Finding and Setting JAVA HOME

Depending on your operating environment, finding the correct value for and then setting the JAVA_HOME environment variable can be either trivially easy or frustratingly difficult.

We'll start with the easier procedures and work our way up the difficulty scale.

Is JAVA_HOME Already Set?

Sometimes your JAVA_HOME environment variable will already be set for other reasons. To check, on the Windows command line do:

```
    C:> echo %JAVA_HOME%
```

or on linux-based systems, do:

```
    % echo $JAVA_HOME
```

If the answer is a directory path, you are good to go.

Finding the Java Home Directory

The JAVA_HOME environment variable needs to point to the installation directory on your system that contains the bin/ directory where the java executable physically resides, *and* the lib/ directory which contains the core java libraries and properties files. For linux users, it is almost certainly *not* the same directory that you get if you do "which java" from the command line, because that would be too easy. So if JAVA_HOME isn't already set, your next task is to find the directory to which you need to set it.

Are You Using a Mac?

If so, then the Apple system programmers may have done you a favor and provided a handy little command called "java_home" that returns the value you need. Just do this from the command line:

```
     % java_home
```

and use the result. If you don't have this command handy, you can create one yourself using the code near the bottom of this page, under Sample Perl Script: java_home.

Is Your Java Version 1.7 or Later?

If your Java version is fairly recent, there's a useful command option that lists out a whole bunch of system property settings, including the value of java.home, which is what we need. To see your Java version (Windows or Linux), do:

```
     % java -version
```

If the answer is 1.7 or later, then this command will dump several dozen lines of property settings to your screen:

```
     % java -XshowSettings:properties -version
```

Scroll through the output to find the line (about a dozen lines from the beginning of the listing) containing "java.home", and use the value part of that line.

Note that java dumps this listing to *stderr* rather than *stdout*, so the usual pipe through more will not have the desired effect. You are going to have to either scroll or, on linux-based systems, redirect error output to a file and then grep through that.

Are You, or Do You Know, a Friendly Java Programmer?

The value needed is the value of the java.home property, which is available from inside a java executable if you know what you are doing. A java programmer should be able to write a reasonably short program to print this value for you on demand, so you never have to bother him or her again. I'm not a java programmer, so if you happen to have a program like this you'd like to donate to the cause, please get in touch.

Alternately, if you have the privileges you can update your java installation to the most recent version and use the "Java Version 1.7 or later" method, above.

Are You Ready to Do It the Hard Way?

As a last resort, you'll have to dig through system directories, trying to find the java installation files. This can be tricky, since the installation might be for the Java Developers Kit (look for directories starting with "jdk"), or the Java Runtime Environment (look for directories starting with "jre"), or it might be referenced as the Java Virtual Machine (look for directories starting with "jvm").

Search functions are your friend, here. On Windows machines, first look for a "Java/ directory in your "Program Files", "Program Files (x86)" or equivalent system directory. If you find one, look down into it until you find the subdirectory containing the bin/ and lib/ directories. If that fails, try searching for the java executable itself, and use the name of the directory that contains the bin/ directory that, in turn, contains the java executable.

On linux-based systems, where there is much greater scope for variability, try running the find command on directories like /usr/lib or /usr/share. (You should have permission to read any directories that might contain java libraries you'd reference, so you can ignore "Permission denied" messages.)

Setting JAVA_HOME

The JAVA_HOME environment variable can be set temporarily, so that when a script or batch file finishes, or you close your command window, it goes away; or as part of your regular environment. The syntax is roughly the same either way, but where you set the value changes.

On Windows Systems

To set the value temporarily from the command line, use the set command:

```
        C:>set JAVA_HOME="C:\Program Files (x86)\Java\jre"
```

where spacing and capitalization count, and everything inside the quotes should be replaced with the path you found via one of the methods in the previous section. The double quotes are required if your path contains any imbedded blanks.

To create and set a JAVA_HOME variable as part of the default environment, use the method appropriate to your particular flavor of Windows described here:

- How to set the path and environment variables in Windows, by ComputerHope.com

Then insert the path you found previously.

On Linux-based Systems

You will need to know what shell you are using and what the appropriate way to set environment variables in that shell is. The two most popular shells are the Bourne shell and its relatives, and different variations on the C-shell. We'll show syntax for both of those here. (Do "echo $SHELL" from the command line to see what shell you are running if you don't know, and "man *shell*" on the shell name to see what type it is.)

In Bourne-type shells, it takes two steps to set an environment variable that can be seen by any programs you then run. For example:

```
% JAVA_HOME=/usr/lib/jvm/java-1.7.0-openjdk-1.7.0.71.x86_64/jre
% export JAVA_HOME
```

C-shell-type shells can do this in a single step:

```
% setenv JAVA_HOME /usr/lib/jvm/java-1.7.0-openjdk-1.7.0.71.x86_64/jre
```

and there's an alternate syntax that works just as well:

```
% set JAVA_HOME=/usr/lib/jvm/java-1.7.0-openjdk-1.7.0.71.x86_64/jre
```

You can include these lines in wrapper scripts, or you can set JAVA_HOME in your shell resource file, so that it becomes an automatic part of your login environment.

Alternately, if you are on a Mac, or a friendly programmer has written you a routine that performs the same function as the Mac java_home program, you can replace the explicit directory listed in the code samples above with a call to this routine. For example, in the C-shell:

```
% setenv JAVA_HOME `java_home`
```

or, alternately:

```
% set JAVA_HOME=`java_home`
```

The same method works for Bourne-type shells.

The back-ticks (the character under the tilde character on most US keyboards) indicate that the command inside the ticks should be run and the result saved as the value of JAVA_HOME. This way, when the java installation is upgraded, you will automatically get the correct JAVA_HOME value without having to edit your resource or script files.

Sample Perl Script: java_home

Here's a quick and dirty Perl script that can be used to get the value for JAVA_HOME from the property list available via the *−XshowSettings* option of Oracle Java version 1.7 and later.

```
#!/usr/bin/perl

# Routine to get system property settings from default java installation
# and return the value of java.home (for setting JAVA_HOME environment
# variable).
# 20 March 2015, A.C.Raugh
#
```

```
#========================================================

open (INP, "java -XshowSettings:properties 2>&1 |") ||
  die "Could not open input pipe to get Java settings, ";

while ($line = <INP>)
  { if ($line =~ /java\.home./)
      { chomp $line;
        $line =~ s/^\s*java\.home\s*=\s*//;
        print ($line,$/);
        close(INP);
        last;
      }
  }
exit;
```