Installation

Portlet Stage-Emploi version 1.0.1

Installation et paramétrage du portlet
Auteur : Lucie DENGREVILLE - Dominique HOUDET-JOLY

Pré requis

Installation du package "esup-portail"

Référez-vous à la documentation de Esup-portail. Le portlet a été installé sur les versions 2.5 et 2.6 ALM de Esup-Portail, mais n'a pas encore été testé pour la version Esup-portail 2.6 DLM.

Librairies du portail

Vous trouverez dans le répertoire utils les librairies indispensables pour l'execution de l'application Stage-Emploi. Vous devez les ajouter si elles sont absentes, par exemple dans ${tomcat.home}/common/lib pour jsp-api.jar et servlet-api.jar et dans ${portal.lib} pour portlet-api.jar

SGBD

Ce portlet utilise le SGBD MySql, il a été testé avec les versions MySQL 4.0.24_Debian-10sarge2-log, 5.0.51, 5.0.22 et 5.1.28 de ce SGBD.

Annuaire Ldap

L'application Stage-Emploi n'accède pas directement aux applications de gestion de l'établissement , mais uniquement à l'annuaire LDAP, en lecture (pas de modification de LDAP depuis l'application). Les informations dans LDAP sont récupérées depuis Apogée (pour les étudiants) ou depuis Harpège (pour les personnels de l'université). En plus des deux unités d'organisation recommandées par SUPANN (ou=people et ou=groups), il faut prévoir dans LDAP:

- une unité facultative ou=composante. Celle-ci n'est pas indispensable dans l'application si les libellés des groupes sont bien présents dans l'unité ou=groups.
- une unité ou=Etape où sont stockées les codes et libellés des étapes d'étude

Voici les attributs utilisés dans l'application :

- eduPersonPrimaryAffiliation récupéré lors de la connection au portail pour l'utilisateur connecté. -> cet attribut doit être obligatoirement renseigné et récupéré par esup-portail (à ajouter dans le fichier de mapping attributs esup-portail/ attributs ldap -personDirectory.xml- si n'existe pas) eduPersonPrimaryAffiliation est également directement utilisé par accès à LDAP lors de la recherche de personnes
- uid = identifiant unique (dans ou=people)
- sn = nom (dans ou=people, tiré d'Harpège ou d'Apogée)
- givenName = prénom (dans ou=people, tiré de Harpège ou d'Apogée)
- mail (dans ou=people)
- suppanCivilite = civilité (pour les personnels dans ou=people, tiré d'Harpège)
- type de personnel (ITARF, Ens. Chercheurs, Ens. 2nd degré, Ass. prof.(mi-temps), ASU... dans ou=people, tiré d'Harpège)
- telephoneNumber pour les personnels (dans ou=people)
- campus (pour les personnels, dans ou=people)
- bâtiment (pour les personnels, dans ou=people)
- supannEtuId pour les étudiants = numéro d'étudiant (dans ou=people, tiré d'Apogée)
- supannEmpId pour les personnels = numéro de l'employé (dans ou=people, tiré d'Harpège)
- code affectation (code du service d'affectation pour un personnel ou code de l'UFR pour un étudiant, correspond à cn dans ou=groups, tiré d'Harpège ou d'Apogée)
- libellé affectation =description (recherché dans ou=composante à Rennes2 mais peut être configuré par ou=groups si les libellés y sont bien présents, tiré d'Harpège)
- code étape pour les étudiants , dans ou=people , correspond à cn dans ou=groups, tiré d'Apogée)
- libellé étape = description (dans ou=groups et ou=etape)

Certains attributs doivent obligatoirement être renseignés : les attributs indispensables sont, pour ou=people : identifiant (uid), nom (sn), prénom (givenName), affiliation (eduPersonPrimaryAffiliation), affectation (code affectation), et également, pour les étudiants, étape d'étude (code étape) et numéro étudiant (suppanEtuId).

Contenu après décompression

Voici les différents fichiers présents depuis le répertoire racine

- INSTALL : *un document texte qui résume les procédures d'installation*.
- build.xml : *c'est le fichier utilisé lors des différentes directives ant*. **A ne pas modifier**.
- build.properties : *c'est un fichier de propriétés qui paramètre le fonctionnement de l'installation. Toutes les personnalisations propres au deploiement du portlet dans votre portail sont paramétrés depuis ce fichier*. **A modifier obligatoirement**
- properties/config-example.properties : *c'est un fichier de propriétés qui paramètre le fonctionnement du portlet. Toutes les personnalisations propres à l'université sont paramétrés depuis ce fichier.* **A modifier obligatoirement**
- properties/log4j-example.properties : *c'est un fichier d'exemple dans lequel sont paramétrées les propriétés concernant la gestion des logs du portlet. Il sert de base pour la création du fichier log4j.properties utilisé par Stage-Emploi* **A modifier obligatoirement**
- properties/esup-pstage-pubchan-example.properties : *c'est un fichier d'exemple pour le déploiement de Stage-Emploi dans le portail. Il sert de base pour la création du fichier esup-pstage-pubchan.xml.* **A modifier obligatoirement**
- properties/custom/sizes.xml : *c'est un fichier représentant les différentes tailles (effectifs) d'établissement d'accueil.* **Peut être modifié.**
- properties/custom/status.xml :*c'est un fichier représentant les différentes statuts juridiques d'établissement d'accueil.* **Peut être modifié.**
- properties/custom/covers.xml :*c'est un fichier représentant les différents choix de d'affiliation à la sécurité sociale dans une convention.* **A modifier selon les langues d'éditions définies (avec restriction - voir dans le fichier pour plus d'informations).**
- properties/custom/healthAuthorities.xml : *c'est un fichier représentant les différents choix de caisses d'assurance maladie dans une convention.* **Peut être modifié.**
- properties/custom/languageCodes.xml : *c'est un fichier permettant de définir les différentes langues d'édition des conventions..* **A modifier selon les langues d'édition des conventions souhaitées.**
- properties/custom/languages-example.xml : *c'est un fichier d'exemple permettant de configurer le texte de la convention et de l'avenant lors de la production du document sous forme de fichier pdf.* **A modifier**
- properties/custom/functions.xml : *c'est un fichier représentant les différentes fonctions APEC (et les codes correspondant) utilisées dans les offres de stages.* **A éviter de modifier.**
- properties/custom/formations.xml : *c'est un fichier définissant la liste des famille de formations dispensées à l'université (ex. : SPORT, TRADUCTION...)* **Peut etre modifié**
- properties/custom/levels.xml : *c'est un fichier définissant la liste des niveaux formations (ex. bac+2, Licence (bac+1,+2,+3)...) pour les offres de stage et d'emploi.* **Peut être modifié.**
- properties/custom/contracts.xml : *c'est un fichier représentant les différents choix de types de contrat pour une offre d'emploi.* *Peut être modifié.*
- properties/custom/themes.xml : *c'est un fichier représentant les différents choix de thématique de stage pour une convention de stage.* **Peut être modifié.**
- properties/custom/civilities.xml : *c'est un fichier représentant les différentes traductions des civilités pour une convention de stage.* **Peut être modifié.**
- properties/custom/basicChoices.xml : *c'est un fichier représentant les différentes choix de bases possibles pour certains champs.* *Peut être modifié avec réserve.*

- properties/custom/candidatures.xml : *c'est un fichier représentant les différentes choix possibles pour le mode de candidature dans une offre de stage.* **Peut être modifié.**

Installation

- Télécharger esup-portlet-PStage-<version>.zip sur le site du projet [|http:_sourcesup.cru.fr/frs/?group_id=257]
- Décompresser le fichier dans un répertoire de travail

> ⚠ Pour configurer cette application, vous devez adapter un certain nombre de fichiers de configuration. Vous devez recopier chaque fichier d'exemple <nomFichier>-example.<extension> en <nomFichier>.<extension> avant de les adapter.

- Adapter le fichier build.properties
- Adapter le fichier properties/config.properties
- Adapter le fichier properties/messages/messages_fr.properties
- Adapter le fichier webpages/portlet.xml si nécessaire
- Adapter le fichier properties/custom/languageCodes.xml
- Adapter le fichier properties/custom/languages.xml
- Adapter le fichier properties/custom/covers.xml
- Adapter les autres fichiers properties/custom/*.xml si nécessaire
- Initialiser la Base de données

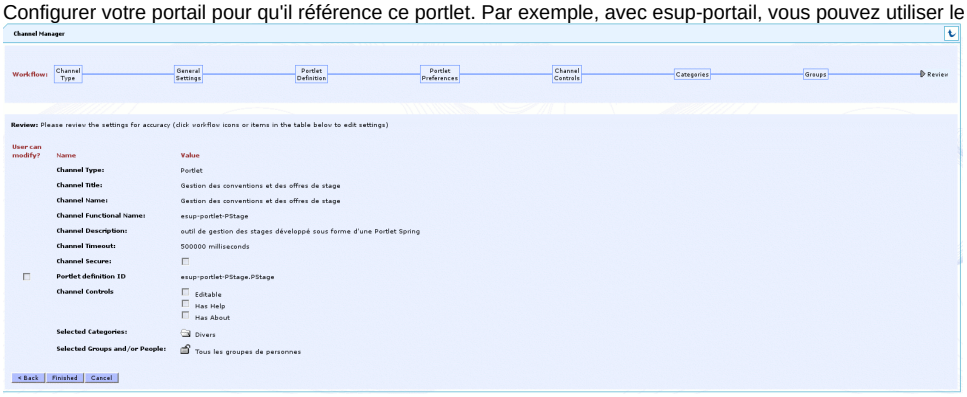
> ⚠ Ce portlet utilise le SGBD MySql, il a été testé avec les versions MySQL 4.0.24, 5.0 et 5.1 de ce SGBD.

- Créer la base de données StageEmploi (ou même nom que celui indiqué dans votre config.properties)
- Lancer le script de création des tables de l'application PStage que vous trouverez dans le répertoire db : **base_StageEmploi_v101.sql**
- Lancer les scripts de remplissage des tables spécifiques aux codes NAF (codes APE) que vous trouverez dans le répertoire db (à lancer dans cet ordre): **naf2008_n1.sql**,**naf2008n5.sql**
- Créer un utilisateur ayant les droits sur la base StageEmploi Le nom de cet utilisateur et son mot de passe seront ceux à indiquer dans votre fichier *config.properties*

> ⓘ Si vous aviez déjà installé une version 0.7.6 du portlet Stage-Emploi, et que vous ne souhaitiez pas perdre les données contenues dans votre base de données, reportez vous à la mise à jour du portlet .

- Configurer deux nouveaux contextes au niveau du serveur d'application. Pour cela, il faut modifier le fichier server.xml publié dans {tomcat.lib}/conf en ajoutant les contextes suivants.
  Pour le premier contexte, modifez le *docBase* en indiquant le chemin complet du répertoire de déploiement du portlet.
  <Context path="/esup-portlet-PStage" docBase="leChemin/esup/webapps/esup-portlet-PStage"crossContext="true" reloadable="false" > < /Context> Dans le second contexte ci-dessous, le *docBase* doit être modifié et doit correspondre au path indiqué dans votre fichier config. properties pour la variable logos.dir (ne pas modifier la valeur de la variable *path*).<Context path="/esup-portlet-PStage/data" docBase="leChemin /logos" crossContext="true" reloadable="false" />

- Configurer votre portail pour qu'il référence ce portlet. Par exemple, avec esup-portail, vous pouvez utiliser le "channel manager" :



> ⚠ le "Portlet definition ID" est très important. Dans notre cas, il s'agit de **esup-portlet-PStage.PStage**. L'identifiant esup-portlet-PStage doit être identique à celui indiqué dans le contexte de server.xml au niveau du serveur d'application et PStage doit être identique au nom du portlet (portlet-name) du fichier webpages/portlet.xml.

- Modifier le fichier properties/personDirectory.xml du portail pour qu'il récupère l'attribut eduPersonPrimaryAffiliation de l'annuaire LDAP.

```
<bean id="uPortalLdapAttributeSource" class="org.jasig.portal.services.persondir.support.
LdapPersonAttributeDaoImpl">
...
```

```
<property name="ldapAttributesToPortalAttributes">
 <map>
 ...
 <entry key="eduPersonPrimaryAffiliation">
 <value>eduPersonPrimaryAffiliation</value>
 </entry>

 ...
 </map>
</property>
</bean>
```


Pour que l'application Stage-Emploi fonctionne correctement, l'attribut eduPersonPrimaryAffiliation doit être renseigné pour toutes les personnes dans votre annuaire LDAP. La valeur de cet attribut sera utilisé pour connaître le profil de l'utilisateur connecté (étudiant, enseignant, ou employé), c'est pourquoi il est également important que ce soit correctement renseigné.

- Feuille de Styles : L'application Stage-Emploi utilise la feuille de style spécifique des portlets située au niveau du portail (ex . : esup_portlet.css) afin d'homogénéiser l'outil avec la charte graphique de l'université. Les styles utilisés sont les suivants (à ajouter dans le fichier css si nécessaire) :
  - portlet-background-light
  - portlet-background-highlight
  - portlet-background-content
  - portlet-label
  - portlet-font
  - portlet-section-text
  - portlet-msg-info
  - portlet-msg-warning
  - portlet-msg-error
  - portlet-table-header
  - portlet-table-text
  - portlet-table-caption
  - portlet-form-button
  - portlet-form-input-field
  - portlet-menu-item
- Deployer l'application: lancer la commande **ant deploy**

Configuration du Portlet

Le fichier de configuration config.properties

Le fichier properties/config.properties est utilisé pour configurer le portlet. Vous trouverez un exemple dans le fichier properties/config-sample.properties sur lequel il faut se baser pour créer le fichier properties/config.properties

Configuration de la base de données propre au portlet Stage-emploi

```
#########################################################
# fichier de configuration de l'application StageEmploi#
#########################################################

##
# Paramètres de connexion à la base de données StageEmploi
##

# Driver
db.driver=com.mysql.jdbc.Driver

# URL
db.url=jdbc:mysql:_localhost:3306/StageEmploi
# User
db.user=nomUser
```

```
# Password
db.password=password
```

- db.driver : driver jdbc utilisé par le portlet pour accéder à la base de données
- db.url :URL jdbc de connexion /nom du serveur :port/ nom de la base de données
- db.user : nom de l'utilisateur pour interroger et modifier la base de données
- db.password : mot de passe pour interroger et modifier la base de données

Configuration des paramètres de connection à l'annuaire LDAP

```
#####################################################
# Noms de paramètres LDAP dans l'appli mappés avec les
# paramètres de connexion à la base ldap
#(utilisés dans portlet-pstage-dao.xml)
############################################

# URL
ldap.url=ldap:_ldap.univ.fr:389/
# baseDn
ldap.base=dc=ldap,dc=uhb,dc=fr
# UserName
ldap.userName=cn=machin,ou=System,dc=ldap,dc=uhb,dc=fr
# Password
ldap.password=password
```

- ldap.url : url de l'annuaire LDAP de l'université
- ldap.base : baseDN
- ldap.userName nom de l'utilisateur pouvant interroger l'annuaire
- ldap.password : mot de passe pour interroger et modifier l'annuaire LDAP (important : StageEmploi ne modifie pas les données de LDAP)

```
# filtre recherche de personnes dans ldap
ldap.people=ou=People

# filtre recherche de groupes dans ldap
ldap.group=ou=Groups

# filtre recherche de composantes dans ldap
ldap.composante=ou=Composante

# filtre recherche des etapes dans ldap
ldap.step=ou=Etape
```

- ldap.people : unité d'organisation de ldap où sont stockées les personnes
- ldap.group : unité d'organisation de ldap où sont stockés les groupes de personnes (la notion de groupe ~~contenant des membres des groupes~~ n'est pas utilisée dans l'application, vous pouvez indiquer la même valeur que pour ldap.composante : seuls l'identifiant et le libellé de chaque groupe sont récupérés par l'application)
- ldap.composante : unité d'organisation de ldap où sont stockés les groupes de personnes (peut être identique à ldap.group si les tous libellés des groupes y sont biens présents)
  *ldap.step : unité d'organisation de ldap où sont stockées uniquement les étapes d'étude. Si votre annuaire LDAP ne comporte pas cette unité d'organisation, vous devez la créer, en particulier si vous choisissez une gestion des centres par étape d'étude (criterium.university. center=ETAPE). Si vous choisissez une critère par UFR ou ETABLISSEMENT, le ldap.step peut être identique à ldap.composante.

```
# codes des ufrs ldap existants à l'université
ldap.ufr=901,902,903,904,905,701,801,802,804,805,959,955,962

# regroupement des départements dans chaque ufr
ldap.ufrEtDep=901:90101,90102,90103,90104-902:90202,90203,90204,90205-903:90301,90302,90303,90304-904:
90401,90402,90403-905:90501-804:90403,90402,90401-805:90501-955:95500
```

- ldap.ufr : codes des ufrs ldap existants à l'université, séparés par une virgule
- ldap.ufrEtDep : regroupement des départements dans chaque ufr: à rentrer sous forme -> code de l'ufr suivi de deux points ":" et de la liste des départments composants cet ufr séparés par une virgule ",". Les ufr sont séparés par un tiret "-". exemple -> codeUfr1:codeDep1,codeDep2, codeDep3-codeUfr2:codeDep4,codeDep5-codeUfr3:codeDep6

```
#affiliation personne dans esup-portail
esup.affiliation=eduPersonPrimaryAffiliation

#valeur de l'attribut esup.affiliation pour un employe, un prof ou un etudiant dans esup-portail
esup.employee=employee
esup.faculty=faculty
esup.student=student
```

- esup.affiliation : nom de l'attribut de esup-portail dans lequel est récupéré l'affiliation de la personne connectée au portail. Cet attribut correspond à celui mappé à l'attribut eduPersonPrimaryAffiliation de LDAP dans le fichier personDirectory.xml du portail
- esup.employee : valeur de l'attribut esup.affiliation pour un employé (ITRF, ASU...)
- esup.faculty : valeur de l'attribut esup.affiliation pour un enseignant
- esup.student : valeur de l'attribut esup.affiliation pour un étudiant

```
#nom de l'attribut spécifiant l'affiliation d'une personne dans ldap
ldap.affiliation=eduPersonPrimaryAffiliation

#valeur de ldap.affiliation pour un etudiant
ldap.student.affiliation=student

#valeur(s) de l'attribut ldap.affiliation pour un administratif
#si plusieurs valeurs possibles, les séparer par une virgule
#exemple : ldap.employee.affiliation=employee,affiliate
ldap.employee.affiliation=employee

#valeur de l'attribut ldap.affiliation pour un enseignant
ldap.faculty.affiliation=faculty
```

- ldap.affiliation : nom de l'attribut de LDAP dans lequel est stocké l'affiliation des personnes.
- ldap.student.affiliation : valeur de l'attribut ldap.affiliation pour un étudiant.
- ldap.employee.affiliation : valeur(s) de l'attribut ldap.affiliation pour un employé. Si plusieurs valeurs existent dans votre annuaire ldap pour définir les employés, séparez les valeurs par une virgule.
- ldap.faculty.affiliation : valeur de l'attribut ldap.affiliation pour un enseignant.

```
#identifiant personne dans ldap
ldap.uid=uid

#nom personne
ldap.name=sn

#prenom personne
ldap.firstName=givenName

#mail personne dans ldap
ldap.mail=mail
```

- ldap.uid : nom de l'attribut de LDAP dans lequel est stocké l'identifiant d'une personne.
- ldap.name : nom de l'attribut de LDAP dans lequel est stocké le nom d'une personne.
- ldap.firstName : nom de l'attribut de LDAP dans lequel est stocké le prénom d'une personne.
- ldap.mail : nom de l'attribut de LDAP dans lequel est stocké le mail d'une personne.

```
#numero etudiant
ldap.student.id=supannEtuId


#code ufr d'inscription d'un etudiant
ldap.student.ufr=attuhbaffectation

#code etape d'inscription d'un etudiant
ldap.student.step=attuhbetp
```

- ldap.student.id : nom de l'attribut de LDAP dans lequel est stocké le numéro d'un étudiant.
- ldap.student.ufr : nom de l'attribut de LDAP dans lequel est(sont) stocké le(s) code(s) des UFR dans le(s)quel(s) est inscrit un étudiant.
- ldap.student.step : nom de l'attribut de LDAP dans lequel est(sont) stocké le(s) code(s) des étapes d'étude dans la(es)quelle(s) est inscrit un étudiant.

```
#code d'affectation d'un personnel
ldap.member.affectation=attuhbaffectation

#type de personnel
ldap.member.type=attuhbtype

#telephone personnel
ldap.member.phone=telephoneNumber

#campus personnel
ldap.member.campus=attuhbcampus

#bureau personnel
ldap.member.room=roomNumber

#batiment personnel
ldap.member.building=buildingName

#civilite personnel
ldap.member.civility=supannCivilite

#type de personnel appartenant à ldap.faculty.affiliation mais ne pouvant être tuteur de stage
#(ex. : lecteur, moniteur) : les valeurs sont séparées par une virgule ","
ldap.faculty.nonTutor=Moniteur,AMN,Lecteur (échange),Lecteur (personnel)
```

- ldap.member.affectation : nom de l'attribut de LDAP dans lequel est stocké le code d'affectation d'un personnel.
- ldap.member.type : nom de l'attribut de LDAP dans lequel est stocké le type d'un personnel.
- ldap.member.phone : nom de l'attribut de LDAP dans lequel est stocké le numéro de téléphone d'un personnel.
- ldap.member.campus : nom de l'attribut de LDAP dans lequel est stocké le campus d'un personnel.
- ldap.member.room : nom de l'attribut de LDAP dans lequel est le numéro du bureau d'un personnel.
- ldap.member.building : nom de l'attribut de LDAP dans lequel est le nom du bâtiment où se trouve le personnel.
- ldap.member.civility : nom de l'attribut de LDAP dans lequel est la civilité d'un personnel.
- ldap.faculty.nonTutor : valeurs de ldap.member.type pour les personnes dont la valeur de ldap.affiliation est celle indiquée dans ldap.faculty. affiliation (qui sont donc classés dans les enseignants) mais ne pouvant être tuteur d'un stage. Les valeurs sont séparées par une virgule ",".

```
#libelle composante dans ldap.composante
ldap.composante.libelle=attuhbcompLibelle

#code composante dans ldap.composante
ldap.composante.code=cn

#libelle groupes dans ldap.group
ldap.group.libelle=description

#codes groupes dans ldap.group
ldap.group.code=cn

#libelle etapes dans ldap.etape
ldap.step.libelle=attuhbcompLibelle

#codes etapes dans ldap.etape
ldap.step.code=cn
```

- ldap.composante.libelle : nom de l'attribut de LDAP dans lequel est stocké le libellé d'une composante stockée dans ldap.composante.
- ldap.composante.code : nom de l'attribut de LDAP dans lequel est stocké le code d'une composante stockée dans ldap.composante.
- ldap.group.libelle : nom de l'attribut de LDAP dans lequel est stocké le libellé d'un groupe stocké dans ldap.group.
- ldap.group.code : nom de l'attribut de LDAP dans lequel est stocké le code d'un groupe stocké dans ldap.group.
- ldap.step.libelle : nom de l'attribut de LDAP dans lequel est stocké le libellé d'une étape stockée dans ldap.etape.
- ldap.step.code : nom de l'attribut de LDAP dans lequel est stocké le code d'une étape stockée dans ldap.etape.

Année universitaire

```
#jour et mois de début d'une année universitaire
# pour année universitaire commençant le 1er octobre, noter :
#start.year.day=01
```

```
#start.year.month=10
start.year.day=01
start.year.month=10
```

- start.year.day : jour de début d'année universitaire (de 01 à 30 ou 31 selon le mois défini dans start.year.month)jour de début d'année universitaire (de 01 à 30 ou 31 selon le mois défini dans start.year.month). Par exemple pour une année universitaire débutant le 1er octobre, mettre 01.
- start.year.month : mois de début d'année universitaire (de 01 à 12). Par exemple pour une année universitaire débutant le 1er octobre, mettre 10.

Les variables start.year.day et start.year.month sont utilisées pour le calcul automatique de l'année universitaire des conventions de stages et des offres de stage et d'emploi.

Répertoire de stockage des logos

```
#repertoire où seront stockés les logos des centres de gestion (chemin absolu)
#ce répertoire doit être le même que celui indiqué dans le fichier conf/server.xml de Tomcat par ajout du
contexte :
#exemple : <Context path="/esup-portlet-PStage/data" docBase="/home/lucie/pstage/logos" crossContext="true"
reloadable="false" />
logos.dir=/home/lucie/pstage/logos
```

- logos.dir : chemin absolu du répertoire où seront stockés les logos des centres de gestion. Ce répertoire doit être le même que celui indiqué dans le fichier conf/server.xml de Tomcat par ajout du contexte :

  ```
  <Context path="/esup-portlet-PStage/data" docBase="/home/lucie/pstage/logos" crossContext="true"
  reloadable="false" />
  ```

Configuration pour l'administration de l'application

```
#critère de création des centres de gestion :
#criterium.university.center=
#criterium.university.center=UFR
criterium.university.center=ETAPE

#uid (de ldap) des administrateurs principaux de l'appli,les valeurs sont séparées par une virgule ","
mainAdmin=machin_m,truc_b
```

- criterium.university.center : critère de création des centres de gestion Décommentez la ligne correspondant au critère choisi : UFR ou ETAPE dans le cas où l'établissement a plusieurs centres de gestion. Dans ce cas, par défaut, il existe aussi un critère ETABLISSEMENT proposé lors de la création d'un centre. Dans le cas où vous souhaitez un seul centre de gestion pour l'ensemble de l'établissement, décommentez la ligne criterium.university.center=. Dans tous les cas, un seul centre ETABLISSEMENT pourra être créé.
- mainAdmin : uid (de ldap) des administrateurs principaux de l'application,les valeurs sont séparées par une virgule ",". Ces administrateurs seront les seules personnes à pouvoir créer et supprimer des centres de gestion.

Le fichier portlet.xml

Le fichier webpages/portlet.xml permet de configurer le démarrage de la partie Spring spécifique au portlet. La majeur partie de ce fichier ne doit pas être modifiée. Seule l'information entre les deux balises <user-attribute> peut être modifée en fonction de l'attribut ldap récupéré depuis le portail : vous devez ainsi indiquer la même valeur que celle saisie pour esup.affiliation dans votre fichier properties/config.properties.

Ainsi, si dans votre fichier properties/config.properties vous avez saisi esup.affiliation=eduPersonPrimaryAffiliation, indiquez eduPersonPrimaryAffiliation entre les deux balises <name>.

```
<user-attribute>
 <description>affiliation de la personne</description>
 <name>eduPersonPrimaryAffiliation</name>
</user-attribute>
```

⚠ Vous ne devez pas modifier les autres parties du fichier portlet.xml.

Le fichier messages_fr.properties

Le fichier properties/messages/messages_fr.properties permet de configurer certains messages que vous souhaiteriez personnaliser. Les variables doivent exister (ne pas les mettre en commentaire) mais peuvent ne rien contenir (vous pouvez indiquer "variable="), sauf pour les variables mail.error.student et mail.error.member.

```
##
# messages de l'application
##

#page d'accueil#
home.welcome=Bienvenue sur l'espace Stage-Emploi
student.welcome=Cet outil est accessible <strong>\u00E0 tous les \u00E9tudiants de l'Universit\u00E9 Rennes 2.
offer.need.validated=Vous devez valider cette offre avant de pouvoir la diffus\u00E9e.

offer.need.contact=Au moins un contact doit \u00EAtre d\u00E9fini pour enregistrer cette offre.

#codeNAf pour etablissment d'accueil
structure.codeNaf.info=Vous pouvez trouver le num\u00E9ro de SIRET et le code NAF de l'\u00E9tablissement
d'accueil sur des sites internet sp\u00E9cialis\u00E9s sur les soci\u00E9t\u00E9s.

#duree maximum pour un stage conseillé
convention.max.duration=Selon la loi sur l'\u00E9galit\u00E9 des chances, un stage conseill\u00E9 ne peut
d\u00E9passer six mois de date \u00E0 date.

#message indiqué lors de la creation d'un service
formulaire.add.service=Si l'\u00E9tablissement est trop petit pour pouvoir identifier un service d'accueil, ne
rien saisir pour le nom du service.


#mail a indiquer lors du message d'erreur (erreur base de donnee, faterror ou portleterror) lorsque c'est
# un etudiant qui est connecte
mail.error.student=stages-emplois-etudiant@listes.univ.fr

#mail a indiquer lors du message d'erreur (erreur base de donnee, fatalerror ou portleterror) lorsque c'est
# un personnel qui est connecte
mail.error.member=stages-emplois-gestionnaire@listes.univ.fr
```

- home.welcome : message d'accueil pour tous les utilisateurs. S'affiche dans le style "portlet-label".
- student.welcome : message d'accueil spécifique aux étudiants (affiché en plus du home.welcome). S'affiche dans le style "portlet-msg-info".
- offer.need.validated : message à afficher aux gestionnaires souhaitant diffuser une offre qui n'a pas été validée.
- offer.need.contact : messsage à afficher aux gestionnaires souhaitant enregistrer une offre dont aucun contact n'a été défini.
- structure.codeNaf.info : message à afficher lors de la création d'un établissement dans une convention. Permet par exemple d'indiquer la façon de trouver le codeNaf ou le code siret de l'établissement.
- convention.max.duration : indication de la loi concernant la durée maximum d'un stage conseillé.
- formulaire.add.service : message à afficher lors de la création d'un service.Par exemple : "Si l'\u00E9tablissement est trop petit pour pouvoir identifier un service d'accueil, ne rien saisir pour le nom du service".
- mail.error.student : adresse du courriel où les étudiants doivent prévenir qu'ils ont rencontré un problème technique (cette adresse de courriel est indiquée lors d'une erreur base de données, erreur fatale ou erreur portlet).
- mail.error.member : adresse du courriel où les gestionnaires doivent prévenir qu'ils ont rencontré un problème technique (cette adresse de courriel est indiquée lors d'une erreur base de données, erreur fatale ou erreur portlet).

Le fichier languageCodes.xml

Le fichier properties/custom/languageCodes.xml permet de définir les différentes langues d'édition des conventions de stage. Il contient le code et le libellé de chaque langue. Ces codes de langues seront ceux à utiliser dans les fichiers properties/custom/languages.xml, properties/custom/covers.xml et properties/custom/civilities.xml.

⚠ Ce fichier est à personnaliser avec les langues d'édition souhaitées.

```
<languages>
 <language id="fr">Français</language>
 <language id="en">Anglais</language>
...
</languages>
```

Pour chaque balise <language>, on a :

- id = identifiant de la langue, qui doit être le même que celui apparaissant dans languages.xml, covers.xml et civilities.xml
- libellé correspondant entre les deux balises<language></language>

Le fichier languages.xml

Le fichier properties/custom/languages.xml est utilisé pour stocké le texte de la convention dans toutes les langues d'édition nécessaires à votre établissement.

> ⚠ Vous devez impérativement modifier ce fichier pour que les conventions soient adaptées à votre établissement. Les identifiants présents dans les balises language (ex. : _<language id="fr">_) doivent obligatoirement correspondre au code de langue définis dans le fichier properties/custom/languageCodes.xml. Référez-vous au fichier properties/custom/languages.dtd dans lequel est définie la dtd de languages.xml.

Le fichier basicChoices.xml

Le fichier properties/custom/basicChoices.xml permet de définir les différents de choix base. Ces choix sont en particulier utilisés pour les champs "offre pourvue" et candidat local dans une offre de stage.Chaque type de choix est défini dans une balise <choice>.

> ⚠ Ce fichier est à personnaliser si nécessaire.

```
<choices>

 <choice id="0">non</choice>
 <choice id="1">oui</choice>
 <choice id="2">non spécifié</choice>

</choices>
```

Pour chaque balise <choice> on a :

- id= identifiant du choix qui sera stocké dans la base
- le libellé du choix doit être placé entre les balises <choice></choice>, c'est lui qui apparaitra à l'écran
  L'ordre dans lequel sont classés les choix sera l'ordre d'apparition à l'écran.

> ⚠
> - le choix avec id="0" doit forcement correspondre à non
> - le choix avec id="1" doit forcement correspondre à oui

Le fichier candidatures.xml

Le fichier properties/custom/candidatures.xml permet de définir les libellés des modes de candidature possibles lors de la saisie d'une offre d'emploi. Chaque mode est défini dans une balise <candidature>.

> ⚠ Ce fichier est à personnaliser si nécessaire.

```
<candidatures>

<candidature id="1">envoi par courriel</candidature>
<candidature id="2">envoi par courrier postal</candidature>
<candidature id="3">contact téléphonique</candidature>
```

```
<candidature id="4">lettre manuscrite</candidature>
<candidature id="5">indifférent</candidature>

</candidatures>
```

Pour chaque balise <candidature> on a :

- id= identifiant du mode de candidature qui sera stocké dans la base
- le libellé du mode de candidature doit être placé entre les balises <candidature></candidature>, c'est lui qui apparaitra à l'écran
  L'ordre dans lequel sont classés les choix sera l'ordre d'apparition à l'écran.

Le fichier civilities.xml

Le fichier properties/custom/civilities.xml permet de définir les libellés des civilités dans les différentes langues d'édition des conventions. le choix en français est celui qui apparaitra à l'écran les choix dans les autres langues ne sont utilisés que pour l'impression Les identifiants présents dans les balises language (ex. : *<language id="fr">*) doivent obligatoirement correspondre au code de langue définis dans le fichier properties/custom/languageCodes.xml.

⚠ Ce fichier est à personnaliser selon les langues d'édition choisies dans languageCodes.xml

L'ordre dans lequel sont classés les choix sera l'ordre d'apparition à l'écran.

```
<civilities>
<language id="fr">
 <civility id="M">M.</civility>
 <civility id="MME">MME</civility>
 <civility id="MLLE">MLLE</civility>
</language>
<language id="en">
 <civility id="M">Mr.</civility>
 <civility id="MME">Mrs.</civility>
 <civility id="MLLE">Ms.</civility>
</language>
...
</civilities>
```

Pour chaque balise <language>, on a :

- id = identifiant de la langue, qui doit être le même que celui apparaissant dans languageCodes.xml
- les balises <civility> avec un id = identifiant qui sera stocké dans la base et et le libellé correspondant entre les deux balises<civility></civility>
  les identifiants et libellés des balises <civility> pour chaque langue doivent correspondre à ceux indiqués pour le Françaisl

Le fichier contracts.xml

Le fichier properties/custom/contracts.xml permet de définir les libellés des types de contrats possibles lors de la saisie d'une offre d'emploi. Chaque type de contrat est défini dans une balise <contract>.

⚠ Ce fichier est à personnaliser si nécessaire.

```
<contracts>

 <contract id="1">CDD</contract>
 <contract id="2">CDI</contract>

</contracts>
```

Pour chaque balise <contract> on a :

- id= identifiant du type de contrat qui sera stocké dans la base
- le libellé du contrat doit être placé entre les balises <contract></contract>, c'est lui qui apparaitra à l'écran
  L'ordre dans lequel sont classés les choix sera l'ordre d'apparition à l'écran.

Le fichier covers.xml

Le fichier properties/custom/covers.xml permet de définir les divers types d'affiliation à la sécurité sociale dans les différentes langues d'édition des conventions.Les identifiants présents dans les balises language (ex. : _<language id="fr">_) doivent obligatoirement correspondre au code de langue définis dans le fichier properties/custom/languageCodes.xml.De plus un identifiant de type d'affiliation doit correspondre au même type d'affiliation pour toutes les langues d'édition.

⚠ Ce fichier est à personnaliser selon les langues d'édition choisies dans languageCodes.xml

⚠ Le type d'affiliation correspondant à id=4 doit nécessairement correspondre au type "étudiant étranger" (en effet, dans le cas de l'identifiant 4, le numéro de sécurité sociale et la caisse de maladie ne sont pas obligatoires, puisqu'il s'agit d'un étudiant étranger)

```
<covers>
 <language id="fr">
 <cover id="1">en qualité d'ayant droit d'assuré social</cover>
 <cover id="2">en qualité d'étudiant(e)</cover>
 <cover id="3">par une assurance volontaire</cover>
 <cover id="4">étudiant étranger</cover>
 </language>
 <language id="en">
 <cover id="1">standard cover</cover>
 <cover id="2">student cover</cover>
 <cover id="3">other optional cover</cover>
 <cover id="4">foreigner student</cover>
</language>
...
</covers>
```

Le fichier formations.xml

Le fichier properties/custom/formations.xml permet de définir les libellés des familles de formations souhaitées lors de la saisie d'une offre de stage ou d'emploi.

⚠ Ce fichier est à personnaliser si nécessaire selon les formations dispensées à l'université. .

```
<formations>
 <formation id="1">AMENAGEMENT/URBANISME/ENVIRONNEMENT</formation>
 <formation id="2">ART/PATRIMOINE</formation>
 <formation id="3">COMMERCE INTERNATIONAL/MARKETING</formation>
...

</formations>
```

Chaque formation est définie dans une balise <formation>. Pour chaque balise <formation> on a :

- id= identifiant de la formation qui sera stocké dans la base
- le libellé de la formation doit être placé entre les balises <formation></formation>, c'est lui qui apparaitra à l'écran.
  L'ordre dans lequel sont classés les choix sera l'ordre d'apparition à l'écran.

Le fichier functions.xml

Le fichier properties/custom/functions.xml permet de définir les fonctions lors de la saisie d'une offre de stage ou d'emploi. Ce classement correspond aux fonctions définies par l'APEC.

ⓘ il est déconseiller de modifier ce fichier.

```
<functions>
 <mainFunction id="010" title="Direction générale">
 <function id="011">--Direction d'entreprise</function>
 <function id="012">--Adjoint conseil de direction</function>
 </mainFunction>
 <mainFunction id="020" title="Production">
 ...
 </mainFunction>
 ....
</functions>
```

Les balises <mainFunction> correspondent aux grands types de fonctions dans lesquels sont regroupées les fonctions.

Pour chaque balise <mainFunction> on a :

- id apec =identifiant qui sera stocké dans la base
- title = libellé de ce grand type de fonction, c'est lui qui apparaitra à l'écran
  Dans chaque balise <mainFunction> sont placées plusieurs balises <function> avec :

- id apec= identifiant de la fonction qui sera stocké dans la base
- le libellé apec de la fonction doit être placé entre les balises <function></function>, c'est lui qui apparaitra à l'écran. Ce libellé doit contenir les deux tirets avant le libellé proprement dit : cela permet de différencier les fonctions des grands types de fonctions lors du choix de la fonction
  L'ordre dans lequel sont classés les choix sera l'ordre d'apparition à l'écran.

Le fichier healthAuthorities.xml

Le fichier properties/custom/healthAuthorities.xml permet de définir les choix de caisses d'assurance maladie pour les conventions.

Ce fichier est à personnaliser si nécessaire.

```
<authorities>
 <authority id="CPAM">
 <lib>Caisse Primaire d'Assurance Maladie</lib>
 <info>Pour les étudiants, les travailleurs salariés, etc</info>
 </authority>
 <authority id="MSA">
 <lib>Mutualité Sociale Agricole</lib>
 <info>Pour les salariés ou exploitants agricoles</info>
 </authority>
....
</authorities>
```

Pour chaque caisse on a dans la balise <authority> avec id = identifiant de la formation qui sera stocké dans la base. Dans chaque balise authority, on a :

- balise <lib> = le libellé qui va s'afficher dans le choix de la caisse et qui sera imprimé sur la convention
- balise <info>= information qui ne s'affiche que lors du choix de la caisse
  L'ordre dans lequel sont classés les choix sera l'ordre d'apparition à l'écran.

Le fichier levels.xml

Le fichier properties/custom/levels.xml permet de définir les niveaux de formations souhaités dans les offres de stage ou d'emploi.

Ce fichier est à personnaliser si nécessaire.

```
<levels>

<level id="1">Bac+1</level>
<level id="2">Bac+2</level>
```

```
<level id="3">Bac+3</level>
<level id="4">Bac+4</level>
<level id="5">Bac+5</level>
<level id="6">Bac+8</level>
<level id="7">Licence (bac+1,2,3)</level>
<level id="8">Master (bac+4,5)</level>
<level id="9">Tous niveaux</level>

</levels>
```

Pour chaque balise <level>, on a ::

- id= identifiant qui sera stocké dans la base
- le libellé du niveau doit être placé entre les balises <level></level>, c'est lui qui apparaitra à l'écran
  L'ordre dans lequel sont classés les choix sera l'ordre d'apparition à l'écran.

Le fichier sizes.xml

Le fichier properties/custom/sizes.xml permet de définir les choix des tailles (effectif) d'établissement d'accueil.

⚠ Ce fichier est à personnaliser si nécessaire.

```
<sizes>
 <size id="0">Non connu</size>
 <size id="1">1 à 9</size>
 <size id="2">10 à 49</size>
 <size id="3">50 à 199</size>
 <size id="4">200 à 499</size>
 <size id="5">500 et plus</size>
</sizes>
```

Pour chaque balise <size>, on a :

- id= identifiant qui sera stocké dans la base
- le libellé de l'effectif doit être placé entre les balises <size></size>, c'est lui qui apparaitra à l'écran.
  L'ordre dans lequel sont classés les choix sera l'ordre d'apparition à l'écran.

Le fichier status.xml

Le fichier properties/custom/status.xml permet de définir les choix des statuts juridiques d'établissement d'accueil.

⚠ Ce fichier est à personnaliser si nécessaire.

ⓘ Le statut juridique avec l'identifiant 7 doit forcément correspondre au type "autre", c'est à dire à des établissements qui ne possèdent pas de numéro de SIret. En effet, le numéro de siret n'est pas obligatoire pour les établissements d'accueil dont le statut juridique est défini comme étant de type 7. En revanche, le numéro Siret est obligatoire pour tous les autres cas.

```
<statusList>

 <status id="0">Non connu</status>
 <status id="1">Entreprise privée</status>>
 <status id="2">Administration publique d'état</status>
 <status id="3">Administration territoriale</status>
 <status id="4">Administration hospitalière</status>
 <status id="5">Entreprise publique</status>
 <status id="6">Association</status>
 <status id="7">Autre (profession libérale, ONG...)</status>

</statusList>
```

Pour chaque balise <status>, on a :

- id= identifiant qui sera stocké dans la base
- le libellé du statut juridique doit être placé entre les balises <status></status>, c'est lui qui apparaitra à l'écran.
  L'ordre dans lequel sont classés les choix sera l'ordre d'apparition à l'écran.

Le fichier themes.xml

Le fichier properties/custom/themes.xml permet de définir les choix de thématique de stage dans une convention de stage

⚠ Ce fichier est à personnaliser selon les thématiques souhaitées.

```
<themes>
 <theme id="1">AMENAGEMENT/URBANISME/ENVIRONNEMENT</theme>
 <theme id="2">ART/PATRIMOINE</theme>
 <theme id="3">COMMERCE INTERNATIONAL/MARKETING</theme>
...

</themes>
```

Pour chaque balise <theme>, on a :

- id= identifiant qui sera stocké dans la base
- le libellé de la thématique doit être placé entre les balises <theme></theme>, c'est lui qui apparaitra à l'écran.
  L'ordre dans lequel sont classés les choix sera l'ordre d'apparition à l'écran.

Autres fichiers personnalisables

Les messages d'erreurs de l'application peuvent être modifiés si nécessaire dans le fichier properties/messages/error_fr.properties : veillez toutefois à ne pas supprimer ou dupliquer un message d'erreur.

Certains messages de l'applications peuvent être modifiés si nécessaire dans le fichier properties/messages/messages_fr.properties : veillez toutefois à ne pas supprimer ou dupliquer un message.

Les libellés des rubriques apparaissant lors de l'impression des offres sous format Pdf peuvent être modifiés si nécessaire dans le fichier properties /messages/offer.properties : veillez toutefois à ne pas supprimer ou dupliquer un identifiant.

Déploiement

Préparation du déploiement : modifier les chemins d'installation du fichier build.properties.

Si vous aviez déjà installé le portlet, lancez la commande **ant undeploy**, pour effacer toute trace du canal dans l'environnement de production.
Si vous aviez déjà compilé le portlet, lancez la commande **ant clean**, pour effacer toute trace dans le répertoire build.
Lancement de la commande **ant deploy**, pour déployer le canal dans les bons répertoires.
Vous pouvez également lancer la commande **ant all** pour effectuer toutes ces étapes en une seule fois .

L'administrateur peut maintenant publier le portlet dans le portail. Pour cela, il y a 2 méthodes :
La plus simple étant de publier le canal de manière normale, par l'interface WEB prévue à cet effet.
La seconde étant d'utiliser la commande **ant pubchan** (Cette méthode remplace toute la démarche de publication).
Recopier le fichier properties/esup-pstage-pubchan.xml dans le répertoire {uPortal}/properties/chanpub du portail.
Ensuite, lancez la commande **ant pubchan -Dchannel=esup-pstage-pubchan.xml** depuis le répertoire "racine esup".

Mise à jour depuis la version 0.7.6

Si vous avez déjà installé la version antérieure 0.7.6 du portlet Stages-Emplois, vous pouvez mettre à jour sans tout réinstaller.

La version 0.8 correspond à la mise en oeuvre des codes NAF 2008 définis par l'INSEE en janvier 2008.

ⓘ Un code NAF 2003 (version des codes NAF utilisée par les versions de Stages-Emplois antérieures à 0.8) correspond en général à plusieurs codes NAF 2008 : la mise à jour des codes NAF implique donc une suppression des codes NAF déjà stockés dans votre base de données, puis

une saisie de ces codes NAF dans la version 2008, sauf pour quelques codes qui sont automatiquement mis à jour avec la nouvelle nomenclature par le script (concerne environ 120 codes dont le taux de basculement d'un code NAF 2003 à un code NAF 2008 est de 100%).

Pour mettre à jour le portlet Stages-Emplois, vous devez :

- Lancer le script de mise à jour de la base de données : db_07_to_08.sql
- Lancer les scripts de remplissage des tables spécifiques aux codes NAF (à lancer dans cet ordre) : naf2008_n1.sql et naf2008_n5.sql
- Appliquer le patch de mise à jour de l'application : patch_stageEmploi_076_to_08.diff en lançant la commande dans votre répertoire PStage-0.7.6 :

```
patch -p1 < patch_stageEmploi_076_to_08.diff
```

- Redéployer l'application par la commande **ant all**