IPM 13/14 – P1
Introduction to Java

Licenciatura em Ciência de Computadores

Miguel Tavares Coimbra

Summary

- Introduction to Java
  - Why Java?
  - Characteristics
  - Syntax
  - Eclipse IDE

Why Java?

- We have been studying Human-Machine Interaction.
- We have talked about the design triangle:
  - Objective, Technology, User
- But how do we test our solution?
  - We need technology.
  - Java is simple. (although we could also have chosen other languages...)

Our course and Java

- Tutorials will involve programming in Java.
- More specifically with:
  - Java AWT
  - Java Swing
- Why?
  - Easy.
  - Close to other languages you have learned (C).
  - Simple libraries for graphical user interfaces.
- How?
  - We will use the Eclipse IDE.

Generic characteristics of Java - I

- The nice stuff
  - Abstract machine (runs on every OS).
  - Memory management.
  - Object-oriented.
  - Vast API
- Simple Graphic libraries.
  - AWT
  - Swing

Generic characteristics of Java - II

- The not so nice stuff
  - API instability.
  - API complexity (sometimes...).
  - Low efficiency
  - Half as fast as C
- Some references:
  - Tutoriais e FAQs da Sun http://java.sun.com/docs/books/tutorial/
  - Java Linux http://www.blackdown.org/java-linux.html

Basic syntax

- Similarity with C/C++:
  - Instructions syntax
  - Variable declarations
  - Expressions
  - Explicit conversions (type casting)
- Specific characteristics:
  - Exclusively object-oriented
    - All code is inside a *class.*
    - Everything is an *object*.
    - Exception: Intrinsic data types (real numbers, booleans and characters).
  - Memory management
    - Garbage collection

More syntax

- The *main* ‘function’ executes the program.
- Instructions end with “;”
- A *class* is equivalent to a C *module*.
- The delimiters for comments are:
  - /* and */
  - // until the end of the line.

Hello World in Java

```
/*
   Hello World in Java
*/
class ola
{
   public static void main(String[] arg)
        { // let's write "hello World"
                System.out.println("hello World");
        }
}
```

Variables, expressions and control

- Variables are declared and used like in C
- Expressions like in C except:
  - *Operator* + is overloaded.
- Execution control resembles C (boolean tests)
- Initial parameters are an Array of Strings

```
// ECHO in java
class echo
{
    public static void main(String[] arg)
     {
    int i;
    for(i=0; i<arg.length; i++)
            System.out.print(arg[i]+“ “ );
    System.out.println("");
    }
}
```

Methods

- Sub-routines are used just like in C.
- They are called ***Methods***, and are inside classes.
- Need to declare types for arguments and return value.

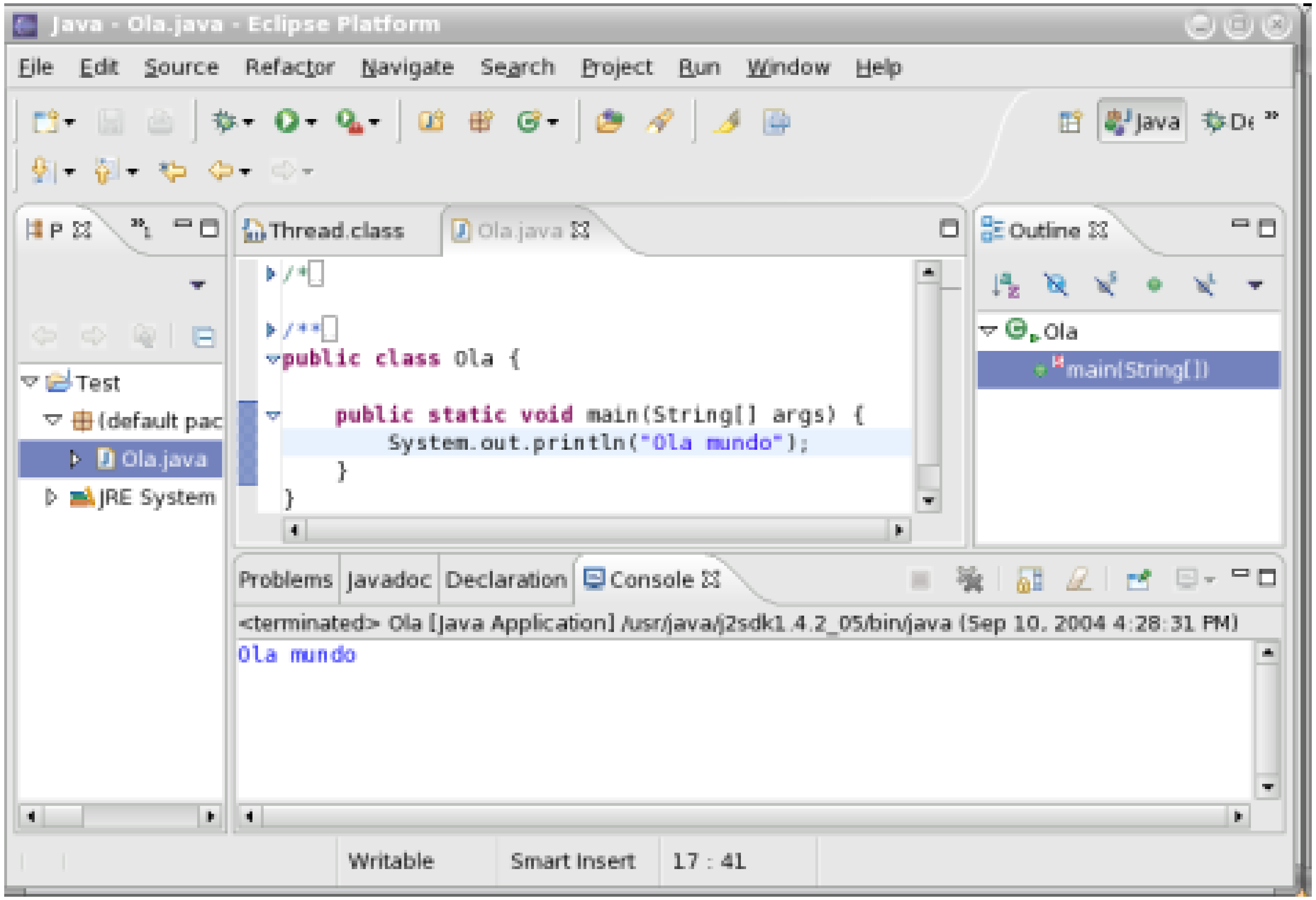
```
// Factorial
class fact {
        public static void main (String args[])
        {
        int n;
        System.out.println("n\tn!");
        for(n=0; n<10; n++)
                System.out.println(n+"\t"+fact(n));
        }

        int fact(int n)
        {
                if(n==0) return 1;
                else return n*fact(n-1);
        }
}
```

Java IDE

- Now that you are an expert in Java
  - How do we program with it?
- We could use command-line compiling
  - But we are not going to...
- We will use an **Integrated Development Environment** (IDE).
  - Easier to edit code, easier to execute and debug, easier to get help... **Easier!**
  - This is not a programming course! We are interested in HCI.

Eclipse IDE



```
/*
/**
public class Ola {

    public static void main(String[] args) {
        System.out.println("Ola mundo");
    }
}
```

```
<terminated> Ola [Java Application] /usr/java/j2sdk1.4.2_05/bin/java (Sep 10, 2004 4:28:31 PM)
Ola mundo
```

What are we missing?

- What are *Objects*?
  - Referencing
  - Visibility
- What are Classes?
  - Extensions
  - Interfaces
- Graphic Toolkits
  - AWT
  - Swing

Resources

1. Developer Resources for Java Technology
   http://java.sun.com/
2. Essentials of the Java programming language
   http://java.sun.com/developer/onlineTraining/Programming/BasicJava1/