NAME

javavm - convenient wrapper for switching Java VMs

SYNOPSIS

javavm [*arguments* ...]

DESCRIPTION

The Java VM wrapper provides a convenient system for switching between different Java VMs. It also provides symbolic links in */usr/local/bin* to allow the use of the Java executables without having to add the specific Java VM executable directories to the PATH environment variable.

The **javavm** utility itself is currently a synonym for "java". However, this behaviour is deprecated and is provided only for compatibility with the previous version of **javavm**. This functionality may be removed in a future version and should not be relied upon.

By default, **javavm** will select the most "native" and up to date version of the Java VM when a given symbolic link is used, invoking and passing the arguments to the matching executable within the chosen Java VM. The choice of Java VM may also be influenced by using environment variables to constrain the version, vendor and operating system of the Java VM.

This selection process is usually achieved through the use of */usr/ports/Mk/bsd.java.mk*. However, if this is not present then **javavm** will use its own internal selection process which is designed to behave almost identically.

ENVIRONMENT

JAVA_HOME

The presence of this variable in the environment when executing **javavm** will override all other considerations regarding the Java VM to be used and the Java VM located at *${JAVA_HOME}/bin/java* will be used.

This variable is set by **javavm** when executing the actual Java VM and will be available to it and all of its child processes.

JAVA_OS

A space delimited list of operating systems. The selected Java VM must have been created for one of the operating systems in the list.

Currently allowed operating system values are ‘native’ and ‘linux’.

JAVA_VENDOR

A space delimited list of Java VM vendors. The selected Java VM must have been released by one of the vendors in the list.

Currently allowed vendors are ‘openjdk’, ‘oracle’, and ‘sun’.

JAVA_VERSION

A space delimited list of versions of the Java VM that may be used. By appending a ‘+’ to a version, any Java VM with a version greater than or equal to the given version will be used.

Currently allowed versions are ‘1.7’, ‘1.7+’, ‘1.8’, ‘1.8+’, ‘1.9’, ‘1.9+’, ‘7’, ‘7+’, ‘8’, ‘8+’, ‘9’, ‘9+’, ‘10’, ‘10+’, ‘11’, ‘11+’, ‘12’, ‘12+’, ‘13’, ‘13+’, ‘14’, ‘14+’, ‘15’, ‘15+’, ‘16’, ‘16+’, ‘17’, ‘17+’, ‘18’, and ‘18+’.

JAVAVM_OPTS

The contents of this environment variable will be passed to the invoked Java VM as options. For more information on environment variables which can be used to set options see */usr/local/etc/javavm_opts.conf.dist*.

JAVAVM_FALLBACK_ONLY

If this variable is set then instead of selecting the Java VM based on */usr/ports/Mk/bsd.java.mk* only the internal selection process is used. This may be useful to achieve consistent results for Java VM selection across multiple hosts, where some have the ports collection installed and others do not. However, this option, when used with scripts installed by a port, may result in Java VM selection inconsistent with that intended by the script author.

JAVAVM_DRYRUN

When this variable is set, no Java VM is invoked. Instead, the Java VM wrapper prints out the following information:

JAVA_HOME

The value of the JAVA_HOME environment variable which the Java VM wrapper would have set before invoking the Java VM.

JAVAVM_CONF

The Java VM wrapper configuration file being used.

JAVAVM_OPTS_CONF

The Java VM wrapper option configuration file being used.

JAVAVM_PROG

The Java VM that would have been invoked.

JAVAVM_OPTS
The options that would have been passed to the invoked Java VM. It is important to note that this variable may not be the same as the JAVAVM_OPTS environment variable due to processing of the Java VM wrapper option configuration file.

JAVAVM_COMMAND
The full command line that would have been used to invoke the Java VM.

FILES

/usr/local/etc/javavms
The location of the Java VM wrapper configuration file.

/usr/local/etc/javavm_opts.conf
The location of the Java VM wrapper option configuration file.

/usr/ports/Mk/bsd.java.mk
The file usually used, along with **make**, to select the Java VM to be used.

EXAMPLES

/usr/local/bin/java
Execute the most up to date and "native" Java VM registered with **javavm**.

JAVA_VERSION=8+ */usr/local/bin/javac MyClass.java*
Compile MyClass.java with a registered Java VM's javac that is at least version 8.

JAVA_OS=native */usr/local/bin/java* **-jar** *MyApp.jar*
Execute MyApp with the most up to date native Java VM that is registered with **javavm**. This is necessary if MyApp uses JNI, for instance.

JAVA_VERSION="8 11" */usr/local/bin/java* **-jar** *MyApp.jar*
Execute MyApp with either a Java VM that is either version 8 or version 11.

JAVAVM_DRYRUN=yes */usr/local/bin/java*
Don't invoke the Java VM, but print out information about what would have been done. This could be used in a script to determine the JAVA_HOME that the Java VM wrapper will use, for instance:

JAVA_HOME=‘env JAVAVM_DRYRUN=yes /usr/local/bin/java | grep ’^JAVA_HOME’ | cut

-c11-‘

SEE ALSO

checkvms(1), make(1), manvm(1), registervm(1), unregistervm(1), javavm_opts.conf(5), javavms(5)

BUGS

The internal selection procedure of **javavm** is not fully identical to that used when */usr/ports/Mk/bsd.java.mk* is present. In particular, **javavm** does not respect environment variables such as JAVA_PREFERRED_PORTS which is used by */usr/ports/Mk/bsd.java.mk* and will not use the values of such variables if they are set up as **make** variables in */etc/make.conf*, for example.