Configuration du fichier Domain.xml dans Slide

Configuration du fichier Domain.xml dans Slide

**
Auteur : Thomas Bellembois (Université de Rennes 1) (http://http://)

Introduction

Le fichier **Domain.xml** permet entre autre de configurer (en partie) le serveur Slide. Il permet de créer et configurer des **stores** et de positionner les **droits d'accès** aux ressources par défaut. Ce document a pour but de configurer rapidement les 3 types de stores principaux de Slide ainsi que de positionner les droits par défaut. **Attention** Ce document n'a pas pour objectif d'expliciter les stores Slide en détail. Se référer à la documentation Slide pour celà. Ce document reste tout de même le document de référence pour la configuration du fichier Domain.xml pour le serveur WebDAV ESUP.

Création et configuration des stores Slide

Les stores peuvent être définis comme les moyens de stockage utilisés pour stocker les informations/ressources (i.e. fichiers, droits d'accès, métadonnées...) sur le serveur Slide. On peut définir plusieurs méthodes de stockages pour plusieurs URL différentes. Ex : * stockage des ressources de l'URL /users dans un annuaire LDAP

- stockage des ressources de l'URL /roles dans une base de données MySQL
- stockage des autres ressources en XML

On distingue 3 types principaux de store, le store XML, le store MySQL et le store LDAP. D'autres stores peuvent être développés en fonction des besoins. L'utilisation des stores s'effectue de la manière suivante : * définition du store entre les balises <definition> et </definition> dans des balises <store> </store>

- association du store à une URL du serveur entre les balises <definition> et </definition> dans des balises <scope> </scope>

Store XML

Le store XML (TxXMLFileDescriptorsStore) permet de stocker les informations/ressources dans des fichiers XML. C'est le store par défaut de Slide.

Définition du store dans le fichier Domain.xml

```
<definition>
<store name="xmlstore"> <!-- Nom du store -->
    <nodestore classname="org.apache.slide.store.txfile.TxXMLFileDescriptorsStore">
        <parameter name="rootpath">store/metadata</parameter> <!-- chemin physique de stockage des fichiers XML
-->
        <parameter name="workpath">work/metadata</parameter> <!-- chemin physique de stockage des fichiers XML
-->
    </nodestore>
    <securitystore>
        <reference store="nodestore"/>
    </securitystore>
    <lockstore>
        <reference store="nodestore"/>
    </lockstore>
    <revisiondescriptorsstore>
        <reference store="nodestoSection Bre"/>
    </revisiondescriptorsstore>
    <revisiondescriptorstore>
        <reference store="nodestore"/>
    </revisiondescriptorstore>
    <contentstore classname="org.apache.slide.store.txfile.TxFileContentStore">
        <parameter name="rootpath">store/content</parameter> <!-- chemin physique de stockage des fichiers XML
-->
        <parameter name="workpath">work/content</parameter> <!-- chemin physique de stockage des fichiers XML --
>
    </contentstore>
</store>
...
</definition>
```

Association du store à une URL

```
<definition>
...
<scope match="/" store="xmlstore"/> <!-- tout ce qui se trouve sous "/" sera associé au store main -->
</definition
```

Store JNDIPrincipalStore

Le store LDAP (JNDIPrincipalStore) permet de stocker les informations/ressources dans un annuaire LDAP. La configuration de l'annuaire n'est pas explicitée ici. Elle disponible sur le site de Jakarta Slide.

Définition du store dans le fichier Domain.xml

```
<store name="ldapstore">
    <nodestore classname="org.apache.slide.store.txjndi.JNDIPrincipalStore">
    <!-- Ce paramètre nous spécifie combien de fois, en secondes, le cache de rafraîchissement des threads doit
vérifier si les URIs présents dans le cache ont besoin d'être rafraîchis. La valeur par défaut est "15". -->
        <parameter name="cache.refresh.checkrate">15</parameter>
    <!-- Ce paramètre nous spécifie la fréquence, en secondes, à laquelle les URIs marqués à rafraîchir doivent
être rafraîchis. Cette valeur doit être inférieure à celle des paramètres TimeToLive et TimeToIdle du EHCache,
afin que les items n'expirent jamais. La valeur par défaut est "800" -->
        <parameter name="cache.refresh.rate">600</parameter>

    <!-- Ce paramètre nous spécifie le temps maximum, en millisecondes, que doivent prendre les méthodes de
recherches avant que les URIs qu'elles recherchent ne soient programmés pour un rafraîchissement. En modifiant
ce paramètre, on peut réduire la durée de vie des URI qui sont rarement accèdés et ainsi éviter qu'ils restent
dans le cache inutilement. -->
        <parameter name="cache.refresh.threshold">500</parameter>
            <!-- baseDN -->
        <parameter name="jndi.container">ou=XXXX,dc=XXXX,dc=XXXX</parameter>
            <!-- attribut unique représentant l'utilisateur/roles dans l'annuaire -->
        <parameter name="jndi.attributes.rdn">uid</parameter>
            <!-- filtre LDAP -->
        <parameter name="jndi.search.filter">(objectClass=*)</parameter>
            <!-- scope de la recherche --><!-- SUBTREE_SCOPE | ONELEVEL_SCOPE | OBJECT_SCOPE-->
        <parameter name="jndi.search.scope">SUBTREE_SCOPE</parameter>
            <!-- liste d'attributs à récupérer de l'anuaire -->
        <parameter name="jndi.search.attributes">uid,phoneNumber</parameter>
            <!-- URL de l'annuaire LDAP-->
        <parameter name="java.naming.provider.url">ldap://ldapURL:389</parameter>
        <parameter name="java.naming.factory.initial">com.sun.jndi.ldap.LdapCtxFactory</parameter>
            <!-- Paramètres d'authentification -->
        <parameter name="java.naming.security.principal"></parameter>
        <parameter name="java.naming.security.authentication">simple</parameter>
        <parameter name="java.naming.security.credentials"></parameter>
    </nodestore>
    <securitystore>
    <reference store="nodestore"/>
    </securitystore>
    <lockstore>
    <reference store="nodestore"/>
    </lockstore>
    <revisiondescriptorsstore>
    <reference store="nodestore"/>
    </revisiondescriptorsstore>
    <revisiondescriptorstore>
    <reference store="nodestore"/>
    </revisiondescriptorstore>
    <contentstore>
    <reference store="nodestore"/>
    </contentstore>
</store>
```

Association du store à une URL

```
<definition>
...
<scope match="/users" store="ldapstore"/> <!-- tout ce qui se trouve sous "/users" sera associé au store
ldapstore -->
</definition>
```

⚠ Les store LDAP peut être utilisé pour la gestion des utilisateurs et roles mais en aucun cas pour le stockage de fichiers.

Store MySqlRDBMSAdapter

Le store MySql (MySqlRDBMSAdapter) permet de stocker les informations/ressources dans une base MySql. La configuration de la base n'est pas explicitée ici. Elle disponible sur le site de Jakarta Slide.

⚠ Ce store n'est pas utilisé dans le serveur WebDAV ESUP.

Définition du store dans le fichier Domain.xml

```
<store name="mysqlstore">
    <nodestore classname="org.apache.slide.store.impl.rdbms.JDBCStore">
        <parameter name="adapter">org.apache.slide.store.impl.rdbms.MySqlRDBMSAdapter</parameter>
        <parameter name="driver">com.mysql.jdbc.Driver</parameter>
            <!-- URL de la base de données -->
        <parameter name="url">jdbc:mysql://####/slidebase</parameter>
            <!-- Paramètres d'authentification  -->
        <parameter name="user">connectionLogin</parameter>
        <parameter name="password">connectionPassword</parameter>
        <parameter name="dbcpPooling">true</parameter>
        <parameter name="maxPooledConnections">10</parameter>
        <parameter name="isolation">SERIALIZABLE</parameter>
        <parameter name="compress">false</parameter>
    </nodestore>
    <contentstore>
        <reference store="nodestore" />
    </contentstore>
    <securitystore>
        <reference store="nodestore" />
    </securitystore>
    <lockstore>
        <reference store="nodestore" />
    </lockstore>
    <revisiondescriptorsstore>
        <reference store="nodestore" />
    </revisiondescriptorsstore>
    <revisiondescriptorstore>
        <reference store="nodestore" />
    </revisiondescriptorstore>
</store>
```

Association du store à une URL

```
<definition>
...
<scope match="/roles" store="mysqlstore"/> <!-- tout ce qui se trouve sous "/roles" sera associé au store
mysqlstore -->
</definition>
```

Store HashStore

Le store HashStore transforme les URL du type /home/john en /home/j/jo/john.

Définition du store dans le fichier Domain.xml

```
<store name="hashstore">
        <!-- Branche des homedirs -->
        <parameter name="store-root">files/homedirs</parameter>
        <parameter name="hash-method">hash</parameter>
    <nodestore classname="org.esupportail.webdavserver.store.txhashfile.TxXMLFileDescriptorsStore">
        <!-- Chemin physique du répertoire de stockage des méta-données -->
        <parameter name="rootpath">/data/Partages-test_depot/metadata/store</parameter>
        <!-- Chemin physique du répertoire temporaire des traitements sur les méta-données -->
        <parameter name="workpath">/data/Partages-test_depot/metadata/work</parameter>
    </nodestore>
    <securitystore>
        <reference store="nodestore"/>
    </securitystore>
    <lockstore>
        <reference store="nodestore"/>
    </lockstore>
    <revisiondescriptorsstore>
        <reference store="nodestore"/>
    </revisiondescriptorsstore>
    <revisiondescriptorstore>
        <reference store="nodestore"/>
    </revisiondescriptorstore>
    <contentstore classname="org.esupportail.webdavserver.store.txhashfile.TxFileContentStore">
        <!-- Chemin physique du répertoire de stockage des ressources -->
        <parameter name="rootpath">/data/Partages-test_depot/content/store</parameter>
        <!-- Chemin physique du répertoire temporaire des traitements sur les ressources -->
        <parameter name="workpath">/data/Partages-test_depot/content/work</parameter>
    </contentstore>
</store>
```

Association du store à une URL

```
<definition>
...
<scope match="/files/homedirs" store="hashstore"/>
</definition>
```

Store UPortal

Le store UPortal fait appel à un Web Service afin de récupérer des groupes du type local.0/local101/PAGS_COMPPERS/PAGS_PERS_UR1. Les local.* sont générés automatiquement lors de l'initialisation de base de donnée du portail. Les PAGS eux, sont des groupes définis par l'administrateur. Chaque PAGS regroupe un ensemble de personnes.

Définition du store dans le fichier Domain.xml

```
<store name="uPortalRolesStore">
    <nodestore classname="org.esupportail.webdavserver.store.uPortal.UPortalRolesStoreV2">
        <!-- URL du Web service -->
        <parameter name="uportal.webservice.url">http://portail.univ-rennes1.fr/services/PortalGroups<
/parameter>
        <!-- Paramètre spécifiant si le cache doit être désactivé -->
        <parameter name="uportal.store.cache.disableCache">false</parameter>
        <!-- Durée en minutes après laquelle les Threads en attente doivent être supprimés -->
        <parameter name="uportal.store.cache.cleaningThreadSleepingTimeInMinutes">1440</parameter> <!-- min 1mn
-->
        <!-- Durée en minutes de la durée de vie des informations de l'utilisateur dans le cache -->
        <parameter name="uportal.store.cache.cachedUserInformationMinutesTolive">1440</parameter> <!-- min 1mn
-->
        <!-- Durée en minutes de la durée de vie des ObjectNode dans le cache -->
        <parameter name="uportal.store.cache.cachedObjectNodeMinutesTolive">1440</parameter> <!-- min 1mn -->
        <!-- ?????????????? Déchargement du cache ??????????????????-->
        <parameter name="uportal.store.cache.dumpCache">dumpCache</parameter>
        <!-- ?????????????? Néttoyage du cache ??????????????????? -->
        <parameter name="uportal.store.cache.cleanCache">cleanCache</parameter>
        <!-- Chemin du fichier de mapping des groupes (uPortalStoreGroupMapping.xml) -->
        <parameter name="uportal.store.groupMappingFilePath">/data/webapps/slide/uPortalStoreGroupMapping.xml<
/parameter>
    </nodestore>
    <securitystore>
        <reference store="nodestore"/>
    </securitystore>
    <lockstore>
        <reference store="nodestore"/>
    </lockstore>
    <revisiondescriptorsstore>
        <reference store="nodestore"/>
    </revisiondescriptorsstore>
    <revisiondescriptorstore>
        <reference store="nodestore"/>
    </revisiondescriptorstore>
    <contentstore>
        <reference store="nodestore"/>
    </contentstore>
</store>
```

⚠ A partir de la version 5.2 la classe utilisée, par défaut, pour le uPortalRolesStore est org.esupportail.webdavserver.store.uPortal.UPortalRolesStoreV2. Cependant, la classe org.esupportail.webdavserver.store.uPortal.UPortalRolesStore est toujours disponible.

Association du store à une URL

```
<definition>
...
<scope match="/roles/uPortal" store="uPortalRolesStore"/>
</definition>
```

Store Shib

Le store Shib permet d'évaluer des règles sur des attributs shibboleth.

Définition du store dans le fichier Domain.xml

```
<store name="ShibRolesStore">
    <nodestore classname="org.esupportail.webdavserver.store.shib.ShibRolesStore">
        <!-- Chemin physique du répertoire de stockage des méta-données -->
        <parameter name="rootpath">/data/Partages-test_depot2/metadata/store</parameter>
        <!-- Chemin physique du répertoire temporaire des traitements sur les méta-données -->
        <parameter name="workpath">/data/Partages-test_depot2/metadata/work</parameter>
    </nodestore>
    <securitystore>
        <reference store="nodestore"/>
    </securitystore>
    <lockstore>
        <reference store="nodestore"/>
    </lockstore>
    <revisiondescriptorsstore>
        <reference store="nodestore"/>
    </revisiondescriptorsstore>
    <revisiondescriptorstore>
        <reference store="nodestore"/>
    </revisiondescriptorstore>
    <contentstore classname="org.apache.slide.store.txfile.TxFileContentStore">
        <!-- Chemin physique du répertoire de stockage des ressources -->
        <parameter name="rootpath">/data/Partages-test_depot/shib/content/store</parameter>
        <!-- Chemin physique du répertoire temporaire des traitements sur les ressources -->
        <parameter name="workpath">/data/Partages-test_depot/shib/content/work</parameter>
        <parameter name="defer-saving">true</parameter>
        <parameter name="timeout">120</parameter>
    </contentstore>
</store>
```

Association du store à une URL

```
<definition>
...
<scope match="/roles/shib" store="ShibRolesStore"/>
</definition>
```

Store UserStoreShib

Le store UserStoreShib est une extension du store JNDIPrincipalStore. Ce store permet de renvoyer des objets du type /users/toto@univ-rennes2.fr.

Définition du store dans le fichier Domain.xml

```
<store name="users">
    <nodestore classname="org.esupportail.webdavserver.store.shib.UserStoreShib">
        <!-- See javadoc for JNDIPrincipalStore for description of parameters -->
        <parameter name="cache.refresh.checkrate">15</parameter>
        <parameter name="cache.refresh.rate">600</parameter>
        <parameter name="cache.refresh.threshold">500</parameter>
        <!-- CONFIGURE THE FOLLOWING PARAMETER(S) -->
        <parameter name="jndi.container">ou=people,dc=univ-rennes1,dc=fr</parameter>  <!-- LDAP baseDN-->
        <parameter name="jndi.attributes.rdn">uid</parameter>   <!-- unique attribute representing the user in
the LDAP directory -->
        <parameter name="jndi.search.filter">(departmentNumber=957*)</parameter>
        <parameter name="jndi.search.scope">SUBTREE_SCOPE</parameter>  <!-- LDAP scope : SUBTREE_SCOPE |
ONELEVEL_SCOPE | OBJECT_SCOPE-->
        <parameter name="jndi.search.attributes">uid,displayName</parameter>   <!-- attribute to retrieve in
the LDAP directory -->
        <parameter name="java.naming.provider.url">ldap://ldapglobal2.univ-rennes1.fr:389</parameter>  <!--
LDAP URL -->
        <parameter name="java.naming.factory.initial">com.sun.jndi.ldap.LdapCtxFactory</parameter>
        <parameter name="java.naming.security.principal"></parameter>  <!-- LDAP principal - if needed to bind
-->
        <parameter name="java.naming.security.authentication">simple</parameter>
        <parameter name="java.naming.security.credentials"></parameter>  <!-- LDAP credential - if needed to
bind -->
        <!-- /CONFIGURE -->
    </nodestore>
    <securitystore>
        <reference store="nodestore"/>
    </securitystore>
    <lockstore>
        <reference store="nodestore"/>
    </lockstore>
    <revisiondescriptorsstore>
        <reference store="nodestore"/>
    </revisiondescriptorsstore>
    <revisiondescriptorstore>
        <reference store="nodestore"/>
    </revisiondescriptorstore>
    <contentstore>
        <reference store="nodestore"/>
    </contentstore>
</store>
```

Positionnement des droits d'accès par défaut

On peut positionner des droits par defaut sur des ressources sur serveur. Les droits sont positionné pour des utilisateurs ou groupes d'utilisateurs. De plus, le Domain.xml possède une partie définie entre les balises et , qui au lancement du serveur, permet de créer l'arborescence de base et positionne également les droits spécifiés sur chaque branche.

Structure de l'entrée dans le fichier Domain.xml

```
<objectnode classname="org.apache.slide.structure.SubjectNode" uri="/">
    <!--
    ========================================
    1. A l'URL "/" on associe des permissions
    ========================================
    -->
    <permission action="/actions/action1" subject="XXXX" inheritable="true|false" negative="true|false"/>
    <permission action="/actions/action2" subject="XXXX" inheritable="true|false" negative="true|false"/>
    ...

    <!--
    ========================================
    2. On défini un sous répertoire "subDir1"
    ========================================
    -->
    <objectnode classname="org.apache.slide.structure.SubjectNode" uri="/subDir1">
        <!--
        ============================================================================
        1. On défini les éléments par défaut de /subDir1 ainsi que les permissions
        ============================================================================
        -->
        <revision>
            <property name="XXXX"><![CDATA[<D:href xmlns:D='DAV:'>XXXX</D:href>]]></property>
        </revision>
        <permission action="/actions/action3" subject="XXXX" inheritable="true|false" negative="true|false"/>
        <permission action="/actions/action3" subject="XXXX" inheritable="true|false" negative="true|false"/>
        ...

        <!--
        ========================================================
        2. On défini un sous sous répertoire "subsubDir1" etc...
        ========================================================
        -->
         <objectnode classname="org.apache.slide.structure.SubjectNode" uri="/subsubDir1">
         ... et ainsi de suite ...
         </objectnode>
    </objectnode>
    ....
</objectnode>
```

A une URL (<objectnode>) sont associés :

- des permissions (<permission>)
- d'autres URL (sous répertoires <objectnode>)
- éventuellement des éléments (users, roles ... <revision>)

Exemple

Un exemple de Domain.xml est disponible sur le SVN du projet Esup-WebDAV. L'url est la suivante : http://subversion.cru.fr/esup-webdav-srv