

# La création d'exécutables en Java

par Baptiste Wicht ([home](#))

Date de publication : Le 22 Septembre 2006

Dernière mise à jour :

Ce tutoriel va vous apprendre à créer un exécutable de votre programme Java. Pour cela, vous allez voir qu'il y a plusieurs manières de faire.

- I - Introduction
- II - Employer un simple script
- III - Créer un Jar
  - III-A - Créer un .jar avec Eclipse
    - III-A-1 - FatJar
  - III-B - Créer un .jar avec Netbeans
  - III-C - Créer un .jar avec JBuilder
- IV - Créer un lanceur de votre programme
  - IV-A - JSmooth
  - IV-B - JExeCreator
  - IV-C - Launch4j
    - IV-C-1 - Plugin Launch4J pour Netbeans
  - IV-D - Exe4j
  - IV-E - JavaExe
- V - Compiler votre programme de manière native
  - V-A - GCJ
  - V-B - Excelsior JET
  - V-C - Toba
  - V-D - Manta
- VI - Remerciements

## I - Introduction

Après avoir créé votre programme, vous avez envie de le partager, mais vous ne savez pas sous quelle forme. Jusqu'à maintenant, vous avez utilisé votre EDI pour le lancement de votre programme, mais vous ne pouvez pas le distribuer ainsi.

Vous allez voir dans ce tutoriel qu'il y a plusieurs manières de créer un exécutable de votre programme Java : un fichier BAT, un JAR, un lanceur ou un exécutable natif.

## II - Employer un simple script

La façon la plus basique pour fournir un exécutable de votre programme, est tout simplement d'employer un simple fichier script qui va se charger de lancer votre application.

Pour cela, on va employer deux formats de script :

- Un Batch. C'est un petit exécutable Windows qui permet de lancer des commandes console.
- Un script Shell. C'est un petit exécutable Linux qui permet de lancer des commandes dans le Shell

Le code est le même pour les deux types de scripts, il suffit d'employer la commande java suivi du nom de la classe main, ou alors javaw suivi du nom de la classe si vous ne voulez pas de fenêtre console.

### Script

```
java MaClasseMain
```

Pour créer un tel exécutable, c'est tout simple, il vous suffit de créer un fichier texte, d'y insérer la ligne citée ci-dessus et enfin de renommer votre fichier en NomDuFichier.bat(ou sh) Il sera ensuite exécutable en double cliquant dessus.

### III - Créer un Jar

Un .jar est tout simplement l'exécutable Java par défaut, c'est l'équivalent du .exe pour le C++. Un jar est utilisable comme n'importe quel exécutable sur votre ordinateur pour peu qu'il possède bien la JVM. Un Jar n'est rien d'autre qu'une archive contenant un fichier Manifeste (MANIFEST.MF) que la JVM va aller lire pour savoir quelle classe lancer et ce qu'il faut inclure dans le Class-Path.

On va donc commencer par créer un fichier manifest :

#### MANIFEST.MF

```
Manifest-Version: 1.0
Main-Class: package.ClasseMain
Class-Path: CheminVersRessource1 CheminVersRessource2 ...
```

Notez qu'il finit par un saut de ligne, ce qui est obligatoire. Ce fichier est placé dans le dossier META-INF à la racine de notre projet. Les packages de l'application sont aussi placés à ce niveau.

Le classPath est tout simplement la liste des fichiers à ajouter avec le programme, des apis externes par exemple. Si vous employez d'autres Jar, vous serez obligé de les renseigner dans le classpath pour que votre programme les trouve.

Ensuite, il faut employer la commande jar pour créer votre jar. Elle s'utilise de la manière suivante :

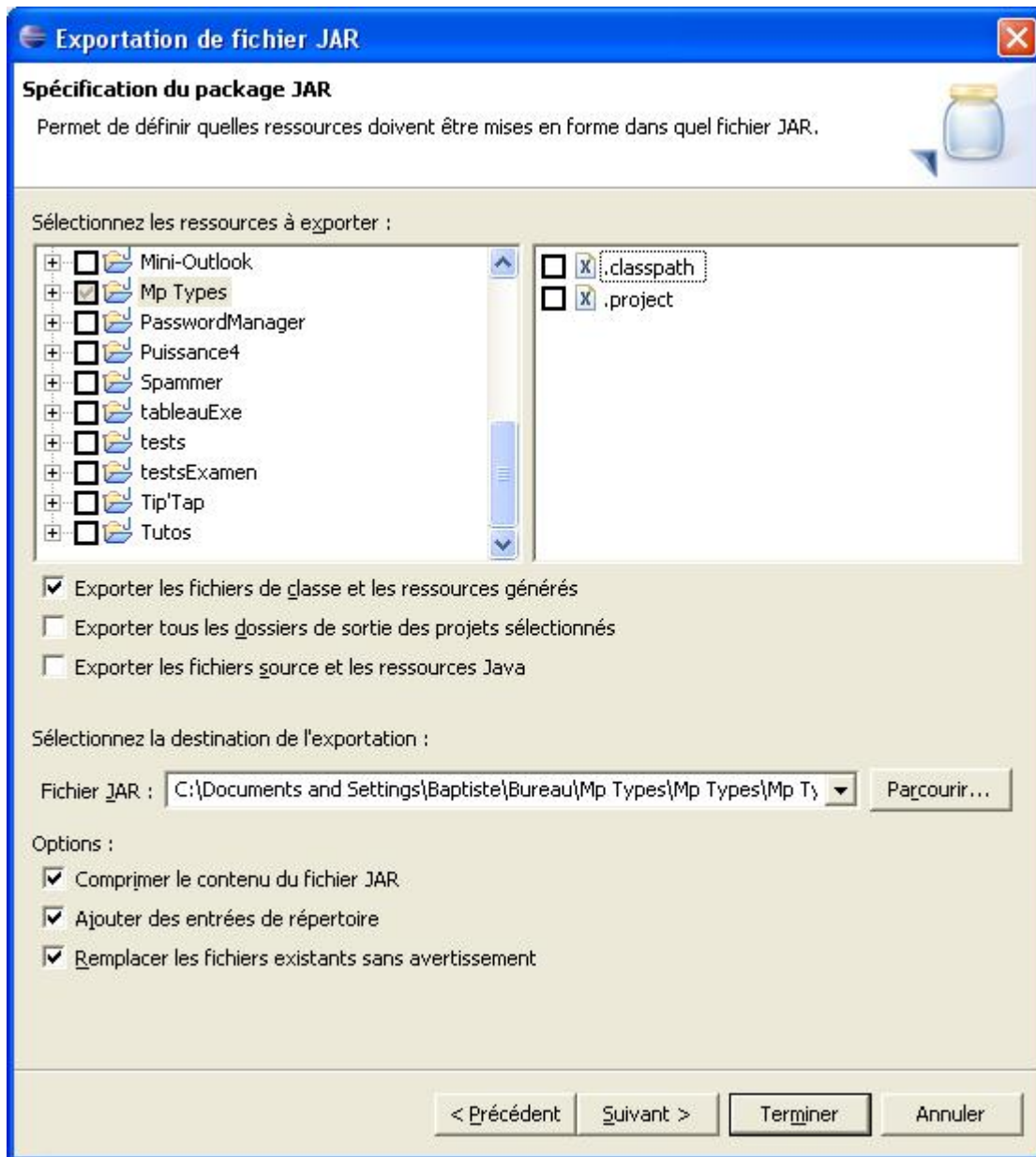
#### Commande Jar

```
jar cvfm CheminDuJar ChdeminDuManifeste
```

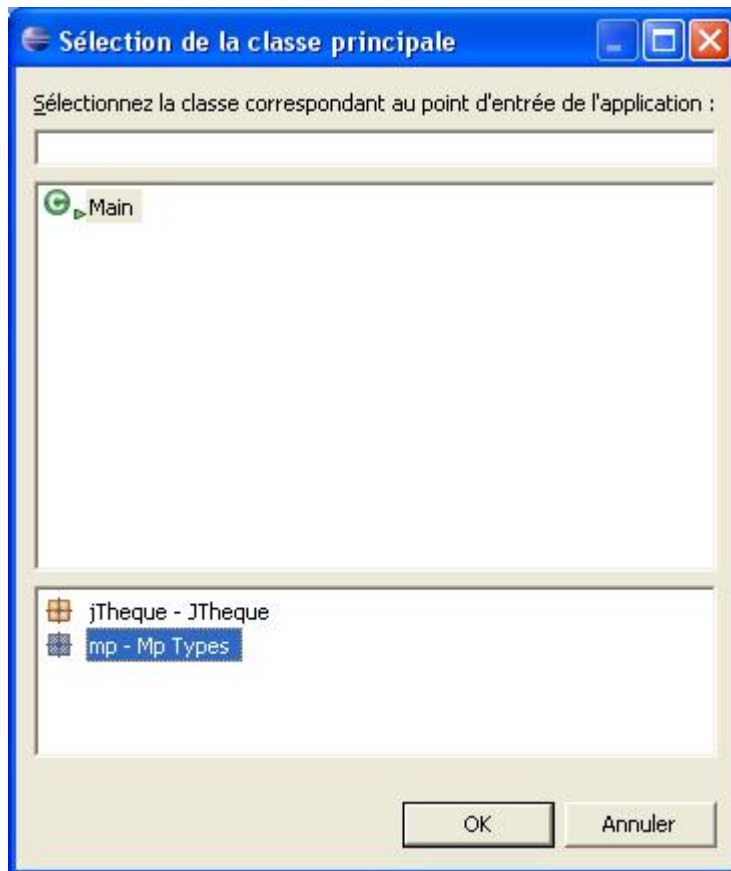
La plupart des Edi permettent de le faire de manière automatisée via une interface graphique, sur laquelle vous n'aurez qu'à remplir quelques options. Ci-dessous, vous trouverez la procédure pour créer un fichier Jar avec les principaux EDI Java.

### III-A - Créer un .jar avec Eclipse

Pour créer un .jar avec Eclipse, il faut cliquer droit sur votre projet, puis exporter et choisir fichier .jar dans la liste. Ensuite, il vous faut renseigner les options d'exportation, qui vont définir si vous voulez mettre les sources dans le Jar, si vous voulez compresser le Jar... Vous devez aussi définir l'emplacement final de ce Jar. Ensuite, il vous faut renseigner la classe Main de l'application à la troisième page de configuration, cliquez sur parcourir et choisissez la classe Main parmi celles proposées.



Choix du projet et des options principales



*Choix de la classe main de votre application*

### III-A-1 - FatJar

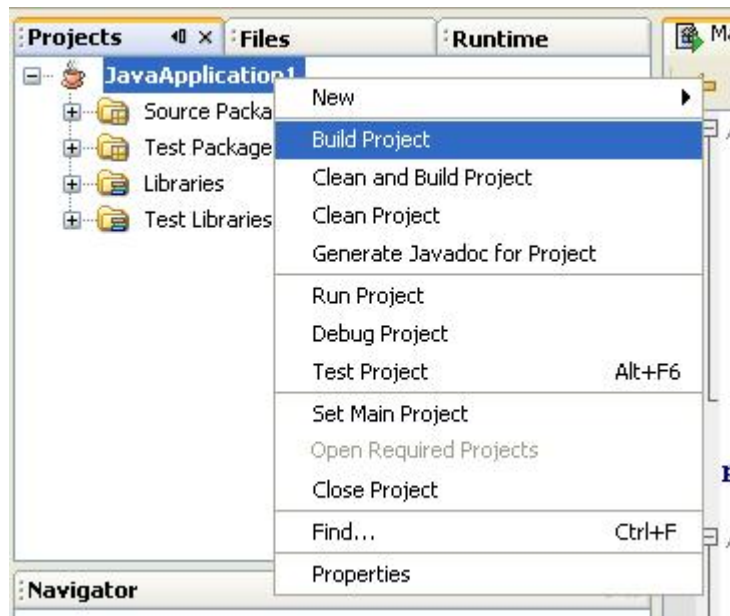
Par défaut, il n'est pas possible d'inclure un Jar à l'intérieur d'un autre, il faut donc passer par le classPath, mais on peut néanmoins le faire grâce au plugin FatJar d'Eclipse, il va créer un nouveau ClassLoader dans le jar pour charger les jar à l'intérieur du principal.

Vous pouvez le télécharger [ici](#).

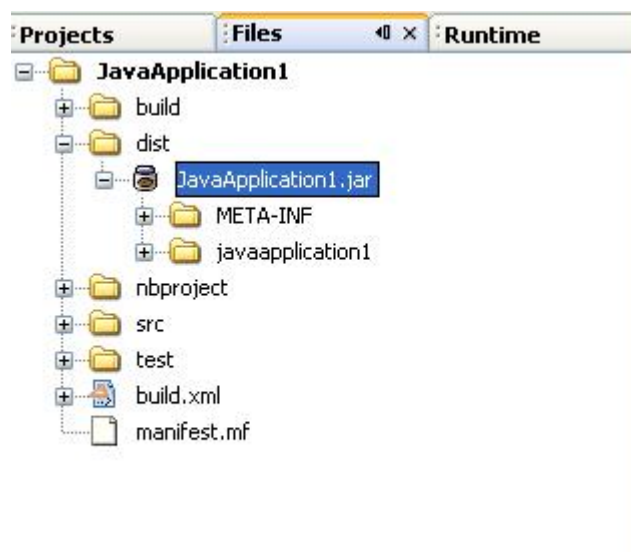
### III-B - Créer un .jar avec Netbeans

Normalement, le jar est créé automatiquement dans le répertoire dist de votre projet à chaque nouvelle compilation.

Si cela n'est pas fait, pour créer un .jar avec Netbeans, il faut cliquer droit sur projet, puis cliquer sur "build Project". Ensuite, le .jar sera automatiquement créé dans le répertoire dist de votre projet.



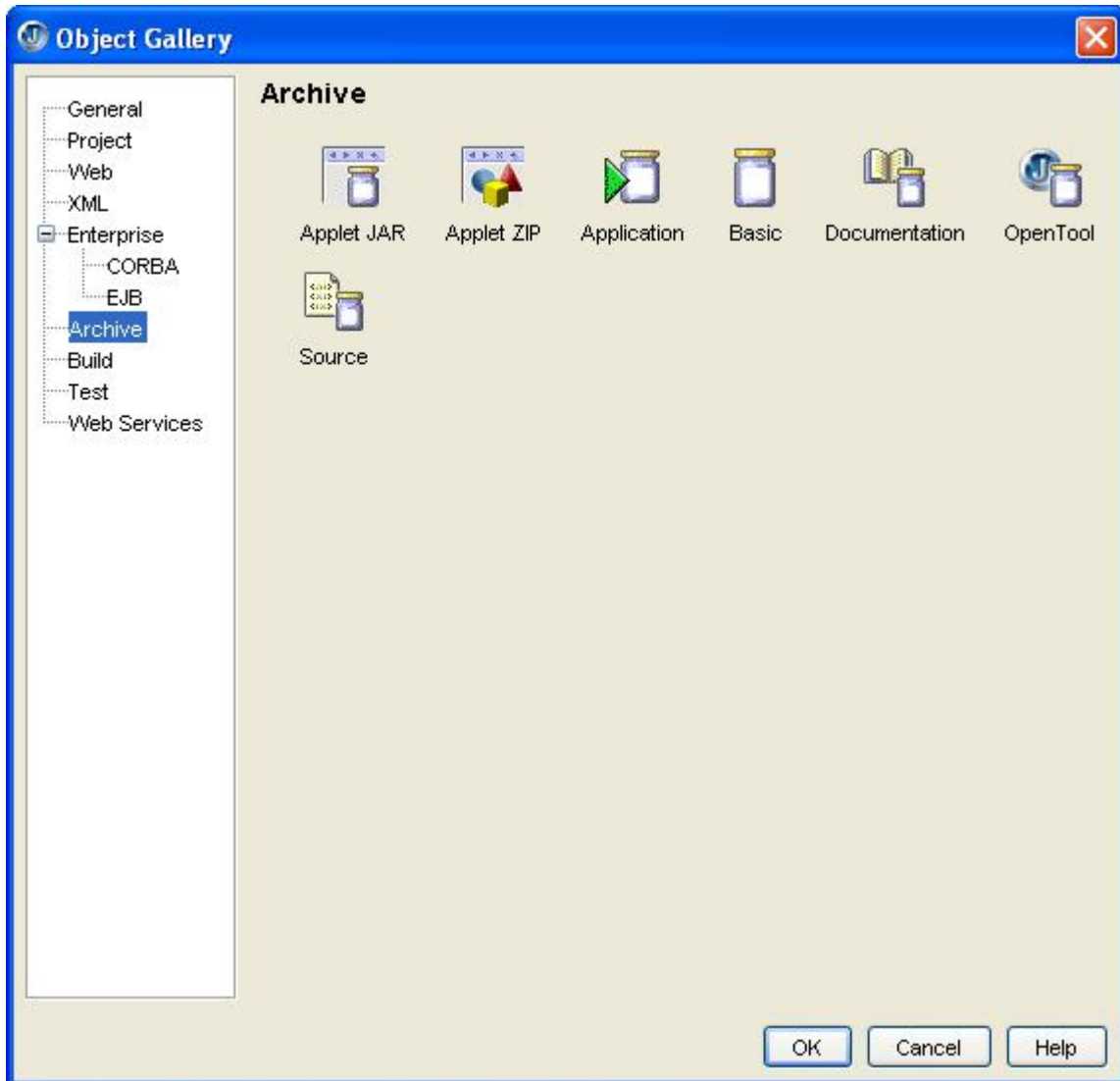
*Build Projet avec Netbeans*



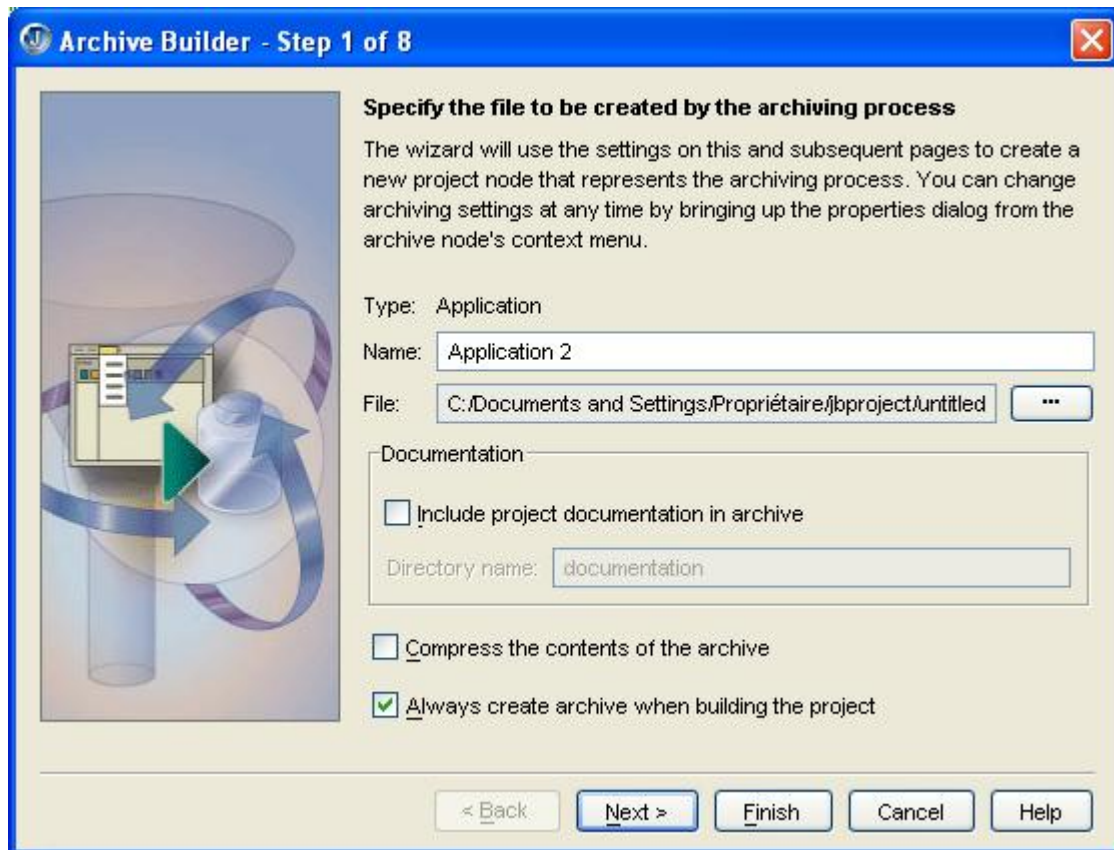
*Jar généré par Netbeans*

### III-C - Créer un .jar avec JBuilder

Pour créer un Jar avec JBuilder, il faut commencer par ouvrir le constructeur d'archives. Ensuite, choisissez "JAR exécutable" dans la liste de formats qui est proposée. choisissez l'emplacement et le nom du Jar, configurez les dépendances, choisissez créer un manifest et choisissez la classe main.



*Création d'une archive avec JBuilder*



*Interface de création de Jar*

## IV - Créer un lanceur de votre programme

Un lanceur est tout simplement un programme natif qui va lancer votre propre programme. Il vous faut donc obligatoirement déjà avoir créé un .jar avant de créer un lanceur. Vous allez me demander à quoi ça sert ? Ca sert tout simplement à rassurer certaines personnes en leur présentant un .exe à la place du .jar. Ca n'a aucun autre avantage sur le .jar, mais aucun désavantage non plus, puisqu'il ne fait que le lancer, cela n'altère aucunement la vitesse de votre programme.

Il y a plusieurs logiciels sur le marché qui permettent de créer des lanceurs de votre programme, je vous en présente quelques uns dans les chapitres qui viennent.

### IV-A - JSmooth

#### Site

Dernière version : **0.9.7**

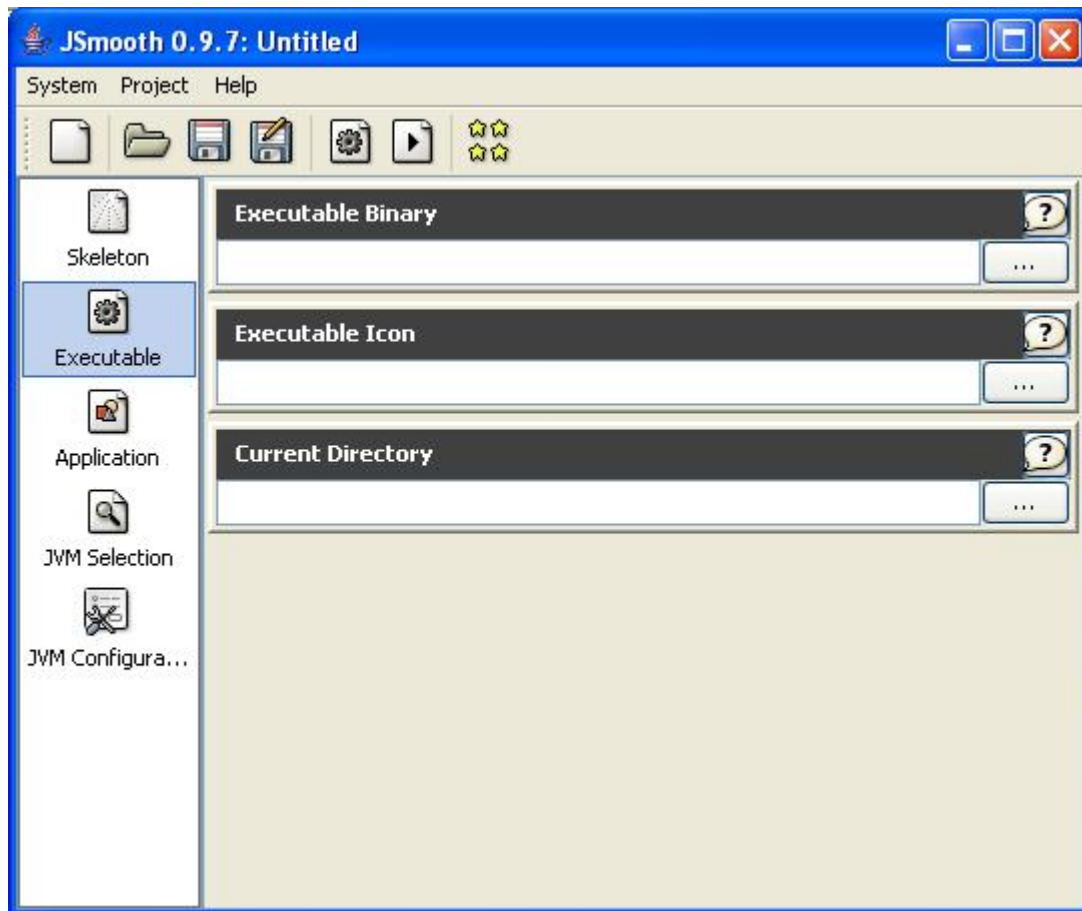
Langue : Anglais

Fonctionnalités :

- S'il n'y a pas de JVM d'installée, le programme va indiquer à l'utilisateur où est-ce qu'il peut en avoir une
- Permet une bonne configuration de la JVM (Memory allocation)

Limitation licence : Aucune (GNU/GPL)

Utilisation : Assez simple, il vous suffit de naviguer dans les différents onglets à gauche et dès que tout est complété, de cliquer sur le bouton build en haut.



Interface de création de JSmooth

## IV-B - JExeCreator

### Site

Dernière version : **1.9.1**

OS supportés : Windows 9x/NT4/2000/XP

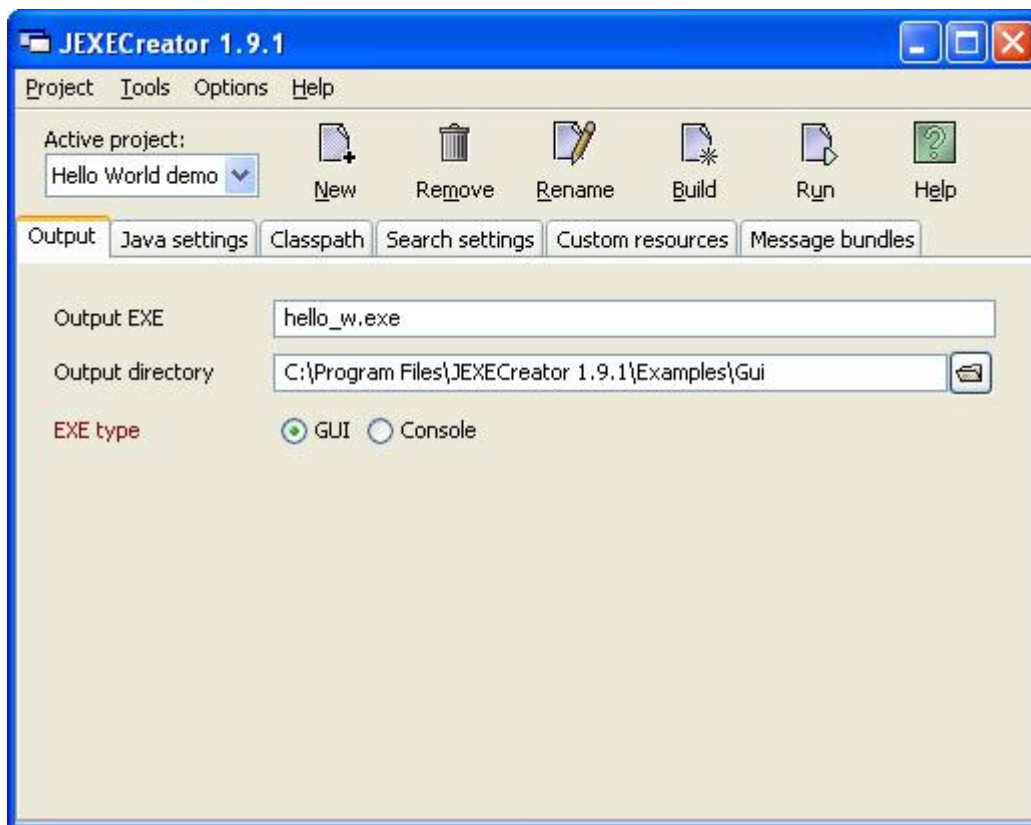
Langue : Anglais

Fonctionnalités :

- S'il n'y a pas de JVM installée, le programme va afficher un message d'erreur avec des informations sur le téléchargement de la JVM
- S'il y a une erreur en cours de programme, une boîte de dialogue avec la trace de l'erreur va s'afficher
- Permet de gérer vos exe sous forme de projet, ainsi vous n'avez plus qu'à ouvrir un ancien projet, mettre à jour les ressources et régénérer le projet pour avoir de nouveau un .exe à jour

Limitation licence : Version d'essai pendant 30 jours

Utilisation : Simple, il vous suffit de remplir les champs de chacun des onglets de l'interface et ensuite de cliquer sur le bouton "Build" pour générer votre exe. Vous pouvez ensuite cliquer sur le bouton "Run" pour tester votre lanceur.



Interface de création de JExeCreator

## IV-C - Launch4j

### Site

Dernière version : **2.1.5**

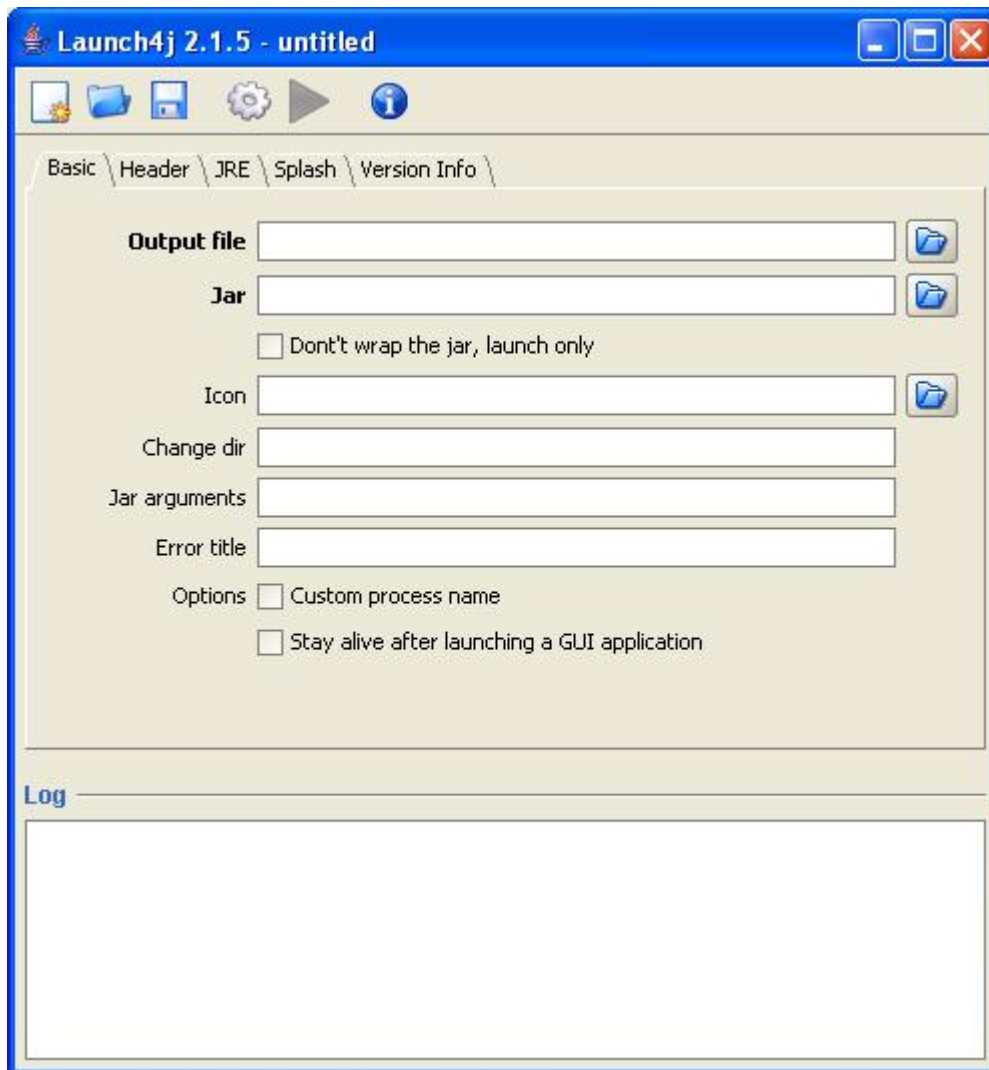
Langue : Anglais

Fonctionnalités :

- Permet d'ajouter un Splash-Screen natif avant le lancement de la JVM avec une image BMP
- Permet de choisir entre une application console et GUI

Limitation licence : Aucune

Utilisation : Très simple, il suffit ici de compléter les champs de chaque onglet un par un. Dès que tout est complété, il vous suffira de cliquer sur "Build Wrapper". Vous pouvez ensuite directement tester le lanceur avec le bouton "test Wrapper"



Interface de création de launch4j

## IV-C-1 - Plugin Launch4J pour Netbeans

Il existe un plugin pour Netbeans qui permet de créer des exécutables natifs directement depuis celui-ci sans passer par un programme externe. Pour télécharger ce plugin, il vous faudra suivre [cette procédure](#).

## IV-D - Exe4j

### Site

Dernière version : **3.1.3**

OS supportés : Windows NT4/2000/XP Linux/Generic Unix/Mac OS x

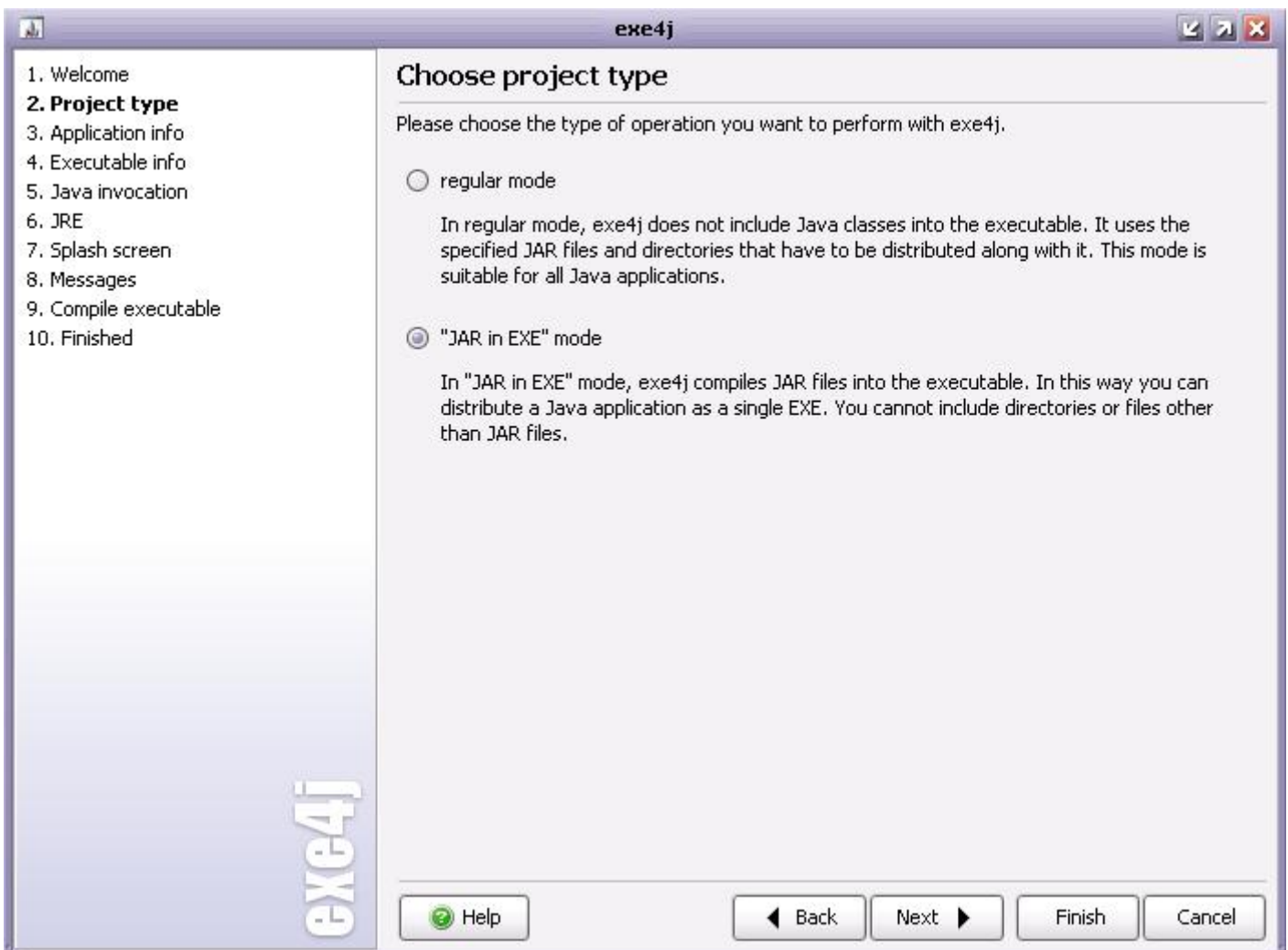
Langue : Anglais

Fonctionnalités :

- Il permet de choisir entre faire une application console ou une application GUI
- Il permet aussi la création simple de services Windows
- Il permet aussi d'inclure des Jar dans des Jar
- Il permet l'affichage d'un Splash-Screen natif pendant le lancement de la JVM
- En cas d'erreur dans le programme, il permet d'afficher une boîte de dialogue avec lesdites erreurs
- Il permet de choisir entre faire un simple lanceur ou alors mettre le .jar dans le .exe

Limitation licence : Si vous n'avez pas payé la licence, un message va s'afficher au démarrage de l'application pour dire que ce programme a été créé avec exe4j.

Utilisation : Encore une fois, très simple, il suffit de suivre chaque étape et de remplir chaque champ l'un après l'autre. En cas de problème, une aide est disponible pour chaque page.



Première page de la création d'un exe avec exe4j

## IV-E - JavaExe

### Site

Dernière version : 2.0

Langue : Français

Fonctionnalités :

- JavaExe est plus basique que les autres programmes, dans le sens où il n'a pas d'interface graphique pour la création, il faut juste employer l'exe donné avec le programme et le nommer comme le Jar
- Il permet soit une application console soit une application Gui
- Il permet de créer un service Windows, mais il faut pour cela modifier un peu votre code
- La package contient aussi une petite application permettant de changer l'icône de votre exe

Limitation licence : Aucune

Utilisation : Très simple, il suffit de renommer le nom du fichier .exe par le nom de votre fichier .jar.

## V - Compiler votre programme de manière native

Utiliser un programme Java implique donc de dépendre d'une JVM. Ce fait déplaît à plusieurs personnes, pour celles-ci, il y a une alternative qui est la compilation de votre code de manière native. Vous ne serez donc dépendant d'aucune JVM, mais cela a certaines limites et inconvénients, une portabilité limitée et des limitations au niveau des classes de Java que certains compilateurs natifs n'incluent pas.

Pour la compilation native de vos programmes java, il existe aussi plusieurs programmes, en voici une partie.

### V-A - GCJ

#### Site

Limitations :

- Support inexistant de Swing
- Faible support de AWT
- Supporte Java 1.4 et quelques parties de Java 1.5
- Problème avec les Apis externes récentes qui ne sont donc pas incluses dans les libs de GCJ

Coût : Gratuit

### V-B - Excelsior JET

#### Site

Limitations :

- Il n'y a aucune limitation à JET, il supporte java 5.0
- JET possède une interface graphique très intuitive
- De plus, Excelsior JET a passé le test de compatibilité de Sun

Coût :

- 4500 \$ pour l'édition Enterprise
- 2300 \$ pour l'édition Professional
- 1200 \$ pour l'édition Standard
- Possibilité d'avoir une version d'essai de 90 jours

### V-C - Toba

#### Site

Limitations : Support de java 1.1 seulement

Coût : Gratuit

## V-D - Manta

### Site

Limitations : Supporte seulement Java 1.1

Coût : Gratuit

## VI - Remerciements

Je tiens à remercier **Elmilouse** pour la relecture de cet article.

