Principles of Software Development

1 – Introduction to Java

Background

- Invented by James Gosling at Sun Microsystems.
- Started out in 1990 as a programming language for consumer electronics (known as “Oak”).
- Redesigned in 1993 for Internet programming (renamed Java).

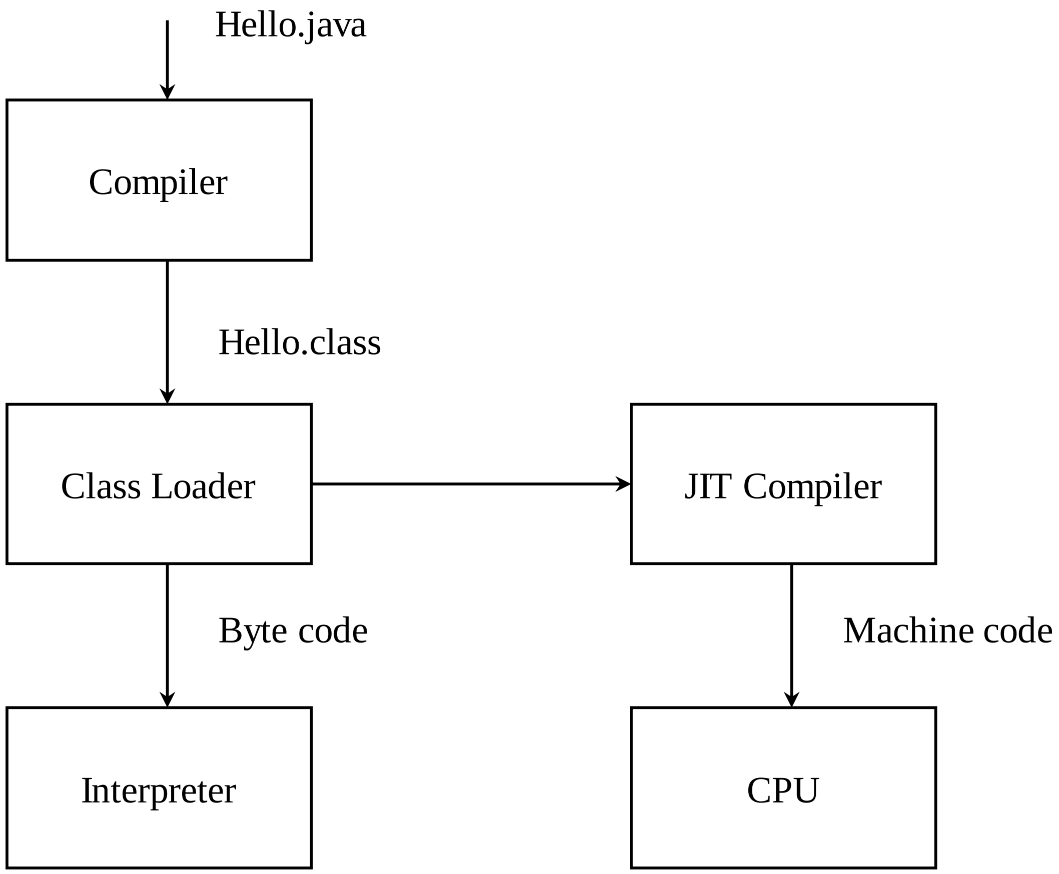
Major Features

- Uses a smaller number of language constructs, comparing to C++.
  - No pointers, structures, operator overloading, multiple inheritance, etc.
  - Does automatic garbage collection.
- Object-Oriented
  - Except for some well-defined primitive data types, everything is an object.
- Distributed
  - Designed to support applications and applets on networks.
- Multithreaded
  - Built-in support for threading and synchronization.

Major Features (continued)

- Interpreted
  - Java source code is compiled to *byte code*, instead of native machine code.
  - To run a Java program, the Java interpreter executes the byte code.
  - Byte code can be run on any system that implements the interpreter and run-time system (the Java Virtual Machine).
  - Java classes can be stored and used anywhere on the network.
- JIT Compiler
  - JIT, "Just in time" compilers, are also available.
- Other:
  - Run-time checking of array and string accesses: keeps accesses within bounds.
  - Java supports linking in of "native" code, written in some other compiled language (*native methods*).

Java Program Execution



Java Commands

- javac: invokes the compiler, converting source code into byte code.
- java: executes byte code, by invoking the Java Interpreter.
- appletviewer: runs applets. Are used for testing applets.
- jdb: invokes the Java Debugger (similar to gdb).
- javap: disassembles .class file
- javadoc: creates a java documentation
- Others.

Compiling and Running Java Programs

- Create your Java source code:
  - Use any text editor (Emacs) to write your java code.
  - Save the file using the .java suffix.
    - Example: **MyProg.java**
- Compile the source code into byte code, using javac command:

```
javac MyProg.java
```

  - This will produce the file MyProg.class
- Run the byte code on the JVM:

```
java MyProg
```

- The JDK and its documentation can be downloaded from Sun Microsystems Web Site.

Programming in Java

Anatomy of a Simple Java Programs

A Simple Java Program

```
public class SimpleJavaProgram {

   public static void main(String[] args) {
        int i;
        Integer j = new Integer(3);
        for (i = 0 ; i < 100; i++) {
           System.out.println(i + ": Hello Java programmers " + j);

           if (i == 3 && j >= 0)
                break;

           j--;
        }

   }
}
```

```
0: Hello Java programmers 3
1: Hello Java programmers 2
2: Hello Java programmers 1
3: Hello Java programmers 0
```

Second Example -Interactive

```
import java.util.Scanner;

public class SimpleJavaProgram2 {
   public static void main(String[] args){
       Scanner scan = new Scanner(System.in);
       System.out.println("Please enter your name: ");
       String name = scan.nextLine();
       System.out.println("Please enter your age: ");
       int age = scan.nextInt();
       System.out.println("Please enter your salary: ");
       double salary = scan.nextDouble();
       System.out.println("Please enter your salary: ");
       System.out.println("Name: " + name + " Age: ”
                                 + age + " Salary: ” + salary);
  }  // END OF MAIN
}    // END OF CLASS DEFINITION
```

```
Please enter your name:
Jim Boss
Please enter your age:
23
Please enter your salary:
3000
Please enter your salary:
Name: Jim Boss Age: 23 Salary: 3000.0
```

More on Scanner

- The scanner can also reads from a string:

```
String input = ”1 2 orange apple ";
Scanner s = new Scanner(input);
System.out.println(s.nextInt());
System.out.println(s.nextInt());
System.out.println(s.next());
System.out.println(s.next());
s.close();
```

- prints the following output:

```
1
2
orange
apple
```

Java Basic Constructs

- Variables
  - Mostly need to be allocated by using new
  - Except for preemptive data type:

    int, double, char, byte, float, boolean, etc…

  - The class objects, and arrays must be always allocated, by using operator new. In the following example x is a reference allocated to on the stack, pointing to an object of class Integer, on the heap:

    Integer x = new Integer(134);

Java basic constructs

- Constants:
  - Java uses the keyword final to declare a constant:

    final double d = 99.99;

    final int x = 22;
  - Objects of Java String class are also immutable objects:

    String s1 = “ABCD”;

    s1 = “XYZ”; // s1 now refers to a different memory space

Java Data Types

Primitive Data Types (continued)

Type	Contains	Bit size	Default values	Value Range
boolean	true or false	1	false	true/
char	Unicode chars	16	\u0000	'\u0000' (or 0) to '\uffff' (or 65,535)
byte	signed integer	8	0	-128 to 127
short	signed integer	16	0	-32768 to 32767
int	signed integer	32	0	-2^{31} to $2^{31} - 1$
long	signed integer	64	0	-2^{63} to $2^{63} -1$
float	floating point	32	0.0	
double	floating point	64	0.0	

If you are interested about the range of float and double please study the a detail discussion at: http://docs.oracle.com/javase/specs/jls/se8/html/jls-4.html#jls-4.2.3

What is Unicode Character

- Unicode is a computing industry standard for the consistent encoding, and representation of text expressed in many of the world's writing systems.
- The latest version of Unicode supports more than 110,000 characters.

```
System.out.println('\u00a5');      // Japan currency Yen -- ¥
System.out.println((char)0x2202); // Greek letter delta -- ∂
System.out.println('\u2202');      // Greek letter delta -- ∂
```

- To find out the decimal values for these hex numbers:

```
System.out.println(0x00a5);       // 165
System.out.println(0x2202);       // 8706
System.out.println(0x0040);       // 64
```

Data Types (continued)

- Each of the data types in the previous slide, except short and byte, have corresponding classes defined in the language:
  - Boolean, Character, Integer, Long, Float, and Double
    - Act as a “wrapper” around the primitive type.
    - Include useful constants and methods.
- Lets take a quick look at the class Integer, on the Oracle website:

  http://docs.oracle.com/javase/8/docs/api/java/lang/Integer.html

Comments and javadoc

- Comments can be specified with:

```
// comment

/* comment */

/** documentation
       comment
  */
```