

Sommaire

Utilisation de Java grâce à Jpackage.org.....	1
Mise en oeuvre générale.....	1
Préparation du home pour la recompilation de RPM.....	2
Récupération des divers archives et SRPM.....	2
Recompilation du RPM.....	3
Ajout de jpackage, section nonfree, pour Mandrivalinux.....	3
Copyright.....	4

Utilisation de Java grâce à Jpackage.org

Utilisation de Java grâce à Jpackage.org
par Mickael

Installez proprement Java sur une distribution à base de RPM

Le Java, *saimalsai proprioetsaipalibre*, mais souvent on a besoin de l'utiliser, pour tout un tas de bonnes raisons. De plus, il existe de très bons logiciels libres écrits en Java, et il fait partie des langages les plus utilisés par l'ASF (Apache Software Foundation). Malheureusement, il y a rarement des paquets de logiciels Java, car ils nécessitent une JVM, une machine virtuelle Java, une espèce d'interpréteur de code assembleur d'un processeur qui n'existe pas (voir Wikipedia pour une description plus détaillée et sans doute plus claire).

Et c'est précisément là le problème : la plupart des JVMs ne sont pas libres, donc les distributions ne les incluent pas. Quant aux JVMs libres, elles ne sont pas assez performantes, aussi bien au niveau de la rapidité d'exécution que du support du langage. Quand un distributeur inclut un paquet de JVM, il met peu de programmes Java qui pourraient en bénéficier, et c'est dommage.

C'est là qu'intervient le projet Jpackage. Il s'agit d'un projet de distribution de RPMs de logiciels Java pour plusieurs distributions. Grâce à eux, installer Tomcat ou Jedit revient à simplement taper `urpmi jedit` ou `yum install tomcat4`. Ils proposent des paquets pour Mandriva, Red Hat, Fedora, et d'autres distributions (mais non testées). Une fois le projet ajouté parmi vos sources de RPMs, vous pouvez donc accéder à eclipse, à ant, et autres logiciels Java habituellement plus complexes à installer.

Toutefois, il reste le problème de la JVM. Malgré les efforts du projet et les tentatives de prise de contacts, Sun (et les autres comme IBM, etc) refusent de laisser des packageurs externes refaire leurs RPMs. Une solution a dû être élaborée par les membres de Jpackage, que je vais expliquer dans ce document.

La problématique de base est la suivante : *"Comment garder la cohérence du système de paquets lorsqu'on veut utiliser des logiciels distribués dans des paquets incorrects, non normalisés, ou inexistantes ?"*. La réponse trouvée est de faire ou refaire les paquets. Cela permet de garantir une intégration correcte avec la distribution, d'être sûr qu'on les retire sans problème, et d'être sûr que tout ne sera pas cassé le jour où le distributeur change tout, comme Sun semble le faire si souvent. Vous trouverez plus d'explications dans la FAQ du projet.

Ce document a été fait en testant sur une distribution Mandivalinux 10.0, mais devrait être suffisamment générique pour d'autres distributions. N'hésitez à me faire parvenir vos contributions.

Mise en oeuvre générale

Vous l'aurez compris, nous allons donc refaire les RPMs du JDK (Java Developer Kit, une JVM + un compilateur Java) afin de pouvoir utiliser jpackage. Le déroulement est le suivant :

- Préparation du home en vue de recompiler le RPM
- Récupération des archives et autres RPMs nécessaires
- Recompile
- Ajout de jpackage comme média urpmi local

Le but à la fin étant de pouvoir utiliser urpmi (ou yum, ou apt) pour installer sans problème un logiciel comme jext.

Préparation du home pour la recompilation de RPM

Sans rentrer dans les détails, il existe deux types de RPM. Les RPM binaires, qu'on voit souvent, qui contiennent les logiciels compilés et prêts à l'emploi, et les RPM sources, qui sont utilisés pour faire les RPMs binaires. Un fichier RPM source, ou src.RPM est un RPM qui contient les sources d'un programme, plus des patches, d'autres fichiers, et un fichier de spécification RPM, appelé spec (car son extension est .spec). La moitié du travail pour faire un RPM consiste à écrire ce fichier, l'autre étant de faire marcher le spec comme il faut, et la troisième moitié étant de le tester, de coller aux règles de la distribution et de faire le support utilisateur et la correction de bugs.

Pour faire un RPM, il existe quelques documentations (RPM.org, qa.mandriva.com, linuxfrench.net), mais pour le cas qui nous intéresse (une "simple" recompilation), seule la partie préparation nous importe.

Pour résumer ces documents, il faut créer une arborescence spéciale destinée aux opérations de RPM dans votre répertoire personnel et dire à RPM d'utiliser ces dossiers :

```
$ mkdir -p ~/rpm/{RPMS/{i586,noarch},SRPMS,SPECS,tmp,BUILD,SOURCES}
$ cat << EOF > ~/.rpmmacros
%_topdir          $HOME/rpm
%_tmppath         /tmp
EOF
```

La dernière chose à faire, c'est d'installer le paquet rpm-build, afin d'avoir les fichiers pour recompiler un RPM.

```
# urpmi rpm-build
```

Récupération des divers archives et SRPM

Première étape, le fichier RPM source du JDK de Sun. Afin de montrer que c'est pas un fichier source comme les autres, il est appelé java-1.4.2-sun-1.4.2.nosrc.rpm. Il faut prendre le fichier source du paquet java-1.4.2-sun-1.4.2.

En général, le SRPM contient les sources du paquet, mais le noeud du problème est justement que seul Sun peut les distribuer. Il faut donc les récupérer à part, sur le site de sun (obtenu du champ Url du paquet). Au moment de la rédaction de cet article, le chemin est :

Choisir "Download", prendre "J2SE v 1.4.2_05 SDK", accepter le formulaire après l'avoir lu, et enfin, choisir "Linux Platform", "self-extracting file" (surtout pas "RPM in self-extracting file" mais bien "self-extracting file").

Le fichier téléchargé (de 30 Mo) doit être déposé dans ~/rpm/SOURCES/.

La dernière étape, c'est d'installer le paquet jpackage-utils. Soit vous récupérez le paquet à la main, soit vous passez par urpmi. Pour l'installation à la main, le paquet est sur le site de jpackage, ou sur les miroirs Mandriva. Pour l'installer, urpmi /le_chemin_vers_le_RPM devrait suffire.

Pour l'installation via urpmi, easy urpmi doit avoir tout ce qu'il faut, il suffit d'ajouter 'jpackage', de la même façon que toutes les autres sources.

Recompilation du RPM

Si tout va bien, vous devez être en mesure de recompiler le RPM. Vérifiez que le fichier `j2sdk-1_4_2_04-linux-i586.bin` est dans `rpm/SOURCES`, et lancez la recompilation, avec la commande :

```
$ rpm --rebuild java-1.4.2-sun-1.4.2.05-3jpp.nosrc.rpm
```

Le RPM va se charger d'accepter la licence que vous avez déjà accepté au moment du téléchargement, et de répondre aux questions de l'installateur de Sun. À la fin, vous devriez voir ça :

```
Checking for unpackaged file(s): /usr/lib/rpm/check-files /tmp/java-1.4.2-sun-1.4.2.04-3jpp-buildroot
Wrote: /home/users/misc/rpm/SRPMS/java-1.4.2-sun-1.4.2.04-3jpp.nosrc.rpm
Wrote: /home/users/misc/rpm/RPMS/i586/java-1.4.2-sun-1.4.2.04-3jpp.i586.rpm
Wrote: /home/users/misc/rpm/RPMS/i586/java-1.4.2-sun-devel-1.4.2.04-3jpp.i586.rpm
Wrote: /home/users/misc/rpm/RPMS/i586/java-1.4.2-sun-src-1.4.2.04-3jpp.i586.rpm
Wrote: /home/users/misc/rpm/RPMS/i586/java-1.4.2-sun-demo-1.4.2.04-3jpp.i586.rpm
Wrote: /home/users/misc/rpm/RPMS/i586/java-1.4.2-sun-plugin-1.4.2.04-3jpp.i586.rpm
Wrote: /home/users/misc/rpm/RPMS/i586/java-1.4.2-sun-fonts-1.4.2.04-3jpp.i586.rpm
Wrote: /home/users/misc/rpm/RPMS/i586/java-1.4.2-sun-alsa-1.4.2.04-3jpp.i586.rpm
Wrote: /home/users/misc/rpm/RPMS/i586/java-1.4.2-sun-jdbc-1.4.2.04-3jpp.i586.rpm
Executing(%clean): /bin/sh -e /tmp/rpm-tmp.6627
+ umask 022
+ cd /home/users/misc/rpm/BUILD
+ cd j2sdk1.4.2_04
+ rm -rf /tmp/java-1.4.2-sun-1.4.2.04-3jpp-buildroot
+ exit 0
```

Vous avez donc , dans `rpm/RPMS/i586/` les RPMs de la JVM. Pour pouvoir exécuter des logiciels en Java, il vous faut `java-1.4.2-sun`. Le RPM `java-1.4.2-sun-devel` contient le compilateur et ce qu'il faut pour commencer à développer en Java. Enfin, `java-1.4.2-sun-plugin` est un plugin Java pour mozilla et konqueror.

Ajout de jpackage, section nonfree, pour Mandrivalinux

Muni de vos RPMs, il va falloir les mettre quelque part pour les utiliser. Pour cela, le plus facile est d'ajouter une source locale pour votre gestionnaire de paquets. Copiez les RPMs dans le dossier de votre choix, on va dire `/var/local/urpmi/jpackage/`. Puis, il faut générer les index à l'aide du programme `genhlist` du paquet `rpmttools`. Enfin, vous devez ajouter la source à `urpmi`.

```
#export DEST=/var/local/urpmi/jpackage
#mkdir -p $DEST
#cp -R ~/rpm/RPMS/i586/java* $DEST
#( cd $DEST; genhlist )
#urpmi.addmedia jpackage_local file://$DEST with ./hdlist.cz
```

Et voilà, maintenant, `urpmi java-1.4.2` vous installera la JVM de Sun, et vous pouvez installer les RPMs de `jpackage`. Si vous préférez une autre JVM ou une autre version, vous pouvez procéder de la même manière. Les paquets sont normalement installables côte à côte en parallèle.

Cette page est issue de la documentation 'pré-wiki' de Léa et a été convertie avec `HTML::WikiConverter`. Elle fut créée par Mickael Scherer le 25/07/2004.

Copyright

Copyright © 25/07/2004, Mickael Scherer



*Ce document est publié sous licence Creative Commons
Attribution, Partage à l'identique, Contexte non commercial 2.0 :*
<http://creativecommons.org/licenses/by-nc-sa/2.0/fr/>