

# Componentes del Lenguaje

Laboratorio de Programación 2009

# Programación Orientada a Objetos

- Un breve repaso

# Así se ve el código Java

```
int age = 23;  
String name = "Diego Torres";  
Singer diego = new Singer(name, age);  
int audience = 5;  
if (x < 20) diego.sing("Tratar de estar mejor");  
  
System.out.println("Señoras y señores con uds:");  
System.out.println(diego.getName);  
  
String number = "3";  
int z = Integer.parseInt(number);  
  
try{  
    readTheFile("myFile.txt");  
}catch (FileNotFoundException e) {  
    System.out.println("File not found.");  
}
```

# Componentes del lenguaje

- En Java existen dos categorías de tipos de datos:

- Tipos primitivos:

- **int** 32-bit complemento a dos
- **boolean** true o false
- **char** 16-bit caracteres Unicode
- **byte** 8-bit complemento a dos
- **short** 16-bit complemento a dos
- **long** 64-bit complemento a dos
- **float** 32-bit IEEE 754
- **double** 64-bit IEEE 754

- Clases o interfaces

# Componentes del lenguaje

- Para los tipos primitivos existen clases “wrappers”:
  - `int` ☐ `Integer`
  - `boolean` ☐ `Boolean`
  - `char` ☐ `Character`
  - ...
- ¿Para qué utilizar las clases wrappers?
- Java no es un lenguaje orientado a objetos “puro”.

# Componentes del lenguaje - Strings


- Clase String:

- No es un tipo primitivo.
- Los String son instancias de la clase **java.lang.String**.
- El compilador trata a los String como si fuesen tipos primitivos del lenguaje.
- La clase tiene varios métodos para trabajar con ellos.
- Como crear uno:  

```
String saludo = "Hola";  
String otroSaludo = new String("¿Cómo andás?");
```

# Componentes del lenguaje

## ARREGLOS

- Los arrays de Java (vectores, matrices, hiper-matrices de más de dos dimensiones) se tratan como objetos de una clase predefinida.
- Ejemplo:
  - `int[] arregloDeEnteros;`
  - Declara un arreglo de enteros pero no inicializa ni aloca memoria.
- Pueden declararse arreglos de más de una dimensión:
  - `int[][] matrizDeEnteros;`
- Se aloca memoria para un arreglo declarado usando **new**:
  - `arregloDeEnteros = new int[5];`
  - `matrizDeEnteros = new int[5][4];`
  - Son “zero-based”. 
- Acceso:
  - `Int[] val = matrizDeEnteros[2];`
  - `matrizDeEnteros[5][3] = 4;`

Error en tiempo  
de ejecución

# Características del Lenguaje

- Las variables de tipo primitivo alocan valores simples
- Las variables de tipos referencia son referencias a objetos
- Los tamaños de una variable definida son independientes de la plataforma.
- La asignación y liberación de memoria está a cargo de la JVM.



# Características del Lenguaje

- Rango de caracteres Unicode
- Son prohibidos **keywords** y valores como **true** y **false**
- Distinción entre mayúsculas y minúsculas
- Palabras que comienzan en minúscula para nombres de variables y con mayúscula para los nombres de clases

# Características del Lenguaje

- Asignación: =

i=i+3;

Otros como C: i += 3; i++;

- Aritméticos: + - \* / %

i+4\*3;

Otros como C: i++;

- Lógicos: & && | || !

(i != null) && (3\*i > 4)

(i>3) & (3\*i > 4)

- Relacionales y Condicionales: > < >= <= == !=

2 >= 3

# Estructuras de Control

**if**                    **if** (x==3) {  
                         System.out.println("Tres");  
                         }

**if / else**           **if** (x==1) {  
                         System.out.println("Uno");  
                         }**else** {  
                         System.out.println("Distinto de uno"); }  
                         }

**while**  
**e**                    **while** (count != -1){  
                         count++;  
                         System.out.println("Faltan "  
                         + count);  
                         }

# Estructuras de Control

## **switch**

```
switch (expresion) {  
  case 1: { sentenciasNro1; break; }  
  case 2: { sentenciasNro2; break; }  
  case 3: { sentenciasNro3; break; }  
  default: {caso default; break; }  
}
```

## **for**

```
for (int i=0; i<9 ; i++ ) {  
  System.out.println("El valor de i es " + i);  
}
```

# Estructura básica de un programa

- Un ejemplo:

```
import java.io.*;

public class Count {

    public static void main (String[] args){
        String in = "Hola que tal";
        char c;
        int i = 0;
        while (i < in.length()){
            c = in.charAt(i);
            System.out.println("El carácter de la posicion"
            + i + "es: " + c);
        }
    }
}
```

# Componentes del lenguaje

- Comentarios:

- Existen 3 tipos:

Por línea: //

- Bloque de código: /\* \*/

JavaDoc: /\*\* \*/

# Componentes del lenguaje

```
package helloWorld ;
```

```
/** Mi primer programa de ejemplo.  
 * Imprime un String con el texto "Hello World"  
 * @author XXX  
 * @version 2.0  
 */
```

```
public class HelloWorld {
```

```
/** Punto de entrada único a la clase y la aplicación  
 * @param args arreglo de argumentos String  
 * @return No retorna valor  
 * @exception No son levantadas excepciones  
 */
```

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

```
}
```

# Componentes del lenguaje

- Ejercicio:
  - Recorrer el siguiente **arreglo** de números sumando en una variable los números con valores **pares** y en otra los que poseen valores **impares**. Recorrerlo utilizando un **for** y luego utilizando un **while**.

**//La operación *modulo* es %**

**int[] arreglo = new int[3];**

**arreglo = {1,2,3,4,5,6,7}**

**int pares;**

**int impares;**



# Ejercicio – Arreglos y Strings

- Al ejecutar un programa en la ventana de comandos, el usuario puede especificar argumentos: `java Hello Hola que tal`
- Los argumentos "Hola", "que" y "tal" se reciben en un arreglo de strings en el main .
- Escribir una clase Echo que en cuyo main imprima en la salida estándar los valores recibidos como parámetros.
- Ahora imprima solamente aquellos que pueden llegar a ser verbos escritos en infinitivo (terminan en ar, er o ir).

# ¿Qué es Eclipse?

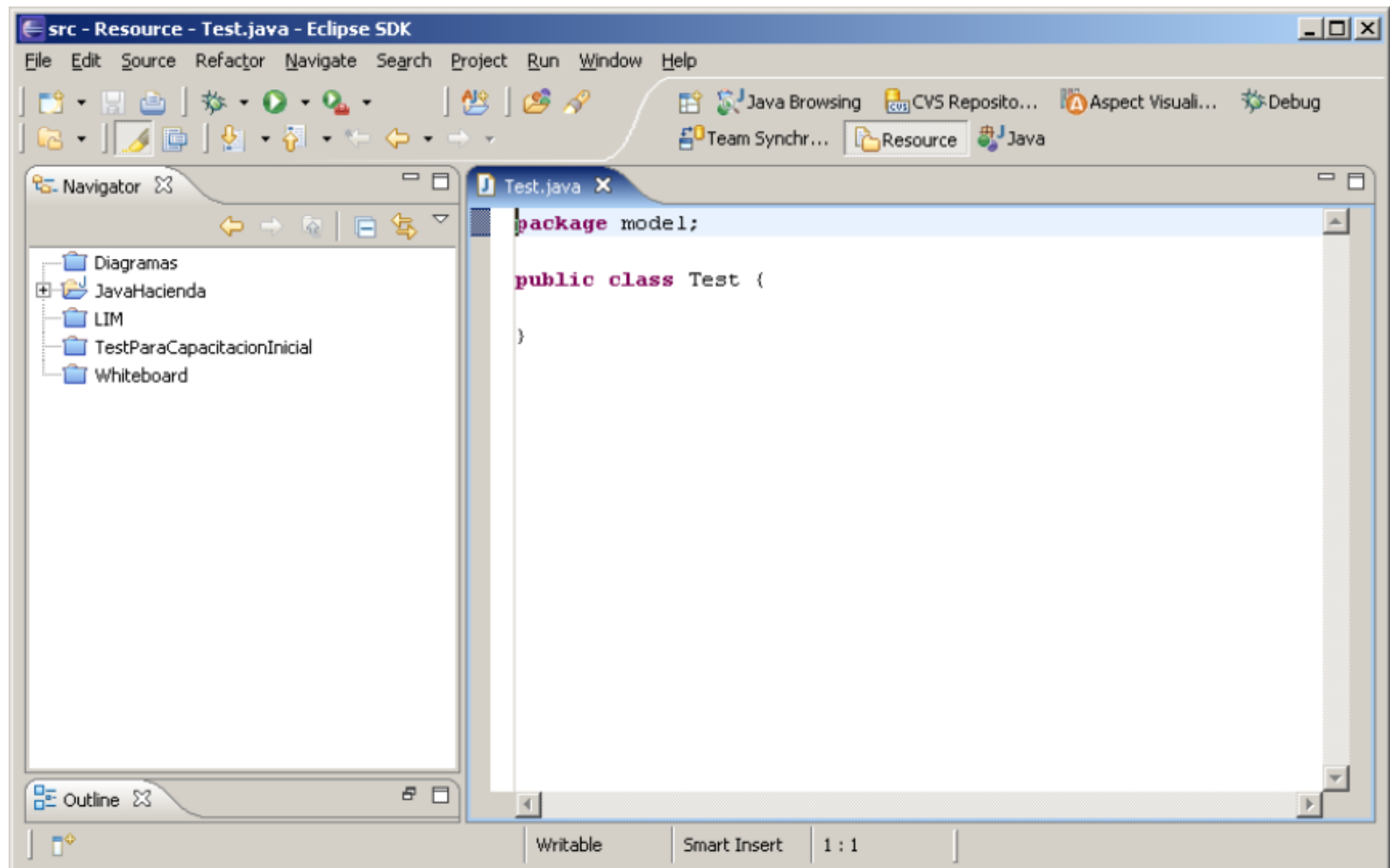
“An IDE for everything and nothing in particular”  
(Una IDE para todo y para nada en particular)

Framework de propósito general basado en plug-ins. La versión base contiene el JDT, plug-in para desarrollo en Java.

Eclipse está desarrollado en Java.

Eclipse es Libre !

# Vista General




# Empezando a programar

- Crear un proyecto Java.
- Selección de JVM / JRE
- Opciones de compilación
- Ubicación del código fuente y el binario

# Crear paquetes y clases

- Varios formas de creación
  - Mediante iconos en la barra de tareas.
  - Menú File/New/Package o File/New/Class
  - Mediante menú contextual utilizando el mouse.


# Ejecutar una clase

- Solo pueden ejecutarse clases que posean el método *main*.
- Eclipse las identifica con 
- Seguir los siguientes pasos.
  - Seleccionar la clase.
  - En el menú: Run/Run as/Java Application

# Ejercicio

- Hola mundo
- Ejercicios de las palabras en el main
- Cómo ejecutar estos programas con parámetros desde Eclipse.

# Ejercicio

- Construya un programa java que reciba como parámetros una serie de palabras. De las palabras se debe retornar cierta información.
- 1) Retornar aquellos verbos en infinitivo. Para ello debe verificar que las terminaciones sean ar er o ir.
- 2) Retornar la frase o las palabras que se reciben como parámetro en el orden inverso al que las reciben.
-  Retornar la cantidad de letras de las palabras. Por ejemplo si los parámetros recibidos son "hola" "que" "tal?" la respuesta debe ser: "4 3 4".
- 4) Eliminar las repeticiones de palabras.



# Compilación

- No hay un botón de compilación.
- Compilación bajo demanda.
- Algunos problemas.
- Solución basada en compilación forzada