MANCHESTER 1824
The University of Manchester

Introduction to Java

BIOL60201
Programming skills

Jean-Marc Schwartz
Faculty of Life Sciences

jean-marc.schwartz@manchester.ac.uk

Outline

1. Introduction to Java
    i. The Java language
    ii. 'Hello World' application
    iii. Variables and operators

Programming languages

- Programming languages can be categorised into three types:
  - Machine language:
    binary codes (strings of 0s and 1s);
    challenging and time-consuming, not portable.
  - Assembly language:
    using symbolic names for instructions and memory addresses;
    easier but still time-consuming, not portable.
  - High-level language:
    symbolic, closer to the English language;
    easier for software development, somewhat portable.

    Examples: Fortran, Pascal, Perl, C, C++, Java...

Programming languages

- Procedural languages contain a series of computational steps to be carried out in chronological order, usually organised into separate procedures or functions.

  Examples: Fortran, Pascal, Perl, C...

- Object-oriented languages encapsulate data and functions needed to manipulate them into structures called 'classes'.

  Examples: C++, Python, Java...

- A class could represent a real life concept or an abstract concept which has various attributes.
- Well-written classes can be reused by several programs.

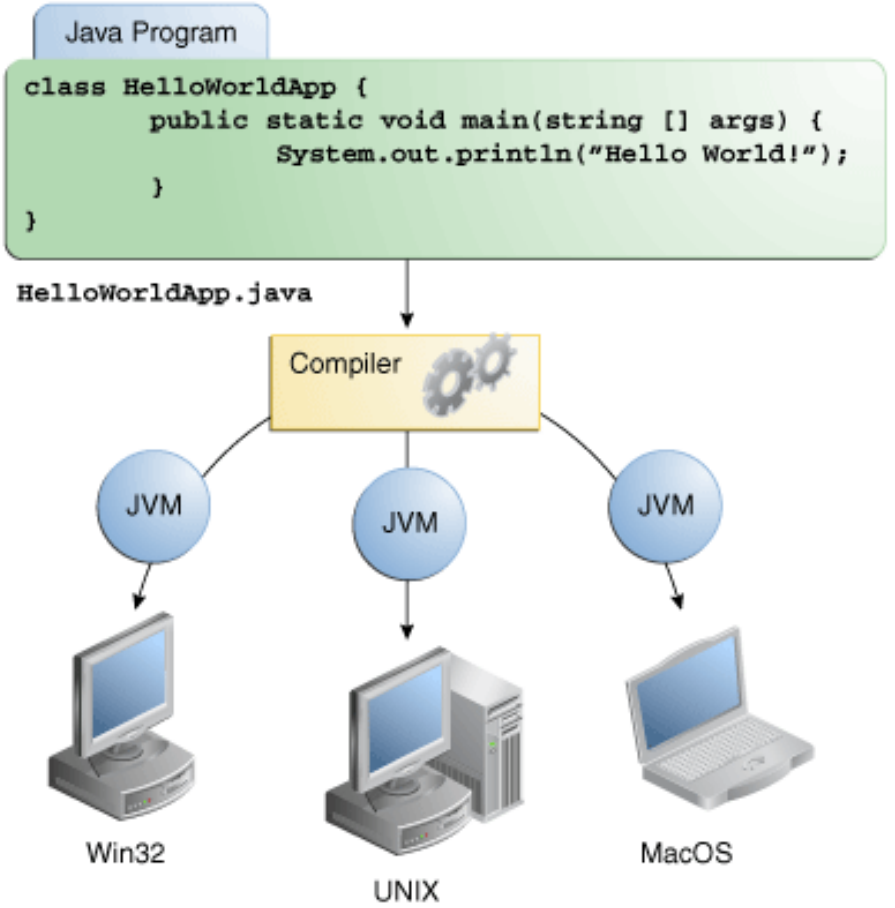
The Java language

- Java was introduced in 1995 by Sun Microsystems as a free, object-oriented language:
  - Syntax identical to that of C++ except for some advanced features.
  - Can be run within a web browser through applets.
  - Portable among most platforms thanks to a Java Virtual Machine.
  - Object-oriented language.
  - Open-source availability.
  - Extensive library of classes for data structures, graphical interfaces and networking.

- In 2009, Sun was acquired by Oracle which now pursues the development of Java.

Java Virtual Machine

- The Java Virtual Machine (JVM) simulates a virtual processor by transforming the compiled byte code into platform specific instructions.



Developing a Java application

- The Java Development Kit (JDK) provides a compiler to develop Java applications and a JVM to run them.

- The Java Runtime Environment (JRE) provides a standalone JVM for users who only need to run compiled Java applications.

- Both can be downloaded for free from:

  http://www.oracle.com/technetwork/java/javase/downloads/

Running a Java application

- Java source code is stored in a text file with the extension *.java*:
  MyProgram.java

- To compile the code, the compiler is called from the command line:
  `javac MyProgram.java`

- The compiler produces a file with the extension *.class* that contains the byte code:
  MyProgram.class

- To run the application, the JVM is called from the command line:
  `java MyProgram`

HelloWorld application

```
/* This program prints "Hello!"
*/

public class HelloWorld {

  public static void main(String args[]) {

    System.out.println("Hello!");
  }
}

// This is a comment: end of the program
```

Types of primitive variables

- Integer data types:
  `byte`, `short`, `int`, `long`
  These types differ in the memory size allocated to store each type, thus in the maximum and minimum allowed values.

- Floating-point data types:
  `float`, `double`

- Character data type (single character only):
  `char`

- Boolean data type (true or false):
  `boolean`

Declaring variables

- `int i = 10;`
- `double x = 1.7;`
- `char letter = 'a';`
- `boolean flag = true;`

Arithmetic operators

- Addition:
```
int sum = i + j;
```
- Subtraction:
```
int difference = i - j;
```
- Multiplication:
```
double product = x * y;
```
- Division:
```
double quotient = x / y;
```
- Remainder after division:
```
int modulus = i % j;
```

Division

- The division of two integers only calculates an integer result.

  Example:

```
int i = 12;
int j = 5;
int k = i / j;   // k equals 2
```

- To obtain a fractional result, variables must be declared as float or double:

  Example:

```
double x = 12.0;
double y = 5.0;
double z = x / y;   // z equals 2.4
```

Casting

- When calculations are performed using mixed data types, lower-precision types are converted to higher precision types.

```
double x = 5.0, z;
int j = 2;
z = x / j;   // z equals 2.5
```

- However, an explicit type casting is necessary to convert to a lower precision type.

```
double x = 5.0, y = 2.0;
int k;
k = (int) x / y;   // k equals 2
```

Operator precedence

- `*`, `%`, `/` have precedence over `+`, `-`

  Example:
  `1 + 2 * 3 - 4` yields 3
  `12 / 2 * 3` yields 18

- Parentheses can be used to force a different order of calculations.

  Example:
  `(1 + 2) * 3 - 4` yields 5
  `12 / (2 * 3)` yields 2

Shortcut operators

- `a++` (or `++a`) is equivalent to `a = a + 1`
- `a--` (or `--a`) is equivalent to `a = a - 1`
- `a+=3` is equivalent to `a = a + 3`
- `a-=3` is equivalent to `a = a - 3`
- `a/=3` is equivalent to `a = a / 3`
- `a%=3` is equivalent to `a = a % 3`

Constants

- When the value of an item should not change during program execution, it is advised to define it as a constant.

```
final int NB_MONTHS = 12;
```

- By convention, names of constants are written in CAPITALS.

- Any attempt to change the value of a constant will generate a compiler error.

User input

- The *Scanner* class provides methods for reading data from the Java console.
- It is defined in the *java.util* package, so programs need to include the following *import* statement:

```
import java.util.Scanner;
```

- A *Scanner* object is then instantiated by:

```
Scanner scan = new Scanner(System.in);
```

- Different types of input are read using different methods:

```
next()          // String
nextInt()       // Integer
nextDouble()    // Double
nextLine()      // Whole line
// etc
```

User input

- Example:

```
import java.util.Scanner;

public class AgeInput {

  public static void main(String[] args) {

    Scanner scan = new Scanner(System.in);
    System.out.print("Enter your age: ");

    int age = scan.nextInt();

    System.out.println("Your age is " + age);

    scan.close();
  }
}
```

Eclipse

- Eclipse is an open-source platform for developing and managing software in Java and other programming languages (Integrated Development Environment).

- It offers a convenient user interface to edit, debug and run Java programs.

- Eclipse IDE for Java Developers can be downloaded for free from:

  http://www.eclipse.org/downloads/

References

- Online Java Tutorial:
  http://docs.oracle.com/javase/tutorial/

- Java Platform Technical Documentation:
  http://docs.oracle.com/javase/8/docs/api/

- Textbook:
  Anderson J, Franceschi H. *Java Illuminated*, Jones & Bartlett Learning, Third Edition (2012).