

What is PATH and CLASSPATH in Java? Path vs ClassPath Example

What is PATH and CLASSPATH in Java

The PATH and CLASSPATH are the two most important environment variables of the Java environment which are used to find the JDK binaries used to compile and run Java in windows and Linux and class files which are compiled Java bytecodes. From my personal experience I can say that PATH and CLASSPATH are the two most problematic things for beginners in Java programming language due to two reasons; first because in most Java courses nobody tells details of what is a PATH and CLASSPATH, What do PATH and CLASSPATH do, What is meaning of setting PATH and CLASSPATH, What happens if we do not set them, Difference between PATH vs CLASSPATH in Java or simply [How Classpath works in Java](#), etc.

These basic question which answers most of the details about PATH and CLASSPATH in Java are mostly not answered until Java programmer itself acquire this knowledge, Things may be changed nowadays but important of PATH and CLASSPATH is still high.

The most common cause of dreaded error

like `java.lang.NoClassDefFoundError` and `java.lang.ClassNotFoundException` is either incorrect or misconfigured CLASSPATH in Java.

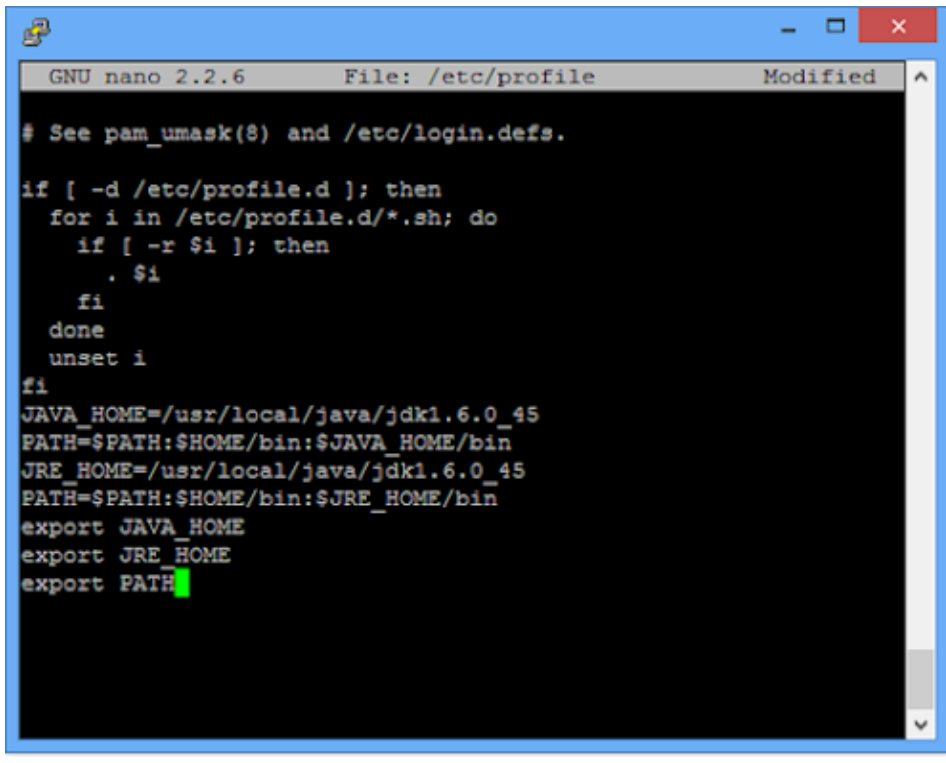
In this article, I'll tell you about the practical difference between PATH and CLASSPATH environment variables, where are they located, and how exactly they are used by the Java compiler and JVM. Once you know this basic detail, you would be able to solve most of the classpath-related problems by yourself.

Difference between PATH and CLASSPATH in Java

Here are some of the common differences between PATH vs CLASSPATH in Java :

1. The main difference between PATH and CLASSPATH is that PATH is an environment variable that is used to locate JDK binaries like the "java" or "javac" command used to run java program and compile the java source file. On the other hand, CLASSPATH, an environment variable is used by System or **Application ClassLoader** to locate and load compile Java bytecodes stored in the .class file.
2. In order to set PATH in Java, you need to include `JDK_HOME/bin` directory in PATH environment variable while in order to set CLASSPATH in Java you need to include all those directories where you have put either your .class file or JAR file which is required by your Java application.
3. Another significant difference between PATH and CLASSPATH is that PATH can not be overridden by any Java settings but CLASSPATH can be overridden by providing command-line option `-classpath` or `-cp` to both "java" and "javac" commands or by using Class-Path attribute in Manifest file inside **JAR archive**.
4. PATH environment variable is used by the operating system to find any binary or command typed in the shell, this is true for both Windows and Linux environments while CLASSPATH is only used by Java ClassLoaders to load class files.

These were some notable differences between PATH vs CLASSPATH in Java and they are worth remembering to debug and troubleshoot Java-related issues. Though, I highly recommend you to join these **best Java Programming courses** to build your fundamentals in Java.



```
GNU nano 2.2.6 File: /etc/profile Modified
# See pam_umask(8) and /etc/login.defs.

if [ -d /etc/profile.d ]; then
  for i in /etc/profile.d/*.sh; do
    if [ -r $i ]; then
      . $i
    fi
  done
unset i
fi
JAVA_HOME=/usr/local/java/jdk1.6.0_45
PATH=$PATH:$HOME/bin:$JAVA_HOME/bin
JRE_HOME=/usr/local/java/jdk1.6.0_45
PATH=$PATH:$HOME/bin:$JRE_HOME/bin
export JAVA_HOME
export JRE_HOME
export PATH
```

How to set PATH and CLASSPATH in Windows and Unix

If you are familiar with DOS operating system and how to use command prompt in Windows or shell in Linux setting PATH and CLASSPATH is a trivial exercise. Both PATH and CLASSPATH are environment variables and can be set using the `export` command in Linux and using `set` keyword in DOS and Windows as shown below:

Command to set PATH in Windows

```
set PATH=%PATH%;C:\Program Files\Java\JDK1.6.20\bin
```

Command to set PATH in UNIX/Linux

```
export PATH = ${PATH}:/opt/Java/JDK1.6.18/bin
```

Look at the difference between the two commands, in Linux use a colon(:) as a separator, and Windows uses a semi-colon(;) as a separator.

Command to set CLASSPATH in windows

```
set CLASSPATH=%CLASSPATH%;C:\Program Files\Java\JDK1.6.20\lib
```

Command to set CLASSPATH in Unix/Linux

```
export CLASSPATH= ${CLASSPATH}:/opt/Java/JDK1.6.18/lib
```

Also, don't forget to include the current directory, denoted by a dot(.) to include in CLASSPATH, this will ensure that it will look first in the current directory and if it found the class it will use that even if that class also exists in another directory which exists in CLASSPATH.

If you using Windows 8 or Windows 10 then you can also follow the steps given here in [this article](#) to set up the JAVA_HOME, PATH, and CLASSPATH for Java programs.