

JNDI : Java et les annuaires

Table des matières

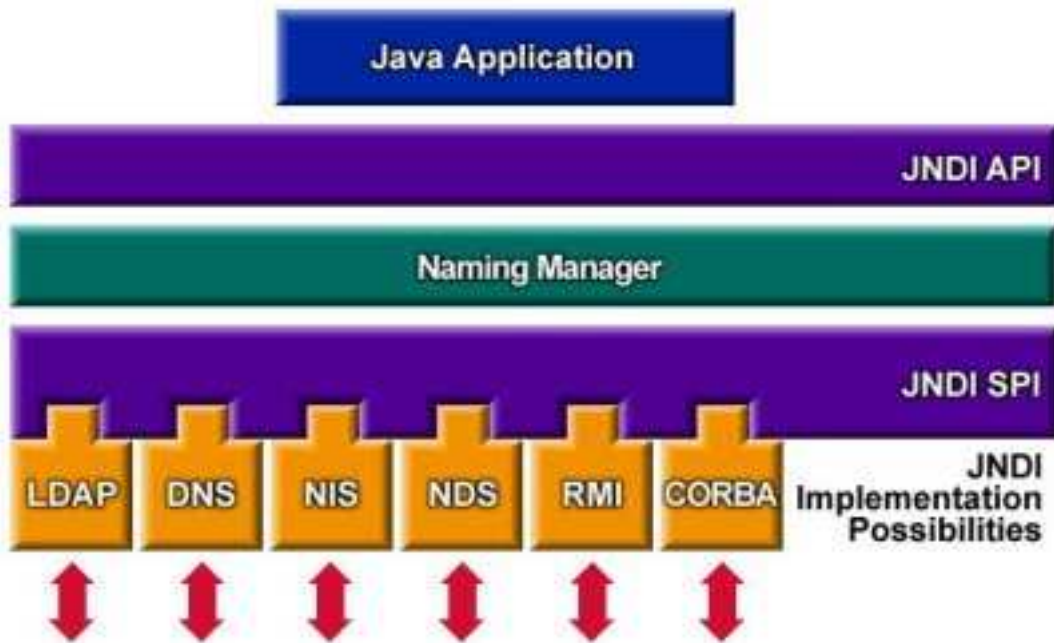
1	Présentation de JNDI	1
2	Architecture JNDI	2
3	Le binding (association)	2
4	Context	3
5	Directory (répertoire)	3
6	Directory (répertoire) (2)	3
7	Les packages JNDI	4
7.1	Package javax.naming	5
7.2	Package javax.naming (2)	5
8	Rechercher un nom dans un annuaire	5
9	Lister les noms d'un annuaire	6
10	Rechercher les associations	6
11	Package javax.naming.directory	7
12	Package javax.naming.directory (2)	7
13	Lister les attributs d'une entrée	7
14	Rechercher des entrées	9

1 Présentation de JNDI

- JNDI = Java Naming and Directory Interface
- C'est une *interface unifiée* vers des annuaires d'entreprise (correspondance nom --> objet)
 - NIS et NIS+
 - DNS
 - NetInfo
 - NDA (Novell Directory Service)
 - Système de fichiers
 - Annuaire Corba et RMI
 - etc.

2 Architecture JNDI

- L'API JNDI est composé de classes et surtout d'interfaces.
- Les pilotes assurent l'interface vers les annuaires.
- Les pilotes DNS, LDAP, RMI et CORBA sont inclus d'office dans la JRE.
- Les autres pilotes doivent être récupérés et ajoutés.



3 Le binding (association)

- C'est une association entre un nom et un objet.
- Dans le NIS un nom d'utilisateur est associé à ses entrées dans la base NIS :

```

"massat"      NIS      |UID=510
               |GID=500
               |shell=/bin/bash
               |home=/home/massat
               |...
    
```

- Le *binding* est aussi une association entre un nom et une référence qui pointe sur l'objet associé à ce nom.

Un exemple dans RMI :

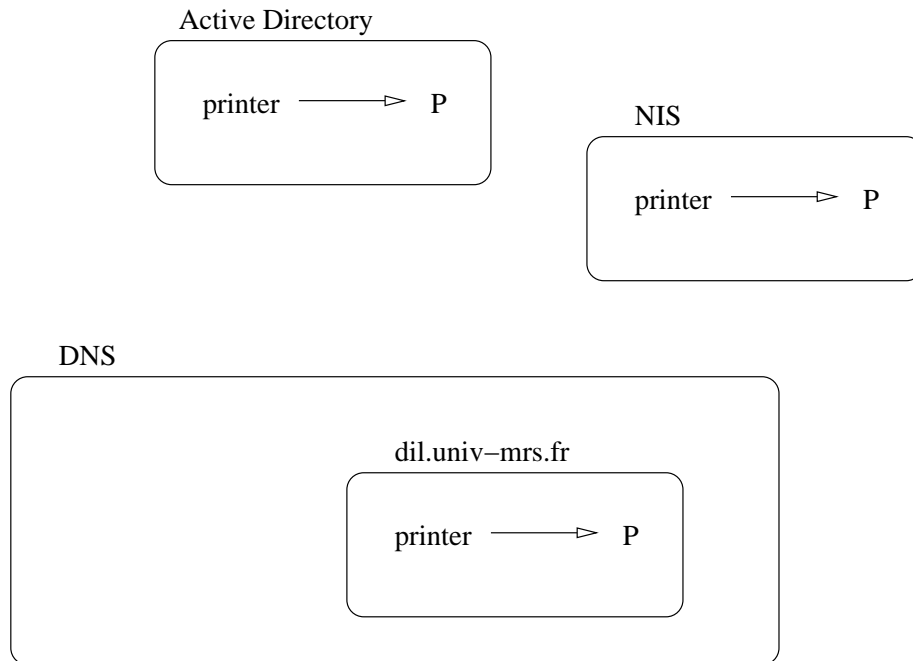
```

"1234AZ13"  Registre RMI      Protocole RMI  Serveur
            ----->  URL RMI  ----->      de
                                                    voiture  --+
                                                    |
                                                    |
            objet voiture <-----+
    
```

L'annuaire se comporte comme un serveur d'objets.

4 Context

- Un *contexte* est un ensemble d'associations entre des noms et des objets.
- Un même nom peut être associé à différents objets dans différents contextes. On parle dans ce cas de noms *locaux*.



associations du nom local « printer » dans différents contextes.

5 Directory (répertoire)

- Dans un répertoire les noms sont associés à des affectations de valeurs à des attributs :

```
nom -----> attribut1 := (valeur1,...,valeurN)
              attribut2 := (valeur1,...,valeurN)
              ...
```

Dans le DNS :

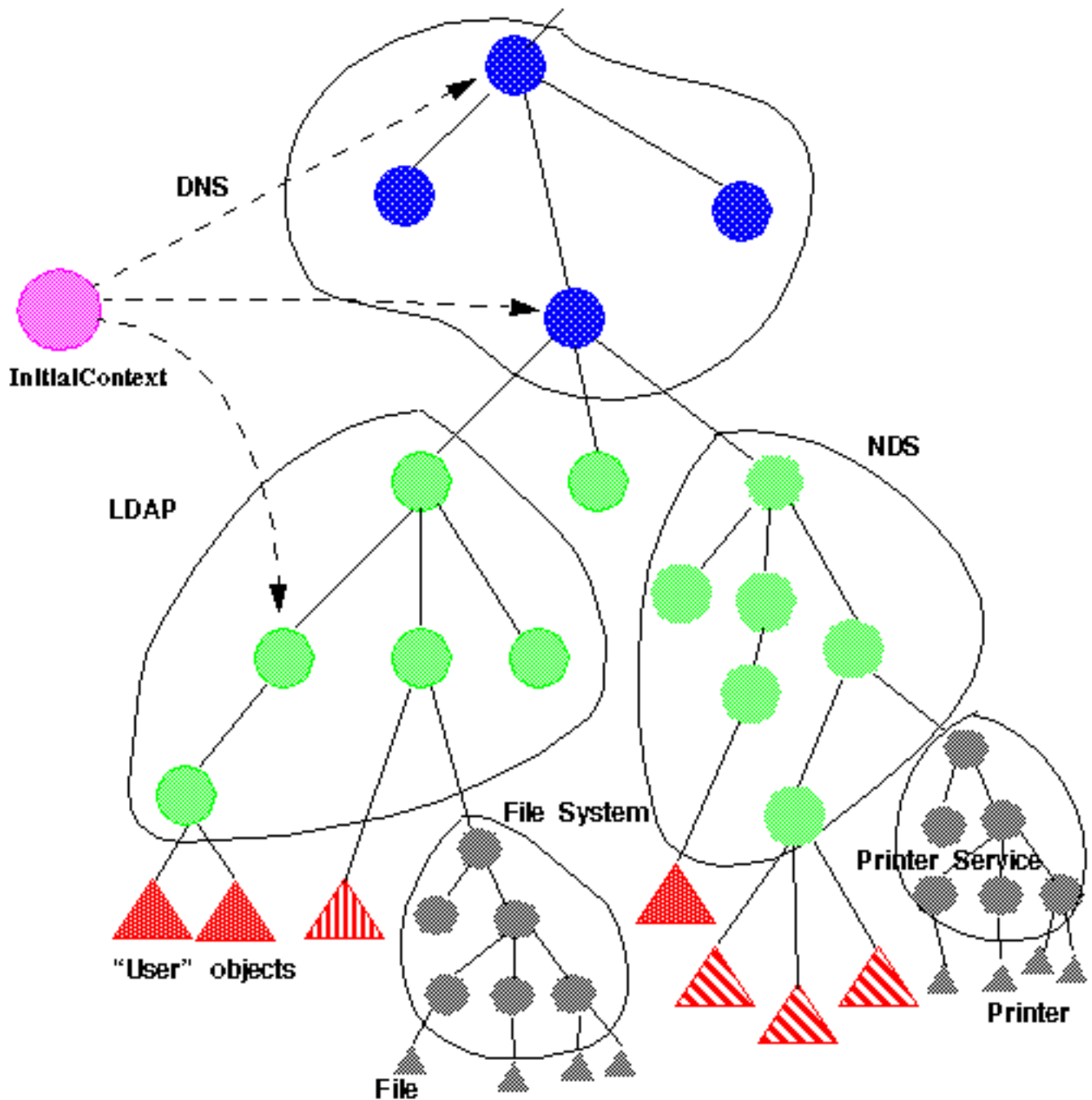
```
dil.univ-mrs.fr -----> SOA: saphir.lidil.univ-mrs.fr
                        dnsmaster.lidil.univ-mrs.fr
                        3171101 21600 7200 ...
                        NS: saphir.lidil.univ-mrs.fr
                           riluminy.univ-mrs.fr
                        MX: sol.dil.univ-mrs.fr
```

contenu du SOA (Start Of Authority) : serveur primaire, mail du responsable, numéro de série, durée de vie, etc.

6 Directory (répertoire) (2)

- Dans un répertoire il est possible d'effectuer des recherches sur la valeur des attributs.
- Les répertoires se caractérisent par une structure hiérarchique.

- Chaque noeud est un contexte,
- Les noms locaux sont interprétés comme des chemins partants d'un noeud vers un autre noeud.



7 Les packages JNDI

- Il existe cinq packages JNDI :

javax.naming Manipulation des associations.

javax.naming.directory Manipulation des répertoires.

javax.naming.event Traitement des évènements en provenance de l'annuaire.

javax.naming.ldap API spécialisée pour les annuaires compatibles LDAP.

javax.naming.spi API de programmation des pilotes JNDI.

7.1 Package javax.naming

- interface Context :
 - Object lookup(Name name)
 - NamingEnumeration<NameClassPair> list(Name name)
 - NamingEnumeration<Binding> listBindings(Name name)
 - bind(Name s1, Object o)
 - rebind(Name s1, Object o)
 - rename(Name s1, Name s2)
 - unbind(Name name)

on peut remplacer Name par String.

- interface Name : gestion d'un nom composé comme

`cn=massat,dc=dil,dc=com`

7.2 Package javax.naming (2)

- classe NameClassPair :
 - String getClassName()
 - String getName()
- classe Binding : étends la classe NameClassPair.
 - Object getObject()
- classe InitialContext : implante l'interface Context.
 - InitialContext(Hashtable<?,?> environment)

8 Rechercher un nom dans un annuaire

Recherche d'un objet dans un système de fichiers :

```

import java.util.Hashtable;
import javax.naming.Context;
import javax.naming.Binding;
import javax.naming.NameClassPair;
import javax.naming.NamingEnumeration;
import javax.naming.InitialContext;

public class ExempleJNDI {
public static void main(String[] args) {
    try {
        Hashtable<String,String> env = new Hashtable<String,String>();
        env.put("java.naming.factory.initial",
            "com.sun.jndi.fscontext.RefFSContextFactory");
        env.put("java.naming.provider.url", "file:///home");

        Context ictx = new InitialContext(env);
        Object o = ictx.lookup(args[0]);
        System.out.print(args[0] + " est ");
        if (o instanceof Context) System.out.println(" un noeud");
        else System.out.println(" une feuille");
    }
    catch (javax.naming.NamingException e) { System.err.println(e); }
}
}

```

9 Lister les noms d'un annuaire

Ce code liste les noms à partir d'un annuaire LDAP :

```

public static void main(String[] args) {
    try {
        Hashtable<String,String> env = new Hashtable<String,String>();
        env.put("java.naming.factory.initial",
            "com.sun.jndi.ldap.LdapCtxFactory");
        env.put("java.naming.provider.url",
            "ldap://localhost/dc=my-domain,dc=com");

        Context ictx = new InitialContext(env);
        NamingEnumeration<NameClassPair> e = ictx.list("cn=userA");
        while (e.hasMore()) {
            System.out.println("name: " + e.next().getName());
        }
    }
    catch (javax.naming.NamingException e) {
        System.err.println("Exception " + e);
    }
}
}

```

Les noms sont relatifs au contexte de base de la recherche (ici cn=userA,dc=my-domain,dc=com).

10 Rechercher les associations

Ce code liste les machines connues dans le domaine DNS args[0] + "." + "univ-mrs.fr" :

```

public static void main(String[] args) {
    try {
        Hashtable<String,String> env = new Hashtable<String,String>();
        env.put("java.naming.factory.initial",
            "com.sun.jndi.dns.DnsContextFactory");
        env.put("java.naming.provider.url",
            "dns://saphir.lidil.univ-mrs.fr/univ-mrs.fr");

        Context ictx = new InitialContext(env);
        NamingEnumeration<Binding> e = ictx.listBindings(args[0]);
        while (e.hasMore()) {
            Binding b = e.next();
            System.out.println("name: " + b.getName());
            System.out.println("object: " + b.getObject());
        }
    }
    catch (javax.naming.NamingException e) {
        System.err.println("Exception " + e);
    }
}

```

11 Package javax.naming.directory

- étend les classes et interfaces de javax.naming.
- interface DirContext étend Context :
 - Attributes getAttributes(Name name)
 - Attributes getAttributes(Name name, String[] attrIds)
 - NamingEnumeration<SearchResult> search(
 - Name name,
 - Attributes matchingAttributes
)
 - NamingEnumeration<SearchResult> search(
 - Name name,
 - Attributes matchingAttributes,
 - String[] attributesToReturn
)

on peut remplacer Name par String.

12 Package javax.naming.directory (2)

- interface Attributes :
 - Attribute get(String attrID)
 - NamingEnumeration<? extends Attribute> getAll()
- interface Attribute :
 - Object get()
 - NamingEnumeration<?> getAll()

13 Lister les attributs d'une entrée

Commençons par les déclarations de classes :

```

import java.util.*;

import javax.naming.Context;
import javax.naming.Binding;
import javax.naming.NamingEnumeration;
import javax.naming.InitialContext;
import javax.naming.NameClassPair;

import javax.naming.directory.Attributes;
import javax.naming.directory.Attribute;
import javax.naming.directory.DirContext;
import javax.naming.directory.InitialDirContext;
import javax.naming.directory.SearchResult;
import javax.naming.directory.SearchControls;

```

Continuons par la récupération du contexte :

```

public class ExempleDirectory {

public static void main(String[] Args)
{
    try {
        Hashtable<String,String> env =
            new Hashtable<String,String>();
        env.put("java.naming.factory.initial",
            "com.sun.jndi.dns.DnsContextFactory");
        env.put("java.naming.provider.url",
            "dns://saphir.lidil.univ-mrs.fr/univ-mrs.fr");

        DirContext ictx = new InitialDirContext(env);
        Attributes attrs = ictx.getAttributes("dil",
            new String[] {"MX", "SOA"});

        System.out.println("Attributs MX et SOA de dil :");
        listerAttributs(attrs);
    }
    catch (javax.naming.NamingException e) { ... }
}

```

Terminons par l'affichage des attributs et de leur valeurs :

```

public static void listerAttributs(Attributes atts) {
    try {
        for(NamingEnumeration e = atts.getAll(); e.hasMore();)
        {
            Attribute a = (Attribute) e.next();
            System.out.println(a.getID() + ":");
            Enumeration values = a.getAll();
            while (values.hasMoreElements()) {
                System.out.println("--> " +
                    values.nextElement().toString());
            }
        }
    }
    catch (javax.naming.NamingException e) { ... }
}

```

L'exécution nous donne

```
Attributs MX et SOA de dil :
SOA:
--> saphir.lidil.univ-mrs.fr. dnsmaster.lidil...
MX:
--> 10 sol.dil.univ-mrs.fr.
```

14 Rechercher des entrées

Recherche en profondeur avec filtre dans un annuaire LDAP :

```
public static void main(String[] args) {
    try {
        Hashtable<String,String> env = new Hashtable<String,String>();
        env.put("java.naming.factory.initial",
            "com.sun.jndi.ldap.LdapCtxFactory");
        env.put("java.naming.provider.url",
            "ldap://localhost/dc=my-domain,dc=com");

        // recherche en profondeur
        SearchControls controle = new SearchControls();
        controle.setSearchScope(SearchControls.SUBTREE_SCOPE);

        String critere = "(|(sn=premier)(sn=deux*))";
        DirContext ictx = new InitialDirContext(env);
        NamingEnumeration<SearchResult> e =
            ictx.search("cn=userA", critere, controle);
        while (e.hasMore()) {
            SearchResult r = e.next();
            System.out.println("name: " + r.getName());
            System.out.println("object: " + r.getClassName());
            listerAttributs(r.getAttributes());
        }
    }
    catch (javax.naming.NamingException e) { ... }
}
```