Canal Esup-Geisha

Manuel d'administration

Table des matières

1 Installation

1.1 Modes de déploiement

Trois modes de déploiement sont disponibles :

- **portlet** : proposé pour les administrateurs de portail, il déploie l'application en une portlet JSR-168, exécutable dans une portail compatible (uPortal, liferay, infoglue, ...),
- **servlet** : proposé pour les administrateurs désireux d'exécuter l'application dans un conteneur de servlet existant (Tomcat, jetty, ...),
- **quick-start** : proposé pour tester l'application, l'application est exécuté dans un Tomcat embarqué.

1.2 Pré-requis

L'installation de l'application nécessite quelques pré-requis :

- Le package fourni contient les sources de l'application qui doivent être compilées. Cette opération requière un kit de développement Java (une machine virtuelle seule ne suffit pas).
- Les commandes d'installation nécessitent l'utilitaire ant. Certaines librairies étant téléchargées automatiquement, les exploitants de serveurs derrière des proxies HTTP doivent configurer ant comme indiqué à l'adresse http://ant.apache.org/manual/proxy.html.
- L'application est configurée par défaut pour se connecter à une base de données MySql. Toutefois, l'utilisation du framework Hibernate permet en principe d'utiliser d'autres formats de base de données (cf.2 Configuration de la connexion à la base de données).
- L'authentification des utilisateurs nécessite l'accès à un annuaire LDAP, ainsi qu'à un serveur CAS (déploiements en servlet ou quick-start).
- L'envoie d'e-mails nécessite l'accès à un serveur SMTP.

1.3 Téléchargement

Décompressez le package zip que vous venez de télécharger. La décompression du package **esup-geisha-x.y.z-t.zip** crée le dossier **esup-geisha-x.y.z**.

> **Notez que les versions successives de l'application doivent être installées dans un dossier commun afin de simplifier les mises à jour.**

1.4 Propriétés du déploiement

Éditez le fichier **build.properties** pour définir les propriétés qui configurent l'installation de l'application.

Une fois ces paramètres définis (voir ci-dessous pour les détails), le comportement de l'application à l'exécution doit être configuré (cf.2 Configuration).

> **Lorsque vous changez le mode de déploiement, il est recommandé de tout nettoyer (commande "ant clean") avant de déployer à nouveau.**

1.4.1 Propriétés communes

La propriété **custom.recover.files** permet de récupérer les personnalisations (cf.3 Personnalisation) réalisées par l'administrateur lors d'une mise à jour (cf.4 Mise à jour) de l'application. Indiquez, au besoin, dans cette propriété les fichiers que vous souhaitez conserver. Par défaut, les fichiers de configuration suivant sont récupérés :

- build.properties,
- properties/config.properties,
- properties/i18n/bundles/Custom_*.properties.

1.4.2 Déploiement en quick-start

Indiquez simplement le déploiement en quick-start de cette façon :

```
quick-start=true
```

Toutes les autres propriétés sont optionnelles. Les propriétés `tomcat.host`, `tomcat.port` and `tomcat.shutdown-port` peuvent être utilisées pour configurer le Tomcat embarqué (valeurs par défaut ci-dessous).

```
tomcat.port=8080
tomcat.shutdown-port=8009
tomcat.host=localhost
```

> **Lors d'un déploiement en quick-start, l'application est déployée dans le dossier `deploy`. Un serveur Tomcat est installé automatiquement et configuré pour pointer vers le contexte de l'application.**

1.4.3 Déploiement en portlet

Indiquez le déploiement en portlet :

```
deploy.type=portlet
```

Indiquez où l'application doit être déployée pour être exécutée par le portail (typiquement dans le dossier `webapps` d'un portail existant) :

```
deploy.home=/usr/local/uPortal/webapps/esup-geisha
```

Ce chemin doit être utilisé par l'administrateur du portail pour configurer le context du portail. Typiquement dans le fichier de configuration `tomcat/conf/server.xml` :

```
   <Host name="localhost"
       appBase="webapps" unpackWARs="true"
       autoDeploy="true" xmlValidation="false"
       xmlNamespaceAware="false">
       <Context path="/esup-geisha"
       docBase="/usr/local/uPortal/webapps/esup-geisha"
       crossContext="true" reloadable="true">
         <Manager pathname="" />
       </Context>
   </Host>
```

La suite du manuel suppose que le dossier de déploiement s'appelle **esup-geisha**. Si vous le nommez autrement, adaptez à votre cas et configurez les variables suivantes en les ajoutant dans le fichier **properties/config.properties** :

```
mediaPath.portlet=/esup-geisha/media
apiPath.portlet=/esup-geisha/api
```

L'application est constituée d'une interface javascript qui utilise des requêtes AJAX. Si un utilisateur laisse le portail et l'application Esup-Geisha ouvert trop longtemps, sa session est invalidée. En revenant, s'il fait une action dans l'interface, la requête AJAX renvoie une erreur lui indiquant qu'il doit se reconnecter. Cependant par défaut, une session sera créée et le JSESSIONID sera transmis côté client dans un cookie avec comme chemin celui de la servlet qui traite ces requêtes AJAX. Après reconnexion, l'utilisateur aura donc une session pour le portail dont l'identifiant sera transmis dans un cookie avec la racine comme chemin, et une session différente pour cette servlet, ce qui pose problème.

Pour éviter cela, il faut configurer le serveur tomcat avec le paramètre "emptySessionPath" à vrai :

```
<Connector port="8009" enableLookups="false" redirectPort="8443" protocol="AJP/1.3"
emptySessionPath="true" />
<Connector port="8443" enableLookups="false" protocol="AJP/1.3" scheme="https"
emptySessionPath="true" />
```

1.4.4 Déploiement en servlet

Indiquez le déploiement en servlet :

```
deploy.type=servlet
```

Indiquez où l'application doit être déployée pour être exécutée par le conteneur de servlet. Typiquement, dans le dossier `webapps` d'un Tomcat existant :

```
deploy.home=/usr/local/tomcat/webapps/esup-geisha
```

Ce chemin doit être utilisé pour configurer le contexte de la servlet. Typiquement dans le fichier de configuration `tomcat/conf/server.xml` :

```
   <Service name="Catalina">
       <Connector     port="8080"
       maxHttpHeaderSize="8192" maxThreads="150"
       minSpareThreads="25" maxSpareThreads="75"
       enableLookups="false" redirectPort="8443"
       acceptCount="100" connectionTimeout="20000"
       disableUploadTimeout="true" emptySessionPath="true" />
       <Engine name="Catalina" defaultHost="helpdesk.domain.edu">
           <Host name="helpdesk.domain.edu"
               appBase="webapps" unpackWARs="true"
           autoDeploy="true" xmlValidation="false"
           xmlNamespaceAware="false">
           <Context
           path="" docBase="/usr/local/tomcat/webapps/esup-helpdesk"
           crossContext="true" reloadable="false">
                   <Manager pathname=""
               className="org.apache.catalina.session.StandardManager" />
               </Context>
           </Host>
       </Engine>
   </Service>
```

1.5 Fichiers de style (CSS)

Les fichiers de style sont automatiquement inclus par l'application en mode *servlet* ou *quick-start*. En mode **portlet**, l'inclusion des fichiers de style doit se faire au niveau du portail. Ajoutez dans un des fichiers CSS du portail :

```
@import "/esup-geisha/media/extjs/css/ext-all.css";
@import "/esup-geisha/media/esup-geisha/esup-geisha.css";
```

1.6 Administration

Le déploiement de l'application s'effectue par la commande **ant deploy**.

Dans les modes portlet et servlet, l'application est démarrée par le serveur Tomcat correspondant. Dans le mode quick-start, le serveur Tomcat embarqué peut être contrôlé par les commandes **ant start**, **ant stop** ou **ant restart**.

2 Configuration

2.1 Préambule

Les fichiers de configuration sont situés dans le dossier **properties**, répartis par domaine. Toutefois, le fichier **properties/config.properties** concentre les principaux paramètres. La modification de ce seul fichier doit suffire dans la majorité des cas.

Pour répondre à des besoins particuliers, il est possible de modifier les autres fichiers. Afin de répercuter ces modifications d'une version à une autre, il faut renseigner les fichiers modifiés dans le paramètre custom.recover.files du fichier **build.properties** (cf.1.4.1 Propriétés communes).

2.2 Paramétrage fonctionnel

Plusieurs fonctionnalités ont été développées pour répondre aux demandes de différents établissements. Leurs utilisations sont configurées par les paramétrages suivants.

- Désactivation globale l'affichage des compteurs globaux dans la partie « saisie du service ».

```
domain.saisie.compteurs=false
```

- Désactivation partielle de l'affichage de certains compteurs pour une population spécifique (masque).

```
# liste, séparée par des virgules, des compteurs affichés par le masque.
# Voir ci-dessous les types de population/contrat concernés.
# Les compteurs possibles sont :
# - serviceAssure : Service assure.
# - serviceCompta : Service comptabilise.
# - serviceStatut : Service statutaire.
# - serviceDu     : Service du.
# - hc            : Heures complementaires.
# - hcNonPayables : Heures complementaires non payables.
# - hcPayables    : Heures complementaires payables.
# - limite        : Limite d'heures complementaires payables (indique à
#                   l'utilisateur qu'il dépasse cette limite si
#                   hcPayables est présent).

domain.masqueCompteurs=serviceAssure


# liste, séparée par des virgules, de types de population/contrat pour
# lesquels les compteurs globaux doivent être masqué.
#
# Les types de population/contrat doivent correspondre à une valeur
# retournée par les procédures stockées dans GEISHA :
# pks_individu.info_type_population(no_individu, d_deb_annee_univ, 'C')
# pks_individu.info_type_contrat_travail(:individu, :annee, 'C')

domain.typesPopMasqueCompteurs=SB,SD,SP
domain.typesCtrMasqueCompteurs=CU,AC,AD,AE,AH,AI
```

- Autoriser ou non la saisie d'une entrée libre pour les champs formation et/ou discipline.

```
domain.formation.libre=true
domain.discipline.libre=false
```

- Ouvrir les parties demandes de dépassement de limite d'HC et validation des services.

```
domain.demande=true
domain.validation=true
```

- Autoriser ou non l'initialisation du service actuel par celui de l'année précédente.

```
domain.saisie.initSAN=true
```

Pour éviter de devoir redéployer l'application à chaque modification d'un de ces paramètres, il est possible de redéfinir leurs valeurs via l'interface d'administration. Cette interface permet également de définir, si besoin, une date d'ouverture et un date de fermeture de la saisie des services. Si une date d'ouverture est définie, la saisie des services sera bloquée avant cette date. Et inversement, si une date de fermeture est définie, la saisie des services sera bloquée après cette date.

2.3 Accès aux bases de données

2.3.1 Base de données de l'application

Une base de données est utilisée pour stocker des informations relatives à l'application seule (donc indépendante de GEISHA) tels que des préférences des utilisateurs, les droits d'administration, etc... Par défaut, l'application dispose de connecteurs MySql et Oracle. Avec MySql, la base doit être de type InnoDB.

La tâche ant **init-data** crée toutes les structures (tables) de la base de données mais la base doit exister. L'administrateur a la charge de créer cette base et de s'assurer que l'utilisateur a les droits d'accès nécessaires.

JDBC. L'accès à la base de données peut être géré directement par l'application. Indiquez pour cela l'**url** d'accès à la base ainsi que le **username** et le **password** pour l'authentification.

```
hibernate.connection.jdbc.url=jdbc:mysql://localhost/esup_geisha
hibernate.connection.jdbc.username=admin
hibernate.connection.jdbc.password=secret
```

JNDI. L'accès à la base de données peut également être géré par le serveur Tomcat. Indiquez pour cela le nom du pool de connexion utilisé (**datasource**). L'utilisation du JNDI est recommandé en production pour des raisons de performance. Il permet de plus aux administrateurs de surveiller la charge liée aux accès à la base de données avec des outils tels que LambdaProbe.

```
hibernate.useJndi=true
hibernate.connection.jndi.datasource=jdbc/esup_geisha
```

Configurez également le pool de connexion Tomcat dans le contexte de l'application :

```
 1 <Resource
 2     name="jdbc/esup_geisha"
 3     auth="Container"
 4     type="javax.sql.DataSource"
 5     username="admin" password="secret"
 6     driverClassName="com.mysql.jdbc.Driver"
 7     url="jdbc:mysql://localhost/esup_geisha"
 8     maxActive="100" maxIdle="10" maxWait="10000"
 9     removeAbandoned="true" removeAbandonedTimeout="60"
10     logAbandoned="true" />
```

2.3.2 GEISHA

La configuration à la base GEISHA se fait de la même manière, pour un connecteur oracle.

```
hibernate.geisha.connection.jdbc.url=jdbc:oracle:thin:@localhost:port:SID
hibernate.geisha.connection.jdbc.username=admin
hibernate.geisha.connection.jdbc.password=secret

hibernate.geisha.useJndi=false
hibernate.geisha.connection.jndi.datasource=jdbc/geisha
```

Si le driver oracle est correctement installé, l'utilisation du JNDI est possible en configurant le pool de

connexion Tomcat dans le contexte de l'application.

```
<Resource
    name="jdbc/geisha"
    auth="Container"
    type="javax.sql.DataSource"
    username="admin" password="secret"
    driverClassName="oracle.jdbc.driver.OracleDriver"
    url="jdbc:oracle:thin:@localhost:port:SID"
    maxActive="100" maxIdle="10" maxWait="10000"
    removeAbandoned="true" removeAbandonedTimeout="60"
    logAbandoned="true" />
```

Remarque : un problème a été observé avec l'utilisation d'un pool JNDI pour la connexion à la base GEISHA. Pour une raison inconnue, les éventuelles décimales du volume horaire ne sont pas correctement enregistrées en base lors de la saisie d'un service. Par conséquent, il est vivement déconseillé d'utiliser un pool JNDI pour la base GEISHA.

2.3.3 Connecteur JDBC

Esup-Geisha embarque par défaut un connecteur MySql et un connecteur Oracle pour Java. Pour utiliser un autre connecteur :

- ajoutez le connecteur (fichier jar) dans le dossier **webapps/WEB-INF/lib**,
- pour assurer la récupération de ce fichier lors d'un changement de version de l'application, renseignez ce fichier dans le paramètre custom.recover.files du fichier **build.properties** (1.4.1 Propriétés communes),
- surchargez les paramètres indiquant le driver JDBC et le dialect Hibernate à utiliser :

```
hibernate.connection.driver_class=com.mysql.jdbc.Driver
hibernate.dialect=org.hibernate.dialect.MySQLInnoDBDialect
```

ou pour la base GEISHA,

```
hibernate.geisha.connection.driver_class=oracle.jdbc.driver.OracleDriver
hibernate.geisha.dialect=org.hibernate.dialect.Oracle10gDialect
```

Par exemple, pour une base Postgresql (version jdbc3) :

```
hibernate.connection.driver_class=org.postgresql.Driver
hibernate.dialect=org.hibernate.dialect.PostgreSQLDialect
```

Adaptez également l'url d'accès à la base :

```
hibernate.connection.jdbc.url=jdbc:postgresql://localhost:5432/esup_geisha
```

2.3.4 Compatibilité validation des services / demande de dépassement

Les fonctionnalités de validation des services et de demande de dépassement nécessitent des modifications de la base GEISHA qui n'ont pas forcément encore été proposées ou pas encore déployées dans votre établissement. Dans ce cas, pour assurer la compatibilité du canal, il faut lui préciser le paramètre suivant :

```
geisha.version=1
```

2.4 Authentification

En mode servlet ou quick-start, l'authentification se fait via un serveur CAS.

```
cas.url=https://cas.domain.edu
```

En mode portlet, l'authentification se fait via le portail. L'identifiant de l'utilisateur est récupéré de l'attribut **auth.portal.uidAttribute** défini par défaut à "uid".

```
auth.portal.uidAttribute=uid
```

2.5 Initialisation

La tâche ant **init-data** initialise les données en enregistrant un administrateur de l'application dans la table des utilisateurs. Renseignez son identifiant LDAP dans le paramètre suivant :

```
init.firstAdministratorId=paubry
```

2.6 Annuaire LDAP

La configuration de l'annuaire LDAP, d'où seront récupérées les informations sur les utilisateurs, se fait classiquement en indiquant l'**url** d'accès au LDAP ainsi que d'éventuels **username** et **password** pour l'authentification (laisser vide pour un accès anonyme).

```
ldap.url=ldap://ldap.domain.edu:389
ldap.userName=
ldap.password=
```

Précisez également la **base** et le **chemin** d'accès au répertoire des utilisateurs.

```
ldap.base=dc=domain,dc=edu
ldap.dnSubPath=ou=people
```

Indiquez ensuite les attributs du LDAP renseignant l'**identifiant LDAP**, le **nom** à afficher, l'adresse **e-mail** et l'**identifiant Harpège** de l'utilisateur.

```
ldap.uidAttribute=uid
ldap.displayNameAttribute=displayName
ldap.emailAttribute=mail
ldap.geishaIdAttribute=supannEmpId
```

Enfin, pour une éventuelle recherche d'utilisateur, vous pouvez préciser l'attribut sur lequel portera la recherche (**searchAttribute**) ainsi que les attributs qui seront affichés dans le résultat de la recherche (**searchDisplayedAttributes**).

> Indiquez dans la liste des attributs (**ldap.attributes**) tous les attributs qui doivent être récupérer du LDAP, donc en particulier les attributs renseignés dans les paramètres précédents.

```
ldap.searchAttribute=cn
ldap.attributes=cn,displayName,employeeType,department,homeDirectory,supannEmpId
ldap.searchDisplayedAttributes=cn,displayName,employeeType,department
```

Vous pouvez votre configuration de l'accès à l'annuaire LDAP en utilisant la tâche ant **test-ldap**.

2.7 Envoi de mails

Configurez l'accès au serveur SMTP en précisant le **host** et le **port** (25 par défaut). Si l'envoi de mail nécessite une authentification, préciser un login et un mot de passe, sinon laisser vide.

```
smtp.host=smtp.domain.edu
smtp.port=25
smtp.user=
smtp.password=
```

L'encodage des mails est défini par défaut à "utf-8" mais vous pouvez changer cette valeur. Précisez également l'adresse e-mail ainsi que le nom qui apparaitront comme expéditeur des e-mails envoyés.

```
smtp.charset=utf-8
smtp.fromEmail=esup-geisha@domain.edu
smtp.fromName=ESUP-Portail Geisha Web
```

Lorsque vous testez l'application, il est conseillé d'intercepter les e-mails pour ne pas spammer les utilisateurs. Par défaut, le bean activant cette interception est **smtpIntercept**. Précisez alors l'adresse mail et le nom du destinataire qui recevra ces e-mails interceptés (le destinataire originale est indiqué dans le mail envoyé).

```
smtp.interceptBean=smtpIntercept
smtp.interceptEmail=maintainer@domain.edu
smtp.interceptName=Maintainer
```

> En production, définissez cette propriété à **null** pour désactiver l'interception des e-mails (commenter la ligne ne suffit pas).

```
smtp.interceptBean=null
```

Vous pouvez tester votre configuration pour l'accès au serveur SMTP en utilisant la commande **ant test-smtp**. Précisez pour cela l'adresse et le nom du destinataire de l'e-mail envoyé durant ce test.

```
smtp.testEmail=maintainer@domain.edu
smtp.testName=Maintainer
```

2.8 Rapports d'exception

Si des erreurs inattendues interviennent durant l'utilisation de l'application, un rapport d'exception est présenté à l'utilisateur. Ce rapport est également enregistré dans le fichier de log. Pour être prévenu automatiquement de cette erreur, vous pouvez recevoir ce rapport par e-mail. Pour cela, indiquez l'**adresse** du destinataire de ces rapports d'exception. Laissez vide pour ne pas recevoir ces rapports.

```
exceptionHandling.email=maintainer@domain.edu
```

2.9 Fuseau horaire

Indiquer le fuseau horaire utilisé sur de votre serveur.

```
timezone=Europe/Paris
```

2.10 Gestion des logs

La gestion des logs se fait via la librairie log4j. Les principaux paramètres sont :

- le **niveau** des logs enregistrés,
- la **sortie** utilisée : par exemple, *stdout* pour la sortie standard, *file* pour un fichier,
- le **patern** des enregistrements.

```
log.level=WARN
log.output=stdout
log.pattern=%d %p [%c] - %m%n
```

Si vous souhaitez enregistrer les logs dans un fichier, certaines valeurs sont définies par défaut :

- le **fichier** utilisé (chemin + nom),
- la taille maximale du fichier,
- le nombre maximum d'archives conservées.

```
log.file=esup-geisha.log
log.maxFileSize=5MB
log.maxBackupIndex=3
```

2.11 Gestion du cache

Les fichiers utilisés par le cache sont stockés dans le dossier défini par le paramètre **cache.path**. Les noms des fichiers peuvent ne pas être spécifique à l'application. Il est donc conseillé d'indiquer un chemin particulier pour l'application afin d'éviter d'éventuels conflit au niveau du cache, notamment avec d'autres applications basées sur le framework esup-commons.

```
cache.path=/tmp/esup-geisha/cache
```

3 Personnalisation

> **N.B. :** Tous les fichiers modifiés pour la personnalisation de l'application doivent être renseigné dans le paramètre custom.recover.files du fichier build.properties (cf.1.4.1 Propriétés communes) afin d'être récupérés lors d'une mise à jour de l'application (cf.4 Mise à jour).

3.1 L'accès aux bases de données

L'accès aux bases de données repose sur Hibernate. De nombreux paramètres peuvent être personnalisé dans les fichiers de configuration :

- /properties/dao/hibernate/hibernate-jdbc.cfg.xml,
- /properties/dao/hibernate/hibernate-jndi.cfg.xml,
- /properties/dao/geisha/hibernate-jdbc.cfg.xml,
- /properties/dao/geisha/hibernate-jndi.cfg.xml.

Reportez-vous sur la documentation d'Hibernate pour connaitre la signification des différents paramètres.

3.2 Les textes

Les textes utilisés dans l'application sont définis dans différents fichiers. Cela concerne aussi bien les textes affichés dans l'interface, les messages d'erreurs envoyés par le serveur ou les patrons de e-mails envoyés.

Les fichiers suivant définissent une partie des textes utilisés :

- `/properties/i18n/bundles/Commons_xx.properties` : textes utilisés par le framework esup-commons,
- `/properties/i18n/bundles/Messages_xx.properties` : textes spécifiques à l'application.

Ces fichiers ne doivent pas être modifiés. Tous les textes correspondants peuvent être surchargés dans les fichiers de personnalisation `/properties/i18n/bundles/Custom_xx.properties` (un par langue). Ces fichiers sont récupérés automatiquement par la commande **ant recover-config**. Il est inutile de les ajouter au paramètre custom.recover.files du fichier build.properties (cf.1.4.1 Propriétés communes).

D'autre part, l'application utilise pour l'interface la librairie javascript ExtJs. Par défaut l'application utilise le fichier de langue français fourni avec l'application. De nombreux fichiers de langue sont fournis avec cette librairie mais normalement, tous les textes utilisés dans l'interface sont définis dans les fichiers spécifiques à l'application situé dans le dossier **/webapp/media/src/geisha/locale**. Suivant la configuration du navigateur de l'utilisateur, l'interface sera en français ou en anglais. Vous pouvez personnaliser ces fichiers à votre convenance mais ces fichiers entrent dans le cadre de la personnalisation des sources javascripts. Reportez-vous à la section 3.4 Les sources javascript pour vous assurez que vos modifications sont correctement pris en compte.

3.3 Le style

3.3.1 ExtJs

La quasi totalité du style de l'application provient de la librairie ExtJs. Nous utilisons actuellement la **version 3.1.1** de cette librairie. Il est possible d'utiliser des thèmes proposés par la communauté ou d'en créer un soi-même mais assurez-vous de la compatibilité avec la version de la librairie. Un thème gris officiel est disponible et semble complet. Un thème « accessibilité » officiel est également disponible. Je le laisse donc dans le package mais je n'en garantie nullement la fiabilité.

Pour utiliser le thème gris proposé dans le package, il faut :

- en mode *servlet* ou *quick-start* : dans le fichier de configuration **/properties/tags/tags.xml**, au niveau de la propriété **stylesheets**, supprimez de la liste le fichier **extjs/css/ext-all.css** et ajoutez à la place les fichiers **ext-all-notheme.css** et **xtheme-gray.css**.

```
        <property name="stylesheets" >
             <description>
                 A list of URLs that will be automatically included in the
                 head part of the output document. Absolute URLs are used
                 as-is, relative URLs are prefixed by property
                 servletMediaPath.
                 Warning: this tag is ignored for portlet installations.
             </description>
             <list>
                 <value>extjs/css/ext-all-notheme.css</value>
                 <value>extjs/css/xtheme-gray.css</value>
                 <value>esup-geisha/esup-geisha.css</value>
             </list>
         </property>
```

- en mode *portlet* : renseignez ces fichiers au niveau des fichiers CSS du portail.

Pour utiliser un thème tiers, ajoutez le fichier du thème choisi (typiquement **xtheme-<nom_du_theme>.css**) dans le dossier `/webapp/media/extjs/css`. Ajoutez également les images associées au thème suivant l'arborescence définie par le thème (généralement dans le dossier `../images/<nom_du_theme>` par rapport au fichier css). Pour plus d'information, voir la documentation à propos des thèmes sur le site d'ExtJs.

Ces fichiers ajoutés au dossier `media` sont automatiquement déployés par la commande **ant deploy**. Toutefois, pour qu'il soit bien pris en compte dans l'application, il faut :

- en mode *servlet* ou *quick-start* : renseignez ces fichiers dans le fichier de configuration **/properties/tags/tags.xml**. Au niveau de la propriété **stylesheets**, supprimez de la liste le fichier **extjs/css/ext-all.css** et ajoutez à la place les fichiers **ext-all-notheme.css** et **xtheme-<nom_du_theme>.css**.

```
        <property name="stylesheets" >
             <description>
                 A list of URLs that will be automatically included in the
                 head part of the output document. Absolute URLs are used
                 as-is, relative URLs are prefixed by property
                 servletMediaPath.
                 Warning: this tag is ignored for portlet installations.
             </description>
             <list>
                 <value>extjs/css/ext-all-notheme.css</value>
                 <value>extjs/css/xtheme-<nom_du_theme>.css</value>
                 <value>esup-geisha/esup-geisha.css</value>
             </list>
         </property>
```

- en mode *portlet* : renseignez ces fichiers au niveau des fichiers CSS du portail.

3.3.2 Esup-Geisha

Quelques suppléments de style sont définis dans le fichier **/webapp/media/esup-geisha/esup-geisha.css**. Cependant, ce fichier est créé à partir du fichier **/webapp/media/ressources/esup-geisha.css** par la commande **ant js-build** (voir la section3.4 Les sources javascript).

3.4 Les sources javascript

L'interface de l'application est développée en javascript. Les sources sont disponibles dans le répertoire **/webapp/media/src**. Ces sources ont été compilées par les développeurs avec la commande **ant js-build**, utilisant l'outil JsBuilder2. L'outil et la commande sont disponibles pour que vous puissiez, au besoin, personnaliser l'interface en modifiant les sources javascript. Cependant, notez que *JsBuilder2* nécessite **java 6**.

La commande **ant js-build** concatène les fichiers sources en suivant le fichier de configuration **/webapp/media/src/esup-geisha.jsb2** pour créer le fichier **geisha-debug.js**. Puis, ce fichier est minifier en utilisant YUI Compressor pour créer le fichier **geisha.js** utiliser en production. Enfin, ces deux fichiers ainsi

que le contenu du dossier **/webapp/media/ressources** sont déployés dans le dossier **/webapp/media/esup-geisha**.

La version minifiée permet d'alléger le fichier. C'est pourquoi ce fichier est utilisé en production. Toutefois, pour tester et déboguer vos modifications, il est conseillé d'utiliser la version **geisha-debug.js**. Pour cela, indiquez le fichier javascript à utiliser dans le fichier de configuration **/properties/tags/tags.xml**, au niveau de la propriété **scripts** :

```
          <property name="scripts" >
               <description>
                   A list of URLs that will be automatically included in the
                   head part of the output document as scripts. Absolute URLs
                   are used as-is, relative URLs are prefixed by property
                   portletMediaPath or servletMediaPath.
               </description>
               <list>
                   <value>functions.js</value>
                   <value>extjs/ext.js</value>
                   <value>esup-geisha/geisha-debug.js</value>
               </list>
           </property>
```

> **N.B. :** Il existe un outil très pratique pour vérifier la qualité de votre code javascript : jslint. Cet outil est également disponible sous la forme d'un plugin pour eclipse.

3.5 Les sources java

Les sources java sont compilés au déploiement de l'application. L'utilisation systématique d'interfaces permet de remplacer facilement l'implémentation d'une ou plusieurs fonctionnalités. Pour cela, il faut remplacer l'ancienne implémentation par la votre dans les fichiers de configuration du dossier **/properties**.

4 Mise à jour

4.1 Récupération des fichiers de configuration

Afin de simplifier la mise à jour de l'application, la commande **ant recover-config** vous permet de récupérer les fichiers de configuration depuis une version précédente. Toutefois, pour cela, il faut que toutes ces versions soient décompressées dans un même dossier et que le nom de dossier résultant de cette décompression (de la forme "esup-geisha-X.Y.Z") ne soit pas modifié.

4.2 Récupérer plus de fichiers

Par défaut, les fichiers récupérés sont les principaux fichiers de configuration :

- build.properties,
- properties/config.properties,
- properties/i18n/bundles/Custom_*.properties.

Comme l'application peut être grandement personnalisée (cf.3 Personnalisation), vous avez pu modifier ou ajouter de nombreux fichiers, que soit au niveau de la configuration, du style ou même des sources de l'application. Pour être récupérés, tous ces fichiers doivent être indiqués dans le paramètre custom.recover.files du fichier build.properties (cf.1.4.1 Propriétés communes).

Si des modifications en profondeur ont été apportées sur l'application, l'administrateur devra s'assurer qu'aucun conflit n'apparait entre ces modifications et la nouvelle version. Au delà d'un certain nombre de modifications, la mise à jour de l'application peut devenir compliquée à cause de ces conflits. Il est alors conseillé de gérer une version de développement avec des outils adaptés.

4.3 Mise à jour de la base de données

Pour assurer la cohérence entre l'application et la structure de la base de données, le numéro de version est stocké en base. Si celui-ci ne correspond pas à la version de l'application déployée, celle-ci est inutilisable et affiche un rapport d'erreur.

Ainsi, lors de la mise à jour de l'application, il est indispensable de mettre à jour également la base de données avec la commande **ant upgrade**.

Ceci concerne la base de données de l'application (base MySql) et non la base GEISHA. L'administrateur doit donc s'assurer que la base GEISHA dont il dispose est compatible avec l'application.