

Programming I

Getting started

1




Department of Computer Science

Programming I: Getting started

- Definitions
- What is programming?
- What is Java?
- Writing your first program

2




Department of Computer Science

Program

- A sequence of instructions carried out by a computer (or *run*, or *executed*).
- The sequence is typically long and complex.
- Running a program will result in millions or billions of instructions being executed.
- The instruction sequence has to be correct or the program will fail.

3



Department of Computer Science

A very simple program!

```
public class Hello
{
    public static void main(String[] args)
    {
        System.out.println("Hello World");
    }
}
```

4

Department of Computer Science



Programmer

- Person who writes programs!
- Responsible for identifying the correct sequence of instructions required and writing them down.

5

Department of Computer Science



Processor

- Microprocessor, CPU, chip.
- The computer hardware that executes program instructions (machine code).
- Intel Pentium™, Sparc, Motorola PowerPC
 - Each has its own specific *instruction set*.

6

Department of Computer Science



Application

- A program that does something useful for the *end user*.
 - Word processor, spreadsheet, Quake II, etc.
- Application programming is the process of *developing* applications.

7

Department of Computer Science



Systems Programming


- Process of developing *operating system* software.
- The operating system controls the computer.
- DOS, Unix, Linux, Windows NT, etc.

8

Department of Computer Science



Software

- More general
- Any computer instructions written to be executed on *hardware*, 
- 2 categories
 - Systems software (for the computer)
 - operating systems,
 - utility programs,
 - Applications software (for users)
- Untouchable (Software) \neq touchable (Hardware)

9

Department of Computer Science



Programming

- The process of writing programs.

– Requires 5 important steps:

- With a pen and paper! {
- Analysis - what is the program meant to do?
 - Design - how is the program structured?
- On a computer {
- Coding - writing the program code.
 - Debugging - fixing errors.
 - Testing - making sure the program does the right thing.

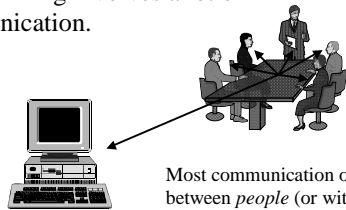
10

Department of Computer Science



Communication

- Programming involves a lot of communication.



Most communication occurs between *people* (or with yourself!)

11

Department of Computer Science



Communication between people

- Speech, writing, drawing, diagrams, etc.
- Very familiar but to talk about a program with other people you need to be quite precise.
- To describe a program to a computer you have to be absolutely precise.

12

Department of Computer Science



With a computer: Programming Language

- A *textual* language used to write a program.
- A language used to “communicate with a computer”.
- The meaning of what you write has to be completely and precisely defined.
- *Java* is a programming language.
- *Implementation*: writing a program.

13

Department of Computer Science 

Syntax

- Syntax describes the *grammatical rules* of a language.
 - Valid words.
 - Punctuation.
 - Sentence construction.
 - Rules of use.
- Programs must be syntactically correct.

14

Department of Computer Science 

Semantics

- Semantics give the *meaning* of what you write with a language.
- A program must be semantically correct to do what you expect.
- A programming language must precisely define the meaning of every *statement* that can be written with it.

15

Department of Computer Science 

Syntax v. Semantics

“This sentence is an elephant.”

- Grammatically correct but no sensible meaning.

“I never tell the truth.”

- Grammatically correct but logically inconsistent.

16

Department of Computer Science



Nonsense programs...

- A program can be syntactically correct but not do anything useful or sensible.
- However, what a program does can always be exactly determined.

is that true?

17

Department of Computer Science



Determinism v. Non-Determinism

- What a deterministic program does *can* always be known.
- The behaviour of non-deterministic programs can't be predicted reliably.

Your programs should be deterministic!

But non-deterministic programs can be written with Java.

18

Department of Computer Science



A Program!

```
public class Hello
{
    public static void main(String[] args)
    {
        System.out.println("Hello World");
    }
}
```

Ordinary text written with an editor.
Called *Source Code*.

Obeys *syntax* rules & *semantics* of Java.

19

Department of Computer Science



Compilation

- A computer cannot directly understand or run source code!
- A translation from source code to processor instructions has to be performed.
- This is known as *compilation*.

20

Department of Computer Science



The Compiler

- Fortunately, a tool called a *Compiler* can translate the text to processor instruction.
- The compiler knows and checks
 - all syntax rules
 - only some of the semantics.
- The compiler is itself a program.

21

Department of Computer Science



Starting to Program

- We want to get you started as soon as possible!
- But at first you will have to take a lot of things on trust.
- We will return to the details later on.

22

Department of Computer Science



Writing a Java Program

- Analyse the problem to solve, with paper and pen!
- Sketch an algorithm, with paper and pen!
- Use an editor to type in or edit the program source code
 - Save the code to a file
 - Compile the file with the Java compiler
 - Run the program and see what happens.
 - Fix the *bugs*!

23

Department of Computer Science



Hello

```
// This is our first program! ← A comment
Public class Hello ← Our program will be called Hello.
{
    public static void main(String[] args)
    {
        System.out.println("Hello World") ;
    }
} ← Save in a file called Hello.java
```

24

Department of Computer Science



Java Application

- Contain a class definition which you can call whatever you want
 - (“Hello” in this program).
 - But it is better (compulsory ☺) if the name is consistent with the effect of the program
- A method called main() is the entry point of the program. It is executed first and has a fixed signature.

25

Department of Computer Science



A second look at Hello

- Comments :
 - Start with // end with the end of the line.
- The program has one class definition which we have called ‘Hello’ and contains one method called main().

```
// This is our first program!  
public class Hello  
{  
    public static void main(String[] args)  
    {  
        System.out.println("Hello World");  
    }  
}
```

26

Department of Computer Science



Comments

- Comments are very important
- A source code without comments is unreadable and confusing

27

Department of Computer Science



Different types of comment

Java documentation

Special comments:

```
/**
```

Text

```
*/
```

Usual comments

```
//Text
```

- Name of authors, date
- Name and goal of the program
- General definition of a class
- Specification of a method
- Explanation of the action of a group of sequences

28

Department of Computer Science



Example

```
/**
 * <dl>
 * <dt>Purpose: First and very simple program to test java.
 * <dd>
 *
 * <dt>Description:
 * <dd> This program displays hello world on a command window.
 *
 * </dl>
 *
 * @author Danny Alexander
 * @author Céline Loscos
 * @version $Date: 2002/09/13$
 */
public class Hello
{
    public static void main(String[] args)
    {
        //Display 'hello world' on the screen
        System.out.println("Hello World");
    }
}
```

29

Department of Computer Science



Java Compiler

- The Java compiler is called *jikes* or *javac*
 - To compile use:
javac Hello.java

This will create a file called Hello.class

Java source code file names must always end with .java

30

Department of Computer Science



Running a Java Program

- To run a Java program use *java*.

- java Hello

Name the program
you want to run.

This is the Java *interpreter* which
actually runs the program.

31

Department of Computer Science



The Java Virtual Machine (JVM)

- Java programs are compiled to *bytecodes* for a *virtual processor*.
- The command `java` runs the JVM which simulates the virtual processor.
- So your program is run by the JVM which is, in turn, run by the real processor.

32

Department of Computer Science



Why?

- A Java program can run on any real processor that can run the JVM.
- “Compile once, run everywhere”

Well almost - a few bugs still
need to be sorted out.

33

Department of Computer Science



So, where did Java come from?

- Designed by a group at Sun Microsystems.
- Originally called Oak - a language for programming consumer devices (Talkie Toaster!).
- Renamed to Java and moved to the web.
- Developed into a full scale application programming language.

34

Department of Computer Science



Summary

- Defined some basic terms.
- Introduced the ideas of a programming language and compilation.
- Seen how to write, compile and run small programs.

35

Department of Computer Science



Course work notes

36

Department of Computer Science



Drawing Shapes

- A number of exercise questions ask you to write programs that draw shapes.
- You are given a complete program which you can copy and then edit.

37

Department of Computer Science



Program meaning

- At the moment you won't understand most of the program!!
- Don't worry.
- Read the comments for guidance. (The lines starting with //)

38

Department of Computer Science



What do you change?

```
// This part of the program does the actual drawing.  
public void paint(Graphics g)  
{  
    // You add/change the statements here to draw  
    // the picture you want.  
    // For example, draw a diagonal line.  
    g.drawLine(0,0,300,300) ;  
}
```

Make changes here. See the notes.

39

Department of Computer Science



Object

- What does `g.drawLine()` mean? What is `g`?
- `g` is a *reference* to an *object*.
- The object knows how to draw.
- This *will* become clear later in the course!

40

Department of Computer Science



Anything else to change?

- Yes, the name of the program.
- See the line that says:
class **Drawing1** extends Panel

↑
Here's the name. Change it!

- Save the program to a *new* .java file using your new name.

41

Department of Computer Science



Yet more to change?

- Yes.
- Wherever you see the name `Drawing1` in the program, replace it with the new name.

Drawing1 drawing = new **Drawing1**() ;

↙ ↘
Change these as well.

42

Department of Computer Science



Drawing something different

// This part of the program does the actual drawing.

```
public void paint(Graphics g)
```

```
{
```

```
    // You add/change the statements here to draw
```

```
    // the picture you want.
```

```
    g.drawRect(150,150,50,50);
```

```
    g.fillRect(20,20,50,50);
```

```
}
```

New lines of code.

The order of the lines give the sequence by which the picture is drawn.

43

Department of Computer Science



What else can you draw?

- g.drawArc(x, y, width, height, startAngle, arcAngle)
- g.drawLine(x1,y1,x2,y2)
- g.drawOval(x,y,width,height)
- g.drawRect(x,y,width,height)
- g.drawString(text, x,y)

A String is a line of text.

44

Department of Computer Science



Drawing a more complicated picture

- Work out how to draw a complicated picture by drawing it using a series of simpler shapes.

Problem decomposition.

45

Department of Computer Science