

1

Elementos de un programa informático

OBJETIVOS DEL CAPÍTULO

- ✓ Conocer qué es un programa, un lenguaje de programación y las diferencias entre lenguajes de programación como Java y C o C++.
- ✓ Reconocer el aspecto de un programa básico en Java y sus características principales.
- ✓ Instalar y utilizar un IDE.
- ✓ Compilar y ejecutar programas sencillos en Java dentro y fuera de un Entorno de desarrollo.
- ✓ Conocer y utilizar fundamentos básicos del lenguaje Java como los tipos de datos, constantes, literales, variables, comentarios, operadores y expresiones.
- ✓ Identificar las ventajas y limitaciones de Java frente a otros lenguajes de programación.

Los compiladores son programas específicos para un lenguaje de programación, los cuales transforman el programa fuente en un programa directa o indirectamente ejecutable por la máquina destino. No es posible compilar un programa escrito en lenguaje Java con un compilador de C porque éste no lo entendería.

El lenguaje máquina que genera Java es un lenguaje intermedio interpretable por una máquina virtual instalada en el ordenador donde se va a ejecutar. Una máquina virtual es una máquina ficticia que traduce las instrucciones máquina ficticias en instrucciones para la máquina real. La ventaja de la misma es que los programas se pueden ejecutar en cualquier tipo de hardware siempre y cuando tenga instalada la máquina virtual correspondiente. Los programas no van a cambiar, lo que cambiará es la máquina virtual dependiendo del hardware (no será igual la máquina virtual de un smartphone que la de un PC).



Compiladores e Intérpretes

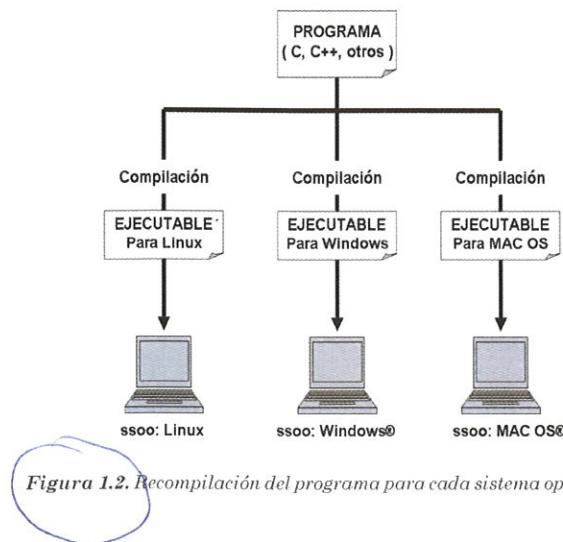
A diferencia de los compiladores, los intérpretes leen línea a línea el código fuente y lo ejecutan. Este proceso es muy lento y requiere tener cargado en memoria el intérprete. La ventaja de los intérpretes es que la depuración y corrección de errores del programa es mucho más sencilla que con los compiladores.

1.1.1 EL LENGUAJE JAVA

Tarea Jardín Tíbble (sept-2016)

Java es uno de los lenguajes más utilizados en la actualidad. Es un lenguaje de propósito general y su éxito radica en que es el lenguaje de Internet. *Applets*, *Servlets*, páginas JSP o JavaScript utilizan Java como lenguaje de programación.

El éxito de Java radica en que es un lenguaje multiplataforma. Java utiliza una máquina virtual en el sistema destino y por lo tanto no hace falta recompilar de nuevo las aplicaciones para cada sistema operativo. Java, por lo tanto, es un lenguaje interpretado que para mayor eficiencia utiliza un código intermedio (*bytecode*). Este código intermedio o *bytecode* es independiente de la arquitectura y por lo tanto puede ser ejecutado en cualquier sistema.



1.1.2 EL JDK

El JDK (*Java Development Kit*), aunque no contiene ninguna herramienta gráfica para el desarrollo de programas, sí que contiene aplicaciones de consola y herramientas de compilación, documentación y depuración. El JDK incluye el JRE (*Java Runtime Environment*) que consta de los mínimos componentes necesarios para ejecutar una aplicación Java, como son la máquina virtual y las librerías de clases.

El JDK contiene, entre otras, las siguientes herramientas de consola:

- **java.** Es la máquina virtual de Java.
- **javac.** Es el compilador de Java. Con él es posible compilar las clases que desarrollemos.
- **javap.** Es un desensamblador de clases.
- **jdb.** El depurador de consola de Java
- **javadoc.** Es el generador de documentación.
- **appletviewer.** Visor de *Applets*.



Importante

Una vez descargado e instalado el JDK hay que modificar los valores de dos variables de entorno:

- Variable PATH. Apunta donde está situado el directorio bin del JDK.
- Variable CLASSPATH. Apunta donde están situadas las clases del JDK.

Podemos descargar y utilizar varios JDK simplemente modificando los valores de ambas variables.

A FONDO

CONOCIENDO LA VERSIÓN DE JAVA

Para conocer la versión de java con la que estamos trabajando basta con ejecutar lo siguiente en una shell o intérprete de comandos:

```
java -version
```

Y aparecerá en la ventana algo parecido a esto:

```
C:\Documents and Settings\JUAN CARLOS>java -version
java version "1.6.0_20"
Java(TM) SE Runtime Environment (build 1.6.0_20-b02)
Java HotSpot(TM) Client VM (build 16.3-b01, mixed mode, sharing)
```



Los comentarios

Existen comentarios de una línea solamente (//) y comentarios multilínea (* *).

- //. Estos comentarios comienzan en la doble barra y terminan hasta el final de la línea.
- /* */ . Estos comentarios comienzan con los caracteres /* y terminan con los caracteres */ y se pueden extender múltiples líneas.

■ La clase *holamundo*

En Java generalmente cada clase es un fichero distinto. Si existieran varias clases en el fichero, la clase cuyo nombre coincide con el nombre del fichero debería de llevar el modificador public (public class holamundo) y es la que se puede utilizar desde fuera del fichero. Las clases tienen el mismo nombre que su fichero .java y es importante que mayúsculas y minúsculas coincidan. La clase abarca desde la primera llave que abre hasta la última que cierra.

```
public class holamundo {  
    ...  
}
```

■ La función o método *main*

```
public static void main (String [ ] args)  
{  
    ...  
}
```

El código Java en las clases se agrupa en métodos o funciones. Cuando Java va a ejecutar el código de una clase, lo primero que hace es buscar el método *main* de dicha clase para ejecutarlo.

El método **main** tiene las siguientes particularidades:

- Es público (**public**). Esto es así para poder llamarlo desde cualquier lado.
- Es estático (**static**). Al ser *static* se le puede llamar sin tener que instanciar la clase.
- No devuelve ningún valor (modificador **void**).
- Admite una serie de parámetros (**String [] args**) que en este ejemplo concreto no son utilizados.

Como puede verse en el ejemplo, el método *main* abarca todo el código contenido entre las llaves.

Una buena opción para empezar a programar en Java es instalar Geany. Geany es un IDE muy liviano y muy intuitivo y su instalación es sumamente sencilla. En Ubuntu Linux se instala ejecutando desde consola el siguiente comando:

```
$ sudo apt-get install geany
```

Una vez instalado el programa, hay que configurar la variable PATH en Windows® (Panel de control -> sistema -> Opciones Avanzadas -> variables de entorno) o las variables JAVA_HOME y JAVA en Linux.



¿Necesitas ayuda para instalar Geany en tu equipo?

En el material adicional del libro tienes un manual paso a paso para instalar Geany y el JDK en Windows® y en Linux.

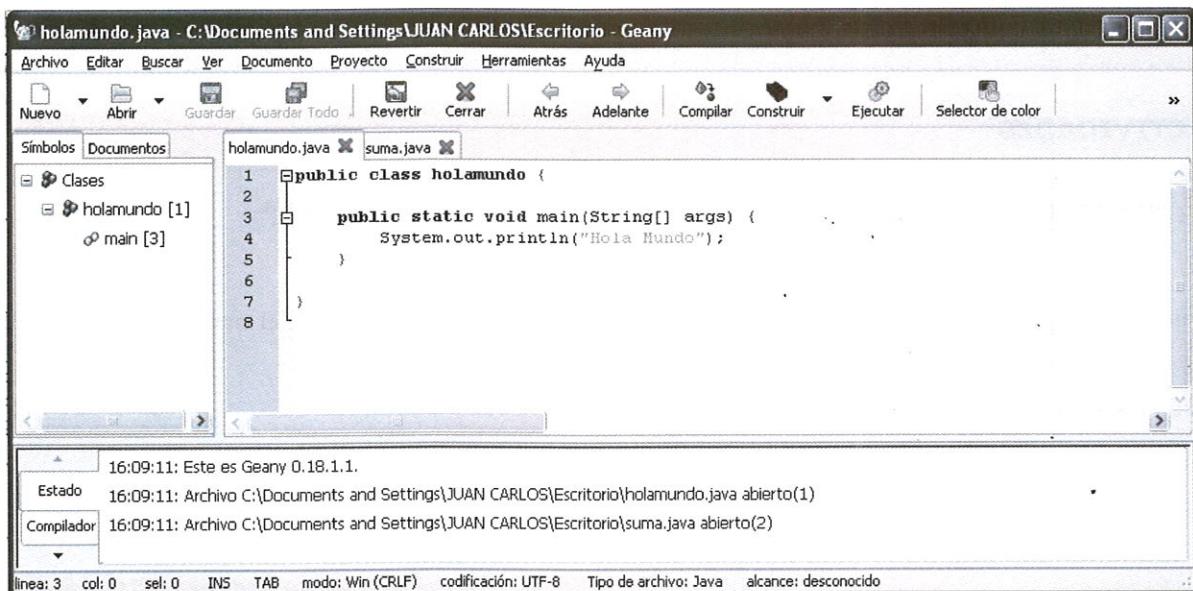


Figura 1.5. Geany. Un entorno de desarrollo ligero y versátil

La secuencia de creación y ejecución de un programa en Java es un proceso que sigue los siguientes pasos: EDITAR → GUARDAR → COMPILEAR → EJECUTAR.

Existen muchos IDE para trabajar con Java. En todos los ejemplos se ha utilizado Geany pero si se quiere algo más potente, una buena opción es Eclipse. Eclipse fue desarrollado primeramente por IBM, aunque actualmente es un IDE de código abierto desarrollado y mantenido por la Fundación Eclipse (<http://www.eclipse.org/>). Eclipse puede utilizarse para Java y añadiendo plugins pueden utilizarse otros lenguajes de programación. Eclipse ha desarrollado numerosas versiones, todas con nombres estelares (Callisto, Europa, Ganymede, Galileo, Helios...). Otra opción no

1.4 TIPOS DE DATOS SIMPLES

Los tipos de datos se utilizan generalmente al declarar variables y son necesarios para que el intérprete o compilador conozca de antemano el tipo de información que va a contener una variable. Los tipos de datos primitivos en Java son los siguientes:

Tabla 1.1. Tipos de datos simples

Tipo de datos	Información representada	Rango	Descripción
byte	Datos enteros	-128 ↔ +127	Se utilizan 8 bits (1 byte) para almacenar el dato.
short	Datos enteros	-32768 ↔ +32767	Dato de 16 bits de longitud (independientemente de la plataforma).
int	Datos enteros	-2147483648 ↔ +2147483647	Dato de 32 bits de longitud (independientemente de la plataforma).
long	Datos enteros	-9223372036854775808 ↔ +9223372036854775807	Dato de 64 bits de longitud (independientemente de la plataforma).
char	Datos enteros y caracteres	0 ↔ 65535	Este rango es para representar números en unicode, los ASCII se representan con los valores del 0 al 127. ASCII es un subconjunto del juego de caracteres Unicode.
float	Datos en coma flotante de 32 bits	Precisión aproximada de 7 dígitos	Dato en coma flotante de 32 bits en formato IEEE 754 (1 bit de signo, 8 para el exponente y 24 para la mantisa).
double	Datos en coma flotante de 64 bits	Precisión aproximada de 16 dígitos	Dato en coma flotante de 64 bits en formato IEEE 754 (1 bit de signo, 11 para el exponente y 52 para la mantisa).
boolean	Valores booleanos	true/false	Utilizado para evaluar si el resultado de una expresión booleanas es verdadero (true) o falso(false).

Las constantes se declaran siguiendo el siguiente formato:

```
final [static] <tipo de datos> <nombre de la constante> = <valor>;
```

Donde el calificador *final* identificará que es una constante, la palabra *static* si se declara implicará que solo existirá una copia de dicha constante en el programa aunque se declare varias veces, el tipo de datos de la constante seguido del nombre y por último el valor que toma.

```
final static double PI=3.141592;
```



Importante

Las constantes se utilizan en datos que nunca varían (IVA, PI, etc.). Utilizando constantes y no variables nos aseguramos que su valor no va a poder ser modificado nunca. También utilizar constantes permite centralizar el valor de un dato en una sola línea de código (si se quiere cambiar el valor del IVA se hará solamente en una línea en vez de si se utilizase el literal 18 en muchas partes del programa).

1.5.2 LOS LITERALES

Un literal puede ser una expresión:

- De tipo de dato simple.
- El valor *null*.
- Un *string* o cadena de caracteres (por ejemplo “Hola Mundo”).

Ejemplos de literales en Java pueden ser ‘a’, 322, 3.1416, “pi” o “programación estructurada”.

1.6 VARIABLES

Una variable no es más ni menos que una zona de memoria donde se puede almacenar información del tipo que deseé el programador.



Las palabras clave

Las palabras clave son las órdenes del lenguaje de programación. El compilador espera esos identificadores para comprender el programa, compilarlo y poder ejecutarlo. Por lo tanto queda PROHIBIDO utilizar palabras clave como (*boolean*, *double*, *long*, *if*, *private*, etc.) utilizadas por el propio Java para nombrar variables dentro de un programa. Tampoco se pueden utilizar caracteres especiales para nombrar variables como (+, -, /, etc.).

**Recuerda**

En Java las variables no pueden declararse fuera de una clase.

Por regla general, en Java, todas las variables que están dentro de un bloque (entre {} y {}) son visibles y existen dentro de dicho bloque. Las funciones miembro de una clase, podrán acceder a todas las variables miembro de dicha clase pero no a las variables locales de otra función miembro.

1.7 OPERADORES Y EXPRESIONES

1.7.1 OPERADORES ARITMÉTICOS

Los operadores aritméticos son utilizados para realizar operaciones matemáticas.

Tabla 1.3. Operadores aritméticos

Operador	Uso	Operación
+	A + B	Suma
-	A - B	Resta
*	A * B	Multiplicación
/	A / B	División
%	A % B	Módulo o resto de una división entera

En el siguiente ejemplo se puede observar la utilización de operadores aritméticos:

```
int n1=2, n2;
n2=n1 * n1;      // n2=4
n2=n2-n1;        // n2=2
n2=n2+n1+15;    // n2=19
n2=n2/n1;        // n2=9
n2=n2%n1;        // n2=1
```

En el siguiente ejemplo se puede observar la utilización de operadores lógicos:

```
int m=2, n=5;
boolean res;
res =m > n && m >= n;//res=false
res =!(m < n || m != n); //res=false
```

1.7.4 OPERADORES UNITARIOS O UNARIOS

Tabla 1.6. Operadores unitarios

Operador	Uso	Operación
~	~A	Complemento a 1 de A
-	-A	Cambio de signo del operando
--	A--	Decremento de A
++	A++	Incremento de A
!	! A	Not A (ya visto)

En el siguiente ejemplo se puede observar la utilización de operadores unitarios:

```
int m=2, n=5;
m++; // m=3
n--; // n=4
```

1.7.5 OPERADORES DE BITS

Tabla 1.7. Operadores de bits

Operador	Uso	Operación
&	A & B	AND lógico. A AND B.
	A B	OR lógico. A OR B.
^	A ^ B	XOR lógico. A XOR B.
<<	A << B	Desplazamiento a la izquierda de A B bits rellenando con ceros por la derecha.
>>	A >> B	Desplazamiento a la derecha de A B bits rellenando con el BIT de signo por la izquierda.
>>>	A >>> B	Desplazamiento a la derecha de A B bits rellenando con ceros por la izquierda.

La precedencia de operadores se resume en la siguiente tabla:

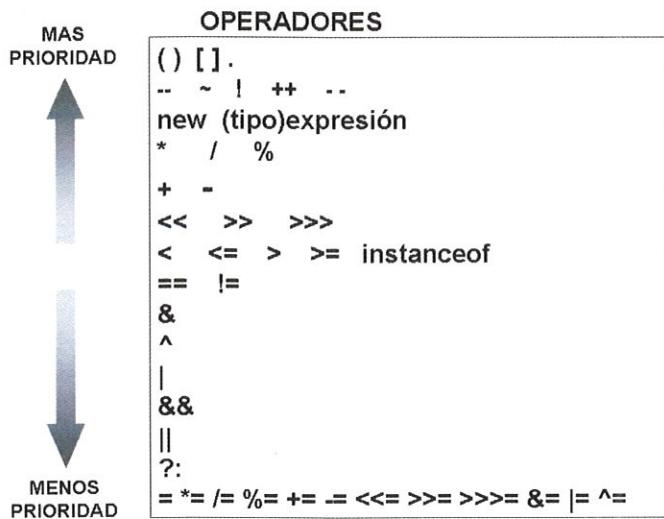


Figura 1.6. Prioridad de los operadores

Imaginemos que se tiene un código como el siguiente:

```
int a = 4;
a = 5 * a + 3;
```

Se desea conocer el valor que tomará a. Para ello se mira en la tabla y se puede observar que el operador * tiene más precedencia que el operador +, con lo cual primero se ejecutará $5 * a$, y al resultado de esta operación se le sumará 3. El resultado de la expresión será 23 y por lo tanto el valor de a será 23 al ejecutar este código.

1.8 CONVERSIONES DE TIPOS (CAST)

Existen dos tipos de conversiones, las conversiones explícitas e implícitas.

- **Conversiones implícitas.** Se realiza de forma automática entre dos tipos de datos diferentes. Requiere que la variable destino (la colocada a la izquierda) tenga más precisión que la variable origen (situada a la derecha).

- 2. El objetivo de este ejercicio es cumplimentar la segunda columna de la siguiente tabla. Como puedes observar ya está cumplimentada pero en los ejercicios propuestos tendrás que comprobar y cumplimentarla tú.

Tabla 1.9. Tabla ejercicio 2

¿Compilará y funcionará el siguiente código?		En caso afirmativo explica que mostrará por pantalla. En caso negativo explica por qué no funciona.
<code>int a = 'a'; System.out.println(a);</code>	<input checked="" type="checkbox"/> Funciona <input type="checkbox"/> No funciona	El código introduce 97 en la variable a que es el valor del código ASCII 'a' y lo muestra por pantalla.
<code>int pi = 3.14; System.out.println(pi);</code>	<input type="checkbox"/> Funciona <input checked="" type="checkbox"/> No funciona	No funciona. No es posible introducir un número real en una variable de precisión entero
<code>double pi = 3,14; System.out.println(pi);</code>	<input type="checkbox"/> Funciona <input checked="" type="checkbox"/> No funciona	Para que funcione basta con cambiar la coma por un punto.
<code>boolean adivina = (1 == 4); System.out.println(adivina);</code>	<input checked="" type="checkbox"/> Funciona <input type="checkbox"/> No funciona	Correcto. Muestra false por pantalla porque los dos valores no son iguales.
<code>boolean adivina = (97 == 'a' == 97); System.out.println(adivina);</code>	<input type="checkbox"/> Funciona <input checked="" type="checkbox"/> No funciona	No funcionará porque la primera parte de la comparación 97 == 'a' genera un booleano, y al comparar un booleano con un entero (97) el compilador dará un error.
<code>boolean adivina = (97 == 'a' == true); System.out.println(adivina);</code>	<input checked="" type="checkbox"/> Funciona <input type="checkbox"/> No funciona	Muestra true por pantalla porque 97 es el código ASCII de 'a' y por lo tanto dará true. Al comparar true con otro valor booleano como true el resultado será true.

- 3. Averigua si las siguientes afirmaciones son verdaderas o falsas:

- En Java generalmente un programa consta de varias clases las cuales se compilan en un único fichero.
- El método *main* puede ser *static* o no. En caso de no ser *static* puede haber varios en un mismo programa.
- Los métodos y funciones difieren en Java en que en los primeros no devuelven ningún valor.
- Es posible hacer **byte a = 200;** El único problema es que como una variable byte solamente almacena hasta el valor 127 la variable a valdrá solo 127.

La solución a este ejercicio está al final de los ejercicios propuestos.

- 4. Realiza un programa en Java que dada dos variables a y b, intercambie los valores de a y b.

Solución:

```
class intercambio {
    public static void main(String[] args) {
        int a= 5, b= 8;
        int tmp;

        tmp=a;
        a=b;
        b=tmp;
        System.out.println("El valor de a ahora es: "+a);
        System.out.println("El valor de b ahora es: "+b);
    }
}
```

EJEMPLO

- 5. Dentro de una clase joven tenemos las variables enteras edad, nivel_de_estudios e ingresos.

Necesitamos almacenar en la variable booleana jasp el valor:

- Verdadero. Si la edad es menor o igual a 28, el nivel_de_estudios es mayor que tres y los ingresos superan los 28.000 (euros).
 - Falso. En caso contrario.
- Escribe el código necesario (2 líneas).

Solución:

```
jasp = false;
jasp = ((edad <= 28) && (nivel_de_estudios > 3) && (ingresos > 28000));
```

- 6. ¿Qué mostrará este programa por pantalla?

```
public class Test {
    public static void main(String[] args) {
        int i=0x100;
        i >>>= 1;
        System.out.println(i);
    }
}
```

Solución:

128

```

    }
}

```

- 4. ¿Qué mostrará el siguiente código por pantalla?

```

int num=5;
num += num - 1 * 4 + 1;
System.out.println(num);
num=4;
num %= 7 * num % 3 * 7 >> 1;
System.out.println(num);

```

- 5. Realiza un programa que calcule la longitud de una circunferencia de radio 3 metros.
- 6. Realiza un programa que calcule el área de una circunferencia de radio 5,2 centímetros.
- 7. Realiza un programa que muestre en pantalla, respetando los retornos de línea, el siguiente texto:
- Me gusta la programación
cada día más.
- 8. (Ejercicio de dificultad alta) Realiza un programa que genere letras aleatoriamente y determine si son vocales o consonantes.
- 9. Cumplimenta la siguiente tabla:

Tabla 1.10. Tabla ejercicio 9

¿Compilará y funcionará el siguiente código?	En caso afirmativo explica qué mostrará por pantalla. En caso negativo explica por qué no funciona.	
<pre> boolean adivina = ((97 == 'a') && true); System.out.println(adivina); </pre>	<input checked="" type="checkbox"/> Funciona <input type="checkbox"/> No funciona	true
<pre> int a=1; int b = a>>>2; System.out.println(b); </pre>	<input checked="" type="checkbox"/> Funciona <input type="checkbox"/> No funciona	0
<pre> int a = 7 4; System.out.println(a); int b = 3 4; System.out.println(b); </pre>	<input checked="" type="checkbox"/> Funciona <input type="checkbox"/> No funciona	7 7