Oracle JDK

Table of Contents

Introduction

Manual installation of Oracle Java (which is preferred to Open Java, which often doesn't work right) is a bit tedious: one needs to find and download the latest release and manually set up all the path variables.

The PPA from webupd8 developer team simplifies our task and provides us with an installer which does all this for us and enables auto-updates.

Installation

Before installing be sure on what's your plan and your Java Environment.

Run the following commands in order to install or update the Oracle Java. The idea is applicable to others JVM.

```
sudo add-apt-repository ppa:webupd8team/java
sudo apt-key adv --keyserver hkp://keyserver.ubuntu.com:80 --recv-keys EEA14886
sudo apt-get update
sudo apt-get install oracle-java8-installer
```

The installer takes a while since it needs to download the latest Java installer from the Oracle site.

Then check your Java Environment. A jinfo should exist populated with your appropriate commands, tools or libraries including a unique priority number.

The Install your new JVM through the **update-alternatives** mechanism.

Setting Java Environment Variables

To automatically set up the Java 8 environment variables, you can install the following package:

```
sudo apt-get install oracle-java8-set-default
```

Below **/etc/profile.d** there are two files **jdk.sh** and **jdk.csh** who contain the environment variables definitions.

Here is the content of **jdk.sh**:

```
export J2SDKDIR=/usr/lib/jvm/default-java
export J2REDIR=/usr/lib/jvm/default-java/jre
export PATH=/usr/lib/jvm/default-java/bin:/usr/lib/jvm/default-java/db/bin:/usr/lib/jvm/default-java/jre
/bin:$PATH
export JAVA_HOME=/usr/lib/jvm/default-java
export DERBY_HOME=/usr/lib/jvm/default-java/db
```

Here is the content of **jdk.csh**:

```
setenv J2SDKDIR /usr/lib/jvm/default-java
setenv J2REDIR /usr/lib/jvm/default-java/jre
setenv PATH /usr/lib/jvm/default-java/bin:/usr/lib/jvm/default-java/db/bin:/usr/lib/jvm/default-java/jre
/bin:${PATH}
setenv JAVA_HOME /usr/lib/jvm/default-java
setenv DERBY_HOME /usr/lib/jvm/default-java/db
```

The **oracle-java8-set-default** package install only those two files.

The system will ensure that one those files will be runned when you open a session.

However the Java Environment variables collapse with the **/usr/bin** and **/etc/alternatives** system.

if we run the **java** command the PATH is resolved against **/usr/bin**. The default **jdk.sh** or **jdk.csh** add paths to solve the java commands, tools or libraries while it is already resolved though the standard **update-alternative** mechanism.

Beyond that it is pretty common to use the **JAVA_HOME**, the OpenJDK could need it as well. Not sure who is using the **J2SDKDIR** and **J2REDIR** environment variables but we keep it for integrity.

Further more Derby is delivered with the Oracle JDK but not with the OpenJDK. It sounds to us that gathering the Java environment variables in a single place is a good practice.

Here is a patched **jdk.sh** and **jdsk.csh** we use who improve the PATH strategy, we favored the **/usr/bin** and **/etc/alternatives** style and narrow the **PATH** strategy. Basically we remove the extra paths and kept the derby path.

Here is the content of our patched **jdk.sh**:

```
export J2SDKDIR=/usr/lib/jvm/default-java
export J2REDIR=/usr/lib/jvm/default-java/jre
export PATH=$PATH:/usr/lib/jvm/default-java/db/bin
export JAVA_HOME=/usr/lib/jvm/default-java
export DERBY_HOME=/usr/lib/jvm/default-java/db
```

Here is the content of our patched **jdk.csh**:

```
setenv J2SDKDIR /usr/lib/jvm/default-java
setenv J2REDIR /usr/lib/jvm/default-java/jre
setenv PATH ${PATH}:/usr/lib/jvm/default-java/db/bin
setenv JAVA_HOME /usr/lib/jvm/default-java
setenv DERBY_HOME /usr/lib/jvm/default-java/db
```

Close and re-open a session and check your environment with a **printenv** command.

Not sure if we need to run this command, however for historical reasons we keep it.

```
update-initramfs -u
```

Troubleshooting

Firewall/cacher

The installer fetches the latest Java binary via wget.

Most of the time, the download fails because either the firewall blocks it or because the cacher doesn't allow it.

Since Java updates are not very frequent, we can turn off the firewall while installing, but the cacher can at least be configured for a permanent bypass.

Create **/etc/apt/apt.conf.d/03java-bypass** with the following contents:

/etc/apt/apt.conf.d/03java-bypass

```
Acquire::http::Proxy {
    download.oracle.com DIRECT;
};
```

This will ensure that downloads from download.oracle.com initiated by pre/post-install scripts do not go through the cacher.

The ideal solution would be to make them cached but that needs a regex hack for apt-cacher-ng (to stop it from rejecting this site as a cacheable source) which I was not able to develop yet.

A workaround has been found for this problem:

According to this http://askubuntu.com/a/503992

One can add the following pattern in the acng.conf file.

```
PfilePattern = .*(\.d?deb|\.rpm|\.drpm|\.dsc|\.tar(\.gz|\.bz2|\.lzma|\.xz)(\.gpg|\?AuthParam=.*)?|\.diff(\.gz|\.
bz2|\.lzma|\.xz)|\.jigdo|\.template|changelog|copyright|\.udeb|\.debdelta|\.diff/.*\.gz|(Devel)?
ReleaseAnnouncement(\?.*)?|[a-f0-9]+-(susedata|updateinfo|primary|deltainfo).xml.gz|fonts/(final/)?[a-z]+32.exe
(\?download.*)?|/dists/.*/installer-[^/]+/[0-9][^/]+/images/.*)$
```

However the trick is to help acng to send a HTTP cookie with each and every request so the following pattern is also necessary in acng.conf:

```
RequestAppendix: Cookie: oraclelicense=a
```

Auto accept license

```
echo oracle-java9-installer shared/accepted-oracle-license-v1-1 select true | sudo /usr/bin/debconf-set-
selections
```

or

```
echo oracle-java8-installer shared/accepted-oracle-licence-v1-1 boolean true | sudo /usr/bin/debconf-set-
selections
```

Architecture

If your host architecture differs from LTSP chroot, you won't be able to install Java this way. Chrooting will make the chroot report the host's architecture to the installer, which causes it to download the wrong binary. Generally, chroots should be kept the same architecture as host, but if they must be different, one can still install Java this way if live-boot from the correct architecture media is done and chroot is entered from it.

Plugin

In some cases, the presence of IcedTea Java Browser plugin (OpenJRE implementation) will prevent Oracle Java plugin from working or even being installed.

It is advised to remove all icedtea packages either prior to Oracle Java installation or after it.

In the latter case, reinstallation of Oracle Java might be required to install the plugin properly.

Note: Oracle Java will **not** work in Google Chrome (since Google changed the plugin implementation system from previous standard) until Oracle issues an updated plugin.

This tutorial only use the Oracle Firefox Java support while Chrome and Chromium do not any more support.

In the **jinfo** file we introduced in the Java Environment Tutorial there are some plugin support described:

```
#plugin xulrunner-1.9-javaplugin.so /usr/lib/jvm/java-8-oracle/jre/lib/i386/libnpjp2.so
plugin firefox-javaplugin.so /usr/lib/jvm/java-8-oracle/jre/lib/i386/libnpjp2.so
#plugin iceape-javaplugin.so /usr/lib/jvm/java-8-oracle/jre/lib/i386/libnpjp2.so
#plugin iceweasel-javaplugin.so /usr/lib/jvm/java-8-oracle/jre/lib/i386/libnpjp2.so
#plugin mozilla-javaplugin.so /usr/lib/jvm/java-8-oracle/jre/lib/i386/libnpjp2.so
#plugin midbrowser-javaplugin.so /usr/lib/jvm/java-8-oracle/jre/lib/i386/libnpjp2.so
#plugin xulrunner-javaplugin.so /usr/lib/jvm/java-8-oracle/jre/lib/i386/libnpjp2.so
```

Only the Firefox Java plugin support is in use.

In the update-alternatives we Install only the Firefox Java plugin :

```
#sudo update-alternatives --install /usr/lib/xulrunner-addons/plugins/libjavaplugin.so xulrunner-1.9-javaplugin.
so /usr/lib/jvm/default-java/jre/lib/i386/libnpjp2.so 1073;
sudo update-alternatives --install /usr/lib/firefox-addons/plugins/firefox-javaplugin.so firefox-javaplugin.so
/usr/lib/jvm/default-java/jre/lib/i386/libnpjp2.so 1073;
#sudo update-alternatives --install /usr/lib/iceape/plugins/iceape-javaplugin.so iceape-javaplugin.so /usr/lib
/jvm/default-java/jre/lib/i386/libnpjp2.so 1073;
#sudo update-alternatives --install /usr/lib/iceweasel/plugins/iceweasel-javaplugin.so iceweasel-javaplugin.so
/usr/lib/jvm/default-java/jre/lib/i386/libnpjp2.so 1073;
#sudo update-alternatives --install /usr/lib/mozilla/plugins/libjavaplugin.so mozilla-javaplugin.so /usr/lib/jvm
/default-java/jre/lib/i386/libnpjp2.so 1073;
#sudo update-alternatives --install /usr/lib/midbrowser/plugins/midbrowser-javaplugin.so midbrowser-javaplugin.
so /usr/lib/jvm/default-java/jre/lib/i386/libnpjp2.so 1073;
#sudo update-alternatives --install /usr/lib/xulrunner-addons/plugins/libjavaplugin.so xulrunner-javaplugin.so
/usr/lib/jvm/default-java/jre/lib/i386/libnpjp2.so 1073;
```

Once you have installed the Firefox Java plugin support you should see the following link:

```
user@machine:/usr/lib/firefox-addons/plugins$ ls -al /usr/lib/firefox-addons/plugins/
total 8
drwxr-xr-x 2 root root 4096 Jul 13 10:26 .
drwxr-xr-x 6 root root 4096 Aug 18 2012 ..
lrwxrwxrwx 1 root root 39 Jul 13 10:26 firefox-javaplugin.so -> /etc/alternatives/firefox-javaplugin.so
```

Then test your Firefox configuration with `Test Java`.

In the update-alternatives we Remove only the Firefox Java plugin :

```
#sudo update-alternatives --remove xulrunner-1.9-javaplugin.so /usr/lib/jvm/default-java/jre/lib/i386/libnpjp2.
so;
sudo update-alternatives --remove firefox-javaplugin.so /usr/lib/jvm/default-java/jre/lib/i386/libnpjp2.so;
#sudo update-alternatives --remove iceape-javaplugin.so /usr/lib/jvm/default-java/jre/lib/i386/libnpjp2.so;
#sudo update-alternatives --remove iceweasel-javaplugin.so /usr/lib/jvm/default-java/jre/lib/i386/libnpjp2.so;
#sudo update-alternatives --remove mozilla-javaplugin.so /usr/lib/jvm/default-java/jre/lib/i386/libnpjp2.so;
#sudo update-alternatives --remove midbrowser-javaplugin.so /usr/lib/jvm/default-java/jre/lib/i386/libnpjp2.so;
#sudo update-alternatives --remove xulrunner-javaplugin.so /usr/lib/jvm/default-java/jre/lib/i386/libnpjp2.so;
```

Once you have removed the Firefox Java plugin support no more symlink should exist:

```
user@machine:/usr/lib/jvm/tools$ ls -al /usr/lib/firefox-addons/plugins/
total 8
drwxr-xr-x 2 root root 4096 Jul 13 11:07 .
drwxr-xr-x 6 root root 4096 Aug 18  2012 ..
```

Reference

http://www.webupd8.org/2015/02/install-oracle-java-9-in-ubuntu-linux.html