



JavaOneSM
Sun's 2003 Worldwide Java Developer Conference™

Sun Open Net Environment (Sun ONE) Directory Client Programming

Ludovic Poitou
Vincent Ryan

Sun Microsystems, Inc.

Agenda

- Introduction
 - Sun ONE Directory Server 5.2
 - Java Naming & Directory Interface™ (JNDI)
- LDAP programming
- DSML programming
- Q & A

Sun ONE Directory Server 5.2

- Powerful, scalable, distributed directory server
- Industry-leading LDAP server
- DS 5.2 release date is June 20, 2003

What's new in DS 5.2?

- 64-bit server application
- IPv4 and IPv6 support
- DSMLv2 support
- Performance and scalability improvements
- Updated and improved server management console
- Simplified migration from versions 4.16 and 5.x

What's new in DS 5.2? (continued)

- Security features
 - Kerberos v5 authentication
 - Multiple password policies
 - Query the access control rights in effect
 - Attribute encryption at the database level
- Replication
 - Multi-master over WAN
 - Supports up to 4 masters
 - Performance improvement
 - Fractional replication
 - Monitoring tools

What is JNDI?

- Java API for accessing naming and directory services
 - Variety of service providers are available (including LDAP, DSML)
- Part of Java 2 Platform, Standard Edition™ (J2SE™) since version 1.3
- J2SE SDK contains
 - JNDI
 - LDAP, DNS, RMI and COS service providers

JNDI/LDAP Booster Pack

- JNDI/LDAP Booster Pack 1.0
 - Extends the functionality of the LDAP service provider in J2SE
 - Available in:
 - Sun ONE Directory Server Resource Kit 5.2
 - Download from:
 - <http://java.sun.com/jndi/>

JNDI/LDAP Features

- Fully-compliant LDAP V3 client (RFC 2251,...)
- Supports DS 5.2 features
- Common LDAP controls and extended op's
- LDAP schema support
- SSL support
- Connection management
- LDAP group object support
- SASL authentication mechanisms

JNDI Support for Controls

- Popular LDAP controls
 - `SortControl`
 - `PagedResultsControl`
 - `VirtualListViewControl`
 - `PersistentSearchControl*`
- DS 5.2-specific LDAP controls
 - `PasswordExpiredResponseControl`
 - `PasswordExpiringResponseControl`
 - `ProxiedAuthorizationControl`
 - `AuthorizationIDControl`
 - `GetEffectiveRightsControl`
 - `VirtualAttributesOnlyControl`
 - `RealAttributesOnlyControl`
- Requires the JNDI/LDAP Booster Pack 1.0

JNDI Support for Extended Ops

- Popular LDAP extended op's
 - `startTlsRequest*`
- DS 5.2-specific LDAP extended op's
 - `BulkImportStartRequest`
 - `BulkImportFinishedRequest`
 - `WhoAmIRequest`
- Requires the JNDI/LDAP Booster Pack 1.0

JNDI Support for Schema

- Use `DirContext.getSchema` to generate a schema tree
 - `AttributeDefinition` context
 - `ClassDefinition` context
 - `SyntaxDefinition` context
 - `MatchingRule` context
- Each LDAP schema string description (RFC 2252) is parsed into a set of attributes
- Operate on contexts using JNDI methods

JNDI Support for SSL

- Use `java.naming.security.protocol` property to activate SSL connections
- Or use `ldaps://` URLs
- Or use `startTlsRequest` extended operation

JNDI Support for Connection Mgmt

- Connection pooling
 - Use `com.sun.jndi.ldap.connect.pool`
- Connection timeouts
 - Use `com.sun.jndi.ldap.connect.timeout`
- Alternative LDAP servers
 - Set `java.naming.provider.url` property to a space-separated list of LDAP URLs
- Control LDAP referral dereferencing
 - Set `java.naming.referral` to `follow` or `throw` or `ignore` (default)

JNDI Support for Connection Mgmt (continued)

- Automatic LDAP server discovery (via DNS)
 - RFC 2782
 - Use `ldap:///<dn>` URLs
 - URL DN is mapped to a DNS domain name
 - Domain's SRV resource record is examined
- Notification of closed connections
 - Use `EventContext.addNamingListener` to register an `UnsolicitedNotificationListener`

JNDI Support for Groups

- All the usual group operations are supported
 - Create and delete a group
 - Find and retrieve a group
 - Add and remove members
 - Test for group membership
 - List a group's members
 - Union, intersection and complement set op's
- Requires the JNDI/LDAP Booster Pack 1.0

JNDI Support for Groups (continued)

- A JNDI group object is backed by an LDAP directory entry
- Static groups
 - Members listed in a multivalued attribute
- Dynamic groups
 - Membership defined by LDAP URL(s)
- Supported LDAP object classes
 - **GroupOfNames** (RFC 2256)
 - **GroupOfUniqueNames** (RFC 2256)
 - **GroupOfURLs**

Code Sample

Activate the JNDI/LDAP provider

```
import javax.naming.*;
import javax.naming.directory.*;

Hashtable env = new Hashtable(7);
// Initialize the initial context factory
env.put(Context.INITIAL_CONTEXT_FACTORY,
        "com.sun.jndi.ldap.LdapCtxFactory");
// Initialize the target server
env.put(Context.PROVIDER_URL,
        "ldap://hostname:389/dc=example,dc=net");

// Optionally, initialize the security properties

// Create a new directory context
DirContext ctx = new InitialDirContext(env);
```

Code Sample

Create a new static group

```
import java.security.acl.Group;  
import com.sun.jndi.ldap.obj.GroupOfUniqueNames;  
  
// Initialize a set of members  
Set s = new HashSet();  
s.add("cn=jack,ou=people,dc=example,dc=net");  
s.add("cn=jill,ou=people,dc=example,dc=net");  
  
// Create the group object  
Group headbangers = new GroupOfUniqueNames(s);
```

Code Sample

Create a new dynamic group

```
import java.security.acl.Group;  
import com.sun.jndi.ldap.obj.GroupOfURLs;  
  
// Initialize a set of members  
Set s = new HashSet();  
s.add("ldap:///ou=people,dc=example,dc=net??sub?  
(employeeType=manager)");  
  
// Create the group object  
Group managers = new GroupOfURLs(s);
```

Code Sample

Bind the group in the directory

```
// Set the java.naming.factory.object and the
// java.naming.factory.state properties to the
// value com.sun.jndi.ldap.obj.LdapGroupFactory.

// Set the java.naming.provider.url property to
// an LDAP URL identifying the LDAP server.

// Create a new directory context
DirContext ctx = new InitialDirContext();

// Bind the group object
ctx.bind("cn=managers,ou=groups,dc=example,dc=net",
        managers);
```

Code Sample

Update a static group

```
import javax.security.auth.x500.X500Principal;  
  
// Retrieve the group object  
Group admins = (Group) ctx.lookup(  
    "cn=admins,ou=groups,dc=example,dc=net");  
  
// Add a new member  
admins.addMember(new X500Principal(  
    "cn=joe,ou=people,dc=example,dc=net"));
```

Code Sample

Test for group membership

```
import javax.security.auth.x500.X500Principal;

// Retrieve the group object
Group admins = (Group) ctx.lookup(
    "cn=admins,ou=groups,dc=example,dc=net");

// Check membership
if (admins.isMember(new X500Principal(
    "cn=joe,ou=people,dc=example,dc=net")) {
    // joe is a member!
}
```

Code Sample

List the members of a group

```
// Retrieve the group object
Group admins = (Group) ctx.lookup(
    "cn=admins,ou=groups,dc=example,dc=net");

// List the members
for (Enumeration members = admins.members();
     members.hasMoreElements(); ) {
    Principal principal =
        (Principal) (members.nextElement());
    if (principal instanceof Group) {
        // group member
        System.out.println(principal.getName());
    } else { // individual member
        System.out.println(principal.getName());
    }
}
```

Group Set Operations

- Union, intersection and complement
 - Members in either group 1 or group 2
 - Use `Collection.addAll`
 - Members in both group 1 and group 2
 - Use `Collection.retainAll`
 - Members in group 1 that are not in group 2
 - Use `Collection.removeAll`

Code Sample

List any members which are in both groups

```
// Retrieve two group objects
Group g1 = (Group) ctx.lookup("cn=sys admins");
Group g2 = (Group) ctx.lookup("cn=dir admins");

// Extract the members of the first group
ArrayList intersection =
    Collections.list(g1.members());
// Intersect with the members of the second group
intersection.retainAll(
    Collections.list(g2.members()));
// Display the resulting set
for (Iterator i = intersection.iterator();
     i.hasNext(); ) {
    System.out.println(i.next());
}
```

Hints and Tips

- Control the recursive expansion of subgroups
 - `com.sun.jndi.ldap.obj.expandGroup`
 - Affects operation of `isMember` and `members`
- Group object must be bound in the directory for all its methods to function
- Omit `server/port#` in URLs of dynamic groups
- Use `sortControl` for a sorted list of members
- To explicitly release resources
 - Call `close` on a group object (LDAP unbind)
 - Call `close` on an enumeration (LDAP abandon)

JNDI Support for SASL

- Supported SASL mechanisms
 - DIGEST-MD5
 - EXTERNAL
 - GSSAPI
- Use `java.naming.security.authentication` property to select a mechanism
- For fine-grained control of SASL, use
 - `java.naming.security.sasl.*` properties
 - `javax.security.sasl.*` properties

GSSAPI SASL Authentication

- Only suited for use of Kerberos v5
- Requires slightly different programming model
 - Kerberos authentication takes place before the LDAP connection
 - Can take advantage of Single Sign-on support
 - Directory applications can use the credentials which have already been established
- 3 steps are required:
 - Authenticate to Kerberos (can leverage SSO)
 - Assume the identity of the principal
 - Perform authenticated JNDI tasks

Code Sample

Authenticate to Kerberos

```
import javax.naming.*;  
import javax.naming.directory.*;  
import javax.security.auth.login.*;  
import javax.security.auth.Subject;  
  
LoginContext lc = null;  
try {  
    lc = new LoginContext("sample",  
                        new SampleCallbackHandler());  
    lc.login();  
} catch (LoginException le){  
    ...  
}  
subject.doAs(lc.getSubject(), new JndiAction(args));
```

Code Sample

Perform authenticated JNDI tasks

```
class JndiAction implements
java.security.PrivilegedAction {

public Object run(){
env.put(Context.INITIAL_CONTEXT_FACTORY,
    "com.sun.jndi.ldap.LdapCtxFactory");

env.put(Context.PROVIDER_URL,
    "ldap://hostname.example.net:389/dc=example,
dc=net");

// Request GSSAPI SASL mechanism
env.put(Context.SECURITY_AUTHENTICATION, "GSSAPI");

DirContext ctx = new InitialDirContext(env);
```

JNDI/DSML

- JNDI/DSMLv2 service providers
 - Provides a DSMLv2 SOAP client and DSMLv2 document processors
 - Available in:
 - Sun ONE Directory Server, Resource Kit 5.2
 - Download from:
 - <http://java.sun.com/jndi/>

JNDI/DSML Features

- Fully-compliant with DSML Version 2 (OASIS DSML 2.0 Specification)
- DSMLv2 service providers
 - DSMLv2 SOAP client
 - Uses the SOAP over HTTP/1.1 binding
 - DSMLv2 request generator
 - Produces DSMLv2 doc using the file binding
 - DSMLv2 search response processor
 - Imports/exports DSMLv2 using the file binding
- LDAP URL handler
 - Generates DSMLv2 from LDAP search results

Code Sample

Activate the JNDI/DSMLv2 provider

```
import javax.naming.*;
import javax.naming.directory.*;

Hashtable env = new Hashtable(7);
// Initialize the initial context factory
env.put(Context.INITIAL_CONTEXT_FACTORY,
        "com.sun.jndi.dsmlv2.soap.DsmlSoapCtxFactory");
// Initialize the target server
env.put(Context.PROVIDER_URL,
        "http://hostname:8080/dsml");

// Optionally, initialize the security properties

// Create a new directory context
DirContext ctx = new InitialDirContext(env);
```

Code Sample

Search for LDAP entries using DSMLv2

```
// Create a new directory context
DirContext ctx = new InitialDirContext(env);
// Initialize the search controls
SearchControls constraints = new SearchControls();
constraints.setSearchScope(
    SearchControls.SUBTREE_SCOPE);

// Perform the DSML search
NamingEnumeration results = ctx.search(
    "dc=example,dc=net", "(objectclass=*)",
    constraints);
// Display the results
while (results.hasMore()) {
    System.out.println(results.next());
}
```

Code Sample

Modify an LDAP entry using DSMLv2

```
// Optionally, initialize the security properties:  
//     java.naming.security.principal  
//     java.naming.security.credentials  
//     com.sun.jndi.dsmlv2.authzid  
  
// Create a new directory context  
DirContext ctx = new InitialDirContext(env);  
  
// Perform the DSML modification  
ctx.modifyAttributes("cn=joe,dc=example,dc=net",  
    DirContext.ADD_ATTRIBUTE,  
    new BasicAttributes("mail", "joe@example.net"))
```

Hints and Tips

- LDAP control values are not specified as XML types in DSMLv2
 - Defined as the `anyType` type
 - The control value is the Base64 encoding of the BER encoding
- Proxy authorization is performed by setting the `com.sun.jndi.dsmlv2.authzid` property
 - DS 5.2 expects the value to be a DN

Summary

- JNDI is an ideal API for directory applications
 - Stable: core component of J2SE
 - Same API for different protocols (e.g., LDAP and DSML)
 - Rich feature set
 - Highly extensible (SPI)
 - Actively supported
 - Tracks new and evolving standards

Further Information on JNDI

- BOF-1933, Java Naming and Directory Interface (JNDI) API
 - <http://java.sun.com/jndi/articles.html>
- JNDI Home Page
 - <http://java.sun.com/jndi/>
- The JNDI Tutorial
 - <http://java.sun.com/jndi/tutorial/>
- JNDI-INTEREST mailing list & archive
 - <http://java.sun.com/jndi/staytouch.html>

Further Information on DS 5.2

- Sun ONE Directory Server Home Page
 - http://www.sun.com/software/products/directory_srvr/home_directory.html

Q&A

JAVA™



JavaOneSM

Sun's 2003 Worldwide Java Developer Conference[®]



java.sun.com/javaone/sf