

PIC 20A

Miscellaneous

David Hyde
UCLA Mathematics

Last edited: March 29, 2020

Outline

PATH

Pseudo-random numbers and Monte Carlo

Fisher-Yates shuffle

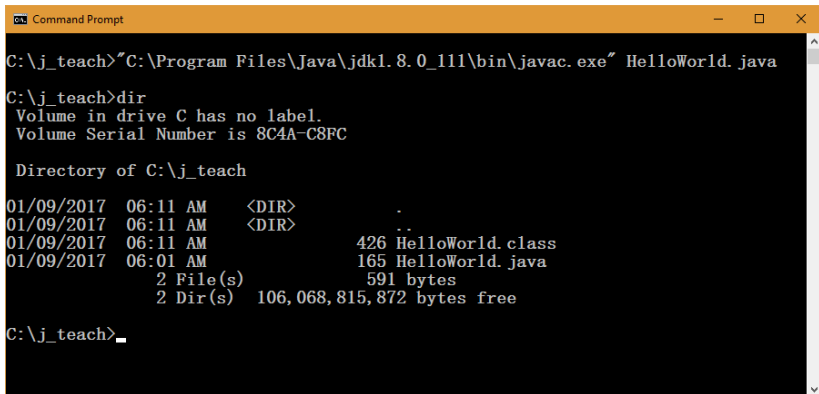
Compilation and JVM

History

Setting the PATH variable

Calling the compiler with its full path

"C:\Program Files\Java\jdk1.8.0_111\bin\javac.exe"
is cumbersome.



```
Command Prompt

C:\j_teach>"C:\Program Files\Java\jdk1.8.0_111\bin\javac.exe" HelloWorld.java

C:\j_teach>dir
Volume in drive C has no label.
Volume Serial Number is 8C4A-C8FC

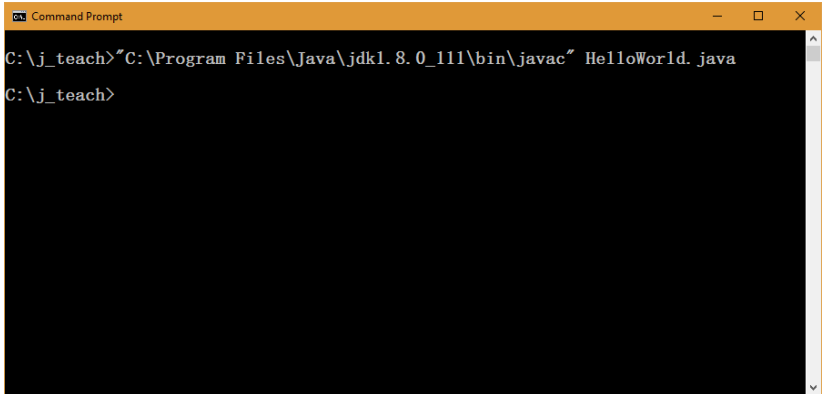
Directory of C:\j_teach

01/09/2017  06:11 AM    <DIR>          .
01/09/2017  06:11 AM    <DIR>          ..
01/09/2017  06:11 AM                426 HelloWorld.class
01/09/2017  06:01 AM                165 HelloWorld.java
                2 File(s)                591 bytes
                2 Dir(s)  106,068,815,872 bytes free

C:\j_teach>_
```

Setting the PATH variable

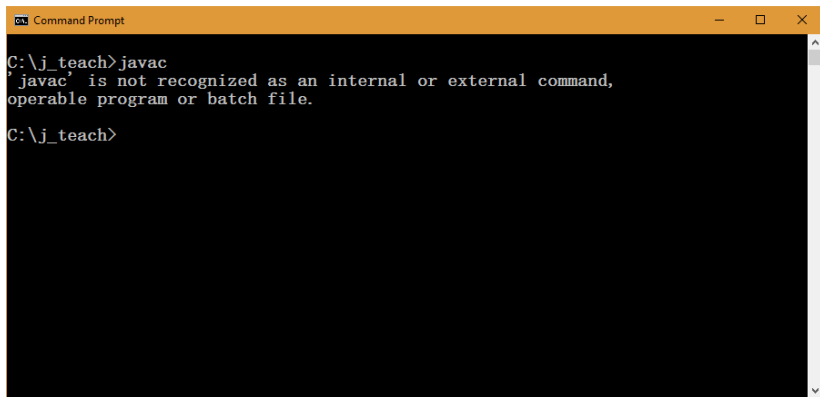
Omit .exe to save you from typing 4 characters.

A screenshot of a Windows Command Prompt window. The title bar is orange and reads "Command Prompt". The window has standard minimize, maximize, and close buttons. The command prompt shows the following text:
C:\j_teach>"C:\Program Files\Java\jdk1.8.0_111\bin\javac" HelloWorld.java
C:\j_teach>
The text is white on a black background. A vertical scrollbar is visible on the right side of the window.

```
Command Prompt
C:\j_teach>"C:\Program Files\Java\jdk1.8.0_111\bin\javac" HelloWorld.java
C:\j_teach>
```

Setting the PATH variable

When you call `javac`, the system looks for `javac.exe` in the current directory *and* in all directories of `PATH`, an environment variable .

A screenshot of a Windows Command Prompt window. The title bar is orange and says "Command Prompt". The window has standard minimize, maximize, and close buttons. The background is black with white text. The prompt is "C:\j_teach>". The user has entered "javac". The response is "'javac' is not recognized as an internal or external command, operable program or batch file." followed by a new prompt "C:\j_teach>".

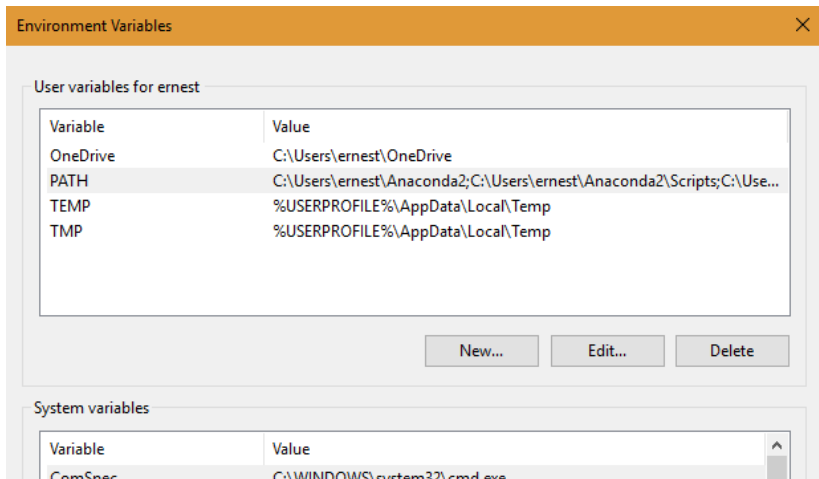
```
Command Prompt
C:\j_teach>javac
'javac' is not recognized as an internal or external command,
operable program or batch file.
C:\j_teach>
```

As it is, `PATH` does not contain the path to `javac`.

Setting the PATH variable

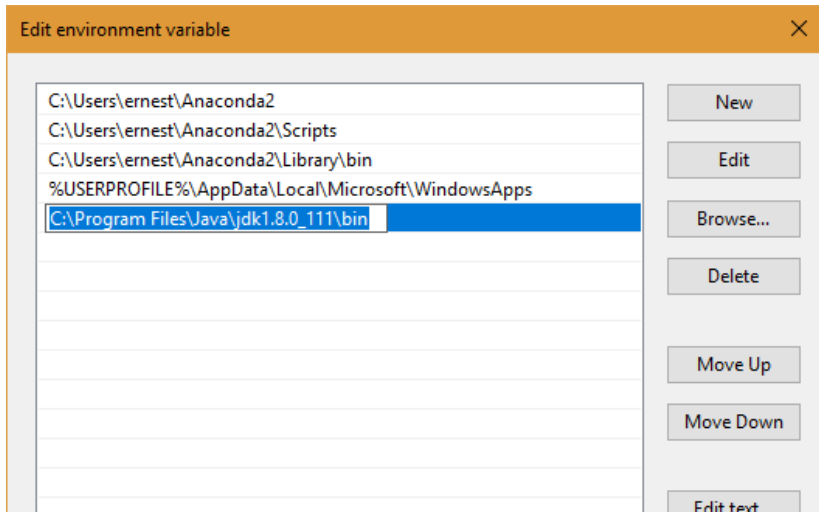
Let's add another path to the PATH variable:

Start→Control Panel→System→Advanced→Environment Variables



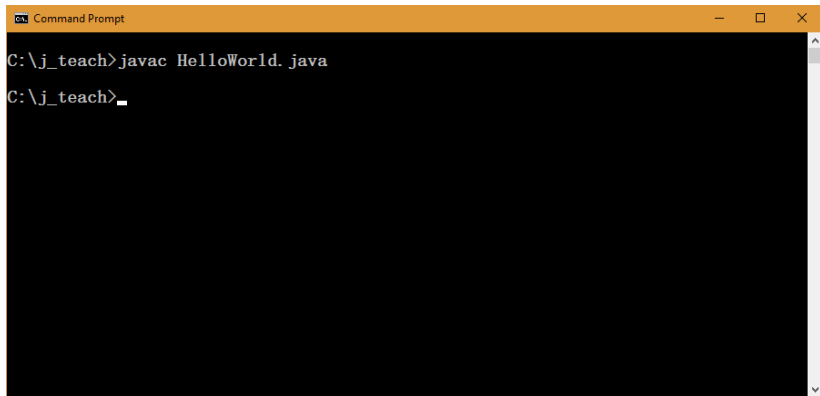
Setting the PATH variable

Select PATH and then Edit. Create enter the path to javac



Setting the PATH variable

Finally, close and reopen the command prompt. Now call `javac` and `java` without referring to its full path.

A screenshot of a Windows Command Prompt window. The title bar is orange and says "Command Prompt". The window has standard minimize, maximize, and close buttons. The command prompt shows the directory "C:\j_teach" and the command "javac HelloWorld.java" being executed. Below that, the prompt "C:\j_teach>" is shown with a cursor, indicating the next command is to be entered.

```
Command Prompt
C:\j_teach>javac HelloWorld.java
C:\j_teach>_
```


Setting the PATH variable

You have the same story with MacOS and Linux: when you call `javac` from the command line, the system (or shell) looks for an executable named `javac` in the current directory and in all directories listed in `PATH`.

Sometimes the Java installation updates the `PATH` variable for you. If not, manually add the path to `javac` and `java` to `PATH`.

Outline

PATH

Pseudo-random numbers and Monte Carlo

Fisher-Yates shuffle

Compilation and JVM

History

Random number generation

The “random numbers” `Math.random()` return are not truly random. They are *pseudo-random* numbers, which are deterministic numbers that look random.

Defining “random” is actually a deep and philosophical exercise, so we avoid it. However, we will say that you can’t generate truly random numbers on a computer.

Pseudo Random numbers

Understanding pseudo-random numbers is hard. Using them is easy. Here's what you need to know to *use* pseudo-random numbers.

There are several ways to generate pseudo-random numbers and they are programmed into *pseudo-random number generators* (PRNG). E.g. linear congruential generator or Mersenne Twister. PRNGs are also called random number engines.

Some PRNGs are bad and some are better than others. For example, `rand()` from `cmath` is a bad PRNG. None are perfect, and all have tradeoffs.

You need a *seed* for PRNG. Given the same seed, the PRNG will produce the same output.

Pseudo Random numbers

```
import java.util.Random;
public class Test {
    public static void main(String[] args) {
        long seed = System.currentTimeMillis();
        Random rand1, rand2;

        rand1 = new Random(seed);
        for(int i=0; i<10; i++)
            System.out.print(rand1.nextInt(10));

        System.out.print("\n");

        rand2 = new Random(seed);
        for(int i=0; i<10; i++)
            System.out.print(rand1.nextInt(10));
    }
}
```

Pseudo Random numbers

The output is different every time the program is run, but the 2 PRNGs produce the same output.

```
...  
rand1 = new Random(seed);  
for(int i=0; i<10; i++)  
    System.out.print(rand1.nextInt(10));  
...  
rand2 = new Random(seed);  
for(int i=0; i<10; i++)  
    System.out.print(rand2.nextInt(10));
```

Pseudo Random numbers

A PRNG always needs a seed. If you don't provide one, the algorithm will create its own seed (usually based on the current time).

```
rand = new Random();  
// "constructor sets the seed to a value  
// very likely to be distinct from  
// other invocation of this constructor"
```

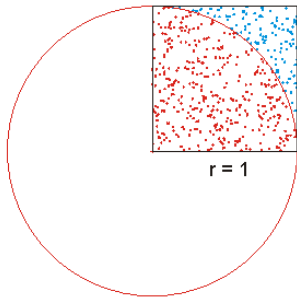
Usually, it's okay to use the automatic seed and not worry about it. Sometimes, however, you want the same pseudo-random numbers, say, for debugging purposes. For that, set the seed explicitly.

Monte Carlo

The *Monte Carlo method* is a fancy name for the following simple idea: Simulate a bunch of random events and take their average to understand the “average behavior”.

Monte Carlo

For example, if x and y are (independently and uniformly) randomly generated on the interval $[0, 1]$, what is the probability of landing within the quarter-circle centered at the origin?



The answer is $(\pi/4)/1$.

Monte Carlo

If we simulate n such points, on average $\pi/4 \cdot n$ will land within the quarter-circle. We can use this to compute π .

```
import java.util.Random;
public class Test {
    public static void main(String[] args) {
        Random rand = new Random(0);
        int n = 1000000; //1 million
        int count = 0;
        for (int i=0; i<n; i++) {
            double x = rand.nextDouble();
            double y = rand.nextDouble();
            if (x*x+y*y <= 1) count++;
        }
        System.out.println( (4.0*count)/n );
    }
}
```

The output is 3.142156.

Outline

PATH

Pseudo-random numbers and Monte Carlo

Fisher-Yates shuffle

Compilation and JVM

History

Fisher-Yates shuffle

The *Fisher-Yates shuffle*, also called the *Knuth shuffle*, randomly permutes an array.

```
a = some array of length n
for i = n-1, ..., 1
    j = randInt 0 <= j <= i
    swap(a[j], a[i])
```

(The Fisher-Yates shuffle makes all $n!$ permutations occur with the equal probability $1/n!$.)

Why does the Fisher-Yates shuffle work?

If you place n objects in a box, shake and mix, and pull out one object at a time, you get a random permutation.

Let's see why the algorithm is exactly the same.

Outline

PATH

Pseudo-random numbers and Monte Carlo

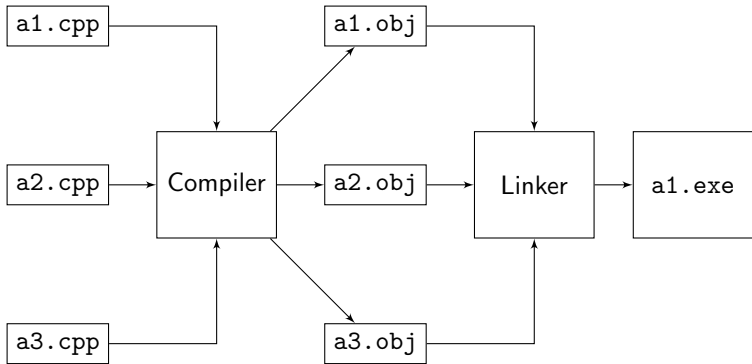
Fisher-Yates shuffle

Compilation and JVM

History

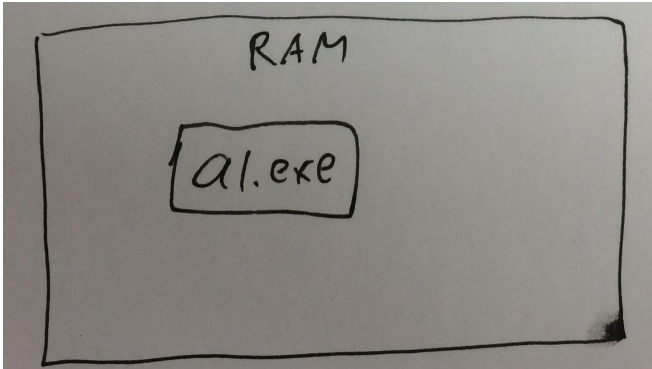
Compiling C++

C++ code compiles into an executable.



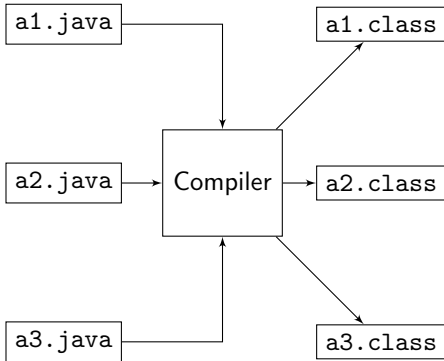
Running C++

To run a C++ program (compiled into an executable) the executable must be loaded into memory.



Compiling Java

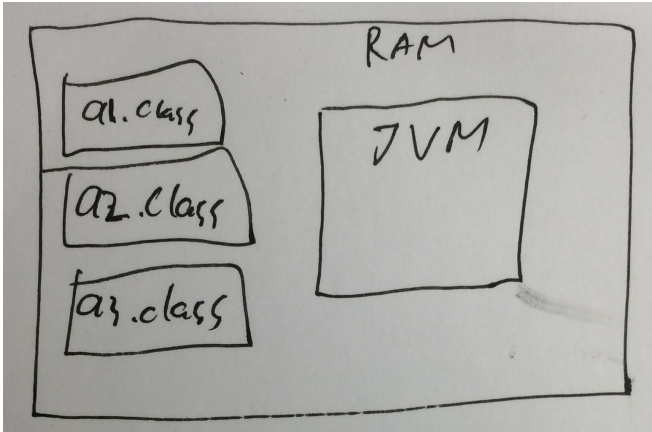
Java doesn't compile into an executable.



There is no “linking” in Java.

Running Java

A Java program (compiled into .class files) is run by loading the .class files into memory and having the Java Virtual Machine (JVM) run them.



Write once, run anywhere

Different systems need their own JVM to run Java. They are written to behave the same across all platforms.

The JVM acts as a buffer between the diverse platforms and the Java programmer. The ideal is “Write once, run anywhere” (WORA).

A programmer can develop Java on a PC and can expect it to run on Java enabled cell phones. This saves programmers the effort of writing a different version of their software for each platform or operating system.

Outline

PATH

Pseudo-random numbers and Monte Carlo

Fisher-Yates shuffle

Compilation and JVM

History

Java history and versions

Sun Microsystems (now acquired by Oracle) developed Java with the following design goals:

- ▶ It must be “simple, object-oriented, and familiar”.
- ▶ It must be “robust and secure”.
- ▶ It must be “architecture-neutral and portable”.
- ▶ It must execute with “high performance”.

Java history and versions

1996 JDK 1.0.

1997 JDK 1.1.

1998 J2SE 1.2. `javax.swing` added.

2000 J2SE 1.3.

2002 J2SE 1.4.

2004 J2SE 5.0. Generics, for each loop, autoboxing/unboxing, varargs, and static imports added.

2006 Java SE 6.

2011 Java SE 7.

2014 Java SE 8. Lambda expressions added.

2017 Java SE 9.

2018 Java SE 10.

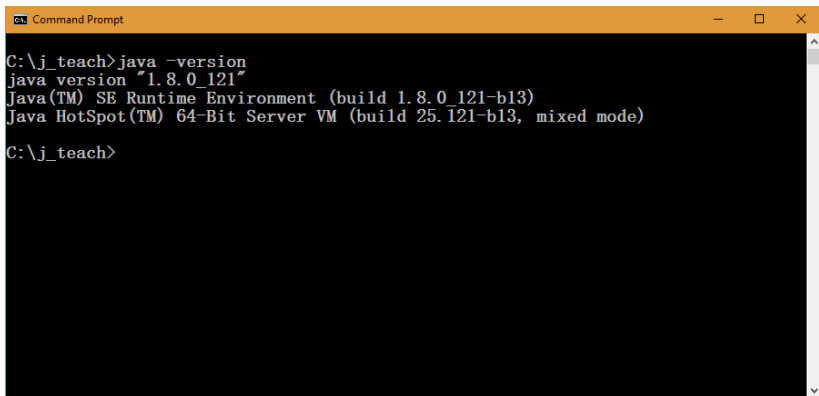
2018 Java SE 11.

2019 Java SE 12.

2019 Java SE 13.

Java history and versions

`java -version` shows the Java version and update number.

A screenshot of a Windows Command Prompt window. The title bar is orange and says "Command Prompt". The window has standard minimize, maximize, and close buttons. The background is black, and the text is white. The text displayed is:
C:\j_teach>java -version
java version "1.8.0_121"
Java(TM) SE Runtime Environment (build 1.8.0_121-b13)
Java HotSpot(TM) 64-Bit Server VM (build 25.121-b13, mixed mode)
C:\j_teach>
There is a vertical scrollbar on the right side of the window.

This system uses Java SE 8 update 121.

Digression: relationship with Javascript

Java and Javascript are entirely unrelated!

Project Mocha, which was renamed LiveScript at the beta phase was named Javascript as a marketing ploy to give JavaScript the cachet of what was then the hot new language.