding WSDL in a UDDI Registry Part 1
How to publish and find WSDL service descriptions
Peter Brittenham, Francisco Cubera, Dave Ehnebuske, Steve Graham, IBM developerWorks
September 2001 (updated September 2002)

Understanding WSDL in a UDDI Registry Part 2
Usage scenarios in publishing and finding WSDL service descriptions
Peter Brittenham, IBM developerWorks
September 2001

Understanding WSDL in a UDDI Registry Part 3
How to publish and find WSDL service descriptions
Peter Brittenham, IBM developerWorks
November 2001

UDDI - A Foundation for Web Services
Thomas A. Bellwood, IBM developerWorks
November 2001

Why UDDI Will Succeed, Quietly: Two Factors Push Web Services Forward
Brent Sleeper, The Stencil Group
April 2001

1.6.3. Related Specifications

SOAP W3C Specification

Web Services Description Language (WSDL) 1.1

Java API for XML Registries (JAXR)

1.6.4. Other Resources

The Java Language Specification
Written by the creators of the Java Programming Language, this online book is considered by many to be the bible for programming in Java. A must read.

Javadoc
Javadoc is the automatic software documentation generator used by Java since it was first released. All code written for this project must be documented using Javadoc conventions.

The Java Code Conventions
This Sun document specifies the de-facto standard way of formatting Java code. All code written for this project must follow these conventions.

Open Source Development with CVS
Written by Karl Fogel, this is an online version of many of the primary chapters from the dead-tree version of his book.

Introduction to CVS
Written by Jim Blandy, this brief introduction gives a first look into CVS. If you have never used CVS before, you'll want to start here.

Version Management with CVS
Written by Per Cederqvist at al, this is the main manual for CVS. It provides details on all documented CVS features.

1.7. jUDDI Bugs

1.7.1. Bug Tracking

Like many other Apache projects, we've begun using JIRA. Our tracking database can be found here:

* http://nagoya.apache.org/jira/secure/BrowseProject.jspa?id=10401

2. Documentation

2.1. jUDDI Installation Guide

2.1.1. Requirements

jUDDI is a pure Java web application and as such can be deployed to any application server or servlet engine that supports version 2.1 or later of the servlet API. If you need an application server, we recommend Jakarta Tomcat. Note also that jUDDI requires Java 1.3 or later. As with any Java web application, deployment to your application server or servlet engine will vary on a product-by-product basis.

Instructions for deploying jUDDI to several application servers have been donated and are available in the HOW-TO section of the jUDDI wiki.

jUDDI also requires an external datastore in which to persist the registry data it manages. Typically this is a relational database management system such as MySQL, Oracle or DB2. Support for several open source and commercial database products are included. See the section below titled Persistence for more information.

2.1.2. Configuration

To properly configure and deploy jUDDI it will be helpful to understand a bit about it's architecture. jUDDI consist of a core request processor that unmarshalls incoming UDDI requests, invoking the appropriate UDDI function and marshalling UDDI responses (marshalling and unmarshalling is the process of converting XML data to/from Java objects).

To invoke a UDDI function jUDDI employs the services of three configurable sub-components or modules that handle persistence (the DataStore), authentication (the Authenticator) and the generation of UUID's (the UUIDGen). jUDDI is bundled and pre-configured to use default implementations of each of these modules to help your registry up and running quickly. These sub-components and a description of the default implementations are described below.

Several public Java interfaces for creating your own DataStore, Authenticator and UUIDGen

module implementations are available. Please see the jUDDI Developer's Guide for more information regarding jUDDI module development.

2.1.3. Persistence (jUDDI DataStore)

jUDDI needs a place to store it's registry data so it should come as no surprise that jUDDI is pre-configured to use JDBC and any one of several different DBMSs to do this.

The process of setting this up is straight forward. Start by creating a new jUDDI database using the instructions for your preferred DBMS which you will find in the 'sql' directory of your jUDDI distribution. At the time of this writing instructions for the following products were available:

* MySQL
* DB2
* HSQLdb (HypersonicSQL)
* Sybase
* PostreSQL
* Oracle
* TotalXML
* JDataStore (Borland)

If support for your DBMS is not listed you might try posting a message to the juddi-user list to see if someone has already developed support for it. You can also try to create the scripts yourself by using the instructions for a supported DBMS as a guide. Please consider contributing your work back to the project so the next person won't have the same issue.

To complete the DataStore set up you'll need to configure a JNDI Datasource with a name of 'jdbc/juddiDB' in the application server or servlet engine that you're deploying to. Datasource setup varies on an product-by-product basis so review documentation for your application server. If you're deploying to Jakarta Tomcat, take a look at Tomcat's JNDI Datasource HOW-TO for assistance.

2.1.4. Authentication (jUDDI Authenticator)

Authenticating a jUDDI publisher is a two-step process. The first step confirms that the ID/password combination provided by the user in a get_authToken request is valid. The default Authenticator implementation simply approves any authentication attempt. It is expected that a typical jUDDI deployment will use an external authentication mechanism. It is our hope that additional jUDDI authentication implementations will be developed by jUDDI users as they determine how they would like authentication to take place in their particular environment. See the jUDDI Developers Guide for more information on

developing a custom jUDDI authentication module for your environment.

The second step confirms that the publisher has been defined to jUDDI. A publisher is said to be defined when a row identifying the publisher exists in the PUBLISHER table of the jUDDI datastore. At the moment the only way to do this is via SQL. An example of defining a new publisher named John Doe would look like this:

INSERT INTO PUBLISHER (PUBLISHER_ID,PUBLISHER_NAME,ADMIN,ENABLED) VALUES ('jdoe','John Doe','false','true');

The PUBLISHER table consists of several columns but only four of them are required and they are defined as follows:

Column Name	Description
PUBLISHER_ID	The user ID the publisher uses when authenticating. IMPORTANT: This should be the same value used to authenticate with the external authentication service.
PUBLISHER_NAME	The publisher's name (or in UDDI speak the Authorized Name).
ADMIN	Indicate if the publisher has administrative privileges. Valid values for this column are 'true' or 'false'. The ADMIN value is currently not used.
ENABLED	Indicate if the publishers account is enabled and eligible for use.

The jUDDI web application will (eventually) be extended to facilitate the Publisher creation process. The value of the ADMIN column in the PUBLISHER table above will be used to determine who has the privilege to create new jUDDI publishers.

2.1.5. UUID Generation (jUDDI UUIDGen)

There's nothing for you to do here but I thought I'd offer a little information about how, why and where jUDDI makes use of UUID generation.

The UDDI specification indicates that each Business, Service, Binding and TModel (Technical Model) is to be uniquely identified by a Universally Unique ID (UUID). Additionally, jUDDI also uses the UUID generator to create AuthTokens.

Generation of a UUID typically requires access to hardware level information that (unfortunately) is not easily accessible from Java. Fortunately, the UUID specification offers

an alternative method for generating these ID's when this hardware information is not present. By default the jUDDI implements this alternative method.

2.1.6. Logging

When deploying jUDDI you may wish to make changes to the juddi.properties and log4j.properties files. These files are located in the juddi webapp's WEB-INF/classes directory. They're here because they need to be in the classpath for jUDDI to locate and load them at runtime. One Log4j property value that you'll most likely want to set is log4j.appender.LOGFILE.File which specifies the name and location of the jUDDI log file.

2.2. jUDDI User's Guide

2.2.1. User's Guide

2.3. jUDDI Developer's Guide

2.3.1. Developer's Guide

3. Downloads

3.1. jUDDI Releases

3.1.1. Releases

Name	Date	Description
0.9rc1	June 16, 2004	Release Candidate #1 for Version 0.9.

For nightly builds or nightly snapshots of the CVS tree see the Interim Drops page.

3.2. jUDDI Interim Drops

3.2.1. Interim Drops

We're not currently creating nightly builds of jUDDI however nightly snapshots of the current CVS tree are available here:

http://cvs.apache.org/snapshots/ws-juddi/

4. Related Projects

5. Misc

5.1. Web Services - jUDDI

5.1.1. Web Services - jUDDI - Who We Are

The jUDDI Project operates on a meritocracy: the more you do, the more responsibility you will obtain. This page lists all of the people who have gone the extra mile and are Committers. If you would like to get involved, the first step is to join the mailing lists.

We ask that you please do not send us emails privately asking for support. We are non-paid volunteers who help out with the project and we do not necessarily have the time or energy to help people on an individual basis. Instead, we have setup mailing lists which often contain hundreds of individuals who will help answer detailed requests for help. The benefit of using mailing lists over private communication is that it is a shared resource where others can also learn from common mistakes and as a community we all grow together.

5.1.2. Active Committers

- Steve Viens <sviens@apache.org>
- Anou Manavalan <anou_mana@apache.org>
- Andy Cutright <acutright@apache.org>
- Anil Saldhana <anil@apache.org>

5.1.3. Committers Emeriti (committers that have been inactive for 3 months or more)

None at this time.

5.2. jUDDI Committer Notes

5.2.1. Updating the jUDDI Web Site

You'll need the appropriate version of [forrest](#) as well as the appropriate version of [ant](#). This document assumes you generally understand how to use these tools.

The site source is located under our source tree at ws-juddi/site. To build the site, simply invoke forrest in ws-juddi/site. The built sources will be located at ws-juddi/build/site. You

should get a clean build, and no broken links. There aren't any known issues with the site.

You must test your builds locally by running forrest on your local box. You can do this by invoking 'forrest run' in ws-juddi/site.

The commit process is a bit complicated. The live site is hosted under the ws-site source tree, -not- the ws-juddi src tree. The ws-juddi/build/site directory hierarchy maps to the ws-site/targets/juddi directory hierarchy. Once you've built the site copy the latter hierarchy to the former, and then commit the changes. The live site is updated about every 6 hours from cvs.

You must -also- commit the changes you've made under ws-juddi/site to persist the changes for the next committer to be able to reproduce your modifications.

To fully clean any site build you've created, you can safely delete everything under ws-juddi/site/build.

5.2.2. Release Process

Release Manager

One committer will be elected or hopefully volunteer to assemble the binary releases and label the source tree.

Digitally Signing Releases

Apache policy requires binary releases be digitaly signed. The Apache process has not been formalized, but a general discussion about creating digital signatures and signing releases is available at http://nagoya.apache.org/wiki/apachewiki.cgi?SigningReleases. This covers some basics about using GnuPG to create key pairs and sign releases. Our goal here is to discuss jUDDI signing requirements, and provide some useful examples to release managers, not discuss digital signatures or encryption technology. Our discussion uses GnuPG, but any compliant software could be used. The examples below come from the GnuPG manual. This discussion is not a subsitute for reading that manual.

Creating a key pair is pretty simple using gpg. Simply invoke gpg and take all the defaults when prompted. You will have to provide a passphrase. Be sure to remember the passphrase since you'll need it anytime you use the key pair. The passphrase should itself be sufficiently secure; it shouldn't simply be a word in a dictionary, should include a mix of digits and alphanumeric characters, etc.

```
        gpg --gen-key
```

You should also generate a revocation certificate. This allows you to declare the key pair

invalid publically, if you ever lose your private key, or it becomes compromised.

```
gpg --output revoke.as --gen-revoke mykey
```

The release manager is responsible for signing the binaries. The release manager must have a public key listed in the 'KEYS' file at the root of our source tree. The release manager must create a detached signature for each binary. This detached signature must be posted along with our binaries, and allow our users to verify the binary's integrity.

```
gpg --output jUDDI.tar.gzip.asc --detach-sig jUDDI.tar.gzip
```

All jUDDI committers are encouraged to create public/ private key pairs and place the public half into our 'KEYS' file at the root of our source tree. jUDDI committers are also encouraged to verify one another's keys and sign them, to help create a web of trust. Verifying a signature and a binary guarantees (in any real sense) the binary was assembled by the person that signed it. However, it does not prove the person signing it can be trusted. A web of trust can be created by signing one another's keys. This allows users and developers to 'trust' the person who created the document-signature pair to provide a secure, safe binary.

5.3. Web Services - jUDDI

5.3.1. Web Services - jUDDI - Contact Us

If you have questions or comments about this site, please send email to:

juddi-dev@ws.apache.org.

If you have questions or comments about the software or documentations on this site, please subscribe to the juddi-user mailing list:

Mailing lists

The jUDDI project is an effort of the Apache Software Foundation. The address for general ASF correspondence and licensing questions is:

apache@apache.org

You can find more contact information for the Apache Software Foundation on the contact page of the main Apache site.

5.4. Web Services - jUDDI

5.4.1. Web Services - jUDDI - Legal Stuff

All material on this website is Copyright © 1999-2004, The Apache Software Foundation.

Sun, Sun Microsystems, Solaris, Java, JavaServer Web Development Kit, and JavaServer Pages are trademarks or registered trademarks of Sun Microsystems, Inc. UNIX is a registered trademark in the United States and other countries, exclusively licensed through X/Open Company, Ltd. Windows, WindowsNT, and Win32 are registered trademarks of Microsoft Corp. All other product names mentioned herein and throughout the entire web site are trademarks of their respective owners.