Installation esup-portlet-mondossierweb

Pré-requis

Pour pouvoir faire fonctionner esup-mondossierweb, il est nécessaire d'avoir installé au préalable :

- **Un Tomcat avec uPortal et esup-portal-ws :** http://sourcesup.cru.fr/projects/esup-portal-ws/

  > ⚠ Attention esup-portal-ws est maintenant intégré dans les dernières versions du package ESUP-Portail. Les instructions données ne sont utiles que pour des anciennes versions du package ESUP-Portail.

  Télécharger uportal-ws-server.jar et le copier dans votre hiérarchie uPortal (/uPortal/lib).
  Activer Axis en modifiant le fichier/uPortal/webpages/WEB-INF/server-config.wsdd, en y ajoutant le code suivant:

```
<service name="UportalService" provider="java:RPC">
<parameter name="allowedMethods" value="getUser,getUserAttributes,getGroupById,getGroupByName,getSubGroups,
getSubGroupsById,getSubGroupsByName,
getRootGroup,getGroupHierarchyById,getGroupHierarchyByName,getGroupHierarchy,getUserGroups,isUserMemberOfGroup"
/>
<parameter name="className" value="org.esupportail.portal.ws.server.UportalService"/>
<parameter name="scope" value="application"/>
</service>
```

* **Le Web Service de l'AMUE :**
déployer apows.war dans Tomcat (qui sera ensuite présent dans webapps/apows).
renseigner ses paramètres de connexion (jdbcUrl, user et password) à Apogee dans 'persistance-SpringContext.xml' présent dans le répertoire 'apows/WEB-INF/classes'. Attention, pour que la mise à jour de l'adresse étudiant fonctionne, le compte utilisé doit avoir les droits d'update sur la base. Cependant, cette mise à jour peut être désactivée dans le fichier 'properties' de l'application.
Penser à activer le wsdl dans WEB-INF/server-config.wsdd. Pour cela décommenter les lignes utiles à la fin du fichier. Voici le résultat:

```
<parameter name="useDefaultQueryStrings" value="true" >
                        <parameter name="qs:wsdl" value="org.apache.axis.transport.http.QSWSDLHandler"/>
                        <parameter name="qs.list" value="org.apache.axis.transport.http.QSListHandler"/>
                        <parameter name="qs.method" value="org.apache.axis.transport.http.QSMethodHandler"/>
                </parameter>
```

Si besoin est (apows ne se trouvant pas dans le répertoire webapp de Tomcat) ajouter le contexte du WS dans le fichier server.xml du répertoire 'conf' de Tomcat, ex :

```
<Context path="/apows" docBase="C:/esupdev/esupdev-2.5-esup-2.1.01/uPortal-quick-start/webapps/apows"
crossContext="true" reloadable="true"/>
```

Installation

- Récupérer la distribution de esup-mondossierweb
- Configurer les bases indispensables au fonctionnement l'application. Suivre les étapes suivantes :
  Pour un déploiement servlet, créer un fichier 'build-servlet.properties' sur la base du fichier d'exemple 'build-servlet-example.properties'. Ce fichier décrit votre déploiement servlet.
  Pour un déploiement portlet, créer un fichier 'build-portlet.properties' sur la base du fichier d'exemple 'build-portlet-example.properties'. Ce fichier décrit votre déploiement portlet.

  > Remarque :
  > Pour l'utilisation de la target 'start', le fichier 'esup-portail.keystore' est disponible dans le répertoire utils/cas du projet esup-mondossierweb

  > **Attention :**
  > **Pour un déploiement portlet au sein d'Uportal, esup-mondossierweb doit être déployé au même niveau qu' UPortal.**

  Renseigner les fichiers (du répertoire 'properties') de configuration : '**application.properties**' et '**log4j.properties**' (dans le repertoire logging) sur la base des fichiers **-example.properties.** Dans le répertoire 'monDossierWeb' créer le fichier **monDossierWeb.xml** sur la base du fichier monDossierWeb-exemple.xml et le renseigner. Dans le répertoire properties/dao/ibatis/mapping créer le fichier **Mellogin.xml** sur la base du fichier Mellogin_example.xml et le renseigner (si possible) en indiquant les requêtes qui vous permettent de récupérer le codetu d'un étudiant à partir de son login et inversement dans Apogée (voir la partie 'Configuration avancée' ci-dessous). D'autres options sont configurables via les beans Spring : leur description se trouve à la fin de la rubrique 'Utilisation'.

Renseigner **configUrlServices.properties** du répertoire 'webapp/WEB-INF/classes' en donnant les urls d'accès au Web Service de l'Amue.
Pour un déploiement servlet, créer le web-servlet.xml sur la base de web-servlet-example.xml en indiquant les bonnes urls du serveur CAS : urls de logout, login, et serviceValidate.
Lancer la tache Ant '_toServlet' ou '_toPortlet' du build.xml suivant le type de déploiement désiré. Vous pouvez ensuite passer au déploiement.

Configuration avancée

- **Attention** : Avant le lancer l'application, il faut impérativement consulter et modifier si necessaire le fichier monDossierWeb.xml, qui contient un paramétrage de tout ce qui est attributs ldap, Apogée, affichage, édition de pdf etc.

COD_ETU, LOGIN et Beans interchangeables :

- Par ailleurs, certaines fonctionnalités sont prévues pour être modifiées suivant les besoins et la configuration rencontrés. Cela a été implémenté sous la forme de Beans Spring configurables (voir la partie Les beans interchangeables ici).
  - Exemple : la récupération du Cod_Etu de l'étudiant à partir de son login (étape effectuée à la connexion d'un étudiant sur la portlet).
    - A l'installation, l'application va rechercher le cod_etu dans le ldap (property '*attributLdapCodEtu*' dans monDossierWeb.xml) à partir du login. Si vous voulez plutôt passer par Apogée, vous devez renseigner le fichier Mellogin.xml en indiquant la requête qui vous permet de récupérer cette info dans Apogée puis remplacer **DaoCodeLoginEtudiantImplLdapBasic** par **DaoCodeLoginEtudiantImplApogeeBasic** dans application.properties. Si aucune des solution ne vous convient, vous avez la possibilité de créer votre propre bean.
    - Cette possibilité a été utilisée par Nancy2. En effet, nous essayons d'abord de récupérer l'information dans le ldap avant d'exécuter une requête dans Apogée.
      - Cette fonctionnalité est implémentée dans la classe : org.esupportail.mondossierweb.dao.**DaoCodeLoginEtudiantImplNancy2.java**. Cette classe, qui implémente l'interface **IDaoCodeLoginEtudiant**, essaye tout d'abord de récupérer la cod_etu dans le ldap (elle va chercher l'attribut "attributLdapEtudiant" configuré dans monDossierWeb.xml) et si cela échoue elle le récupère via une requête SQL.
      - la requête SQL appelée se trouve bien sur dans le fichier properties/dao/ibatis/mapping/Mellogin.xml que nous avons configuré pour nos besoins.
      - La classe **DaoCodeLoginEtudiantImplNancy2** est indiquée dans le fichier **application.properties**, à la place de **DaoCodeLoginEtudiantImplLdapBasic**.
    - Pour implémenter votre propre solution, vous avez plusieures possibilités:
      - 1 Vous pouvez effectuer une requête sur votre base de données qui vous retourne le codEtu en fonction du Login. Dans ce cas, vous n'avez qu'à indiquer la bonne requête dans Mellogin.xml comme indiqué dans la partie 'Installation'. Puis vous indiquez le bean **DaoCodeLoginEtudiantImplApogeeBasic** dans application.properties
      - 2 Votre configuration est plus complexe. Dans ce cas vous devez créer votre propre bean DaoCodeLoginEtudiantImplUniversité qui implémente l'interface IDaoCodeLoginEtudiant. Pour cela ajoutez votre classe dans la package **dao** (là où se trouve DaoCodeLoginEtudiantImplNancy2.java) et référencez là dans application.properties à la place de **DaoCodeLoginEtudiantImplBasic**. Si vous devez créer de nouvelles requêtes sur la base, vous pouvez compléter le fichier de config iBATIS existant (Mellogin.xml). Vous pouvez bien sur vous inspirer de la classe **DaoCodeLoginEtudiantImplNancy2** pour implémenter votre solution.

- Ce fonctionnement est le même pour la récupération du login à partir du CodEtu (utilisé pour créer les adresses mails à partir des CodEtu d'une liste d'étudiant dans la partie 'enseignant' de la portlet) excepté qu'il n'existe pas de classe équivalente pour Nancy2. En effet, nous utilisons la classe **Basic** et la requête qui va ien dans Mellogin.xml.

Rappel : La liste de tous les beans interchangeables se trouve ici

Implémentation des fonctionnalités de Rennes1 :

- Attention : fonctionnalité non disponible à partir de la version 2.2-5 !
- Depuis la version 1.8 vous avez la possibilités d'utiliser des implémentations faites par l'université de Rennes1. Cela permet:
  - L'utilisation du calendrier des examens types Rennes1 (calendrierEtape)
  - L'utilisation du calendrier de rentrée type Rennes1.
  - L'utilisation des habilitations Apogée

- Pour cela il vous faudra :
  - Configurer monDossierWeb.xml : activer les options désirées (les mettre à true) , décommenter le bean basé sur EtudiantAmueR1, décommenter le bean basé sur SecurityHabilitationApogee
  - Configurer specific.xml : Décommenter la déclarations des beans.
  - Il vous faudra également avoir installé le WS de Rennes1.

le CSS :

- Les différentes pages de l'application utilisent autant que possible les feuilles de styles CSS.
- En effet, la(les) feuille(s) de style de votre portail peut ne pas couvrir l'ensemble des styles de l'application ou avoir un rendu non désiré. Pour résoudre ce problème il suffit de surcharger les feuilles de styles de votre portail avec une feuille de style ne modifiant que les styles de la portlet esup-mondossierweb.
- A nouveau à titre d'exemple, le déploiement servlet utilisera la feuille de style suivante : /webapp/media/style/esup-mondossierweb.css. A partir de cette css vous pouvez suivre le déroulement complet des styles à Nancy2. Pour vous, le but ici est de remplacer esup-mondossierweb.css (ainsi que la bannière se trouvant dans media/images/CMonDossierWeb/banniere.gif) par vos propres feuilles de style. La css nancy2 est là à titre d'exemple.