Introduction to Programming (Java)

Roman Szturc (roman.szturc@vsb.cz)

Zdenek Sawa (zdenek.sawa@vsb.cz)

Department of Computer Science
VSB – Technical University of Ostrava

Info for Students

Lectures: Zdenek Sawa (e-mail: zdenek.sawa@vsb.cz, room: A1006)

Consultations: Wednesday 9:00 – 10:30

Exercises: 40 points

Num.	Date	Points
1.	08.–12.03.	5
2.	22.–26.03.	6
3.	05.–09.04.	6
4.	19.–23.04.	7
5.	03.–07.05.	8
6.	17.–21.05.	8

Exam: 60 points

WWW: http://www.cs.vsb.cz/java/

See also: http://www.cs.vsb.cz/benes/vyuka/upr/index.html

References

- Sun Microsystem, Inc., **The Source for Java Technology**, http://java.sun.com
- J. Gosling, B. Joy, G. Steele, G. Bracha: **The Java Language Specification** http://java.sun.com/docs/books/jls/index.html
- JavaTM 2 Platform, Standard Edition, v 1.4.2 API Specification, http://java.sun.com/j2se/1.4.2/docs/api/index.html
- B. Eckel, **Thinking in Java**, http://www.mindview.net/Books/TIJ
- JavaWorld.com, an IDG Communications company, **JavaWorld**, http://www.javaworld.com

Java - Overview

Java is a general-purpose language with the following features:

- is object-oriented (class-based)
- is cross-platform
- is strongly typed
- has garbage collection
- supports concurrecy
- supports exceptions
- security is considered

It was created by James Gosling from Sun Microsystems in 1990. Original name was Oak and it was intended for use in embedded consumer-electronic applications.

Later in 1993 it was renamed to Java and retargeted to Internet applications.

First official implementation (JDK 1.0) was released in 1996.

Creating Java Program (Step 1)

Source code creation

In the Java, each method (function) and variable exists within a class or an object. The Java does not support global functions or variables. Thus, the skeleton of any Java program is a class definition.

Every Java application must contain a main() method. The method is invoked when the application is executed by a Java interpreter.

```
public class HelloWorld {
   public static void main(String[] arguments) {
      System.out.println("Hello, world...");
   }
}
```

Note: A Java class source code **must** be stored in file which name starts with the **class name** appended with the **.java**, so the previous example must be stored in a file called HelloWorld.java.

Creating Java Program (Step 2)

Compilation

Compilation transfers Java source code into Java bytecode. There is a lot of compilers available. The most often used ones are javac and jikes.

File name containing the source code is passed to the compiler. Result bytecode is stored into file whose name is the **class name** appended by the **.class** suffix.

```
$ javac HelloWorld.java
```

Note: Each compiler provides its specific options. The most important are -classpath <path> and -g.

Creating Java Program (Step 3)

Running

Java bytecode can be executed using a Java interpreter. The **class name** is passed as an argument to the interpreter.

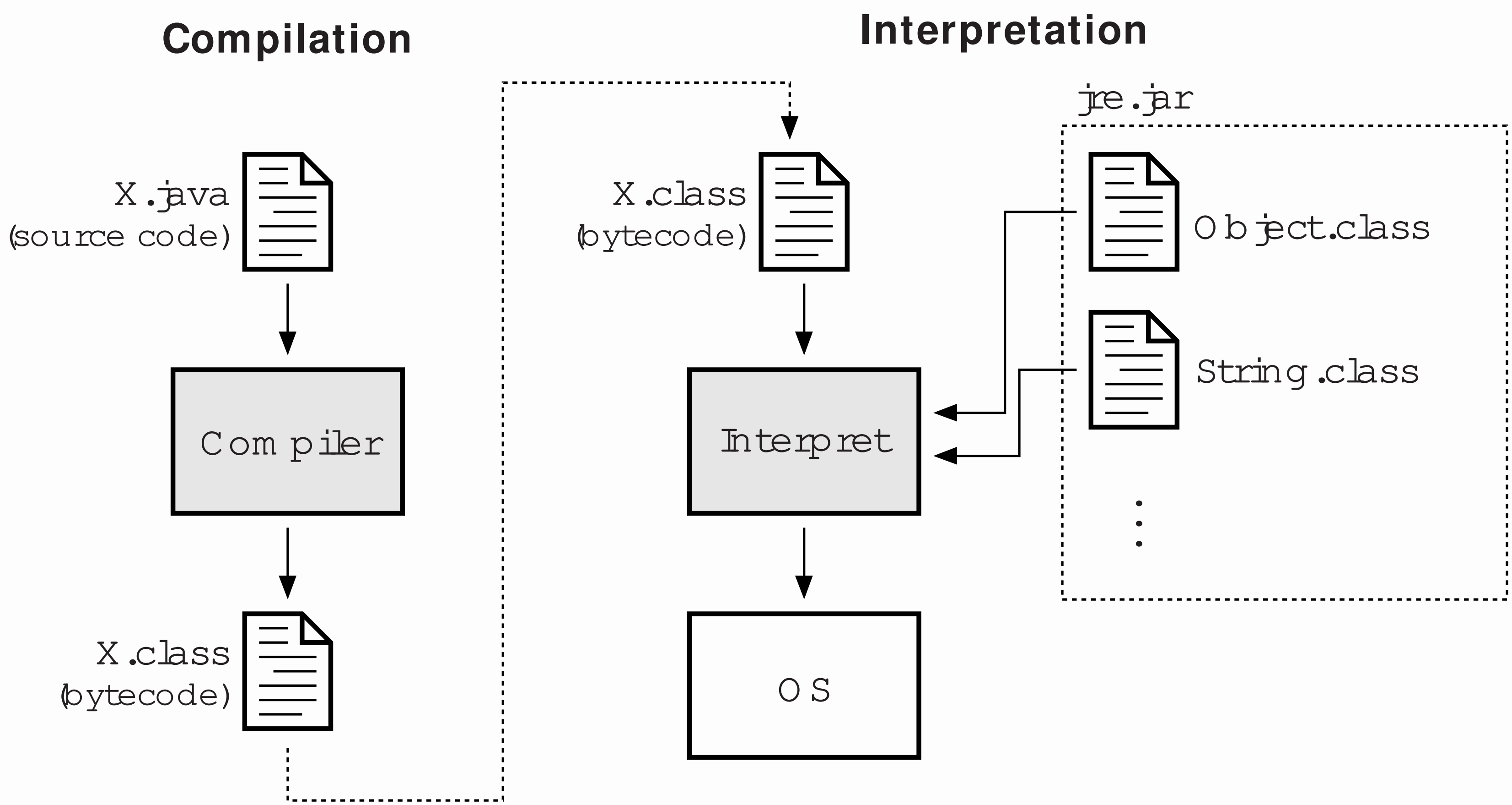
Environment variable CLASSPATH plays important role while executing the Java bytecode. It holds list of directories and libraries containg bytecodes being executed.

```
$ java HelloWorld
Hello, world...
```

Note: It is convenient to include the . (dot) in the CLASSPATH. The . ensures that bytecodes in the current working directory are successfully found by interpreter.

Java Environment

Java combines compilation and interpretation techniques. Java source code is first compiled into an intermediate language called *bytecode*. The bytecode helps make write once, run anywhere possible.



Note: Compilation happens just **once**; interpretation occurs **each time** the program is executed.

Bytecode

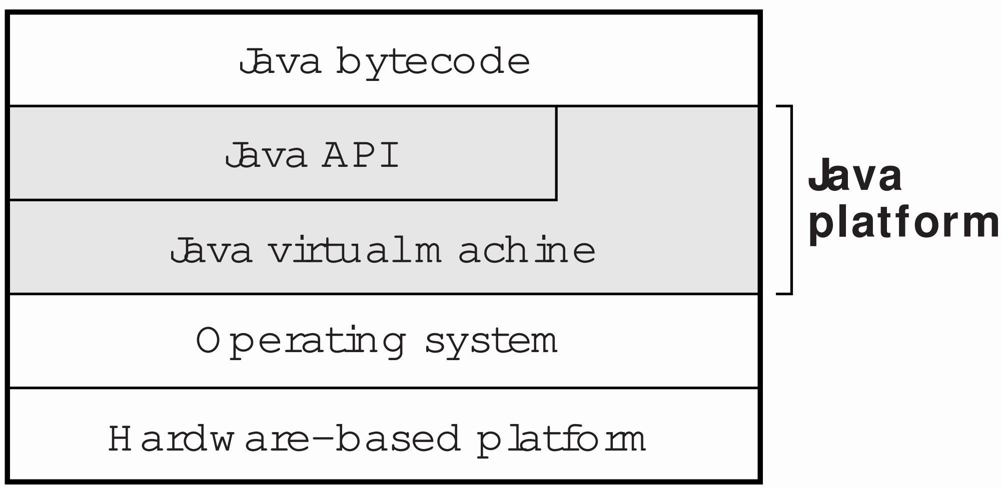
An example of bytecode produced by javap utility.

```
$ javap -c HelloWorld

Compiled from HelloWorld.java
public class HelloWorld extends java.lang.Object {
    public HelloWorld();
    public static void main(java.lang.String[]);
}

Method HelloWorld()
   0 aload_0
   1 invokespecial #1 <Method java.lang.Object()>
   4 return

Method void main(java.lang.String[])
   0 getstatic #2 <Field java.io.PrintStream out>
   3 ldc #3 <String "Hello, world...">
   5 invokevirtual #4 <Method void println(java.lang.String)>
   8 return
```

The Java Platform

A **platform** is the hardware or software environment in which a program runs. The **Java platform** differs from most other platforms in that it's a software-only platform that runs on top of other hardware-based platforms. The Java platform has two components:

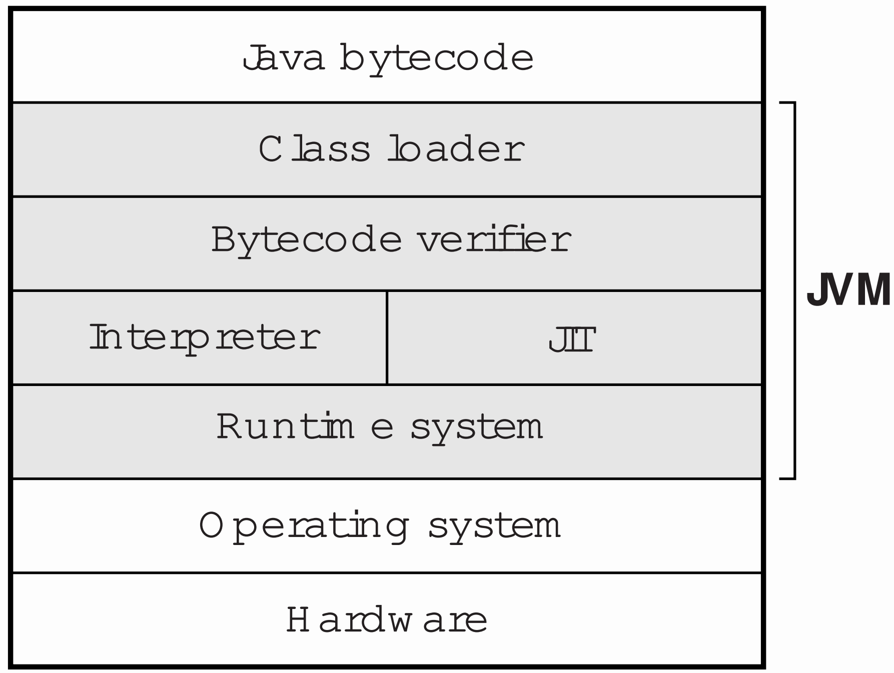
- Java virtual machine,
- Java application programming interface (API).



Note: The Java API is a large collection of **ready- made** software components that provide many useful capabilities. The Java API is grouped into libraries of related **classes** and **interfaces**; these libraries are known as **packages**.

Java Virtual Machine

Java bytecode is machine code for the **Java Virtual Machine** (JVM). Every Java interpreter is an implementation of a JVM.



Note: A Java program can be compiled into bytecodes on any platform that has a Java compiler. The bytecodes can then be run on any implementation of the JVM, regardless of operating system or hardware platform.