

Normalisation des build.xml et build.properties des canaux



Objectif

Le but de ce document est de définir une norme pour le déploiement de canaux via ANT.

Cédric Champmartin Sébastien Gaudin
Université de Nancy 2

	Dates de modification	
Revision 1.0	2 février 2006	Reprise de tout le document original en docbook (Yohan Colmant)

Nous proposons ici d'uniformiser le déploiement de nos canaux, dans le cadre de développements classiques. Par le terme classique, nous entendons 'canal' et pas 'bibliothèques'. Pour cela, vous trouverez ici les fichiers types à utiliser dans le développement de vos canaux : build.properties, build.xml

1. Les conventions suivies

- Conventions de nommage de package : [Règles de nommage des livrables ESUP](#)
- Répertoires d'installation par défaut d'esup :



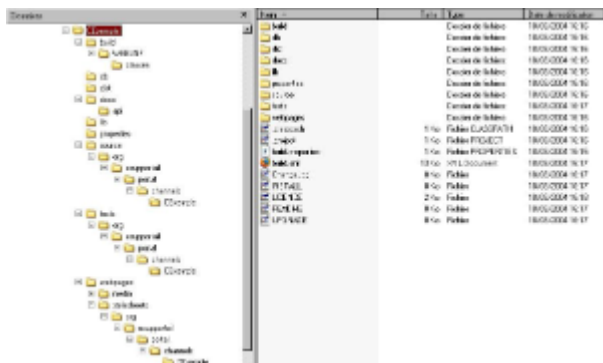
Installation d'un environnement esup-portail de test

2. La démarche d'utilisation

2.1. Point de vue du développeur

2.1.1. Création des fichiers

- Copiez le build.properties et le build.xml dans un répertoire vide.
- Mettez à jour le build.properties avec les informations propres au channel. Il n'est plus nécessaire d'indiquer les bibliothèques utilisées dans le canal, un classpath assez complet est défini dans le build.xml.
- Mettez à jour le nom du package dans le build.xml (première ligne de code du fichier). Ceci sert à reconnaître ce fichier dans Eclipse.
- Avec Eclipse, créer un nouveau projet basé sur ce répertoire.



- Exécutez ant prepare.
- Avec Eclipse, régénérez le projet.
Toute l'arborescence par défaut est créée. Il vous reste à mettre à jour les fichiers INSTALL, UPGRADE, ChangeLog et LICENCE (cliquez sur le lien pour avoir le contenu du fichier "LICENCE" officiel, que vous devrez copier dans votre fichier).

2.1.2. Utilisation courante lors du développement

- Stoppez tomcat.
- Exécutez le ant all, qui va déployer correctement le canal. Pourquoi ant all et pas ant deploy ? La réponse est détaillée un peu plus loin dans ce document.
- Démarrez tomcat.

2.2. Point de vue de l'administrateur d'UPortal

- Lisez le fichier INSTALL à la racine du package.
- Mettez à jour le build.properties avec les chemins locaux d'installation.
- Mettez à jour le fichier de configuration du canal
- S'il le canal contient des librairies vous pouvez les déployer grace à ant.
MAIS si une librairie déjà déployée a le même nom que l'une dans le canal, cette dernière ne sera pas déployée! Pour cela utilisez la commande

```
ant_deploylibs.
```

- Exécutez

```
ant deploy
```

- Déclarez le canal.



Remarque

Nous proposons désormais la target ant deploypubchan, qui permet de déposer les fichiers pubchan dans le répertoire d'uPortal. Exécutez ensuite la commande ant uportal.pubchan du build.xml fourni dans le package esupdev.

- Testez le canal.

3. Détail du build.properties

Ce fichier se compose de 2 parties :

- Les répertoires utiles
- Les informations sur le projet

3.1. Les répertoires

Normalement, sous windows, les trois chemins (Tomcat, uPortal et deploy) sont décrits dans la distribution esup standard , mais libre à chacun de s'organiser comme il veut, donc vous pouvez adapter ici l'installation à votre arborescence locale.

```
tomcat.home   = C:/esupdev/esupdev-2.4/uPortal_2-4-1-quick-start/Tomcat_5-0-28
uportal.home  = C:/esupdev/esupdev-2.4/uPortal_2-4-1-quick-start/uPortal_rel-2-4-1
deploy.home   = C:/esupdev/esupdev-2.4/uPortal_2-4-1-quick-start/webapps/uPortal
```

Deux chemins d'installation doivent être précisés : Là où seront déployées les sources (depuis \$

```
{deploy.home}/WEB-INF/classes/) deploy.tree = org/esupportail/portal/channels
#deploy.tree = fr/univ/portal/channelset
```

là ou seront déployés les fichiers de propriétés (depuis \${deploy.home}/WEB-INF/classes/properties)

```
properties.tree = org_esup
#properties.tree = fr_univ
```

3.2. Informations sur le projet

Le nom du canal va servir à nommer les répertoires spécifiques pour le déploiement du canal (pour les feuilles de style, pour les propriétés ... etc).

```
app.name.home=CExemple
```





Attention

Ce nom est utilisé dans le chemin d'accès au fichier de configuration. Donc si vous le modifiez, pensez à répercuter cette modification dans la classe java faisant appel au fichier de configuration.

La variable `app.version` est utilisée à titre indicatif pour le javadoc et pour dans le nom des fichiers `.tar.gz` et `.zip` générés.

```
app.version=1.0.1-RC-1
```

Le nom du package va servir à nommer les fichiers `.tar.gz` et `.zip` générés.

```
app.name.deploy=esup-canal-exemple
```



Remarque

Dans chacun de ces 2 deux cas, les fichiers sont copiés dans un sous répertoire dont le nom est `$app.name.home`

Les fichiers sont déployés dans les répertoires suivants :

- pour les classes et les feuilles de styles : `$deploy.home\WEB-INF\classes\${deploy.tree}\$app.name.home`
Exemple : `C:\esupdev\esupdev-2.2\Portal_2-2-quick-start\webapps\Portal\WEB-INF\classes\org\esupportail\portal\channels\CExemple`
- pour les fichiers de config : `$deploy.home\WEB-INF\classes\properties\channels\${properties.tree}\$app.name.home`
Exemple : `C:\esupdev\esupdev-2.2\Portal_2-2-quick-start\webapps\Portal\WEB-INF\classes\properties\channels\org_esup\CExemple`
- pour les fichiers statiques : `$deploy.home\media\${deploy.tree}\$app.name.home`
Exemple : `C:\esupdev\esupdev-2.2\Portal_2-2-quick-start\webapps\Portal\media\org\esupportail\portal\channels\CExemple`
- pour les feuilles de style : `$deploy.home\WEB-INF\classes\${deploy.tree}\$app.name.home`
Exemple : `C:\esupdev\esupdev-2.2\Portal_2-2-quick-start\webapps\Portal\org\esupportail\portal\channels\CExemple`



Remarque

Les répertoires en italique dans les exemples ci-dessus sont les répertoires par défaut. Si vous n'utilisez pas ces répertoires, vous pouvez les modifier dans le `build.xml` (les noms sont en début de fichiers).

4. Détail du build.xml

Les "sous-targets" n'ont pas de description. Ceci implique qu'il est possible, sous eclipse, de les masquer.

Voici le détail des targets que vous serez amené à utiliser :

`all` Appel des 3 targets : `clean`, `undeploy` (permet d'effacer de "vieux" fichiers dans l'arborescence), `deploy`. Très utile lors du développement d'un canal.

`prepare` Crée tous les répertoires par défaut dans le canal (docs, tests, source...), peut être utilisé pour obtenir un environnement type de développement.

`clean` Vide les répertoires `build`, `unitBuild` et `dist`.

`undeploy` Supprime tous les éléments (classes, media, properties, stylesheets) mis en place par la target `ant_deploy` mais pas les librairies, ni le fichier correspondant au pubchan

`deploy` Compile et déploie le canal (classes, media, properties, stylesheets) mais pas les librairies, ni le fichier correspondant au pubchan

`compil` Compile le canal dans le `build`

`javadoc` Crée la javadoc

`dist` Crée le jar du canal pour la distribution

`buildtar` Crée le package en `.tar.gz`

`buildzip` Crée le package en `.zip` (zip même si les noms complets des fichiers dépassent 100 caractères)

`junit.init` vide les répertoires nécessaires aux tests junits

`junit.compile` compile les classes junit (par défaut dans le répertoire `buildJunit`)

`junit.start` lance le(s) test(s) junits.



Remarque

- Les fichiers de tests doivent être de la forme :Test*.javaet doivent se trouver dans le répertoire/tests/...du projet pour être pris en compte par cette commande.
- Il se peut que la dernière ligne affichée après l'exécution soit de la forme :
"BUILD FAILED: C:\esupdev\Canaux\CExemple\build.xml:294: Tests failed. Check log and/or reports".
Ceci signifie tout simplement qu'une méthode de l'un des tests unitaires n'a pas été vérifiée.
- Une fois les tests terminés, un rapport est généré. Ouvrez la pageBuildJUnit/rapport/index.html et observez le résultat des tests.
- Pour plus d'informations sur les tests unitaires, allez sur

 cette page

deploylibsPermet de déployer les librairies du canal.MAIS si une librairie déjà déployée a le même nom que l'une dans le canal, cette dernière ne sera pas déployée! Pour cela utilisez la commandeant_deploylibs. Sous eclipse, cette target ne fonctionne pas, car le logiciel ne prend pas en compte l'intervention utilisateur dans la console.

undeploylibsPermet d'enlever les librairies propres au canal.MAIS si une librairie est utilisée pour un autre canal, elle sera tout de même supprimée.Sous eclipse, cette target ne fonctionne pas, car le logiciel ne prend pas en compte l'intervention utilisateur dans la console.

deploypubchanPermet de déposer les fichiers pubchan dans le répertoire d'uPortal. Exécutez ensuite la commandeant_uportal.pubchandubuild.xmlfournit dans le package esupdev.

undeploypubchanPermet d'enlever les fichiers pubchan relatifs au canal du répertoire d'uPortal.

docbook.to.htmlTransforme le docbook en HTML.

 Voir cette documentation plus détaillée.

docbook.to.html.noindexTransforme le docbook en HTML sans générer de table des matières.

 Voir cette documentation plus détaillée.

5. Ce que ce build ne fait pas

- Le déploiement automatique des librairies lors du déploiement du canal.
- La suppression automatique des librairies lors de la désinstallation du canal.
- La création de la base de données spécifique au canal, le chargement des données ainsi que la destruction de cette base.
- La publication du canal.