链滴

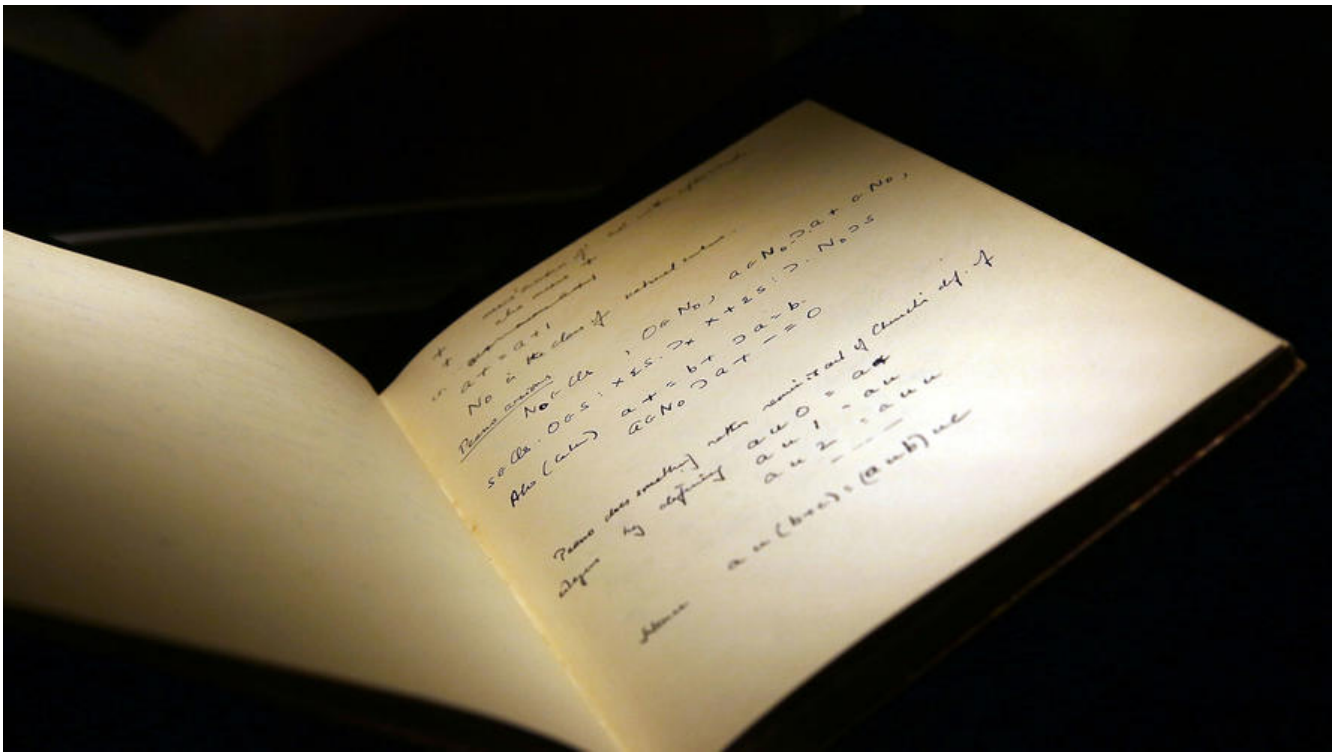
Java 又双叒叕发布新版本，这么多版本如何灵活管理？

作者：9526xu

原文链接：https://ld246.com/article/1574818133354

来源网站：链滴

许可协议：署名-相同方式共享 4.0 国际 (CC BY-SA 4.0)



前言

不知不觉 JDK13 发布已有两个月，不知道各位有没有下载学习体验一番？每次下载安装之后，需要新配置一下 Java 环境变量。等到运行平时的项目又需要切回之前 JDK 版本，这又需要重新环境变量这么重复配置显然非常低效，又不能灵活切换版本。

所幸通过万能 Google 找到解决方案，使用 **jenv** 管理 JDK 版本。

jenv 介绍

熟悉 Ruby 的同学，应该会觉得比较眼熟，jenv 就是 Java 平台的 rbenv 工具。

jenv 是一个命令行工具，可以在 Linux/OS X 平台使用，可以管理多个版本 JDK，方便在多个版本 JD 之间切换，另外其还可以设置 JAVA_HOME 环境变量。

Windows 真的伤不起 sob

使用 jenv 有一个前提，必须提前安装 JDK，其不提供下载安装 JDK 的功能。MacOSX 平台可以提使用 brew 安装 JDK。

jenv 安装

手动安装

首先下载 jenv 源程序

git clone https://github.com/jenv/jenv.git ~/.jenv

然后再将 jenv 命令路径放入 Shell 配置文件中。

若使用 Shell 为 bash：

```
echo 'export PATH="$HOME/.jenv/bin:$PATH"' >> ~/.bash_profile
echo 'eval "$(jenv init -)"' >> ~/.bash_profile
source ~/.bash_profile
```

若使用 Shell 为 zsh

```
echo 'export PATH="$HOME/.jenv/bin:$PATH"' >> ~/.zshrc
echo 'eval "$(jenv init -)"' >> ~/.zshrc
source ~/.zshrc
```

自动安装

若使用 MacOSX，可以直接使用 brew 安装：

```
brew install jenv
```

jenv 校验

安装 jenv 之后，可以运行 jenv doctor 检查是否正确安装。以下为我本机 MacOSX 输出：

```
$ jenv doctor
[OK]    JAVA_HOME variable probably set by jenv PROMPT
[OK]    Java binaries in path are jenv shims
[OK]    Jenv is correctly loaded
```

刚安装小伙伴运行 jenv doctor，可以观察到输出：

```
[ERROR]    Java binary in path is not in the jenv shims.
```

这代表还未正确添加 JDK，需要运行下面指令添加 JDK。

jenv 配置

JDK 配置

运行 jenv add jdk_path,将 JDK 交给 jenv 管理。

```
$ jenv add /Library/Java/JavaVirtualMachines/openjdk-13.0.1.jdk/Contents/Home
openjdk64-13.0.1 added
13.0.1 added
13.0 added
```

可能有些小伙伴并不知道 JDK 安装路径，若是 MacOSX，JDK 是通过 Oracle 提供安装包或者通过 brew 安装，可以通过运行 /usr/libexec/java_home -V 查找 JDK 路径。

```
$ /usr/libexec/java_home -V
Matching Java Virtual Machines (5):
    13.0.1, x86_64:     "OpenJDK 13.0.1"        /Library/Java/JavaVirtualMachines/openjdk-13.0.1.jdk/Contents/Home
    9.0.1, x86_64:      "Java SE 9.0.1" /Library/Java/JavaVirtualMachines/jdk-9.0.1.jdk/Contents/Home
    1.8.0_152, x86_64:  "Java SE 8"     /Library/Java/JavaVirtualMachines/jdk1.8.0_152.jdk/Contents/Home
    1.8.0_111, x86_64:  "Java SE 8"     /Library/Java/JavaVirtualMachines/jdk1.8.0_111.jdk/Contents/Home
    1.7.0_79, x86_64:   "Java SE 7"     /Library/Java/JavaVirtualMachines/jdk1.7.0_79.jdk/Contents/Home

/Library/Java/JavaVirtualMachines/openjdk-13.0.1.jdk/Contents/Home
```

原文链接：Java 又双叒叕发布新版本，这么多版本如何灵活管理？

若不是通过以上方式，或者为 Linux 平台，可以先通过 which java 或者 whereis java 找出命令路径然后通过使用 ls -l 查看命名的实际路径。

```
[root      ~]# which java
/usr/bin/java
[root      ~]# ls -l /usr/bin/java
lrwxrwxrwx 1 root root 26 Oct 23  2017 /usr/bin/java -> /usr/java/default/bin/java
```

JAVA_HOME 配置

有些应用程序将会读取系统 ${JAVA_HOME} 环境变量，通过安装 jenv 插件，切换 JDK 版本时，将同步设置 ${JAVA_HOME} 变量。

```
jenv enable-plugin export
## 运行这个才会生效
exec $SHELL -l
```

jenv 使用教程

jenv versions

查看当前系统 jenv 管理所有 JDK 版本。

```
jenv versions
```

```
$ jenv versions
  system
  1.7
  1.7.0.79
  1.8
  1.8.0.111
  1.8.0.152-ea
  13.0
  13.0.1
* 9.0 (set by /Users/andy.xu/Downloads/.java-version)
  9.0.1
  Home
  openjdk64-13.0.1
  oracle64-1.7.0.79
  oracle64-1.8.0.111
  oracle64-1.8.0.152-ea
  oracle64-9.0.1
  other64-
```

jenv local

通过上面命令，我们知道当前系统所有 JDK 版本之后，通过下面命令切换 JDK 版本。

JDK 版本切换 JDK1.7

jenv local 1.7

这里需要注意的，jenv local 切换 JDK 版本只对当前文件夹有效,如果切换到其他文件夹，将会切换当前默认 JDK 版本。

jenv shell

使用 jenv shell 使切换的 JDK 版本对整个 Shell session 有效。重启 Shell 终端或重新打开新的 Shell 终端，该配置失效。

jenv shell 9.0

jenv global

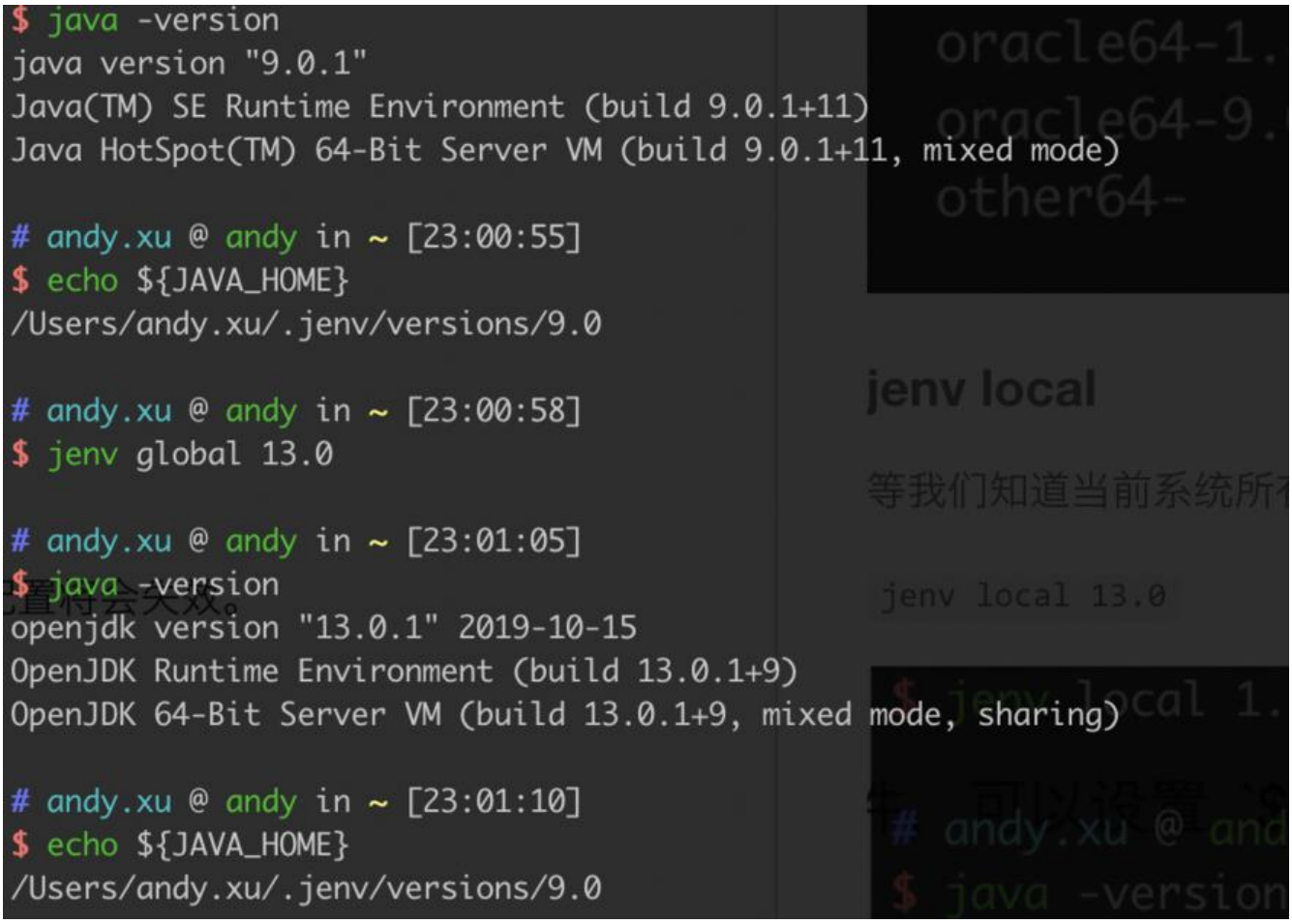
jenv global 将会设置一个全局默认的 JDK 版本，即使重启 Shell 窗口，该配置也不会改变。

jenv global 9.0

jenv 小问题

上面几个是 jenv 经常使用的指令，若想了解 jenv 其他指令，可以通过 jenv commands 查找其他指。

使用 jenv 过程可能会发现，当切换 JDK 版本之后， ${JAVA_HOME} 环境变量没有改变，还是上一个 JDK 版本配置。



这时可以运行 exec $SHELL -l ， ${JAVA_HOME} 将会变成当前版本。

最后

做个小调查，你还在用那个版本 JDK？各位小伙伴可以留言一下，嘿嘿~