La Sénia
Institut d'Educació Secundària
Paiporta

Unión Europea
Fondo Social Europeo
El FSE invierte en tu futuro

Codificando en Java

Curso 2023 – 2024

1. Introducción

Tras el diseño se realiza el proceso de codificación.

El programador recibe las especificaciones del diseño y las transforma en instrucciones escritas en un lenguaje de programación.

A este conjunto de instrucciones se le llama **código fuente**.

En cualquier **proyecto** en el que trabaja un grupo de personas debe haber unas **normas de codificación y estilo**, claras y homogéneas.

Facilitar las tareas de **corrección** y **mantenimiento** de los programas, sobre todo cuando se realizan por personas que no los han desarrollado.

1. Introducción

```java
import java.util.Scanner;

public class Suma {

     public static void main (String[] args){
          int suma = 0;
          int contador = 0;

           while (contador < 10) {
                contador++;
                suma = suma + contador;
           }
           System.out.println("Suma => " + suma);
     }
}
```

```java
import java.util.Scanner;
public class
Suma {
public static

void main (String[] args){
  int suma = 0; int contador = 0;
  while (contador < 10)
  {    contador++;

  suma = suma + contador; }
  System.out.println("Suma => " + suma);
    }
}
```

2. Normas y estilo

2.1 Nombres de ficheros

La extensión para los ficheros de código fuente es *.java*

La extensión para los ficheros compilados es *.class*

2.2 Organización de ficheros

- Cada fichero debe contener una sola clase pública y debe ser la Primera.

- Las clases privadas e interfaces asociados con esa clase pública se pueden poner en el mismo fichero después de la clase pública.

- Las secciones en las que se divide el fichero son:
  - Comentarios
  - Sentencias del tipo *package* e *import*
  - Declaraciones de clases e interfaces

2.2 Organización de ficheros

Comentarios:

- Todos los ficheros fuente deben comenzar con un comentario que muestre:
  - El nombre de la clase
  - Información de la versión
  - La fecha de creación y/o modificación
  - Aviso de derechos de autor

```
/**
* Nombre de la clase
*
* @version
*
* @since
*
* @author
*
*/
```

2.2 Organización de ficheros

Package e import:

- Van después de los comentarios, la sentencia **package** va delante de **import**.

Ejemplo:

```
package paquete.ejemplo;
import java.util.ArrayList;
```

2.2 Organización de ficheros

Clases e Interfaces:

- Comentario de documentación (/** … */)
- Sentencia *class* o *interface*
- Variables estáticas, en este orden: públicas, protegidas y luego privadas.
- Variables de instancia, en este orden: públicas, protegidas y luego privadas.
- Constructores
- Métodos. Se agrupan por su funcionalidad, no por su alcance.

2.3 Identación

- Como norma general se usarán cuatro espacios.
- La longitud de las líneas de código no debe superar 80 caracteres.
- La longitud de las líneas de comentarios no debe superar 70 caracteres.
- Cuando una expresión no cabe en una solo línea: romper después de una coma o antes de un operador, alinear la nueva línea al principio de la anterior.

2.4 Comentarios

- Los comentarios deben contener solo la información que es relevante para la lectura y la comprensión del programa.

- Existen 2 tipos de comentarios:

  De documentación: Están destinados a describir la especificación del código. Se utilizan para describir las clases Java, las interfaces, los constructores, los métodos, ...

  De implementación: Son para comentar algo acerca de la aplicación particular, de qué está realizando el algoritmo, ...

2.5 Declaraciones

- Se recomienda declarar una variable por línea.
- Inicializar las variables locales donde están declaradas y colocarlas al comienzo del bloque.
- En las clases e interfaces:
  - No se ponen espacios en blanco entre el nombre del método y el paréntesis “(”.
  - La llave de apertura “{” se coloca en la misma línea que el nombre del método o clase.
  - La llave de cierre “}” aparece en una línea aparte.

2.6 Sentencias

Cada línea debe contener una sentencia.

Si hay un bloque de sentencias, este debe ser sangrado con respecto a la sentencia que lo genera y debe estar entre llaves aunque solo tenga una sentencia.

2.7 Separaciones

Mejoran la legibilidad del código. Se utilizan:

Dos líneas en blanco: entre las definiciones de clases e interfaces.

Una línea en blanco: entre los métodos, la definición de las variables locales de un método y la primera instrucción, antes de un comentario, entre secciones lógicas de un método para mejorar la legibilidad.

Un carácter en blanco: entre una palabra y un paréntesis, después de una coma, los operadores binarios menos el punto, las expresiones del for, y entre un cast y la variable.

2.8 Nombres

Las normas para asignar nombres a los elementos en Java son las siguientes:

Paquetes: El nombre se escribe en minúscula, se pueden utilizar puntos para reflejar algún tipo de jerarquía. Ejemplo: *java.io*
Clases e interfaces: Deben ser sustantivos descriptivos. La primera letra siempre será en mayúscula. Ejemplo: *Clientes*
Métodos: Se deben usar verbos en infinitivo. Ejemplo: *asignarDestino()*
Variables: Deben ser cortas y significativas. Ejemplo: *sumaTotal*.
Constantes: Se escriben en mayúsculas y si son varias palabras unidas por subrayado. Ejemplo: MAX_VALOR

Resumen

	Regla	Ejemplo
paquetes	todo en minúscula	dominio
clases	empiezan con Mayúscula en singular	Cliente
variables	en minúsculas (camelCase)	cantidadTotal
constantes	en mayúsculas (snake_case)	VALOR_PTAS
métodos	en minúsculas verbos en infinitivo	asignarDestino()

En instrucciones

Una instrucción por línea.

Líneas de separación antes condicionales, bucles, comentarios y métodos.

En expresiones

Poner espacios entre las variables y los operadores;