

1.4 Création d'un projet à partir d'un archetype blank



Bon pour relecture

Sommaire :

- [Création de la structure du projet à partir de l'archetype esup-commons](#)
 - [Création en ligne de commande](#)
 - [Création depuis Eclipse](#)
- [Utilisation du projet](#)

Création de la structure du projet à partir de l'archetype *esup-commons*

Qu'est-ce qu'un archetype Maven ? Dans le contexte de *esup-commons*, un archetype est une structure complète d'un projet à déployer. Toute l'organisation des fichiers, des classes et des projets Eclipse sera mise en place par l'archetype. L'objectif est de proposer une structure standard à tous les développeurs. Le développeur aura "uniquement" à sa charge le remplissage de cette structure.

La structure du projet va être construite à partir d'un archetype maven dont voici les paramètres :

```
archetypeRepository=https://mvn.esup-portail.org/content/repositories/releases
archetypeGroupId=org.esupportail.blank
archetypeArtifactId=esup-blank-archetype
archetypeVersion=numeroVersionArchetype
groupId=domaineMonProjet
artifactId=nomProjet
package=packageMonProjet
version=numeroVersionMonProjet
```

où :

- **numeroVersionArchetype** : numéro de version de l'archetype à utiliser (version 0.2.8 à l'heure où sont écrites ces lignes). Pour connaître le numéro de version de l'archetype, Merci de vous référer au dépôt maven : <https://mvn.esup-portail.org/content/repositories/releases/org/esupportail/blank/esup-blank-archetype/>
- **domaineMonProjet** : nom de domaine du projet à développer. Par exemple : fr.univ pour un projet local ou org.esup-portail pour un projet communautaire
- **nomProjet** : nom de mon projet. Par exemple : portlet-reservation
- **packageMonProjet** : nom du package JAVA correspondant à mon projet. Par exemple : fr.uhp.reservation
- **numeroVersionMonProjet** : numéro de version de mon projet. Par exemple : 1.0-SNAPSHOT
- les autres paramètres ne sont pas à modifier

Il y a 2 méthodes possible pour créer un nouveau projet eclipse à partir de l'archetype blank : Soit directement via l'interface d'Eclipse, soit en ligne de commande puis importation dans Eclipse.



Attention

Nous avons constaté quelques soucis dans la création de projet à partir d'un archetype avec le plugin maven d'eclipse : m2eclipse. Si vous rencontrez des messages d'erreurs indiquant qu'eclipse ne parvient pas à trouver l'archetype (voir erreur ci-dessous) n'hésitez pas à relancer complètement Eclipse.



Pièce jointe inconnue

Toutefois nous vous recommandons d'utiliser la méthode par ligne de commande qui elle fonctionne bien.

Création en ligne de commande

Récupération du projet : se positionner dans le répertoire workspace de Eclipse et lancer la commande suivante :

```
mvn archetype:generate -B -DarchetypeRepository=https://mvn.esup-portail.org/content/repositories/releases
-DarchetypeGroupId=org.esupportail.blank -DarchetypeArtifactId=esup-blank-archetype -
DarchetypeVersion=numeroVersionArchetype
-DgroupId=fr.univ -DartifactId=nomProjet -Dpackage=packageMonProjet -Dversion=numeroVersionMonProjet
```

Une fois l'archetype récupéré, il faut se positionner dans le répertoire parent (là où se trouve le pom.xml), et lancer la commande suivante :

```
mvn eclipse:eclipse
```

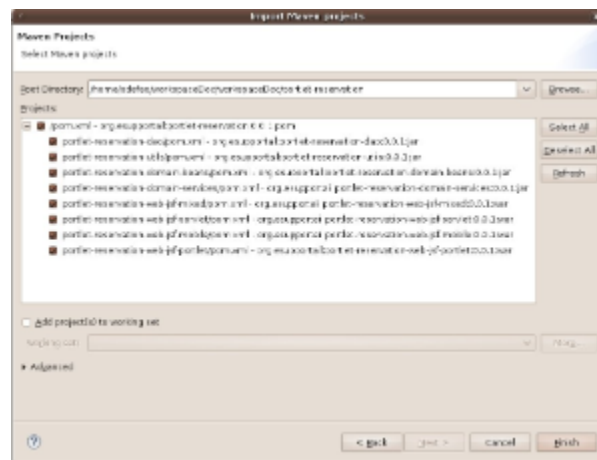
Voilà le résultat dans la console :

```
[INFO] Using as WTP server : Apache Tomcat v8.0
[INFO] Adding default classpath container: org.eclipse.jdt.launching.JRE_CONTAINER
[INFO] File /home/eddefe/workspaceDev/workspaceDev/portlet-reservation/portlet-reservation-web
Additional settings will be preserved, run mvn eclipse:clean if you want old settings t
[INFO] Wrote Eclipse project for "portlet-reservation-web-jef-portlet" to /home/eddefe/workspa
jef-portlet.
[INFO]
[INFO]
[INFO]
[INFO]
[INFO] Reactor Summary:
[INFO]
[INFO] esup-blank ..... SUCCESS [1:18.7m]
[INFO] esup-blank-domain ..... SUCCESS [1:1.66s]
[INFO] esup-blank-domain ..... SUCCESS [0:4.65m]
[INFO] esup-blank-utilities ..... SUCCESS [0:0.41s]
[INFO] esup-blank-domain-services ..... SUCCESS [1:1.89m]
[INFO] esup-blank-web-jef-frontend ..... SUCCESS [1:07.064s]
[INFO] esup-blank-web-jef-services ..... SUCCESS [0:0.89m]
[INFO] esup-blank-web-jef-mobile ..... SUCCESS [0:1.06s]
[INFO] esup-blank-web-jef-portlet ..... SUCCESS [0:4.65m]
[INFO]
[INFO] BUILD SUCCESSFUL
[INFO]
[INFO] Total time: 1 minute 15 seconds
[INFO] Finished at: Mon Feb 20 15:12:44 CET 2011
[INFO] Final Memory: 20M/168M
[INFO]
```

Cette commande va configurer les projets pour que ce soient des projets WTP dans Eclipse. Une fois cette étape terminée, il faut les importer dans Eclipse en tant que projets Maven.

Dans Eclipse : **File => New => Maven => Existing Maven Projects**

Il faut alors choisir le répertoire parent du Blank que l'on vient de récupérer. Eclipse va automatiquement sélectionner tous les modules présents.

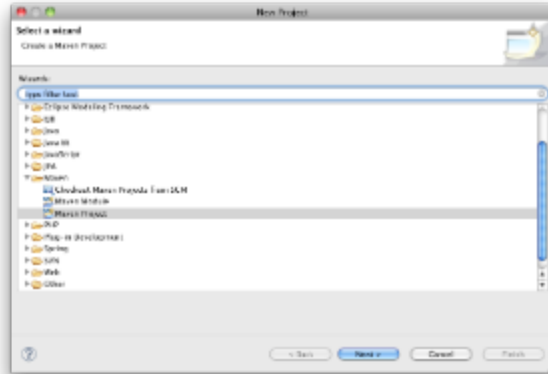


On valide en cliquant sur **Finish**.

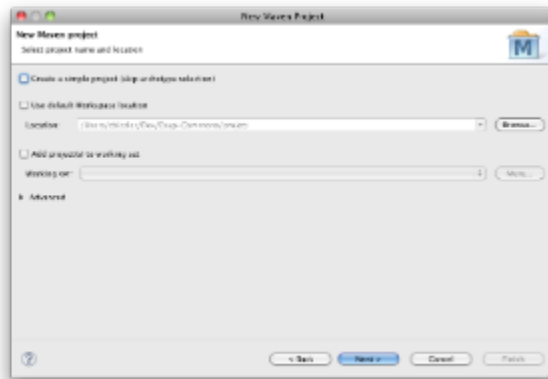
Le projet est maintenant prêt à être utilisé.

Création depuis Eclipse

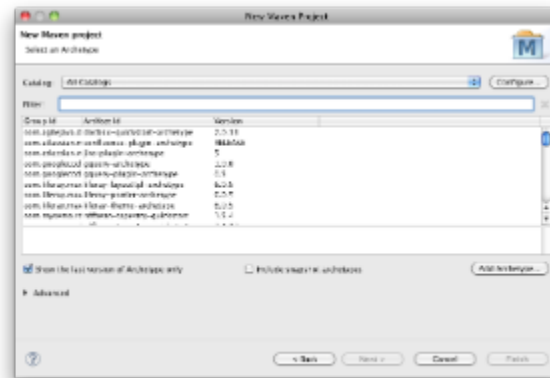
Se rendre dans le menu **File => New => Project...**
Choisir **Maven Project**



Laisser l'emplacement par défaut pour le projet

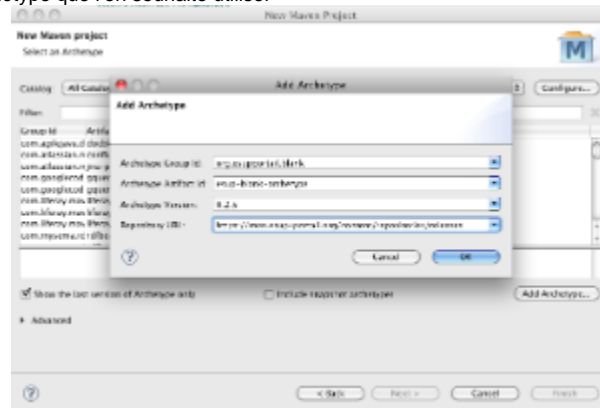


Il n'y a pas d'archétype

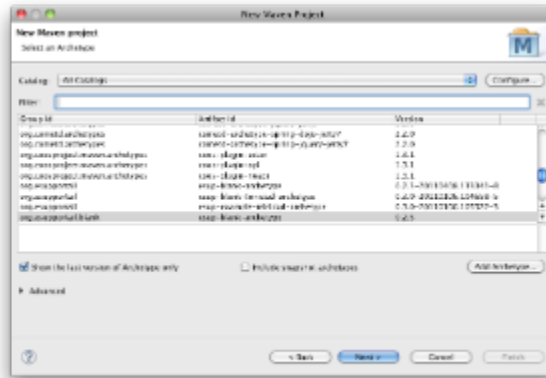


Cliquer sur **Add Archetype**

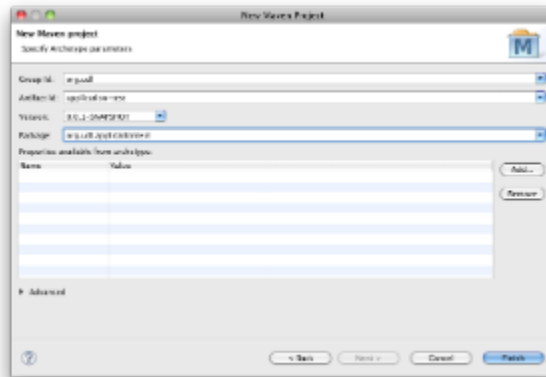
Remplir les champs correspondant à l'archétype que l'on souhaite utiliser



Choisir l'archétype que l'on vient d'ajouter



Créer le projet



Cliquer sur **finish** et le projet est créé avec vos choix.

? Pièce jointe inconnue

Pour préparer un projet, toutes les dépendances doivent être satisfaites. Chaque module Maven dépendant d'un autre, il faut donc compiler chaque module afin de générer les jar qui seront utilisés par les autres modules. C'est le but de la target Install de Maven.

Lancer la tâche Maven via un clic droit sur le projet parent : **Run as => Maven install**

Voilà le résultat dans la console de Eclipse :

```
[INFO] Reactor Summary:
[INFO]
[INFO] spring-blank ..... SUCCESS [0.412s]
[INFO] spring-blank-domain-service ..... SUCCESS [2.452s]
[INFO] spring-blank-dao ..... SUCCESS [0.672s]
[INFO] spring-blank-web ..... SUCCESS [0.494s]
[INFO] spring-blank-domain-service ..... SUCCESS [0.617s]
[INFO] spring-blank-web-jsp ..... SUCCESS [0.146s]
[INFO] spring-blank-web-jsp ..... SUCCESS [4.003s]
[INFO] spring-blank-web-jsp ..... SUCCESS [4.183s]
[INFO] spring-blank-web-jsp ..... SUCCESS [4.003s]
[INFO] BUILD SUCCESS
[INFO]
[INFO] Total time: 20.207s
[INFO] Finished at: Mon Feb 28 14:18:11 CET 2011
[INFO] Final Memory: 5M/183M
[INFO]
```

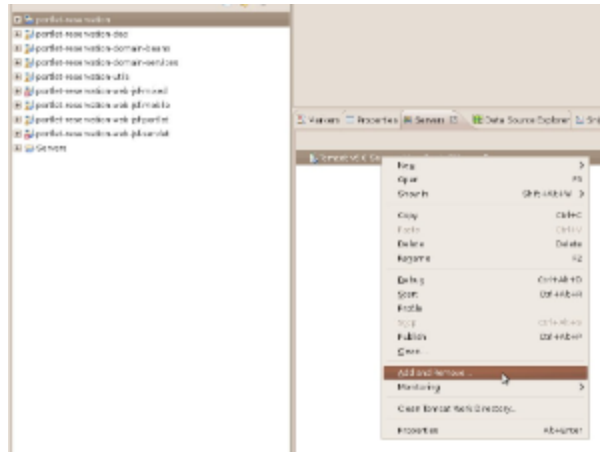
Le projet est maintenant prêt à être utilisé.

Utilisation du projet

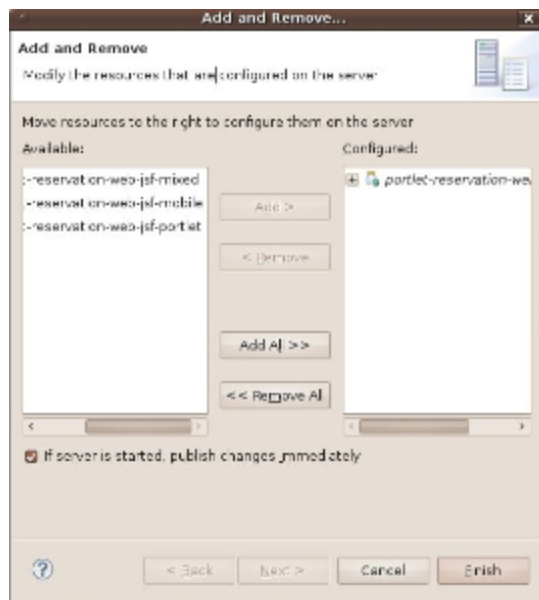
Une fois le projet prêt, vous pouvez supprimer les modules web qui ne vous intéressent pas. Pour rappel, dans cette page : [1.2 Méthodologie de développement](#), vous trouverez toutes les informations pour choisir le module web à garder. Ce n'est pas une obligation de supprimer les modules web que vous ne souhaitez pas utiliser, vous pouvez les garder pour un usage ultérieur. Ils ne poseront pas de problème dans votre travail de développement.

Pour tester votre module et donc votre développement, il faut utiliser les fonctionnalités WTP de Eclipse. Il faut ajouter le module web dans un des serveurs déclarés dans votre Eclipse.

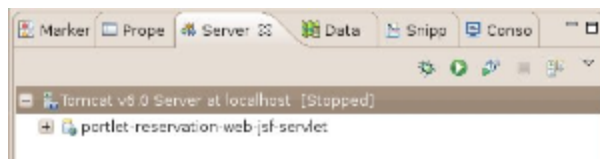
Pour cela, faire un clic-droit sur le serveur que vous souhaitez utiliser, et choisissez "Add and Remove ...".



Vous devez alors choisir le module web sur lequel vous souhaitez travailler.



Et une fois vos choix validés, le contexte de votre module est accessible dans le serveur choisi.



Dans notre exemple, pour y accéder, vous devrez alors taper l'URL : <http://localhost:8080/portlet-reservation-web-jsp-servlet>... Il ne vous reste alors plus qu'à démarrer le serveur.