¿Qué es Java?

- La tecnología Java es un lenguaje de programación orientado a objetos y una plataforma comercializada por primera vez en 1995 por Sun Microsystems.

¿Qué es Java?

- El lenguaje de programación Java fue originalmente desarrollado por James Gosling de Sun Microsystems (la cual fue adquirida por la compañía Oracle) y publicado en 1995 como un componente fundamental de la plataforma Java de Sun Microsystems.

¿Qué es Java?

- El lenguaje en sí mismo toma mucha de su sintaxis de C y C++, pero tiene un modelo de objetos más simple y elimina herramientas de bajo nivel.

- Es independiente de la plataforma. Las primeras implementaciones de Java rezaban:"write once, run anywhere".

- Posee un sistema de administración de memoria automático.

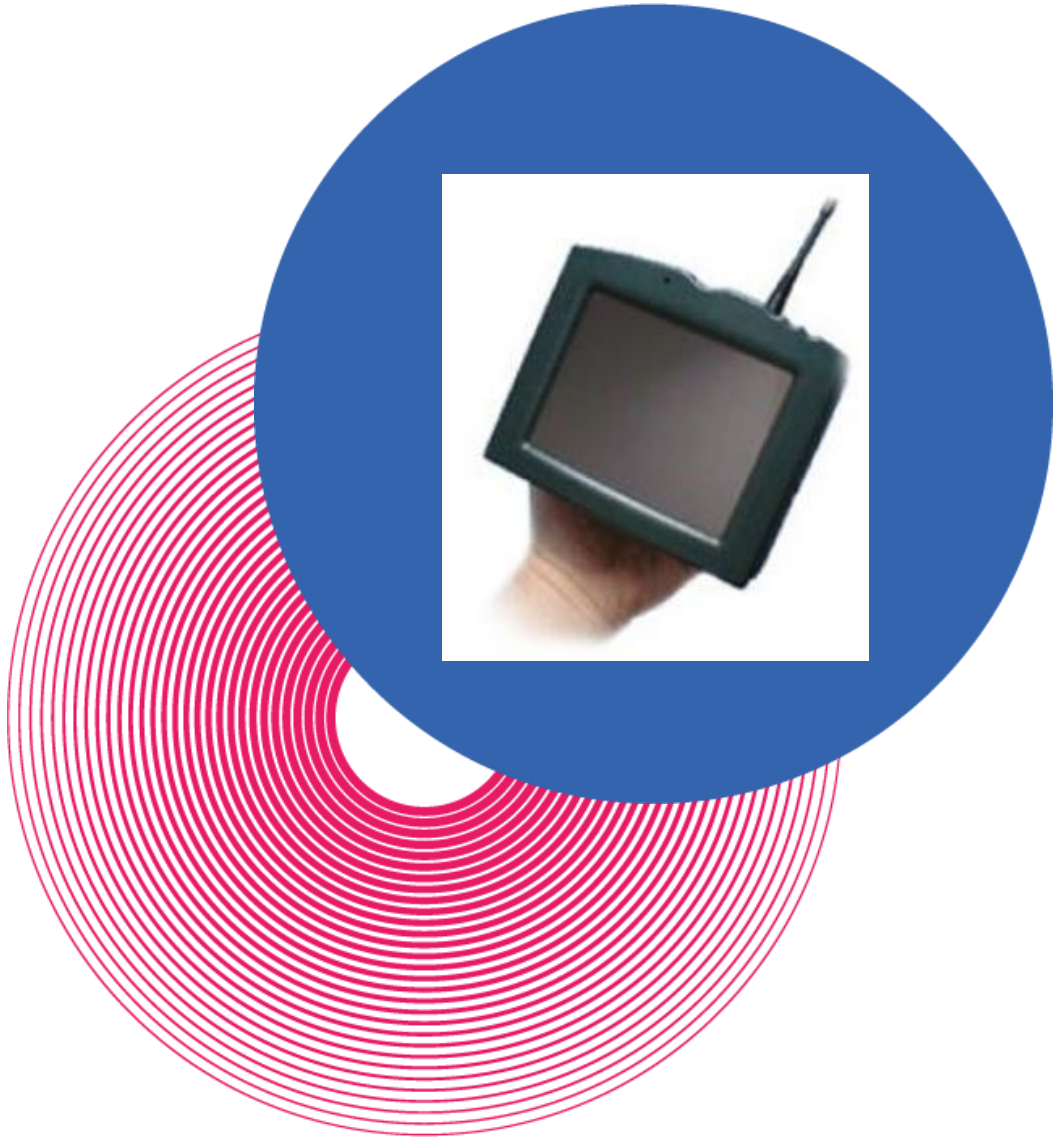
Algo de historia

- Java se creó originalmente como una herramienta de programación para un proyecto set-top-box conocido como *7.
- Fue realizado por un equipo de 13 personas, dirigidas por James Gosling.
- Los objetivos de Gosling eran implementar una máquina virtual y un lenguaje con una estructura y sintaxis similar a C++.

Algo de historia

- En un principio, el sistema *7 no encontró un lugar en el mercado.

- A principios de los noventa, y sin un mercado para su herramienta, Gosling y su equipo se reunieron y notaron que la nueva y popular Internet" tenía exactamente el tipo de configuración de red que ellos habían visionado para la industria de la TV por cable.



Algo de historia

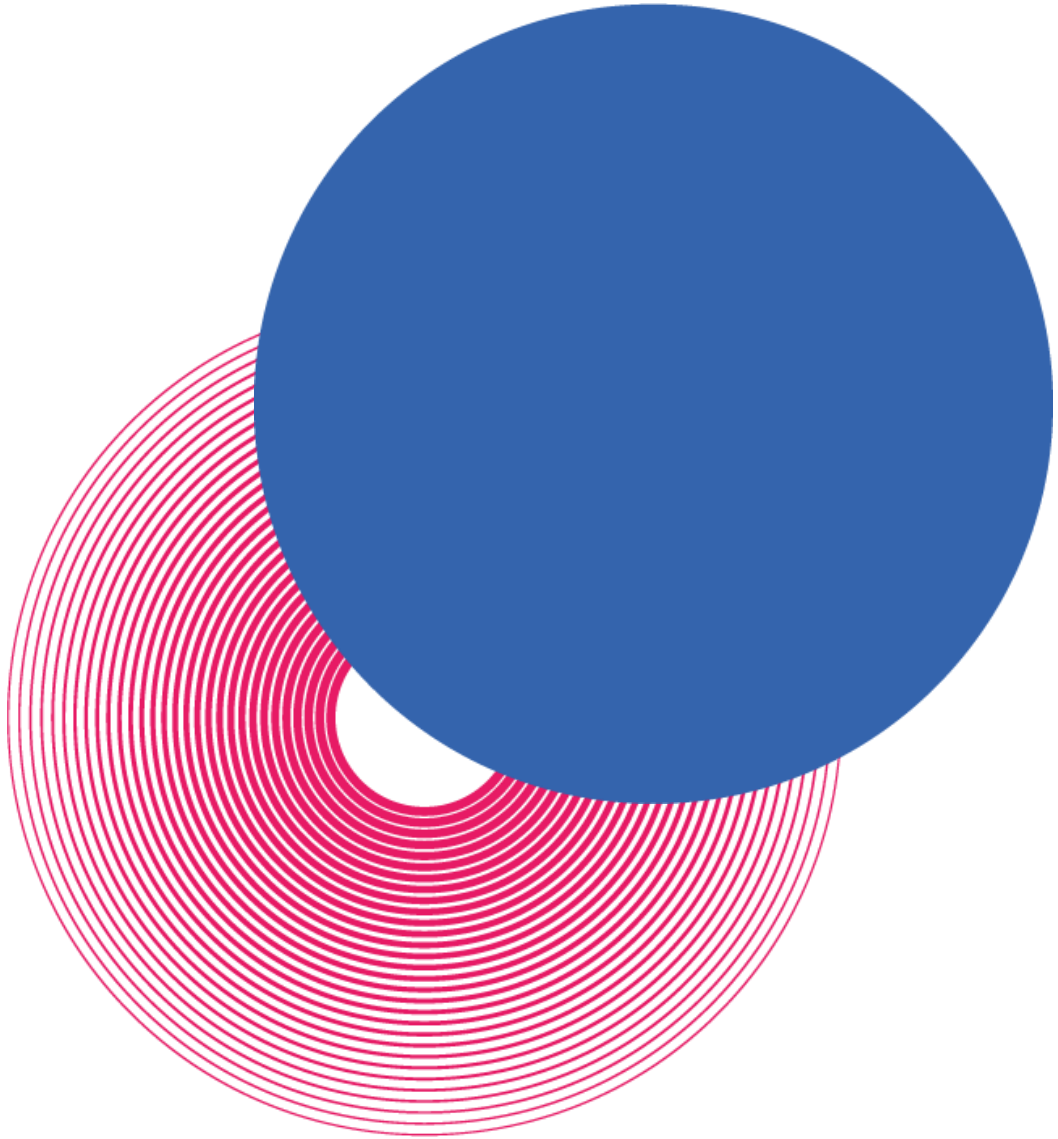
- Con esto en mente Gosling y su equipo crearon el navegador WebRunner y realizaron un demo que mostraba una molécula animada en una reunión de profesionales de la industria de entretenimiento e Internet.

- Todos sabemos como termina esta historia....

¿Por qué Java?

- El lenguaje se denominó inicialmente "Oak". Luego pasó a denominarse "Green" tras descubrir que Oak era ya una marca comercial registrada.

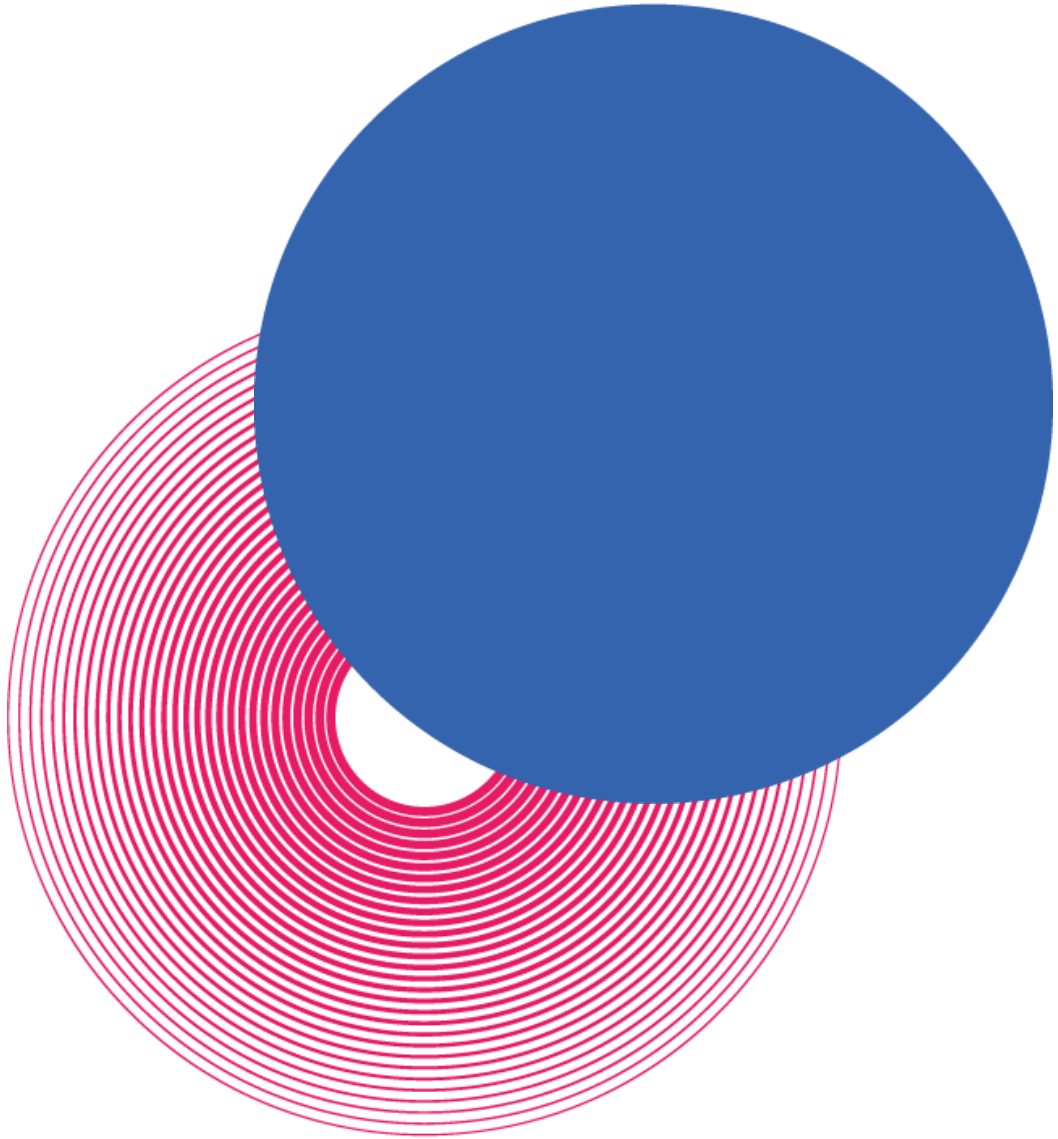
- El término "JAVA" fue acuñado en una cafetería frecuentada por algunos de los miembros del equipo.



¿Por qué Java?

- No está claro si es un acrónimo o no, algunas hipótesis indican que podría tratarse de las iniciales de sus creadores: **J**ames Gosling, **A**rthur **V**an Hoff, y **A**ndy Bechtolsheim. Otras abogan por "**J**ust **A**nother **V**ague **A**cronym".

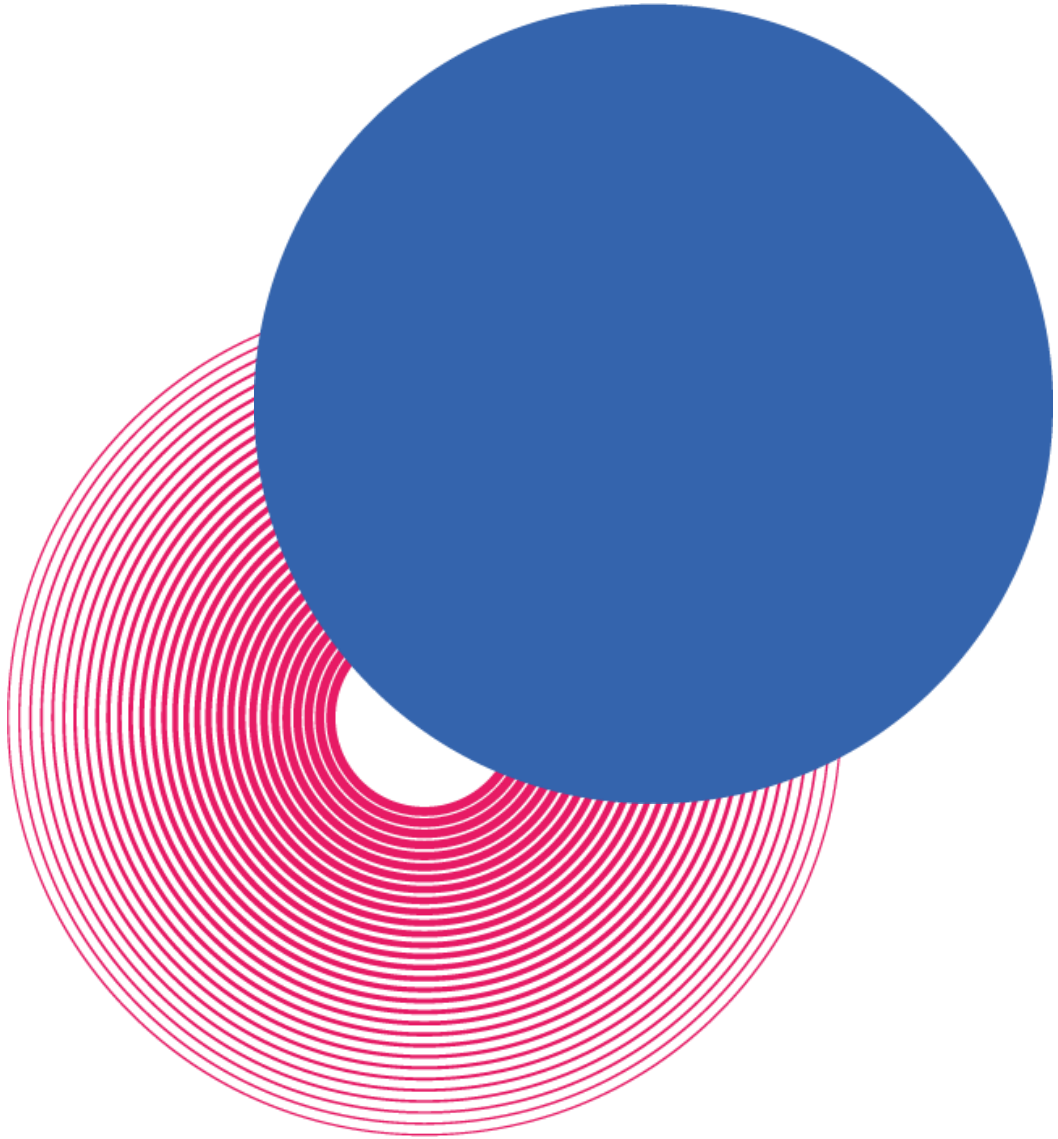
- La hipótesis que más fuerza tiene es la que Java debe su nombre a un tipo de café disponible en la cafetería.



Introducción

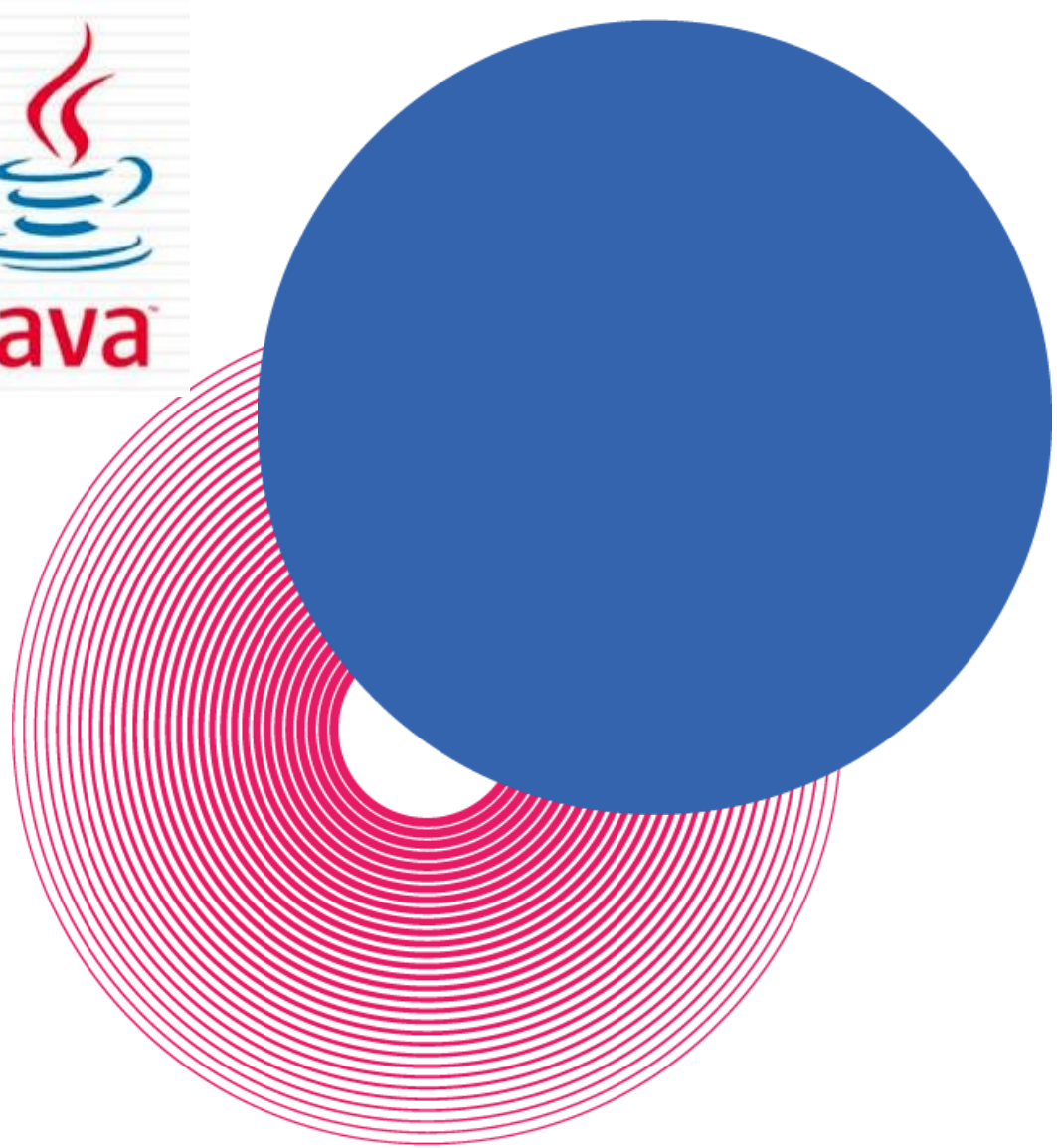
- La plataforma Java es una plataforma de software que se ejecuta sobre la base de varias plataformas de hardware.

- Está compuesto por **JVM** (Java Virtual Machine) y la interfaz de programación de aplicaciones Java (**API**), un amplio conjunto de componentes de software listos para usar, que facilitan el desarrollo y despliegue de aplicaciones.



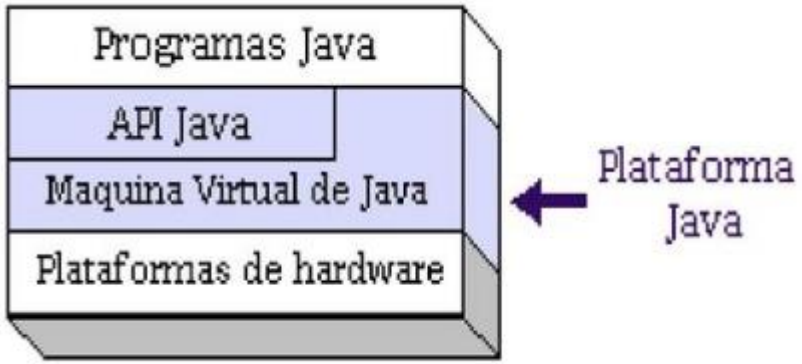
Introducción

- Además de la API Java, toda implementación completa de la plataforma Java incluye:
  - Herramientas de desarrollo para compilar, ejecutar, supervisar, depurar y documentar aplicaciones.
  - Mecanismos estándar para desplegar aplicaciones para los usuarios.
  - Kits de herramientas de interfaz de usuario que permiten crear interfaces gráficas de usuario (GUIs) sofisticadas.
  - Bibliotecas de integración que permiten que los programas accedan a bases de datos y manipulen objetos remotos.



Introducción

- Los programas Java no se ejecutan en nuestra máquina real (en nuestra computadora) sino que Java simula una "máquina virtual" con su propio hardware y sistema operativo.



Introducción

En Java el proceso es:

Del código fuente, se pasa a un código intermedio denominado habitualmente "bytecode" entendible por la máquina virtual Java.

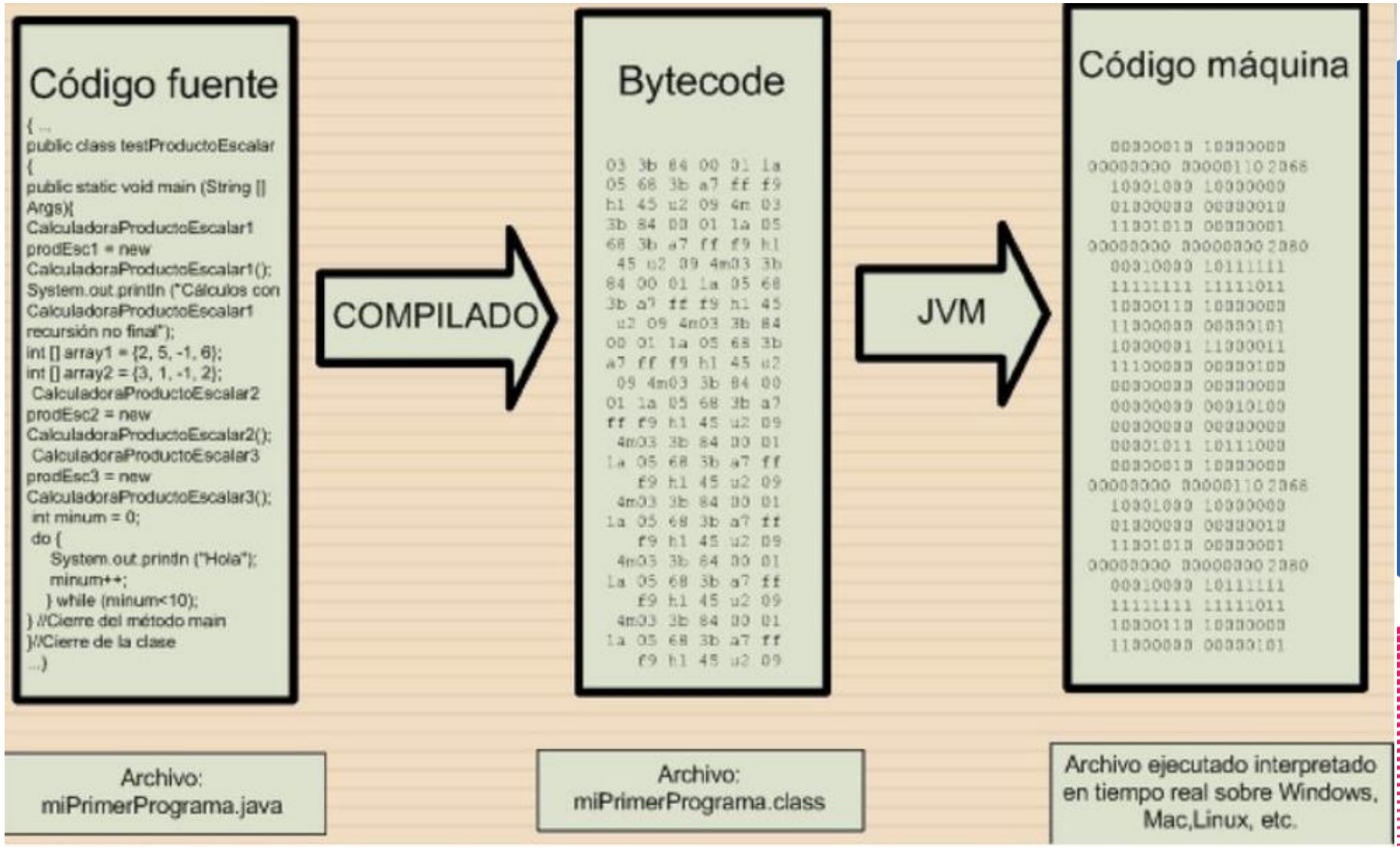
Y es esta máquina virtual simulada, denominada Java Virtual Machine o JVM, la encargada de interpretar el bytecode dando lugar a la ejecución del programa.

Introducción

En Java el código fuente se escribe en un archivo de texto plano con extensión java.

Luego, el código es compilado a archivos .class. Un archivo .class no contiene código nativo a un tipo de procesador, en cambio contiene bytecodes.

Finalmente, la aplicación es interpretada por la máquina virtual de Java, transformando los bytecodes en código nativo en tiempo de ejecución.


Código fuente
{ ...
public class testProductoEscalar
{
public static void main (String []
Args){
CalculadoraProductoEscalar1
prodEsc1 = new
CalculadoraProductoEscalar1();
System.out.println ("Cálculos con
CalculadoraProductoEscalar1
recursión no final");
int [] array1 = {2, 5, -1, 6};
int [] array2 = {3, 1, -1, 2};
CalculadoraProductoEscalar2
prodEsc2 = new
CalculadoraProductoEscalar2();
CalculadoraProductoEscalar3
prodEsc3 = new
CalculadoraProductoEscalar3();
int minum = 0;
do {
System.out.println ("Hola");
minum++;
} while (minum<10);
} //Cierre del método main
}//Cierre de la clase
...)
COMPILADO
Bytecode
03 3b 84 00 01 1a
05 68 3b a7 ff f9
b1 45 u2 09 4m 03
3b 84 00 01 1a 05
68 3b a7 ff f9 b1
45 u2 09 4m03 3b
84 00 01 1a 05 68
3b a7 ff f9 b1 45
u2 09 4m03 3b 84
00 01 1a 05 68 3b
a7 ff f9 b1 45 u2
09 4m03 3b 84 00
01 1a 05 68 3b a7
ff f9 b1 45 u2 09
4m03 3b 84 00 01
1a 05 68 3b a7 ff
f9 b1 45 u2 09
4m03 3b 84 00 01
1a 05 68 3b a7 ff
f9 b1 45 u2 09
4m03 3b 84 00 01
1a 05 68 3b a7 ff
f9 b1 45 u2 09
JVM
Código máquina
00000010 10000000
00000000 00000110 2368
10001000 10000000
01000000 00000010
11001010 00000001
00000000 00000000 2080
00010000 10111111
11111111 11111011
10000110 10000000
11000000 00000101
10000001 11000011
11100000 00000100
00000000 00000000
00000000 00010100
00000000 00000000
00001011 10111000
00000010 10000000
00000000 00000110 2368
10001000 10000000
01000000 00000010
11001010 00000001
00000000 00000000 2080
00010000 10111111
11111111 11111011
10000110 10000000
11000000 00000101
Archivo:
miPrimerPrograma.java
Archivo:
miPrimerPrograma.class
Archivo ejecutado interpretado
en tiempo real sobre Windows,
Mac,Linux, etc.

Introducción

- Todo programa en Java está organizado en clases, estas se codifican en archivos de texto con extensión .java.

- Cada archivo de código fuente .java puede contener una o varias clases, aunque lo normal es que haya un archivo por clase.

Introducción

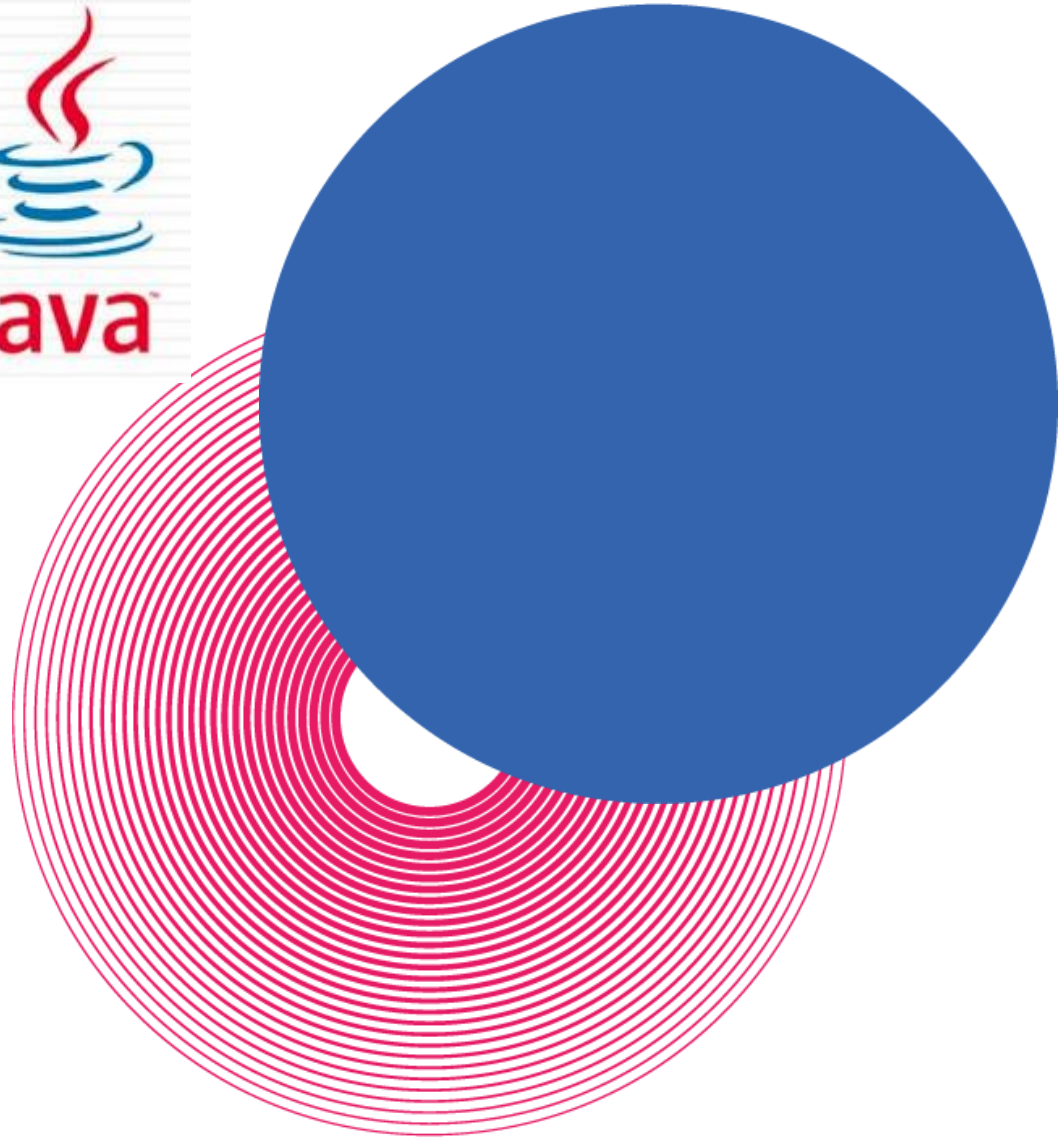
- Cuando se compila un .java se genera uno o varios archivos .class de código binario (bytecodes) que son independientes de la arquitectura.

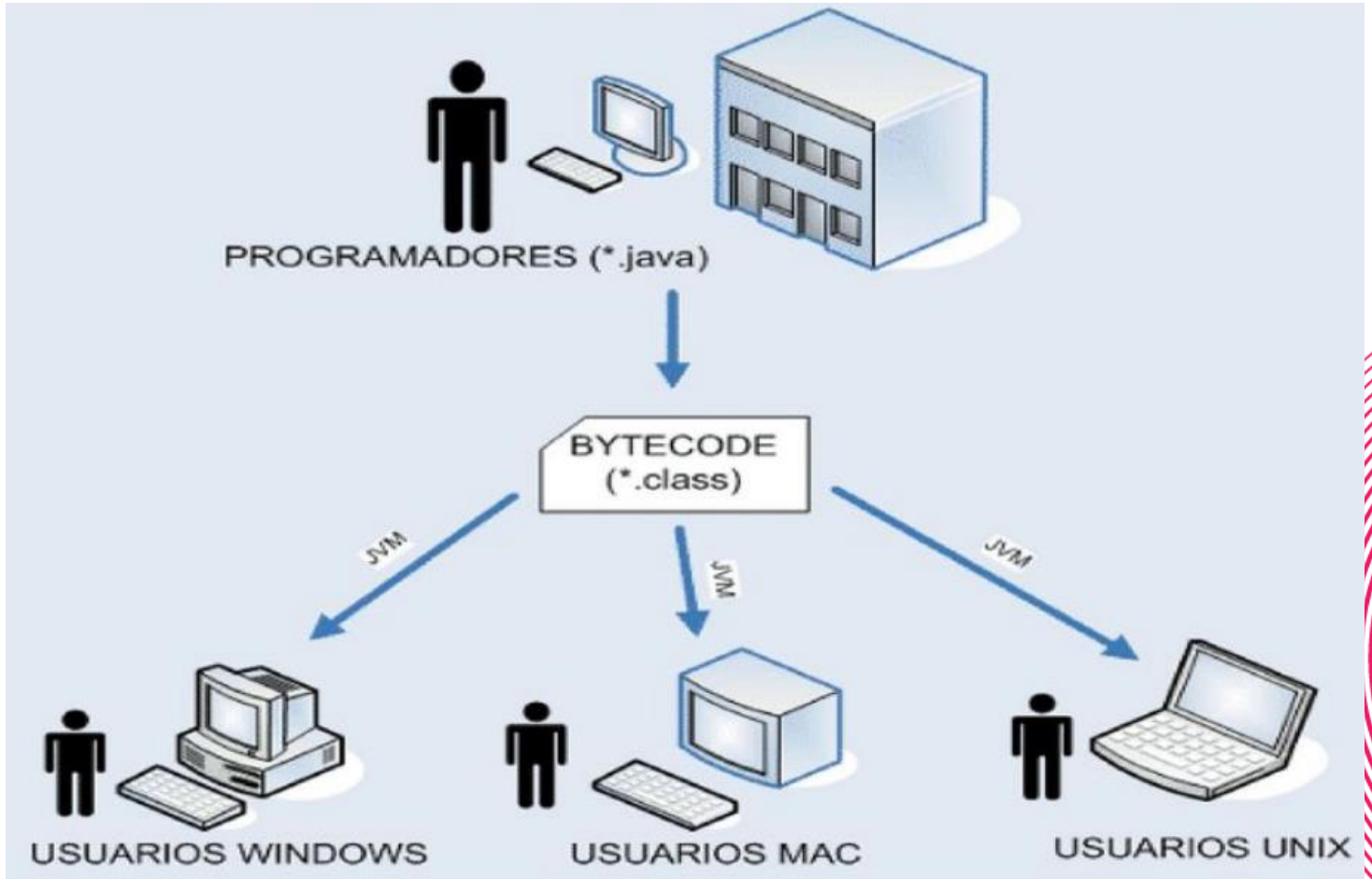
- Esta independencia supone que los bytecodes no pueden ser ejecutados directamente por ningún sistema operativo.

Introducción

- Durante la fase de ejecución es cuando los archivos .class se someten a un proceso de interpretación, consistente en traducir los bytecodes a código ejecutable por el sistema operativo.

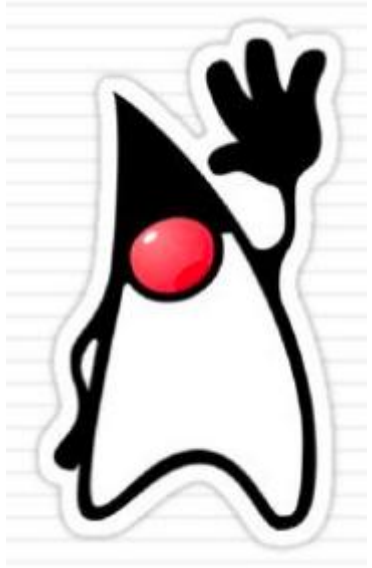
- Esta operación es realizada por la JVM.




PROGRAMADORES (*.java)
BYTECODE
(*.class)
JVM
JVM
JVM
USUARIOS WINDOWS
USUARIOS MAC
USUARIOS UNIX

JDK

- El Java Development Kit (JDK) proporciona el conjunto de herramientas básico para el desarrollo de aplicaciones con Java estándar.



Entorno de ejecución

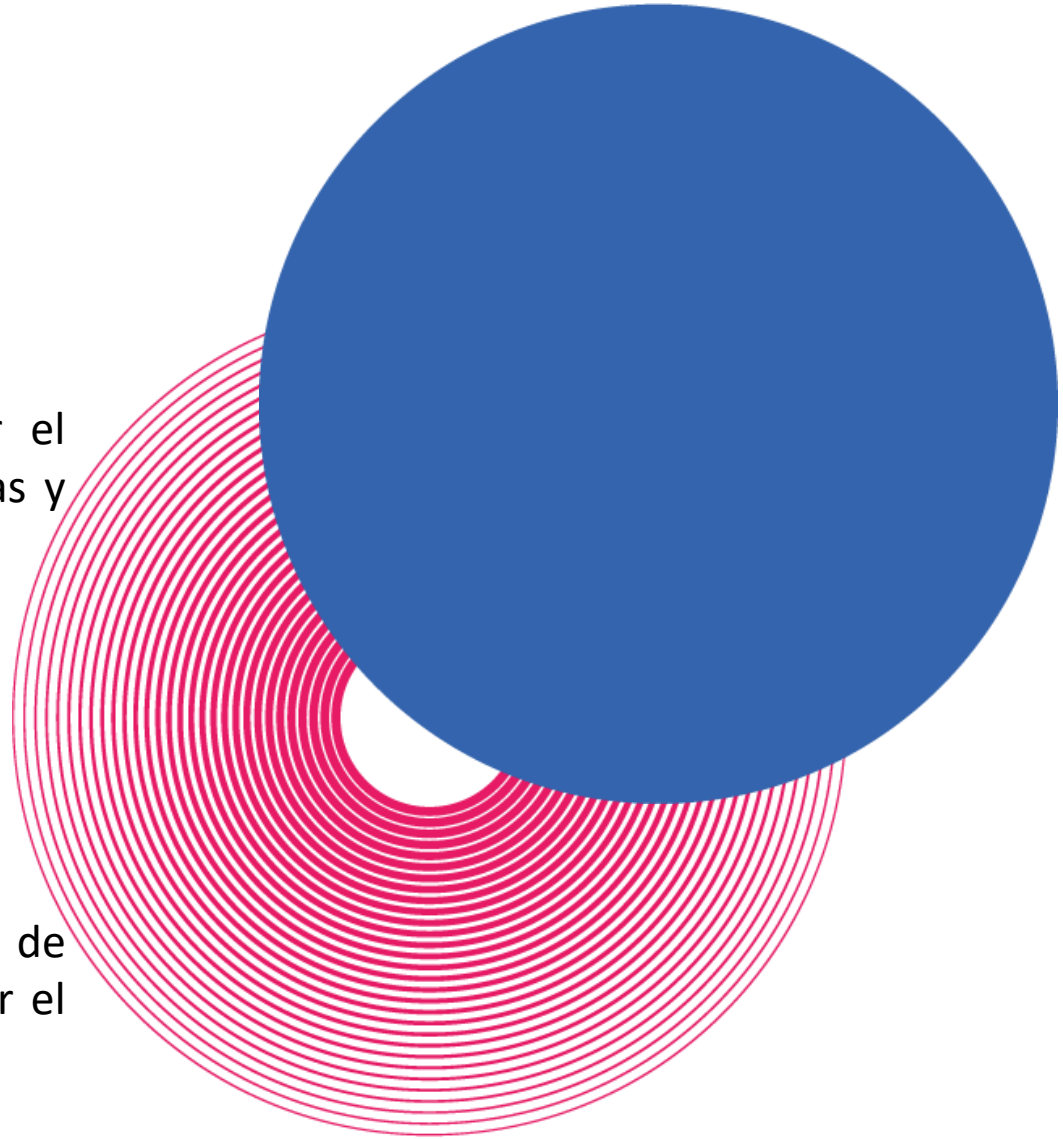
Aunque aún carecemos del conocimiento del lenguaje, podemos iniciar con un sencillo programa en Java que servirá para conocer el procedimiento general que se debe seguir para crear, compilar y ejecutar programas Java.

Entorno de ejecución

Utilizando cualquier editor de texto procedemos a capturar el código (teniendo en cuenta que Java es sensitivo a mayúsculas y minúsculas)

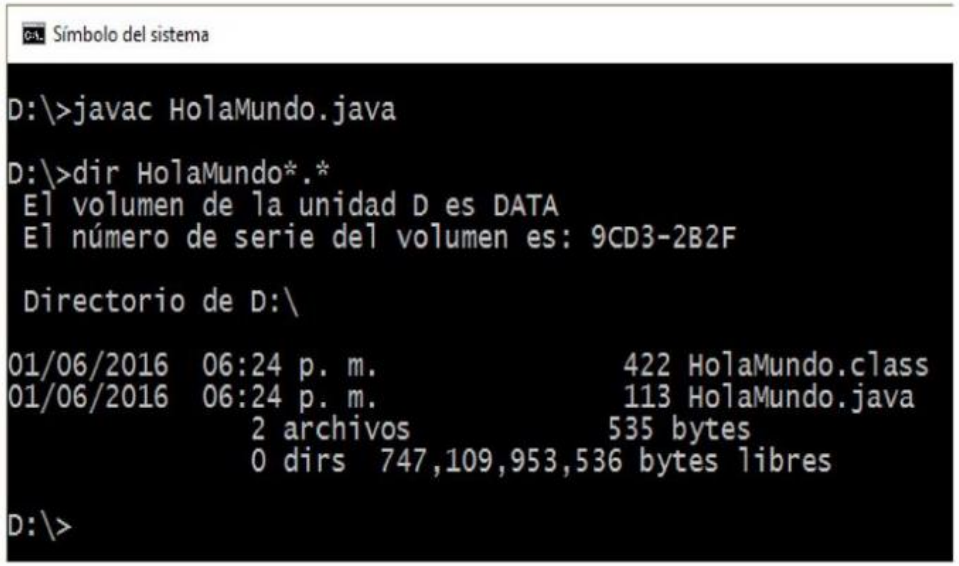
```
public class HolaMundo {
        public static void main(String[ ] args) {
                System.out.println("Hola Mundo');
        }
}
```

Después procedemos a guardar este programa en un archivo de texto llamado HolaMundo.java (el nombre del archivo debe ser el mismo que el de la clase).



Entorno de ejecución

La compilación de un archivo de código fuente .java se realiza a través del comando **javac** del JDK



Entorno de ejecución

Para ejecutar el programa (una vez compilado correctamente), se utiliza el comando java seguido del nombre de la clase que contiene el método main().

La llamada al comando java insta a la máquina virtual a buscar en la clase indicada un método llamado main( ) y procede a su ejecución.

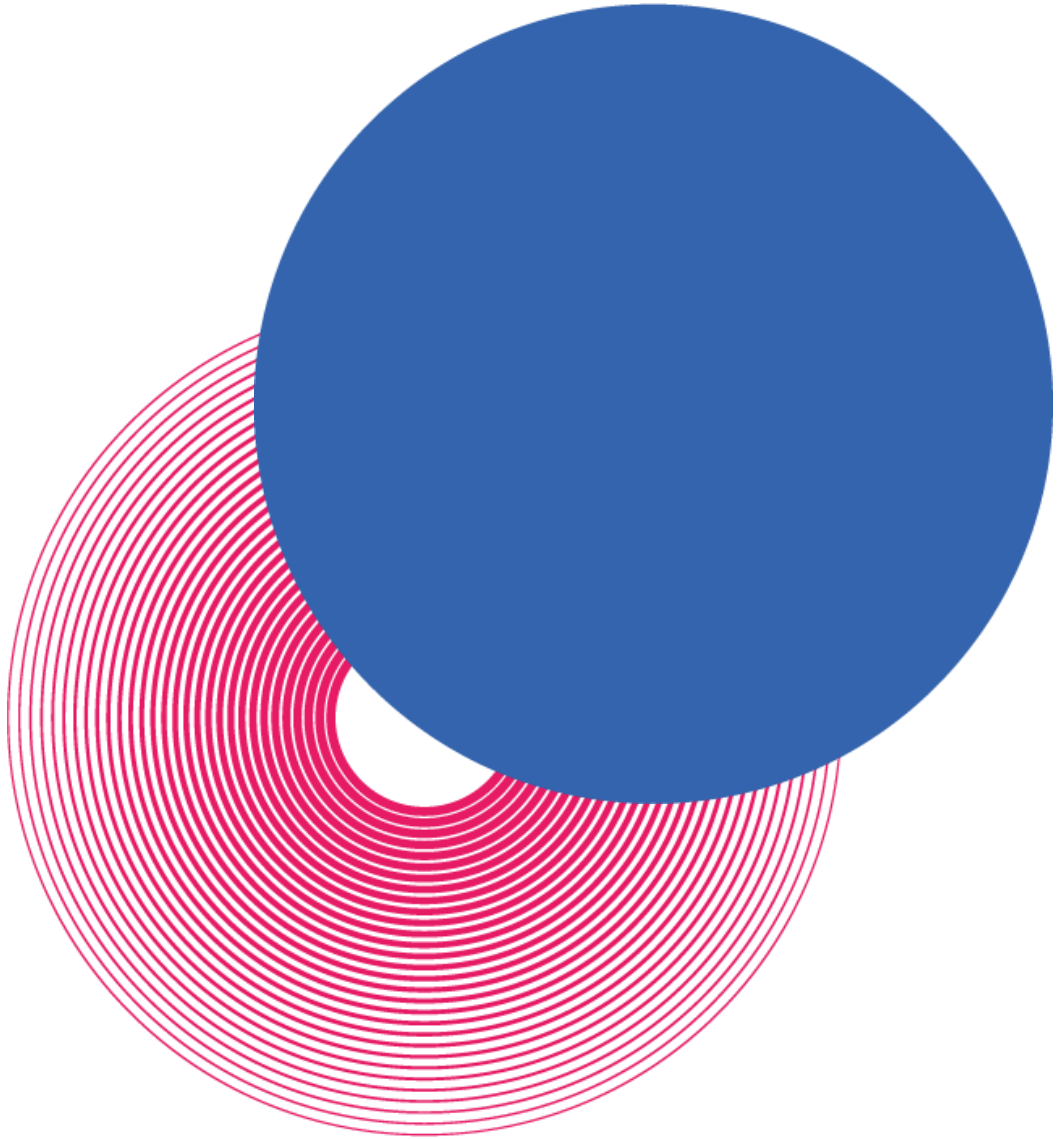
IDE

Un **IDE** (Integrated Development Environment) es una aplicación que facilita el desarrollo de aplicaciones en algún lenguaje de programación.

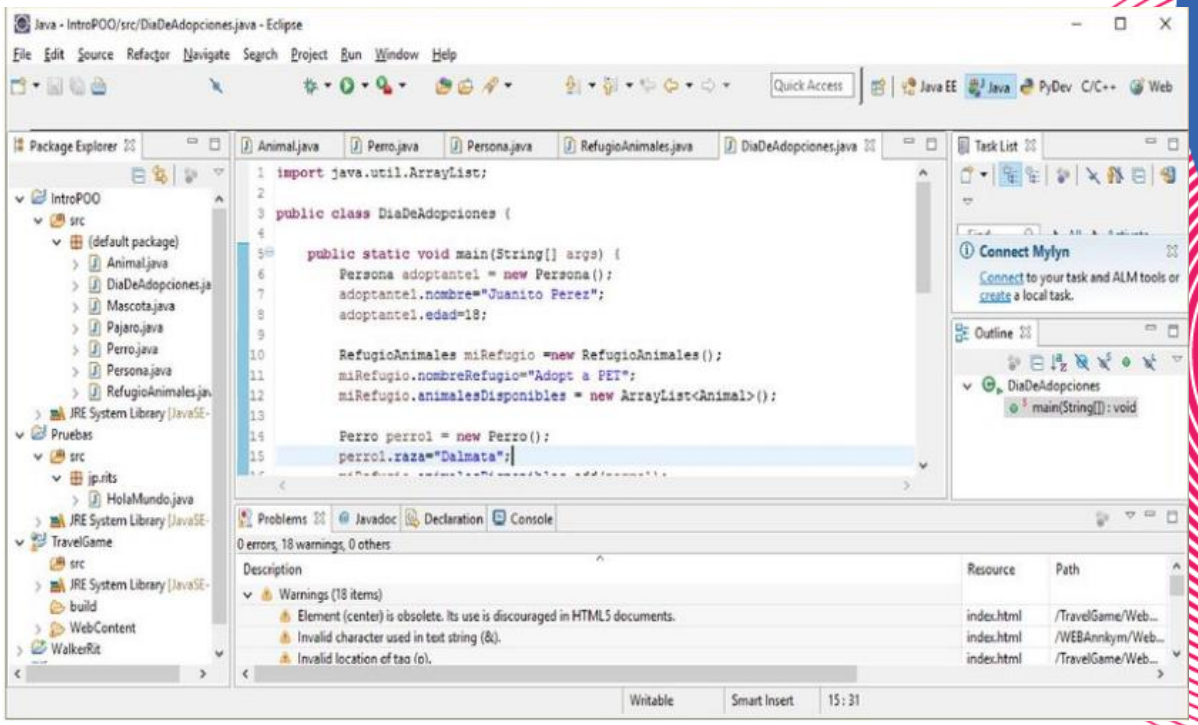
IDE

Un IDE es una interfaz gráfica de usuario diseñada para ayudar a los desarrolladores proporcionando todas las herramientas necesarias para la codificación, compilación, depuración y ejecución.

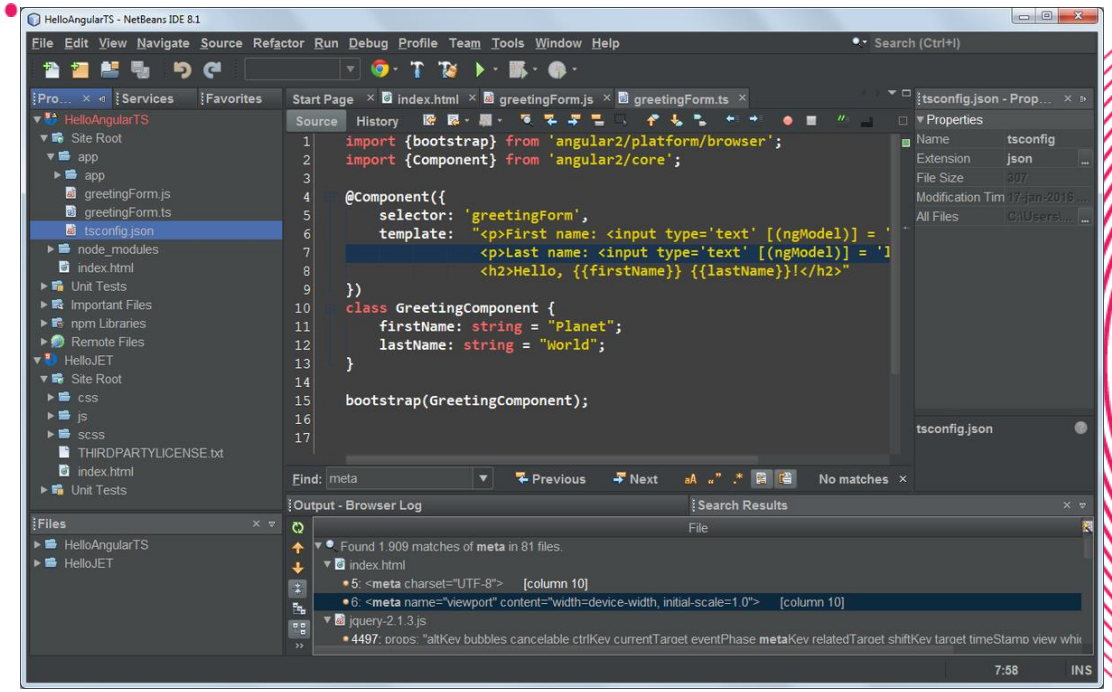
En el mercado existen diversos tipos de IDE, cada uno con características propias, empero, una constante es que permiten manejar las etapas para generar un programa dependiendo del tipo de lenguaje utilizado.



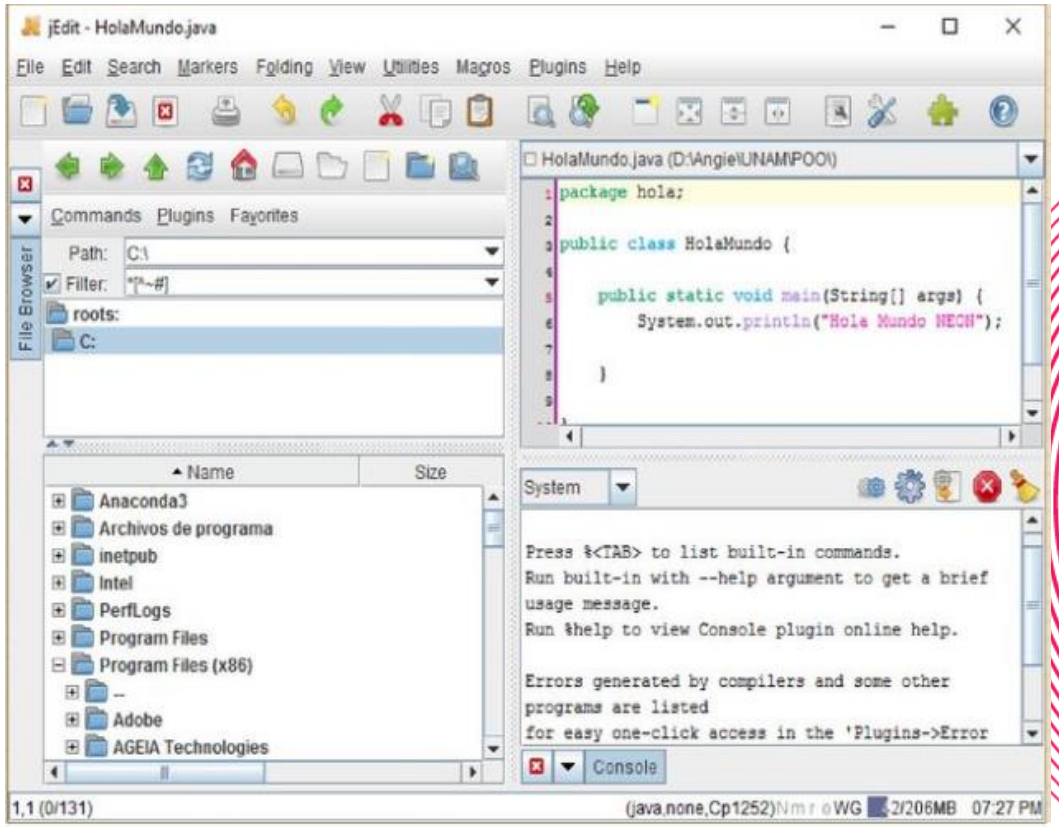
Eclipse


Java - IntroPOO/src/DiaDeAdopciones.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
Java EE
Java
PyDev
C/C++
Web
Package Explorer
IntroPOO
src
(default package)
Animal.java
DiaDeAdopciones.ja
Mascota.java
Pajaro.java
Perro.java
Persona.java
RefugioAnimales.jav
JRE System Library [JavaSE-
Pruebas
src
jp.rits
HolaMundo.java
JRE System Library [JavaSE-
TravelGame
src
JRE System Library [JavaSE-
build
WebContent
WalkerRit
Animal.java
Perro.java
Persona.java
RefugioAnimales.java
DiaDeAdopciones.java
1 import java.util.ArrayList;
2
3 public class DiaDeAdopciones {
4
5 public static void main(String[] args) {
6 Persona adoptante1 = new Persona();
7 adoptante1.nombre="Juanito Perez";
8 adoptante1.edad=18;
9
10 RefugioAnimales miRefugio =new RefugioAnimales();
11 miRefugio.nombreRefugio="Adopt a PET";
12 miRefugio.animalesDisponibles = new ArrayList<Animal>();
13
14 Perro perro1 = new Perro();
15 perro1.raza="Dalmata";
Task List
Connect Mylyn
Connect to your task and ALM tools or create a local task.
Outline
DiaDeAdopciones
main(String[]) : void
Problems
Javadoc
Declaration
Console
0 errors, 18 warnings, 0 others
Description
Resource
Path
Warnings (18 items)
Element (center) is obsolete. Its use is discouraged in HTML5 documents.
index.html
/TravelGame/Web...
Invalid character used in text string (&).
index.html
/WEBAnnkym/Web...
Invalid location of tag (o).
index.html
/TravelGame/Web...
Writable
Smart Insert
15 : 31

NetBeans

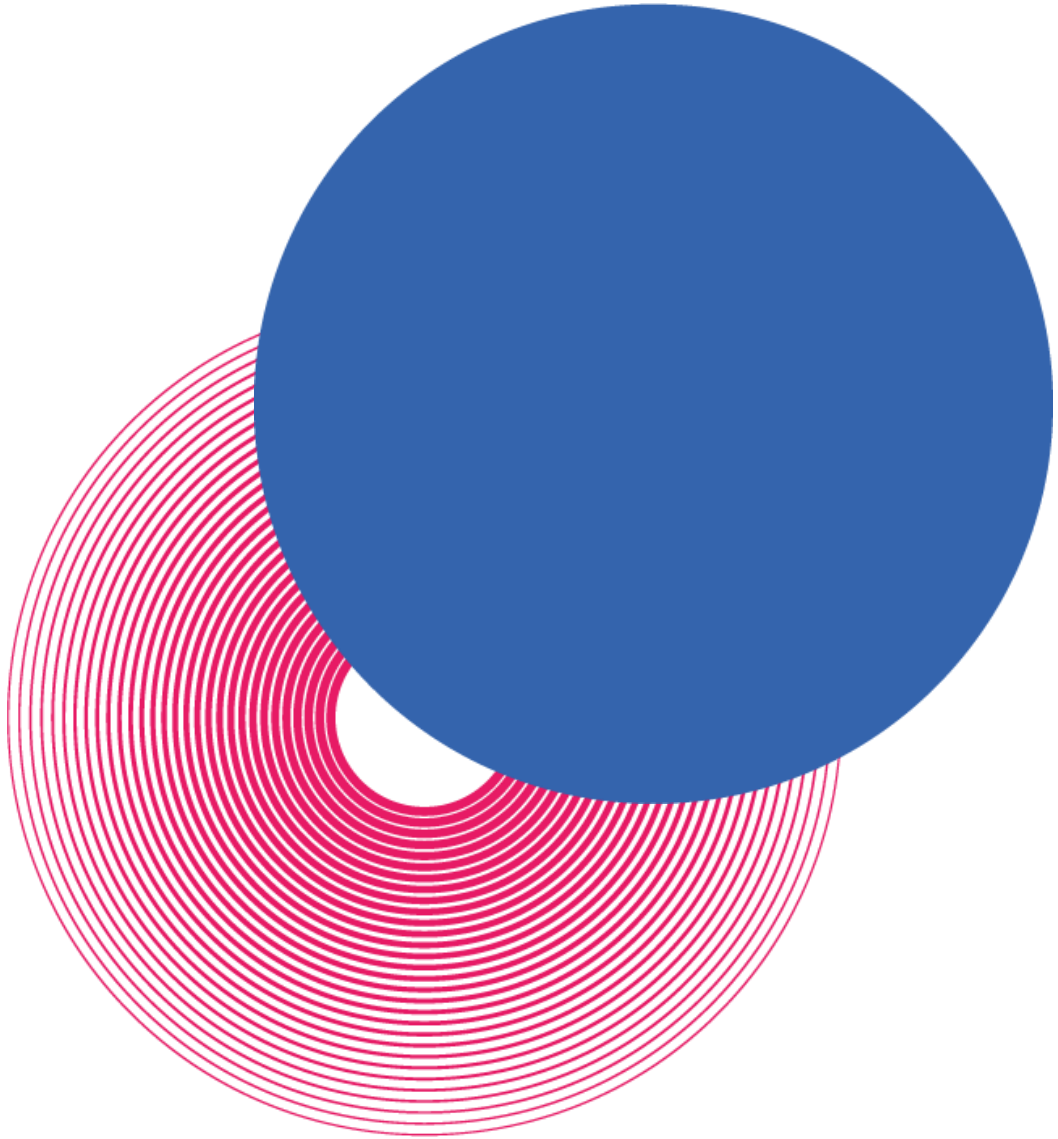

HelloAngularTS - NetBeans IDE 8.1
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Search (Ctrl+I)
Start Page
index.html
greetingForm.js
greetingForm.ts
Source
History
import {bootstrap} from 'angular2/platform/browser';
import {Component} from 'angular2/core';
@Component({
selector: 'greetingForm',
template: "<p>First name: <input type='text' [(ngModel)] = '
<p>Last name: <input type='text' [(ngModel)] = 'l
<h2>Hello, {{firstName}} {{lastName}}!</h2>"
})
class GreetingComponent {
firstName: string = "Planet";
lastName: string = "World";
}
bootstrap(GreetingComponent);
tsconfig.json - Prop...
Properties
Name tsconfig
Extension json
File Size
Modification Tim
All Files
tsconfig.json
Find: meta
Previous
Next
No matches
Output - Browser Log
Search Results
File
Found 1.909 matches of meta in 81 files.
index.html
5: <meta charset="UTF-8"> [column 10]
6: <meta name="viewport" content="width=device-width, initial-scale=1.0"> [column 10]
jquery-2.1.3.js
4497: props: "altKey bubbles cancelable ctrlKey currentTarget eventPhase metaKey relatedTarget shiftKey target timeStamp view whi
Pro...
Services
Favorites
HelloAngularTS
Site Root
app
app
greetingForm.js
greetingForm.ts
tsconfig.json
node_modules
index.html
Unit Tests
Important Files
npm Libraries
Remote Files
HelloJET
Site Root
css
js
scss
THIRDPARTYLICENSE.txt
index.html
Unit Tests
Files
HelloAngularTS
HelloJET
7:58
INS

jEdit


jEdit - HolaMundo.java
File Edit Search Markers Folding View Utilities Macros Plugins Help
Commands Plugins Favorites
Path: C:\
Filter: *[^~#]
roots:
C:
File Browser
HolaMundo.java (D:\Angie\UNAM\POO\)
package hola;
public class HolaMundo {
public static void main(String[] args) {
System.out.println("Hola Mundo MECH");
}
Name
Size
Anaconda3
Archivos de programa
inetpub
Intel
PerfLogs
Program Files
Program Files (x86)
Adobe
AGEIA Technologies
System
Press %<TAB> to list built-in commands.
Run built-in with --help argument to get a brief usage message.
Run %help to view Console plugin online help.
Errors generated by compilers and some other programs are listed
for easy one-click access in the 'Plugins->Error
Console
1,1 (0/131)
(java,none,Cp1252)
07:27 PM

Estructura de un programa

Todo programa Java debe estar escrito en una o varias clases, dentro de las cuales se podrá hacer uso del amplio conjunto de paquetes y clases pre-diseñadas.

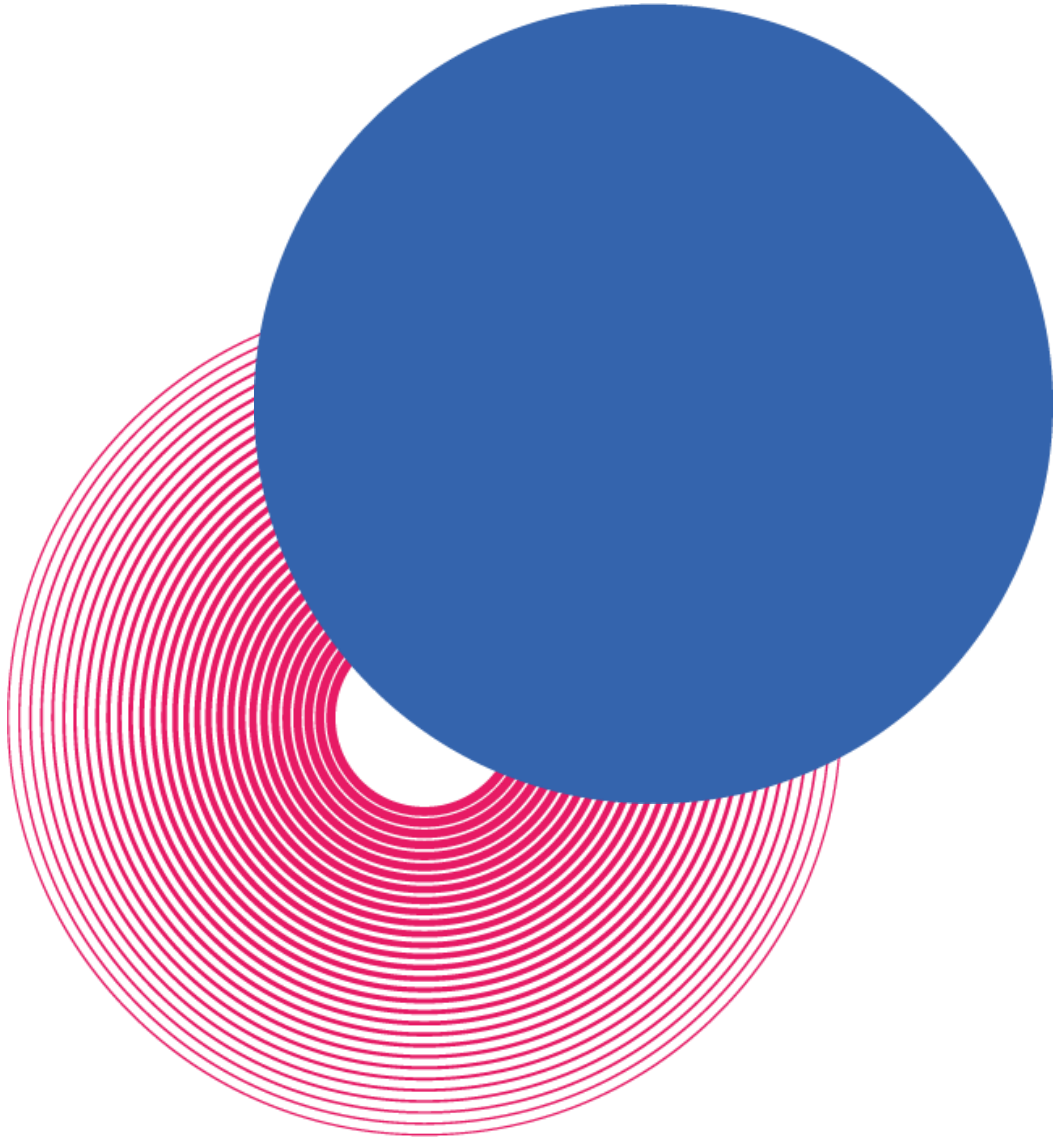


Programación

La sintaxis de Java es muy parecida a la de C y C++, por ejemplo, las sentencias finalizan con ; , los bloques de instrucciones se delimitan con llaves { y }, etc.

Una de las principales características de Java es que es un lenguaje totalmente orientado a objetos.

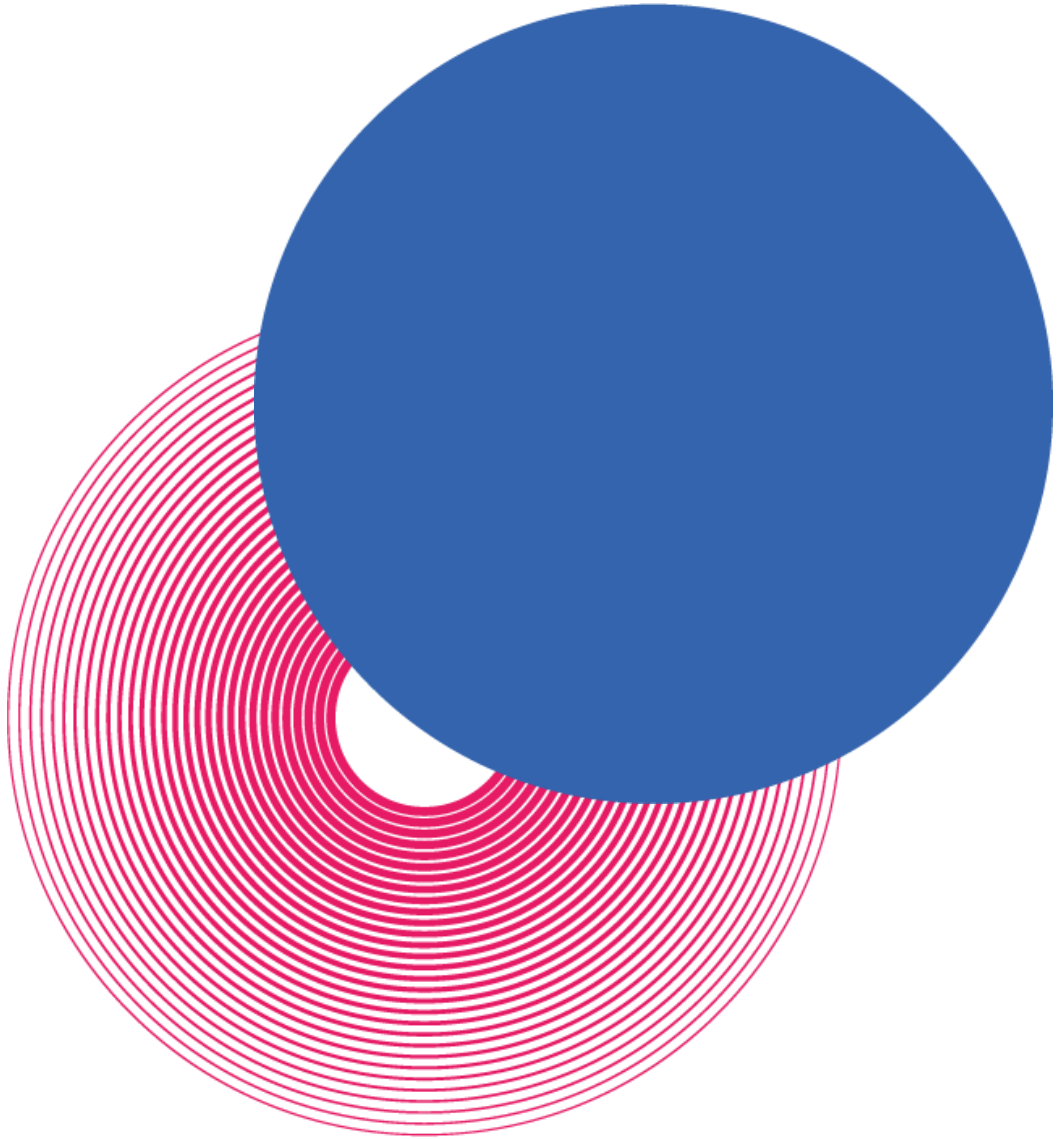
Como tal, todo programa Java debe estar escrito en una o varias clases, dentro de las cuales se podrá hacer uso del amplio conjunto de paquetes y clases pre-diseñadas.



Programación

Un programa en Java, consta de una clase principal (que contiene el método main) y algunas clases de usuario (las específicas de la aplicación que se está desarrollando) que son utilizadas por el programa o clase principal.

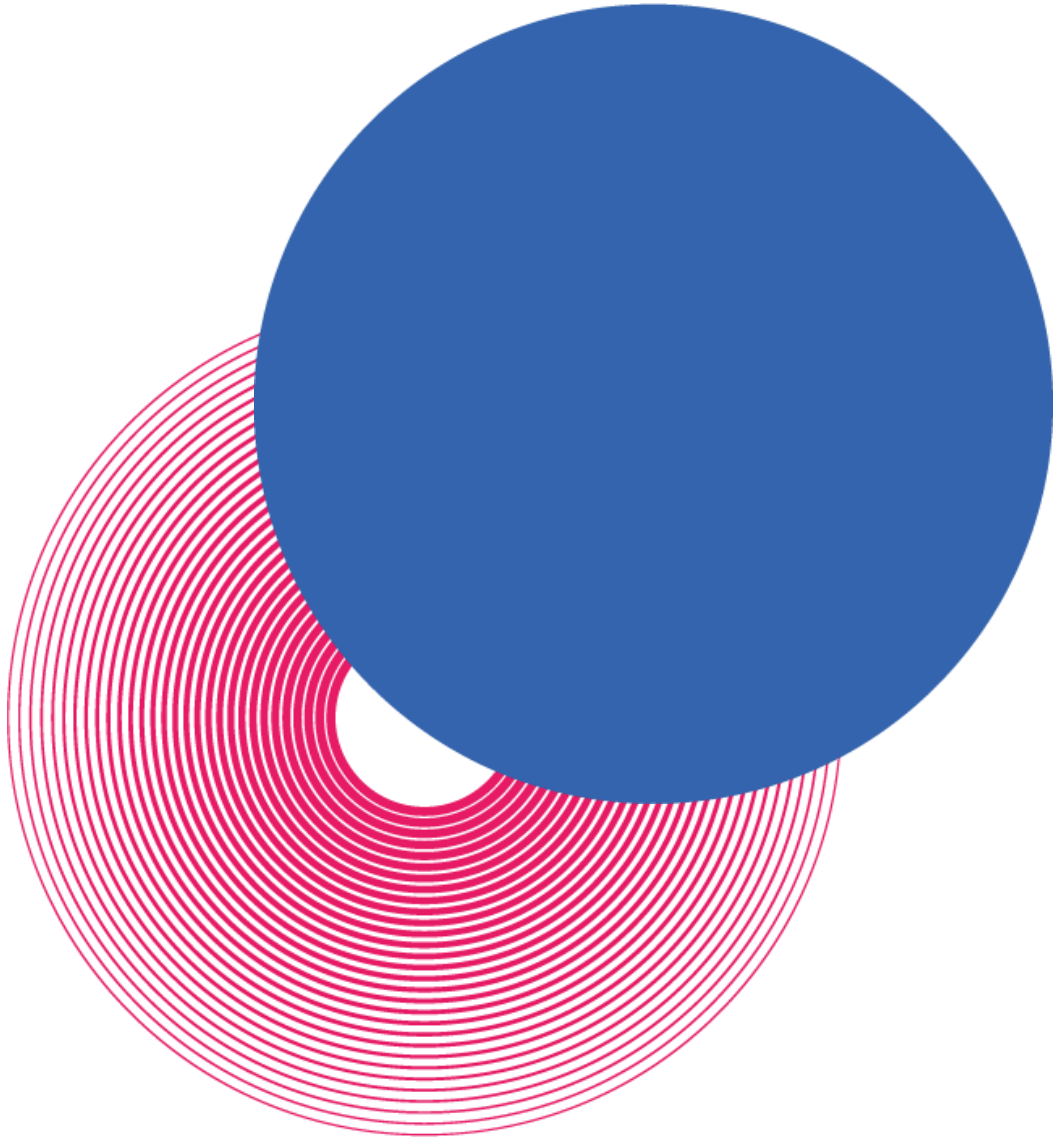
La clase principal debe ser declarada con el modificador de acceso public y la _ palabra reservada class, seguida del nombre de la clase iniciando con mayúscula.



Ejemplo

Ejemplo:

```
public class MiClase {
    public static void main(String[ ] args) {
        System.out.println("Esta es mi clase");
    }
}
```



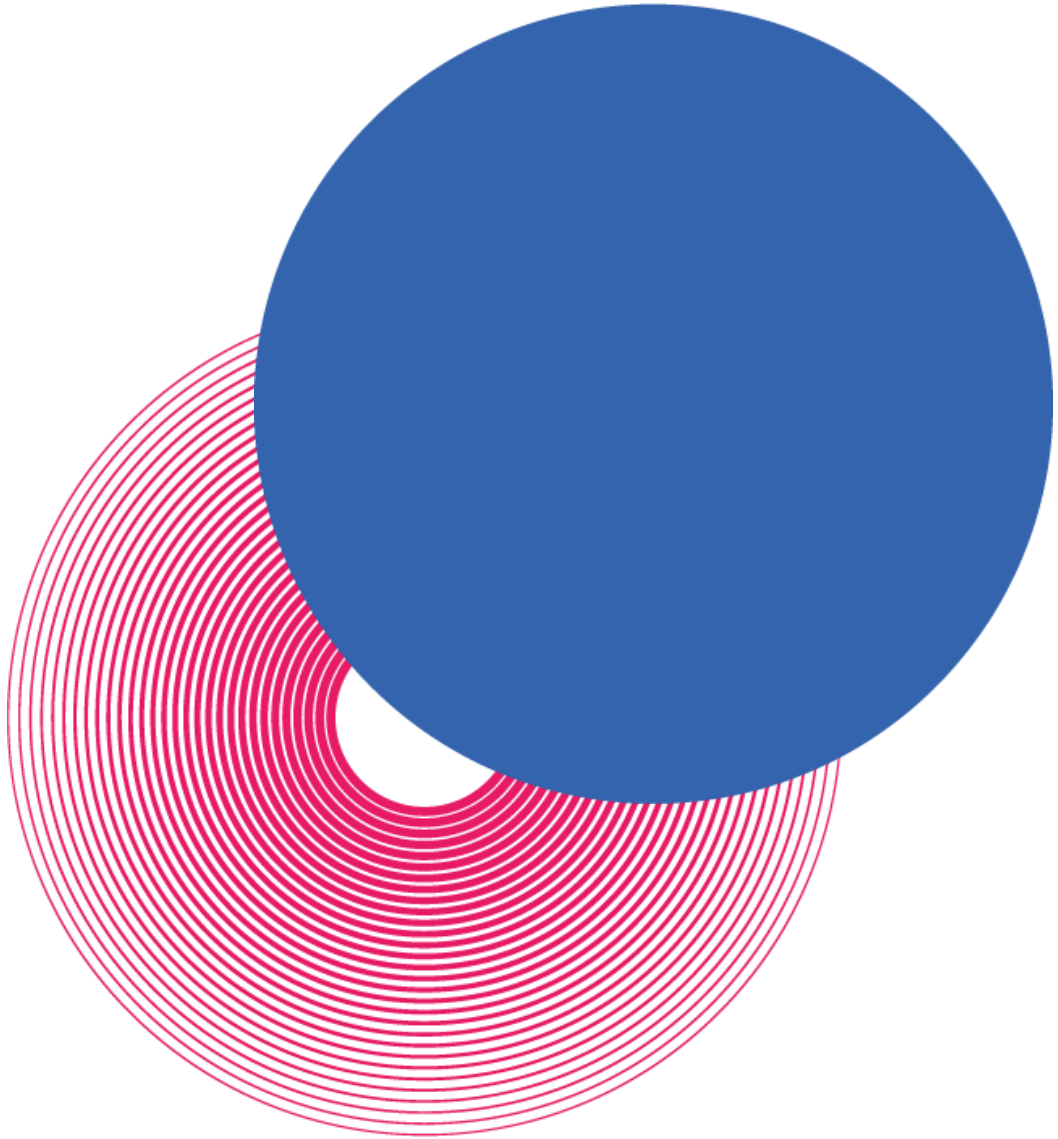
Programación

Un archivo fuente (*.java) puede contener mds de una clase, pero sólo una puede ser public.

El nombre del archivo fuente debe coincidir con el de la clase public (con la extensión *.java).

El nombre del archivo tiene que ser exactamente el mismo
que el de la clase.

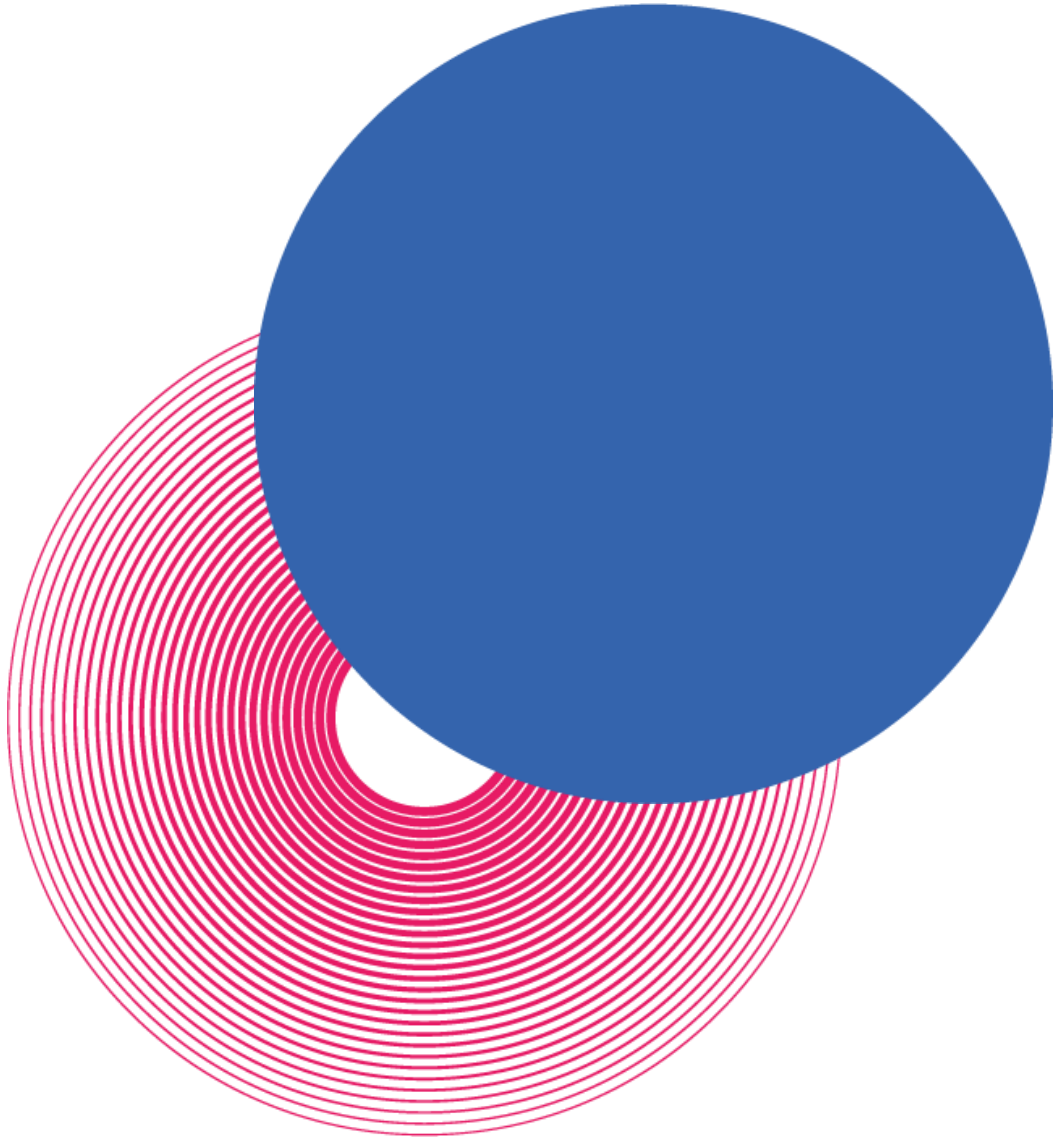
Es importante que coincidan mayusculas y minUsculas ya que MiClase.java y miclase.java serian clases diferentes para Java.



Programación

Una de las primeras cosas que hay que tener en cuenta es que Java es un lenguaje sensitivo a mayúsculas /minúsculas.

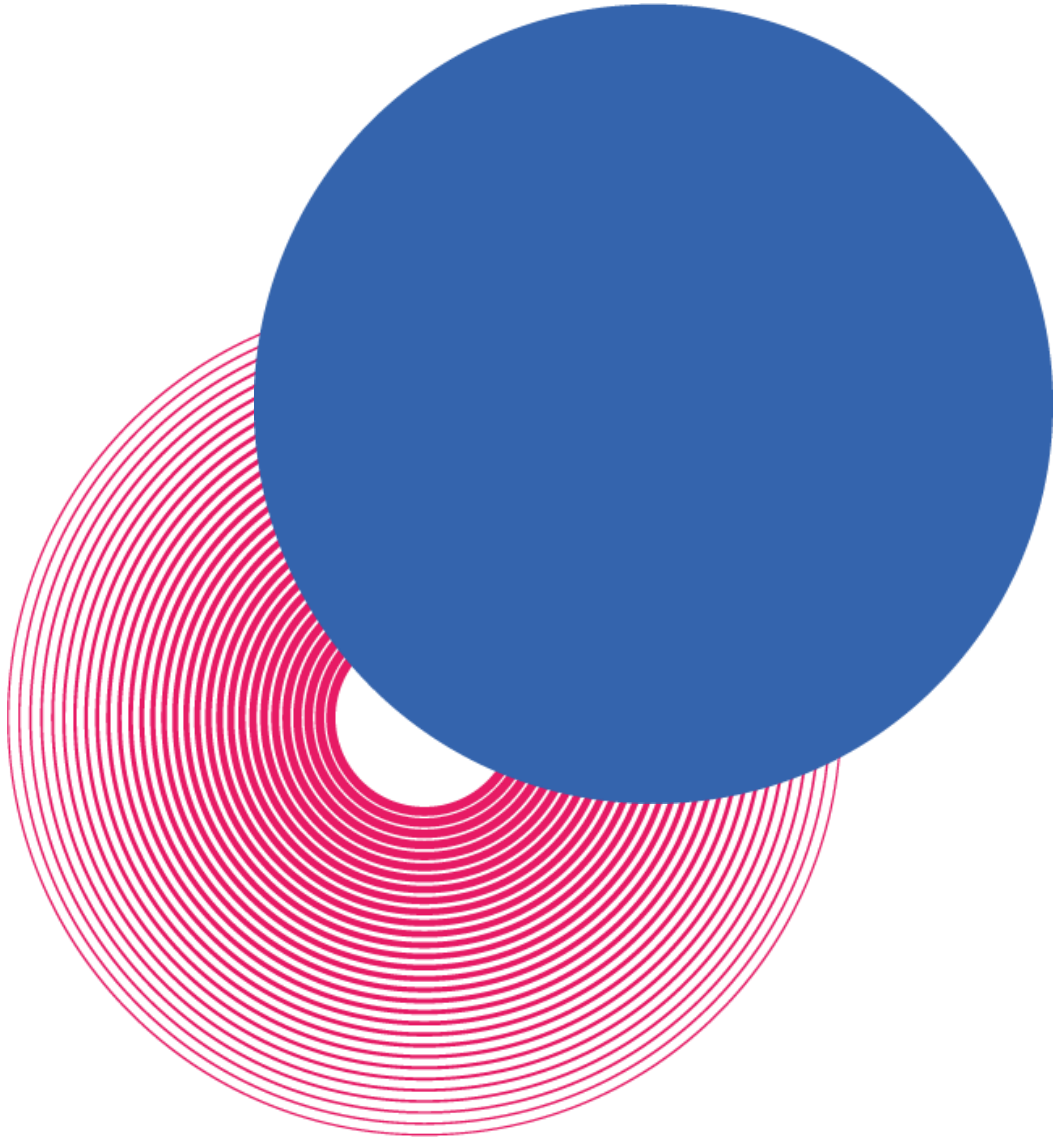
El compilador Java hace distinción entre mayúsculas y minúsculas, esta distinción no solo se aplica a palabras reservadas del lenguaje sino también a nombres de variables y métodos.



El método main

En la clase principal debe existir una función o método estático llamado main cuyo formato debe ser:

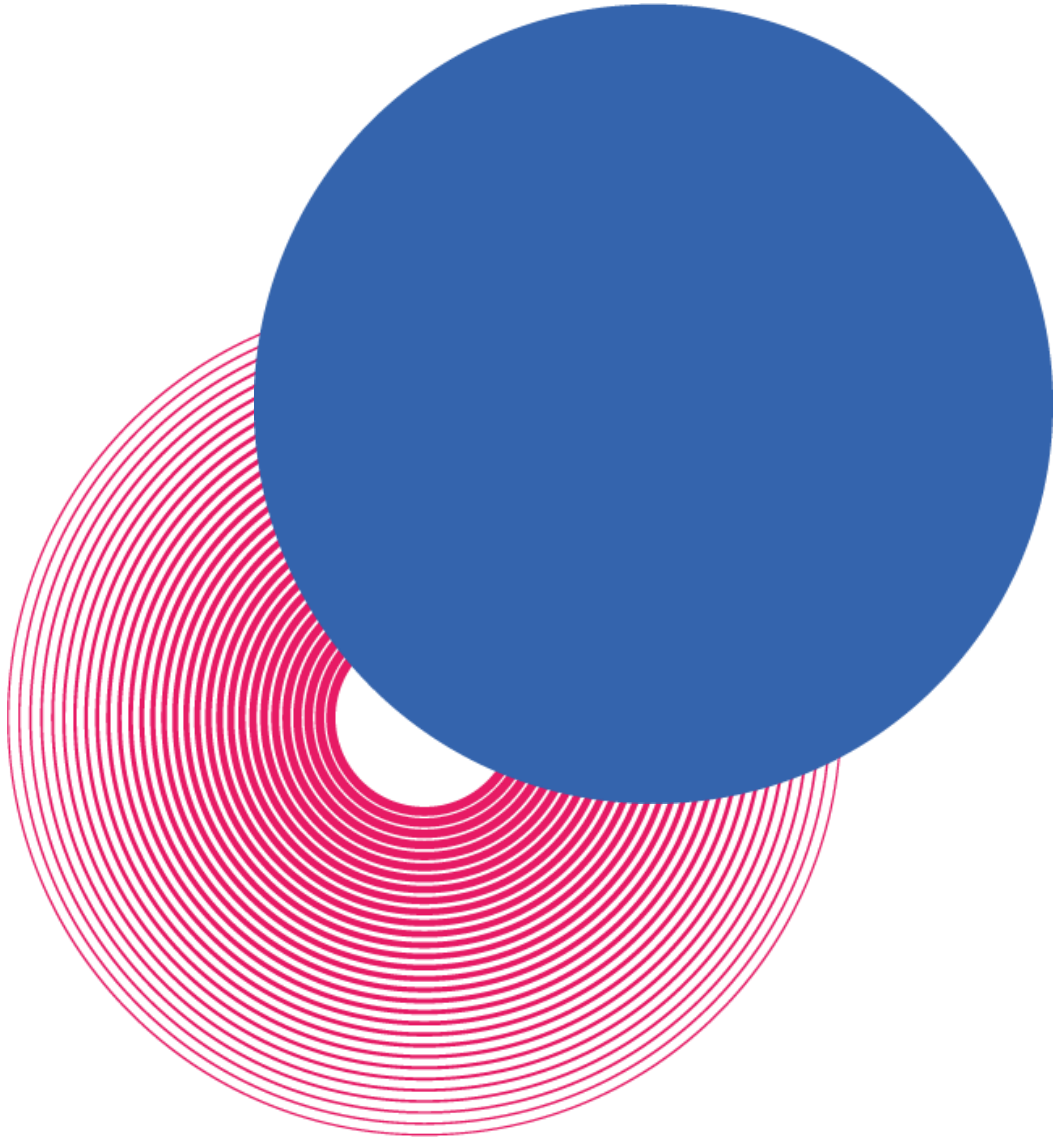
public static void main(String[ ] args)



El método main

El método main es el punto de arranque de un programa Java, cuando se invoca al comando java desde la línea de comandos, la JVM busca en la clase indicada un método estático llamado main con el formato indicado.

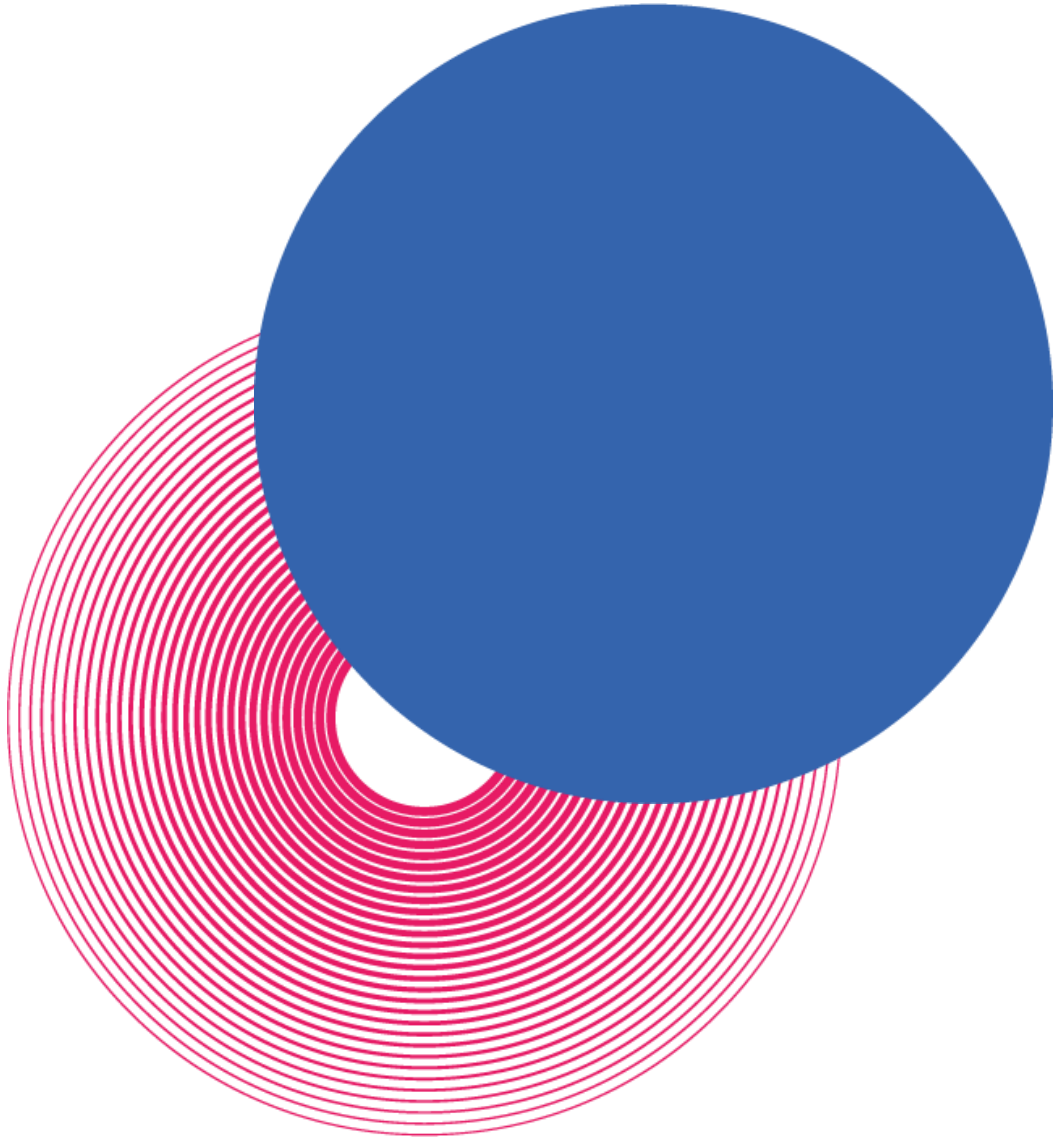
Dentro del código de main pueden crearse objetos de otras clases e invocar sus métodos, en general, se puede incluir cualquier tipo de lógica que respete la sintaxis y estructura de java.



Comentarios

Los comentarios son muy útiles para poder entender el código utilizado, facilitando de ese modo futuras revisiones y correcciones.

Además permite que cualquier persona distinta al programador original pueda comprender el código escrito de una forma más rápida.



Comentarios

En Java existen tres tipos de comentarios:

- Comentarios de una sola línea.

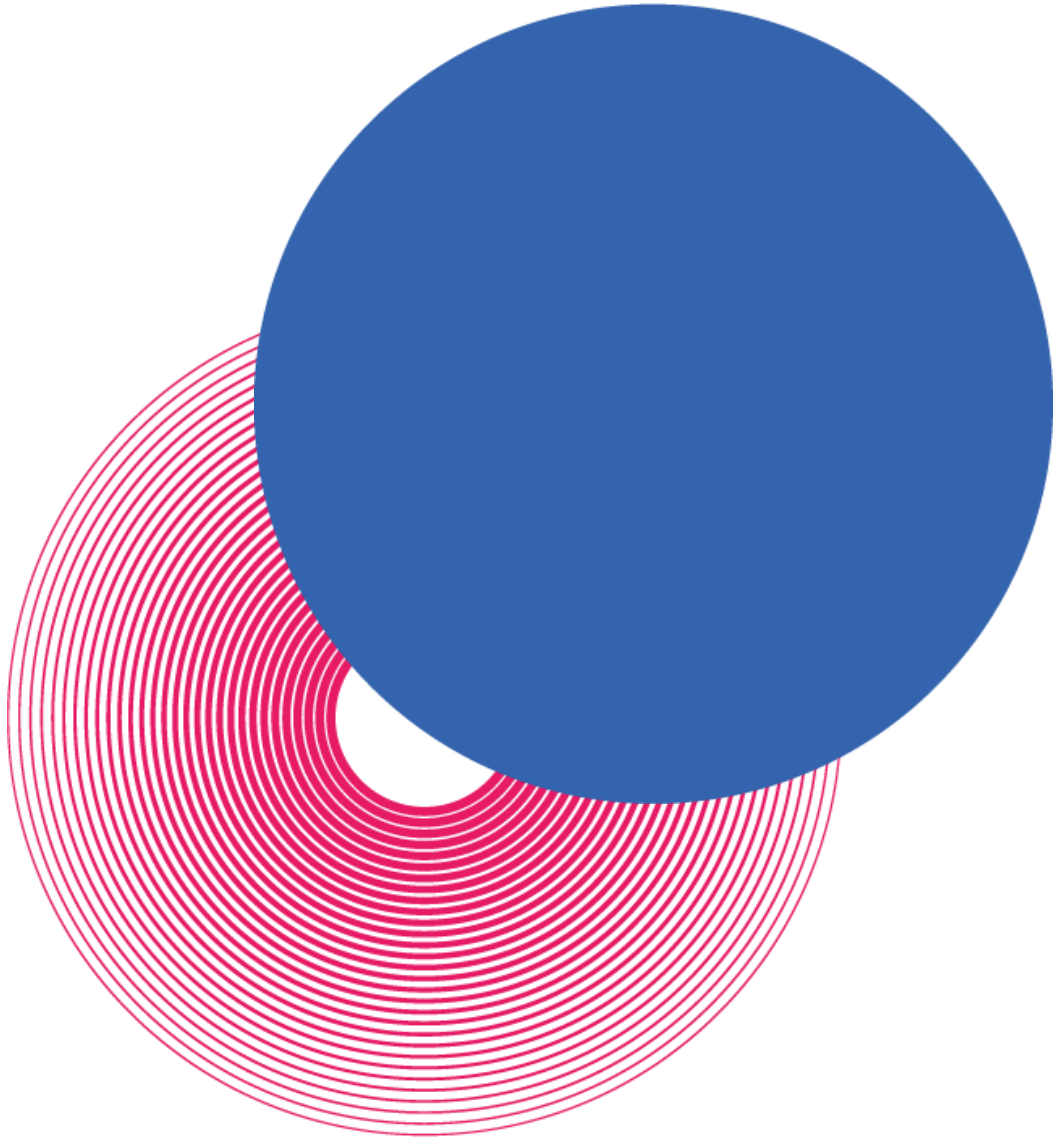
  // Esta es una línea comentada.

- Comentarios de bloques.

  /* Aquí empieza el bloque comentado
  y aquí acaba */

- Comentarios de documentación (JavaDoc).

  / Los comentarios de documentación se realizan de este modo */**



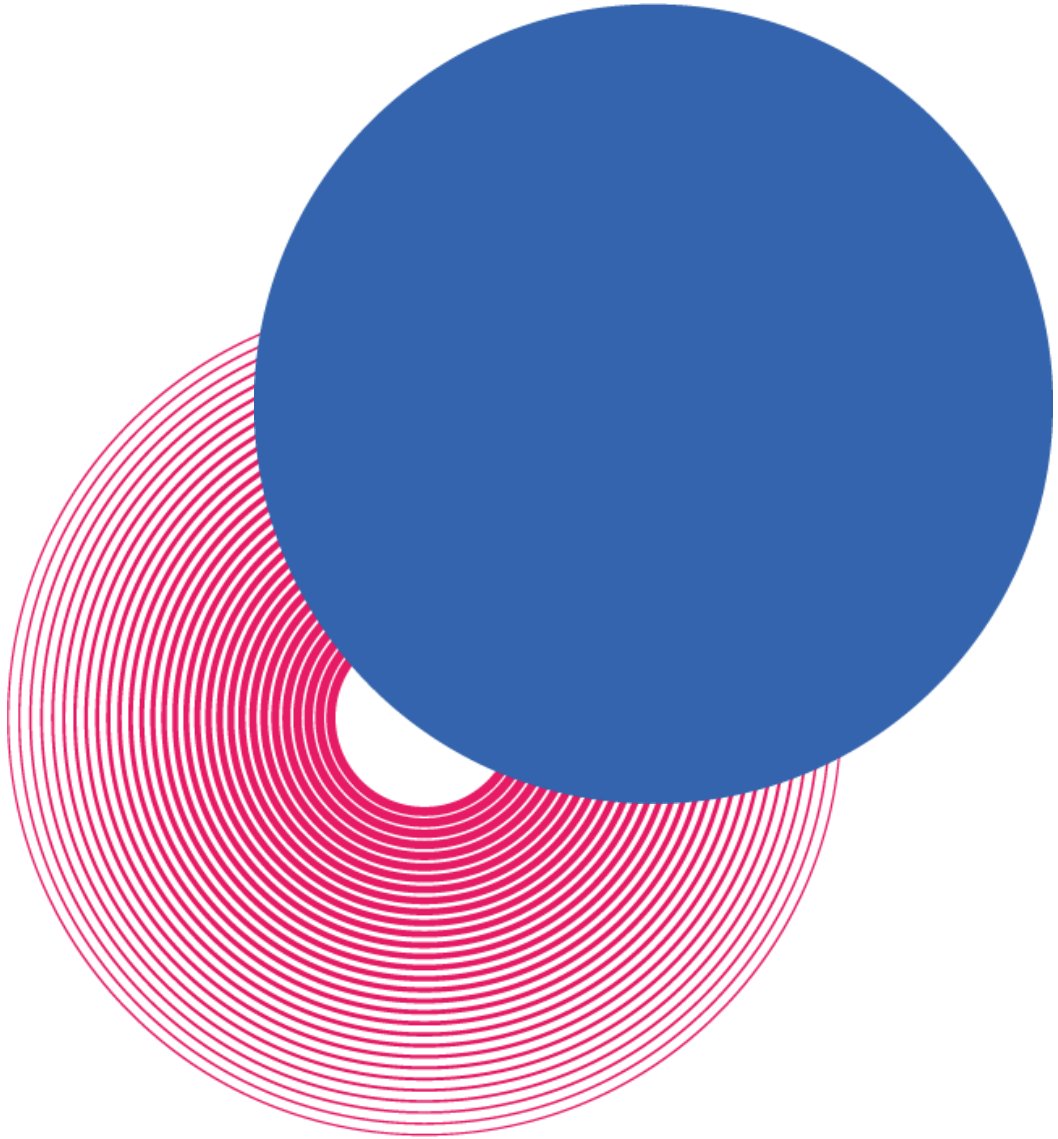
Identificadores

En Java los identificadores comienzan por una letra del alfabeto inglés, un subrayado _ o el símbolo de dólar $.

Los siguientes caracteres del identificador pueden ser letras o dígitos.

No se debe nunca iniciar con un dígito.

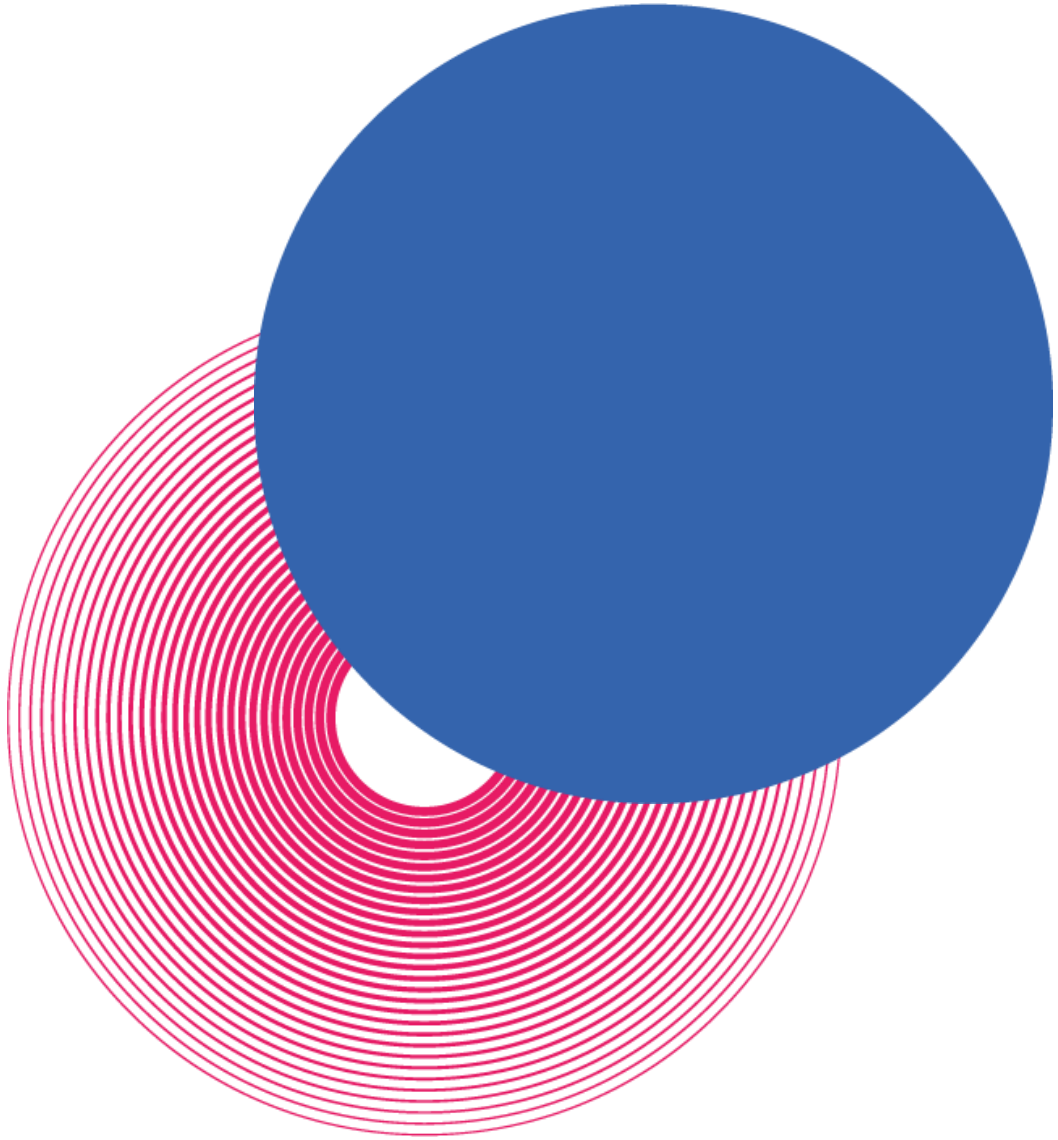
No hay un límite en lo concerniente al número de caracteres que pueden tener los identificadores.



Identificadores

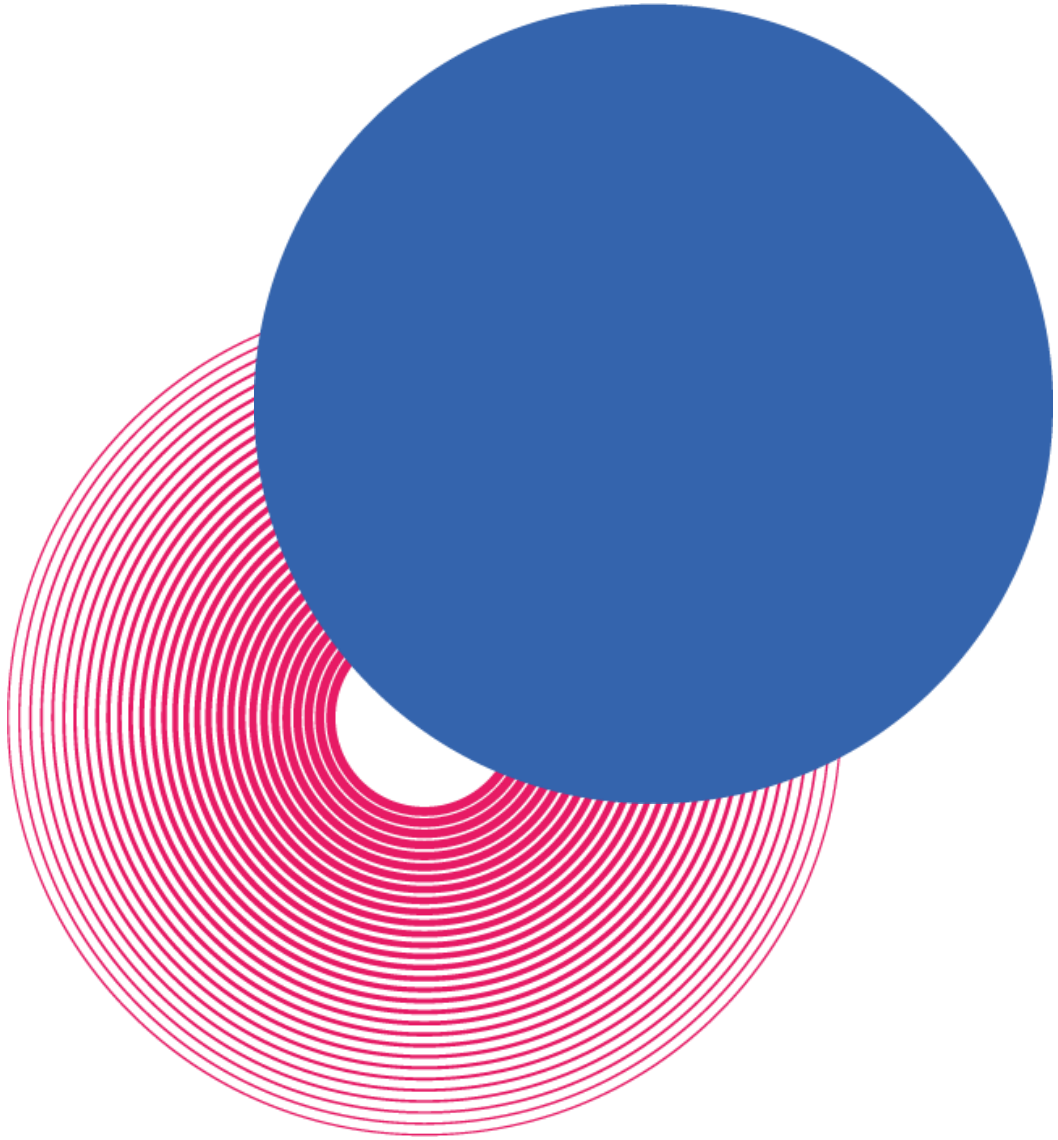
Edad	nombre	_Precio
AÑO0	$cantidad	_$cantidad
num4	bl4nc0	miércoles

Año	año_nacimiento
cantidad_10_1	PrecioVenta
PrIvAdo	máximo



Identificadores

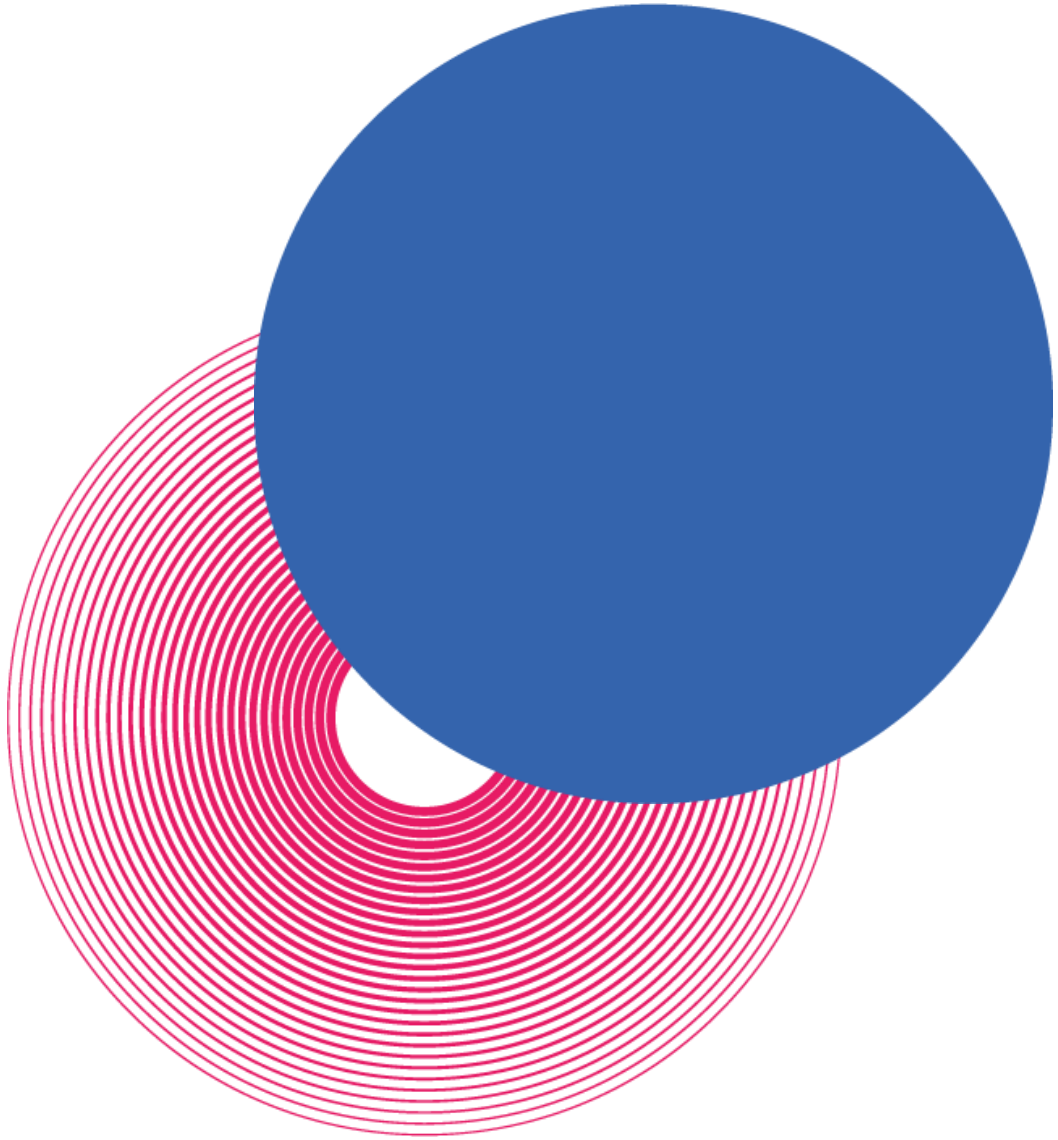
- 4num : Identificador no válido porque comienza por un dígito
- z# : No válido porque contiene el carácter especial #
- "Edad" : No válido porque no puede contener comillas
- Tom's : No válido porque contiene el carácter '
- año-nacimiento : no válido porque contiene el carácter -
- public : no válido porque es una palabra reservada del lenguaje
- __precio:final : no válido porque contiene el carácter :



Reglas

Las reglas del lenguaje respecto a los nombres de variables son muy amplias y permiten mucha libertad al programador, pero es habitual seguir ciertas normas que facilitan la lectura y el mantenimiento de los programas.

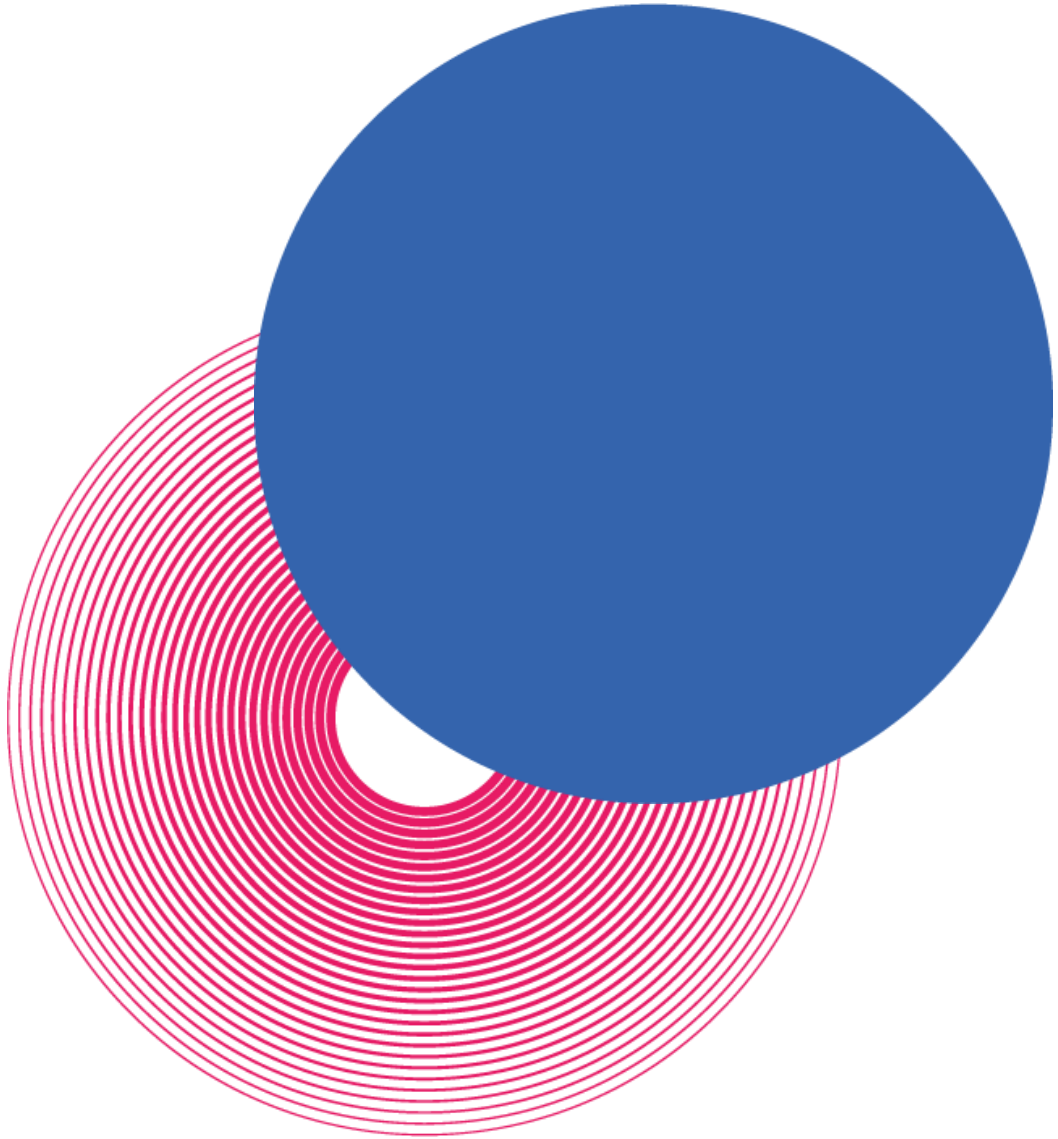
En Java es habitual utilizar nombres con minúsculas, con las excepciones que se indican en los puntos siguientes:



Reglas

- Los nombres de clases e interfaces comienzan siempre por mayúscula.

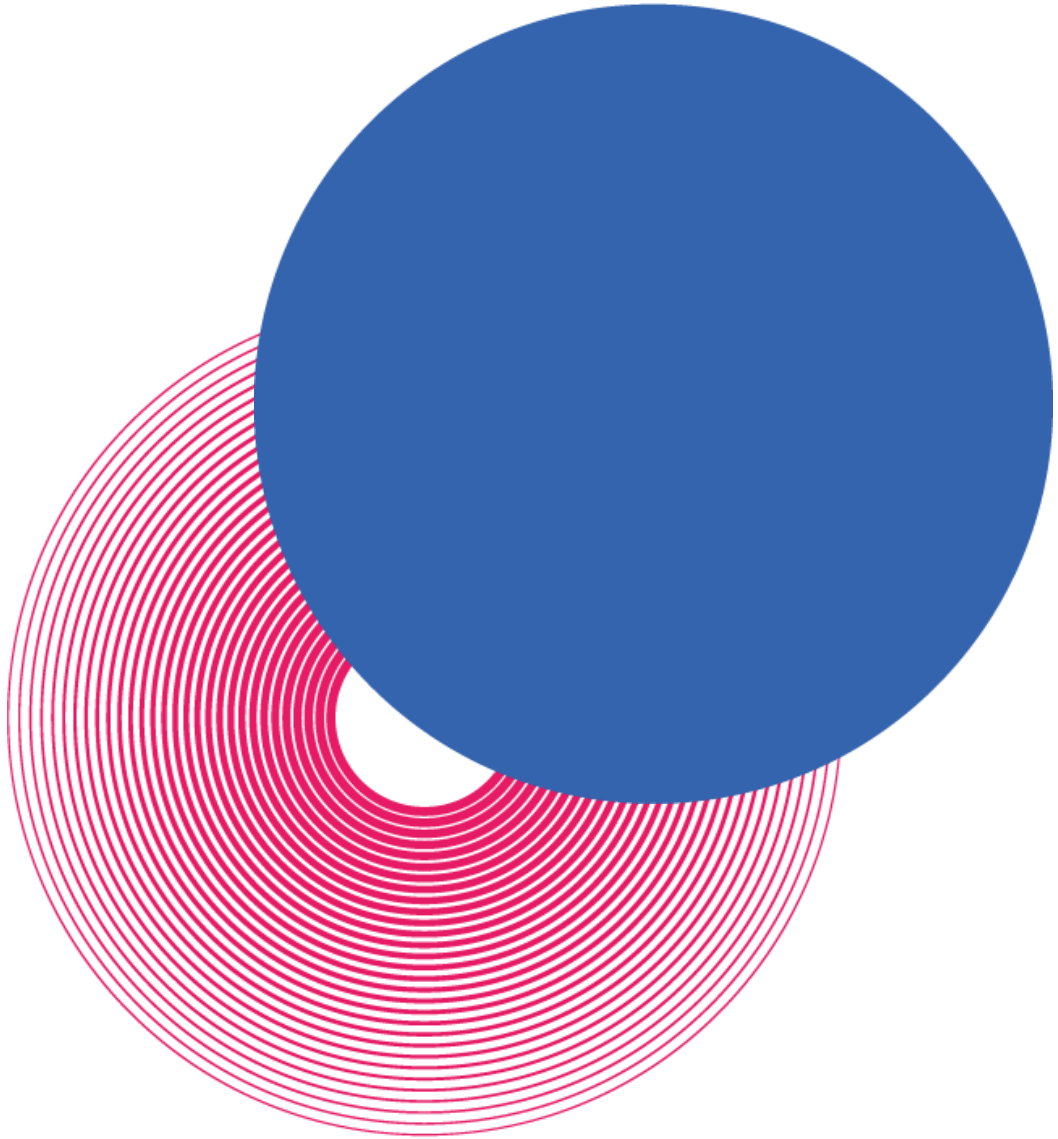
- Cuando un nombre consta de varias palabras es habitual poner una a continuación de otra, poniendo con mayúscula la primera letra de la palabra que sigue (CammelCase).



Reglas

- Los nombres de objetos, métodos, variables miembro y variables locales de los métodos, comienzan siempre por minúscula.

- Los nombres de las variables finales, es decir de las constantes, se definen siempre con mayúsculas.



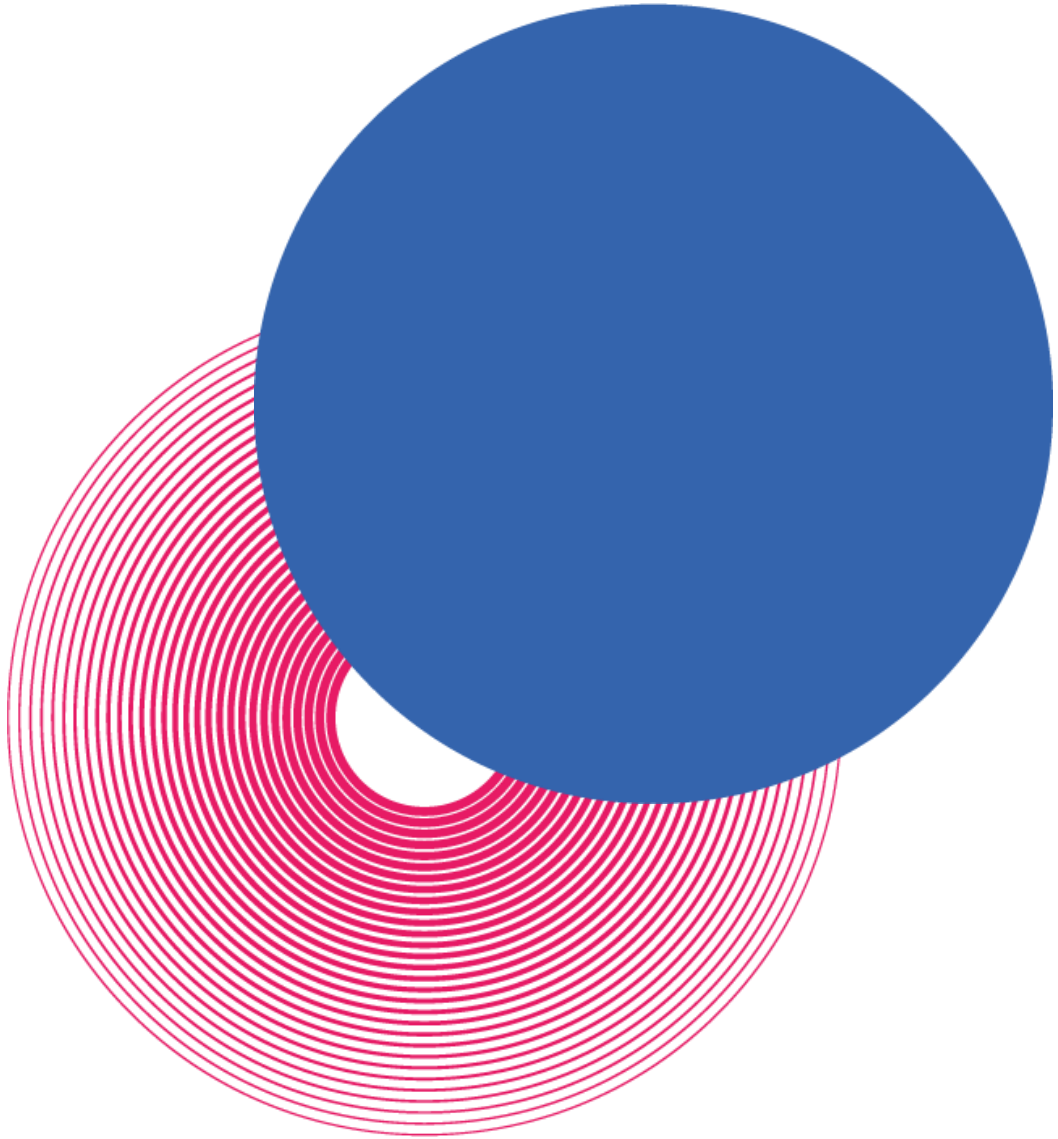
Palabras Reservadas

abstract	boolean	break	byte	case	catch
char	class	const	continue	default	do
double	else	extends	final	finally	float
for	goto	if	implements	import	instanceof
int	interface	long	native	new	package
private	protected	public	return	short	static
strictfp	super	switch	synchronized	this	throw
throws	transient	try	void	volatile	while
assert	enum				

Tipos de datos

En Java existen dos grupos de datos :

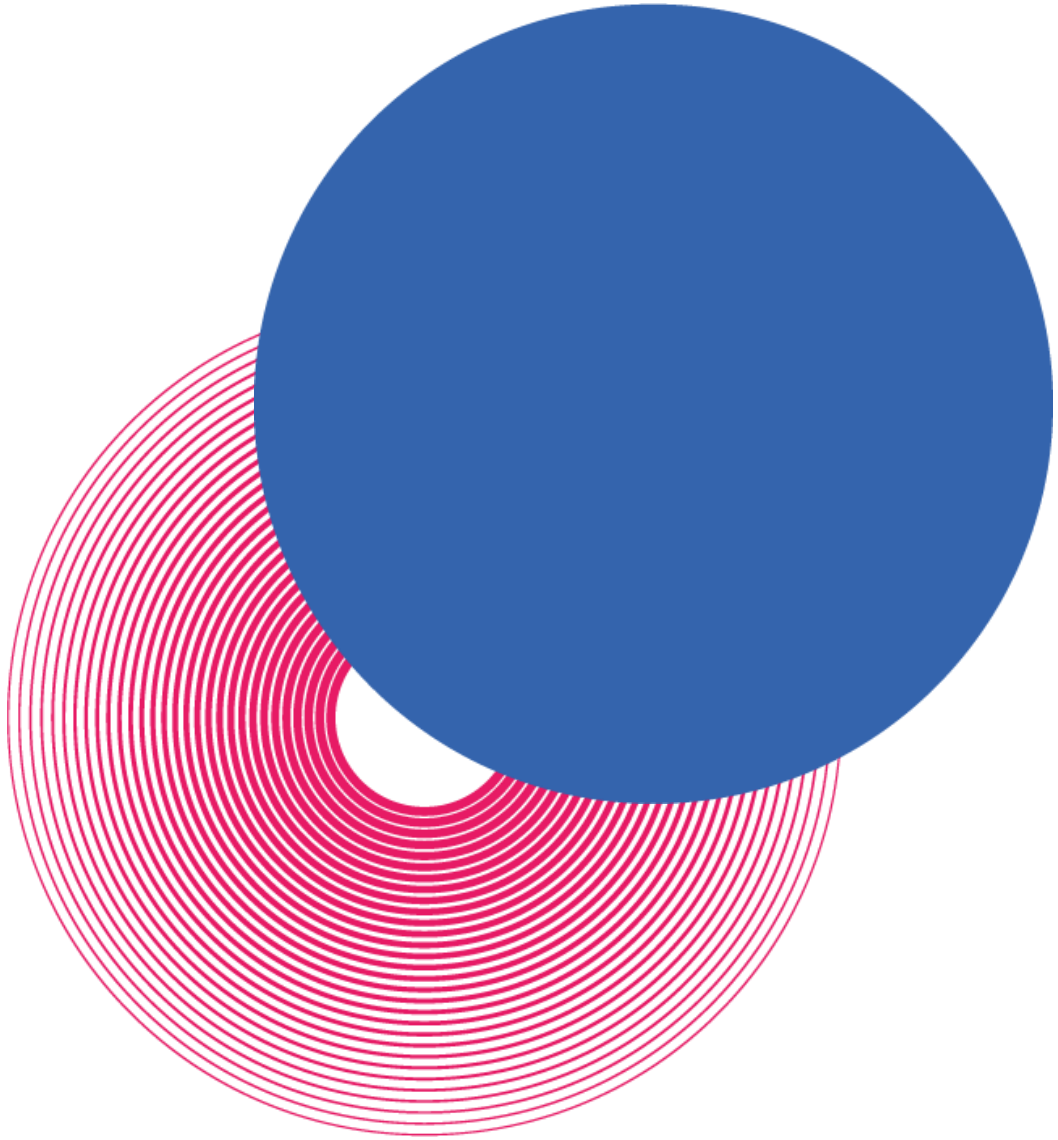
- Tipos primitivos
- Tipos referencia



Tipos de dato Primitivos

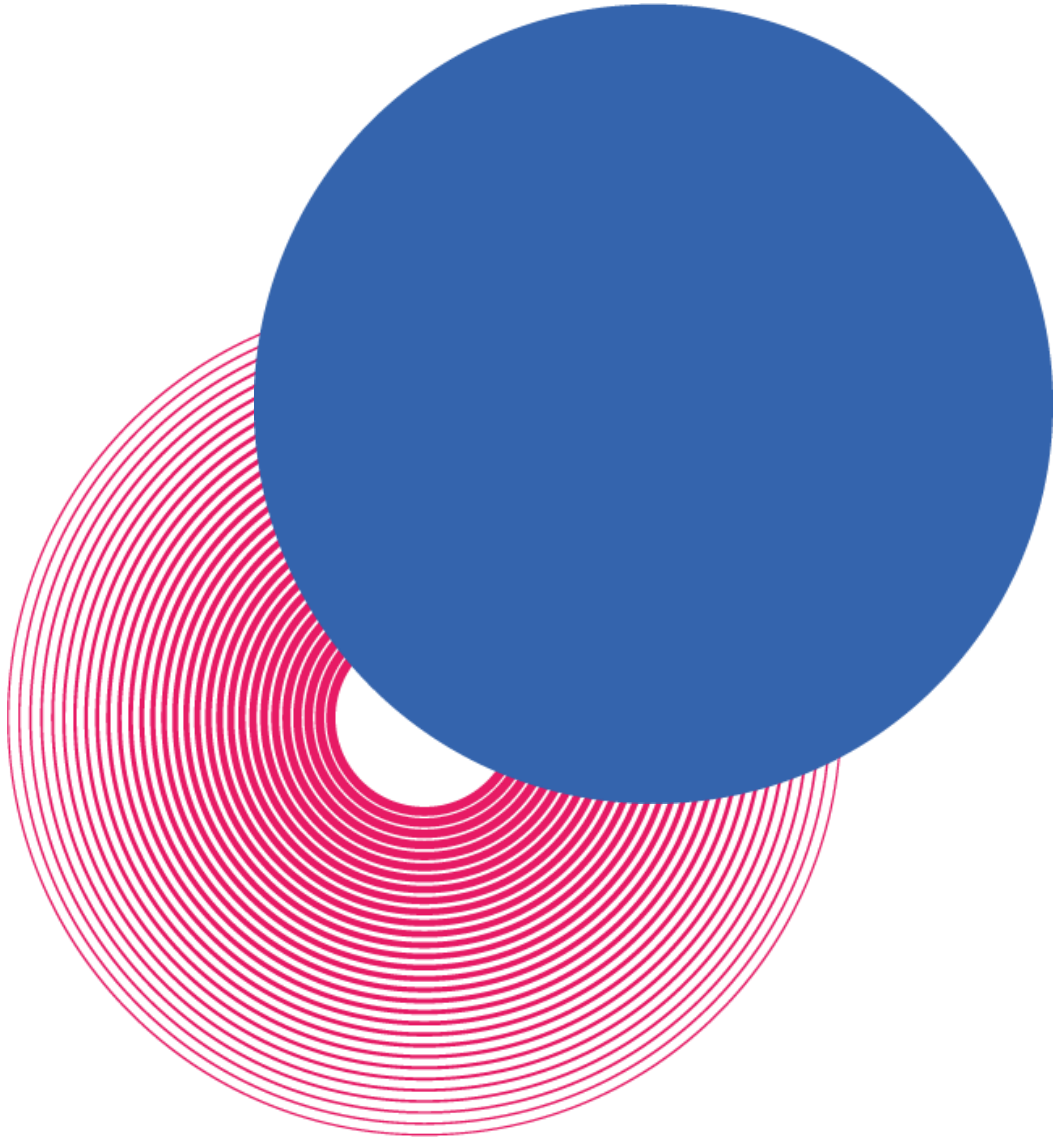
Se llaman tipos primitivos a aquellos datos sencillos que contienen los tipos de información más habituales:

- Valores booleanos
- Caracteres
- Valores numéricos: enteros o de punto flotante



Tipos de dato Primitivos

Tipo	Definición
boolean	`true o false`
char	Carácter Unicode de 16 bits
byte	Entero en complemento a dos con signo de 8 bits
short	Entero en complemento a dos con signo de 16 bits
int	Entero en complemento a dos con signo de 32 bits
long	Entero en complemento a dos con signo de 64 bits
float	Real en punto flotante según la norma IEEE 754 de 32 bits
double	Real en punto flotante según la norma IEEE 754 de 64 bits

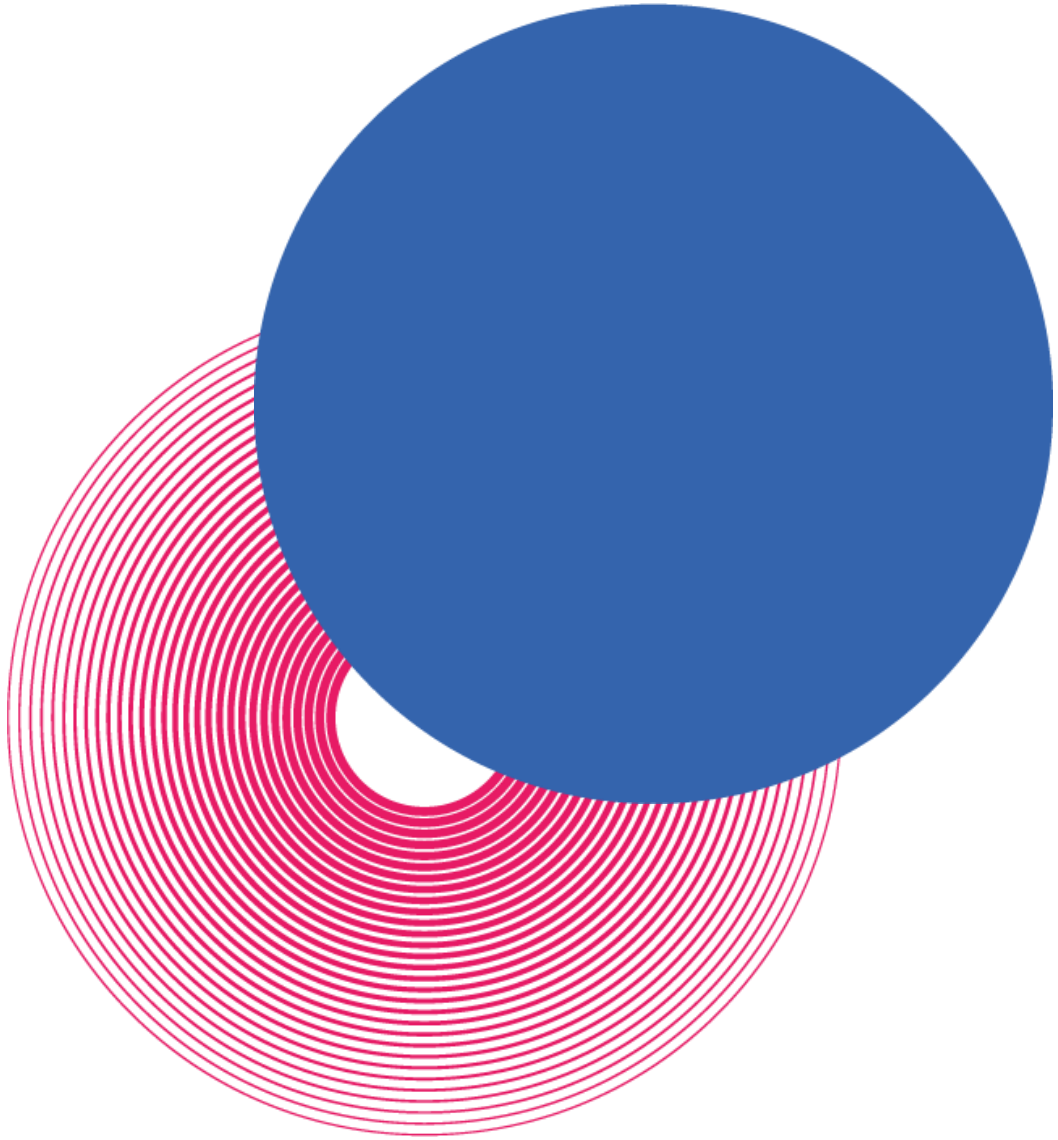


Tipos de dato Referencia

Los tipos de dato referencia representan datos compuestos o estructuras, es decir, referencias a objetos.

Estos tipos de dato almacenan las direcciones de memoria y no el valor en sí (similares a los apuntadores en C).

Una referencia a un objeto es la dirección de un área en memoria destinada a representar ese objeto.

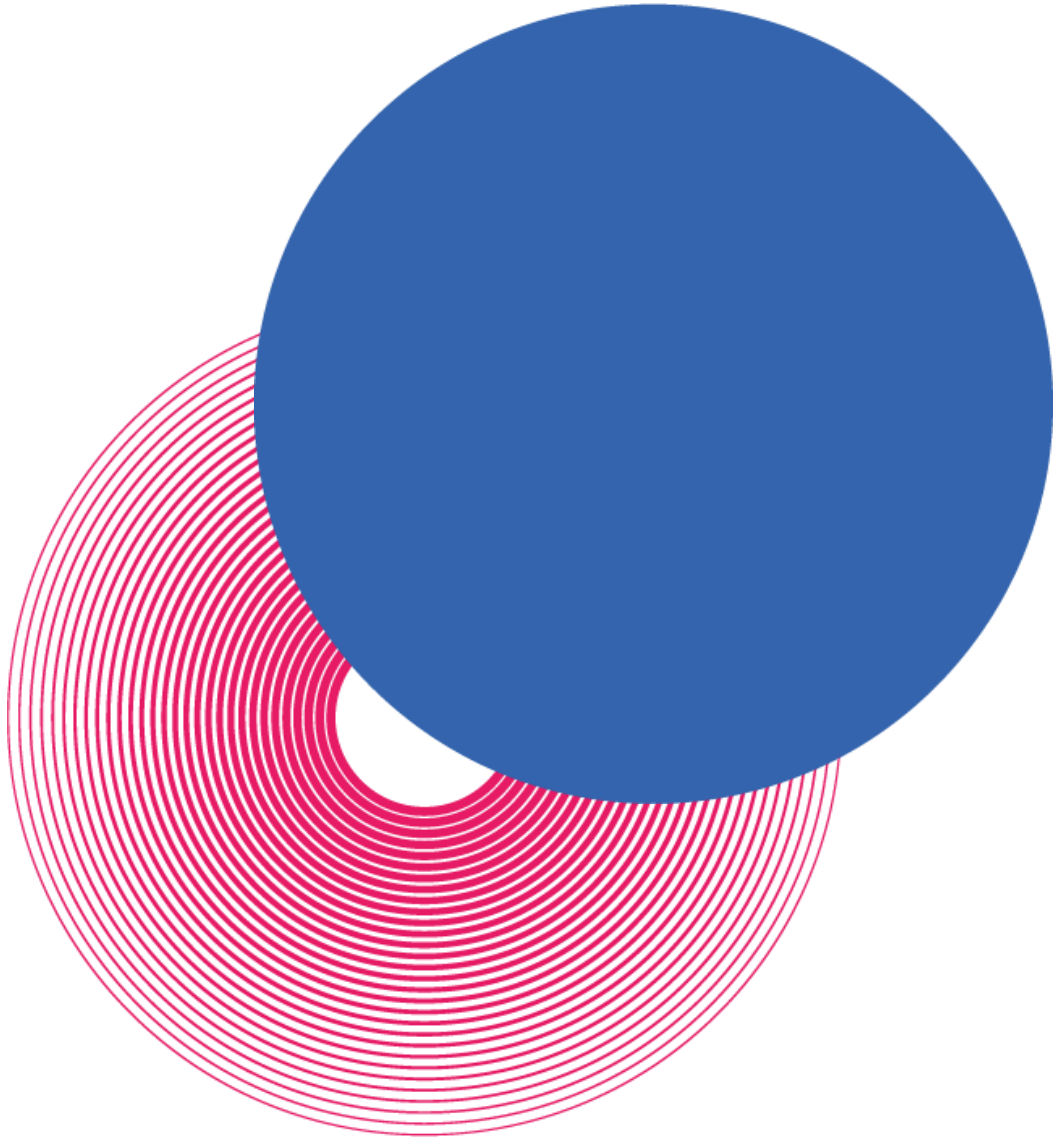


Variables

Una variable es un nombre que contiene un valor que puede cambiar a lo largo del programa.

De acuerdo con el tipo de información que contienen, en Java hay dos tipos principales de variables:

- Variables de tipos primitivos
- Variables referencia

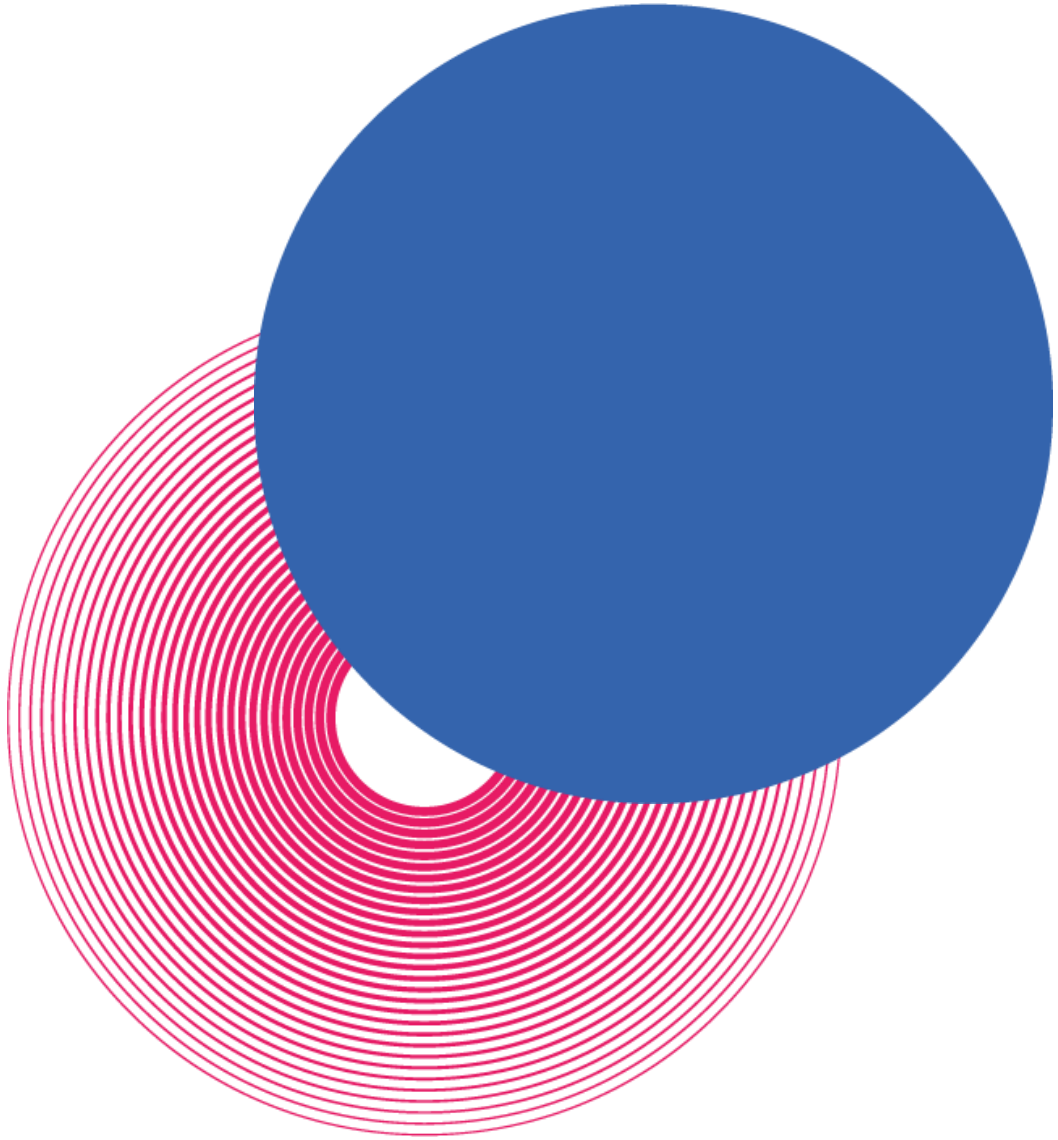


Variables

Desde el punto de vista del papel o misión en el programa, las variables pueden ser:

- **Variables miembro de una clase**: Se definen en una clase, fuera de cualquier método; pueden ser tipos primitivos o referencias.

- **Variables locales**: Se definen dentro de un método o más en general dentro de cualquier bloque entre llaves { }. Se crean en el interior del bloque y se destruyen al finalizar dicho bloque.

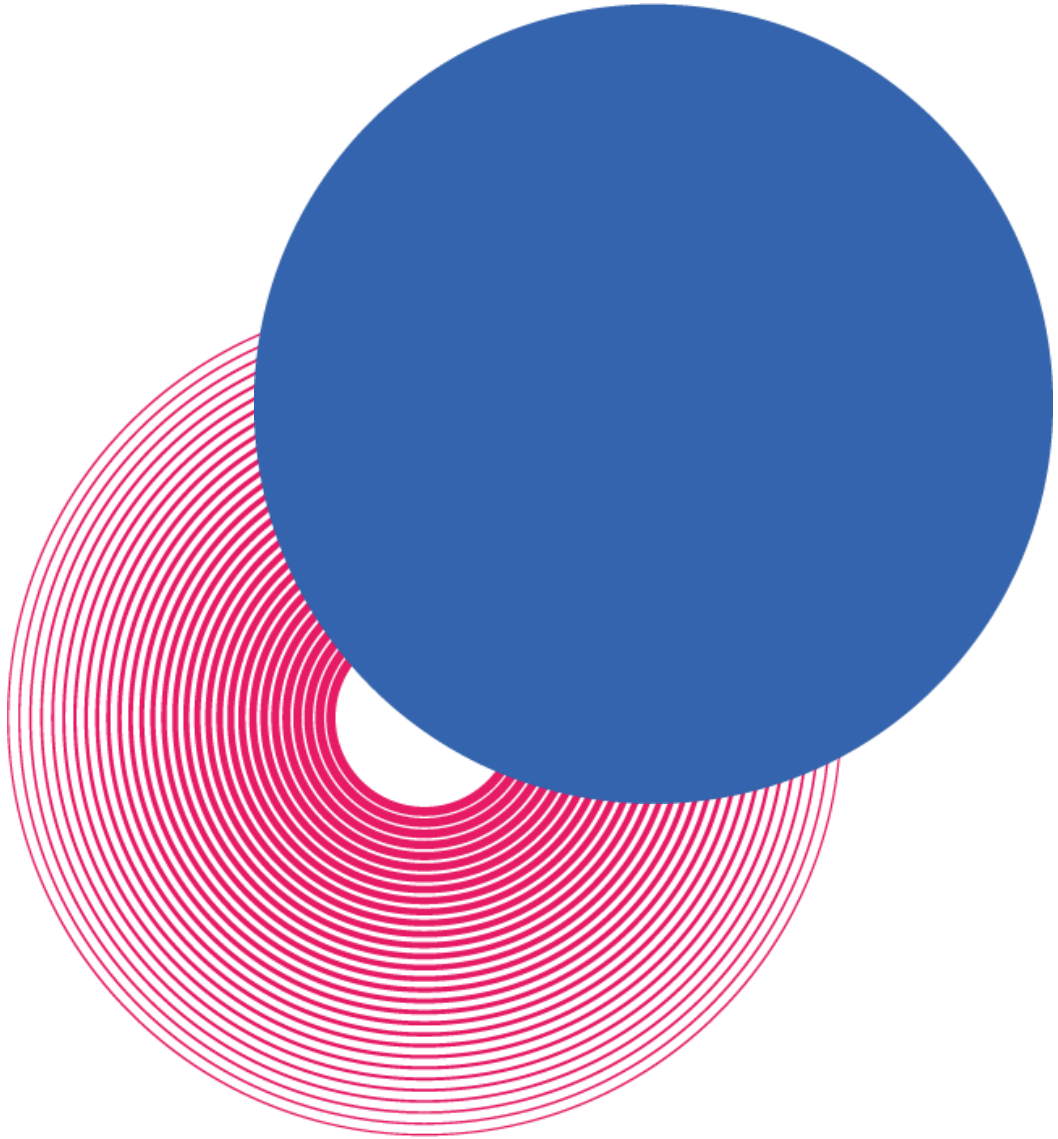


Variables

Una variable se define especificando el **tipo** y el **nombre** de dicha variable.

Estas variables pueden ser tanto de tipos primitivos como referencias a objetos de alguna clase perteneciente al API de Java o generada por el usuario.

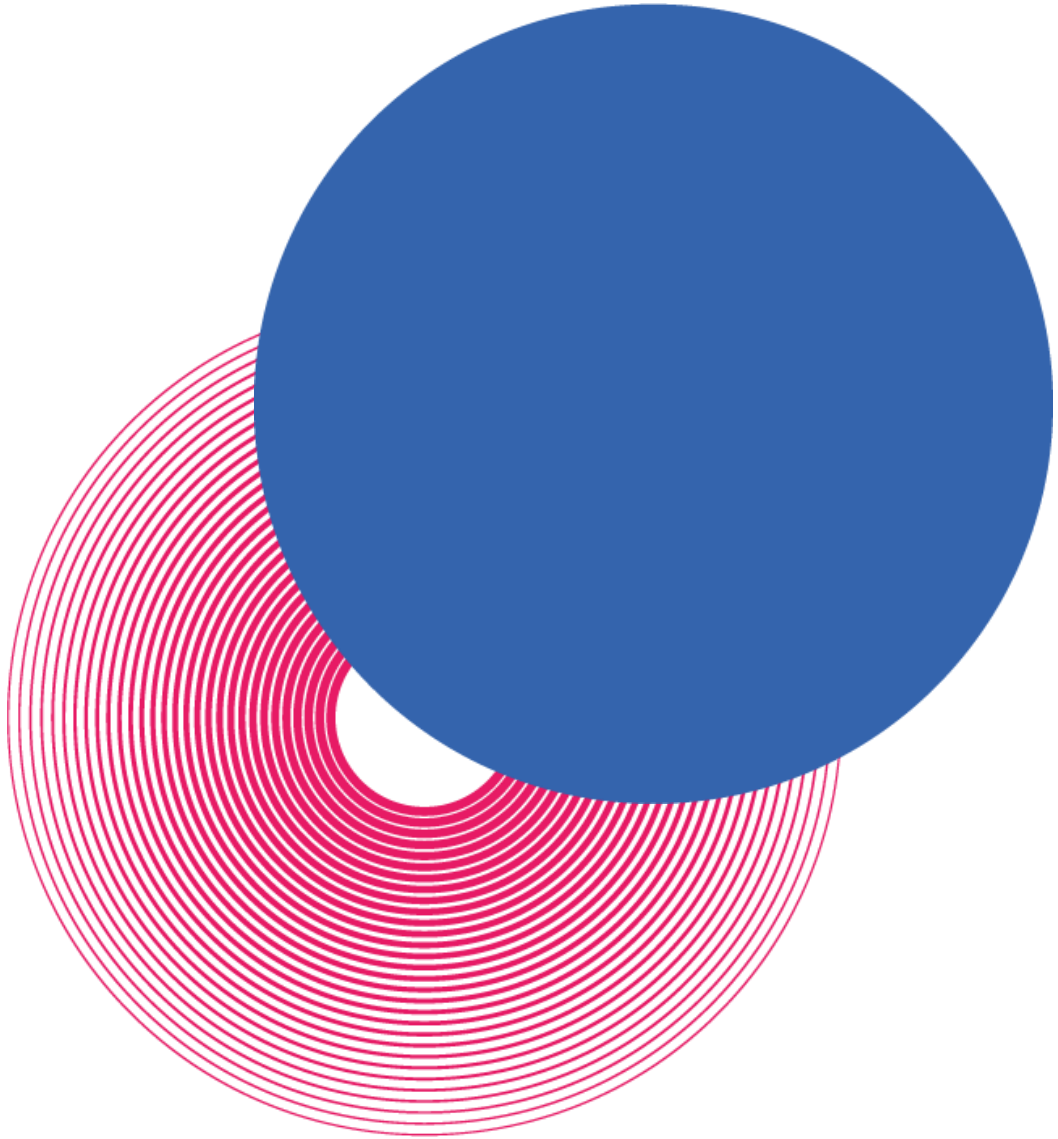
```
tipoDeDato nombreVariable;
```



Variables

Si no se especifica un valor en su declaración, las variables primitivas se inicializan a **cero** (salvo boolean y char, que se inicializan a **false** y '**\0**').

Análogamente las variables de tipo referencia son inicializadas por defecto a un valor especial: **null**.

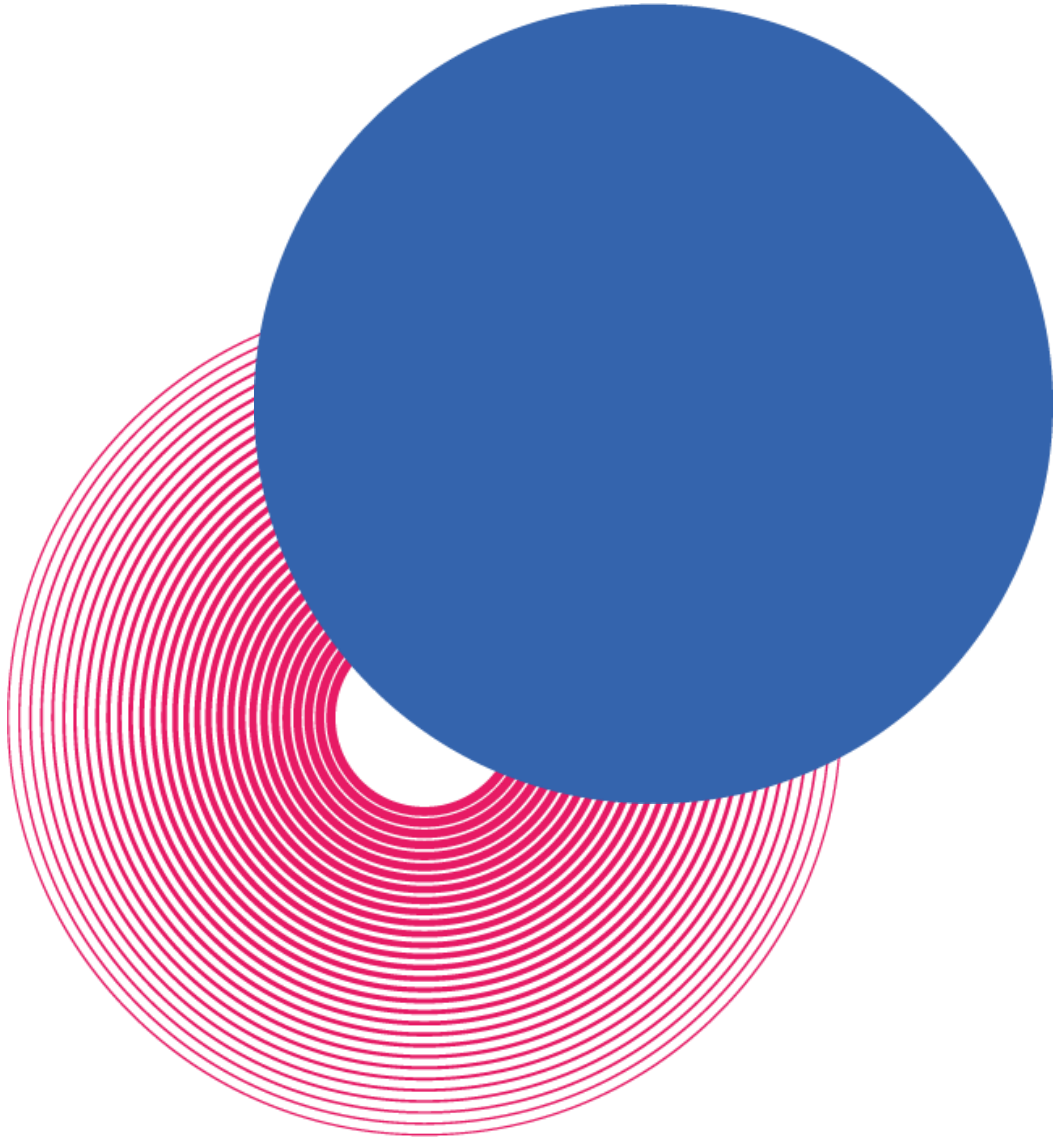


Arreglos

Un tipo particular de referencias son los **arrays** o arreglos, sean éstos de variables primitivas (por ejemplo de enteros) o de objetos.

En la declaración de una referencia de tipo **array** hay que incluir los **corchetes** [ ].

Ejemplo:
int [ ] vector;
vector = **new** int[10];
MyClass [ ] lista=**new** MyClass[5];



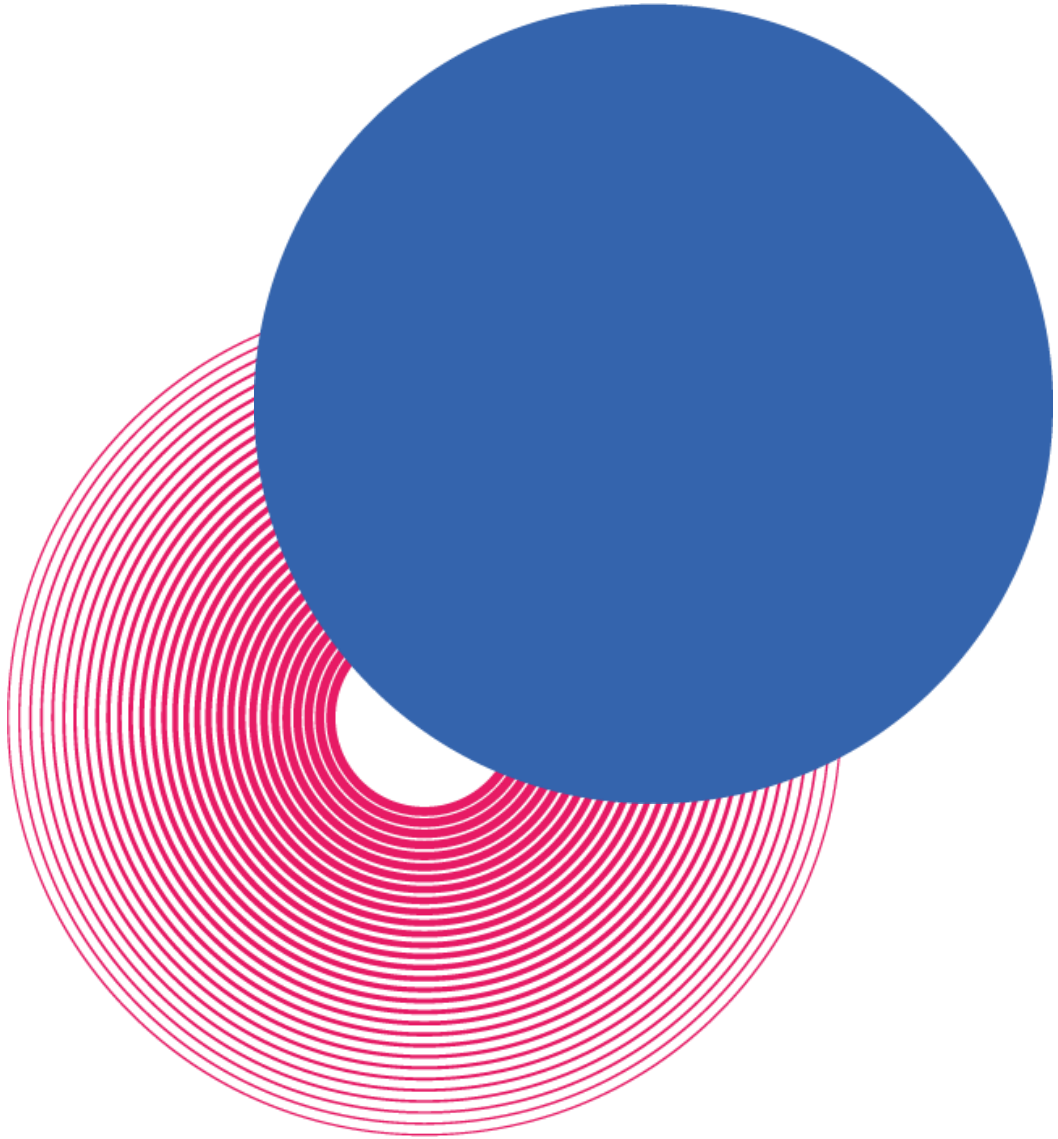
Constantes

Una **constante** es una variable cuyo valor **no puede ser modificado.**

Para definir una constante en Java se utiliza la palabra reservada **final**, delante de la declaración del tipo, de la siguiente manera:

final tipoDato nombreDeConstante = valor;

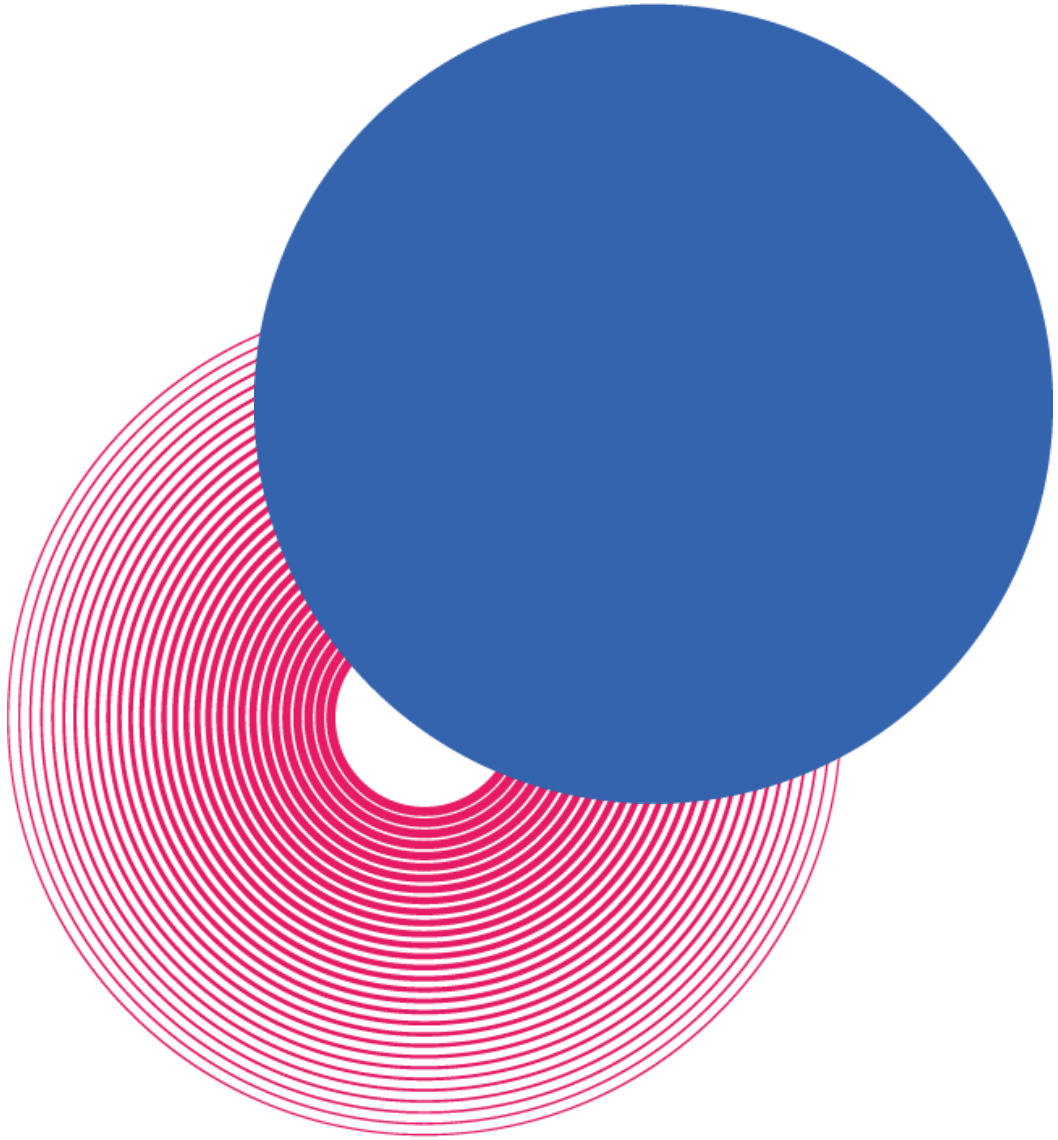
Ejemplo:
final double PI = 3.1416;



Entrada y Salida por consola

Una de las operaciones más habituales que tiene que realizar un programa es intercambiar datos con el exterior.

Para ello el **API** de Java incluye una serie de clases que permiten gestionar la entrada y salida de datos en un programa, independientemente de los dispositivos utilizados para el envío/recepción de datos.

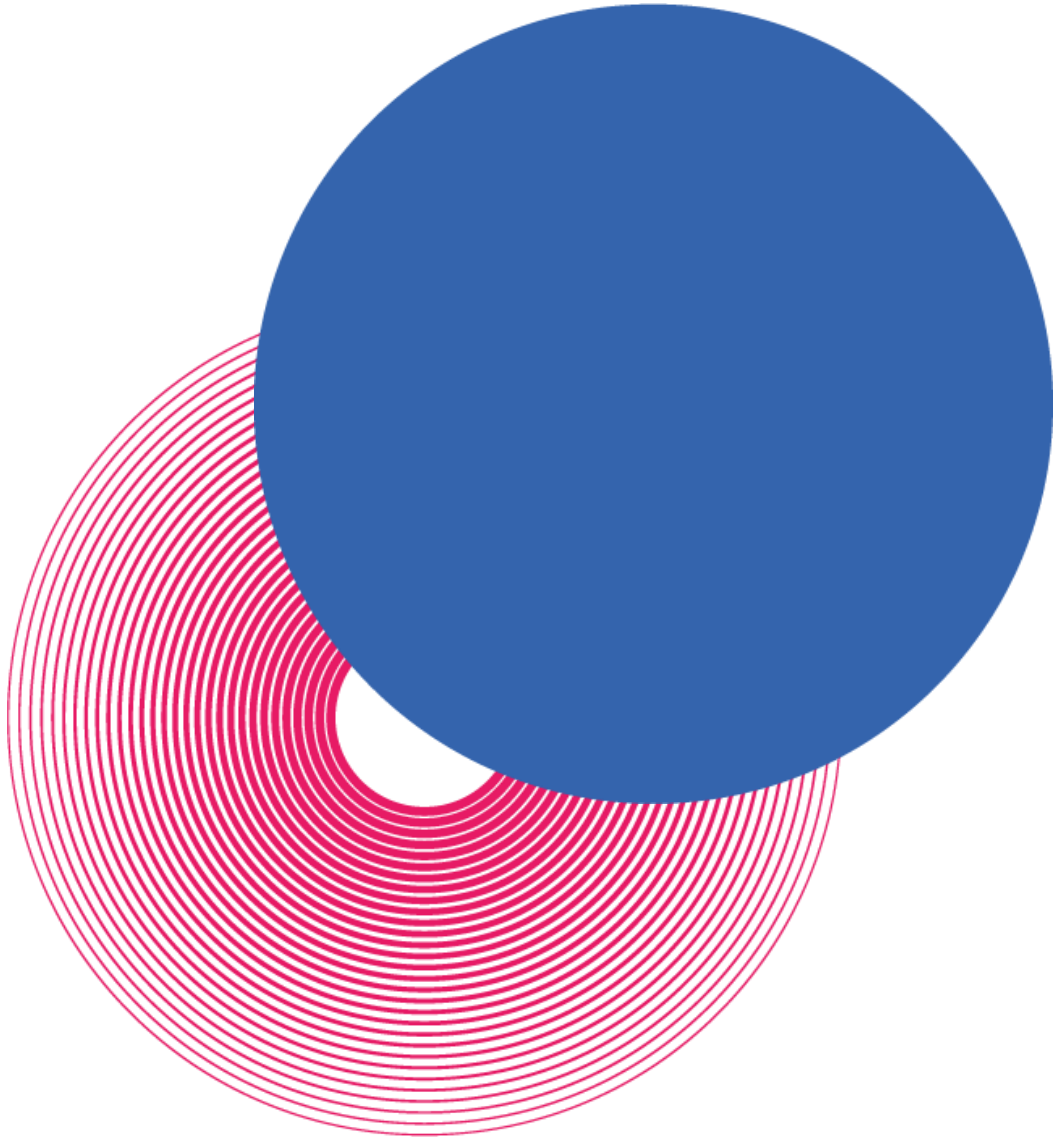


Salida por consola

Para el envío de datos al exterior se utiliza un flujo de datos de impresión o **print stream**.

Esto se logra usando la siguiente expresión:

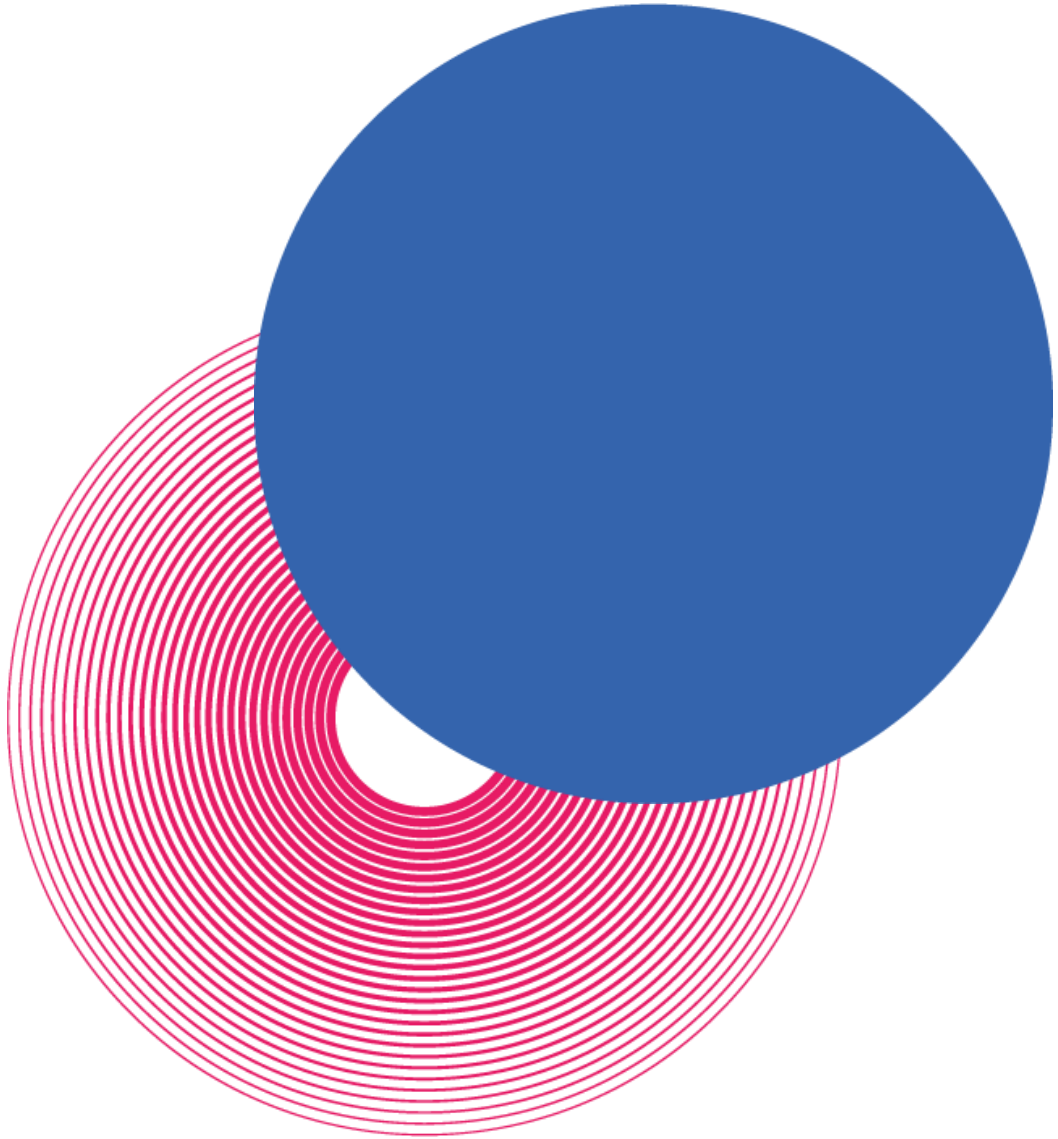
System.out.println(“Mi mensaje”);



Entrada por consola

Para la recepción o lectura de datos desde el exterior se utiliza un flujo de datos de entrada o **input stream**.

```
Scanner sc = new Scanner (System.in);
String s = sc.next(); //Para cadenas
int x = sc.nextInt(); //Para enteros
float y = sc.nextFloat(); //Para enteros
```



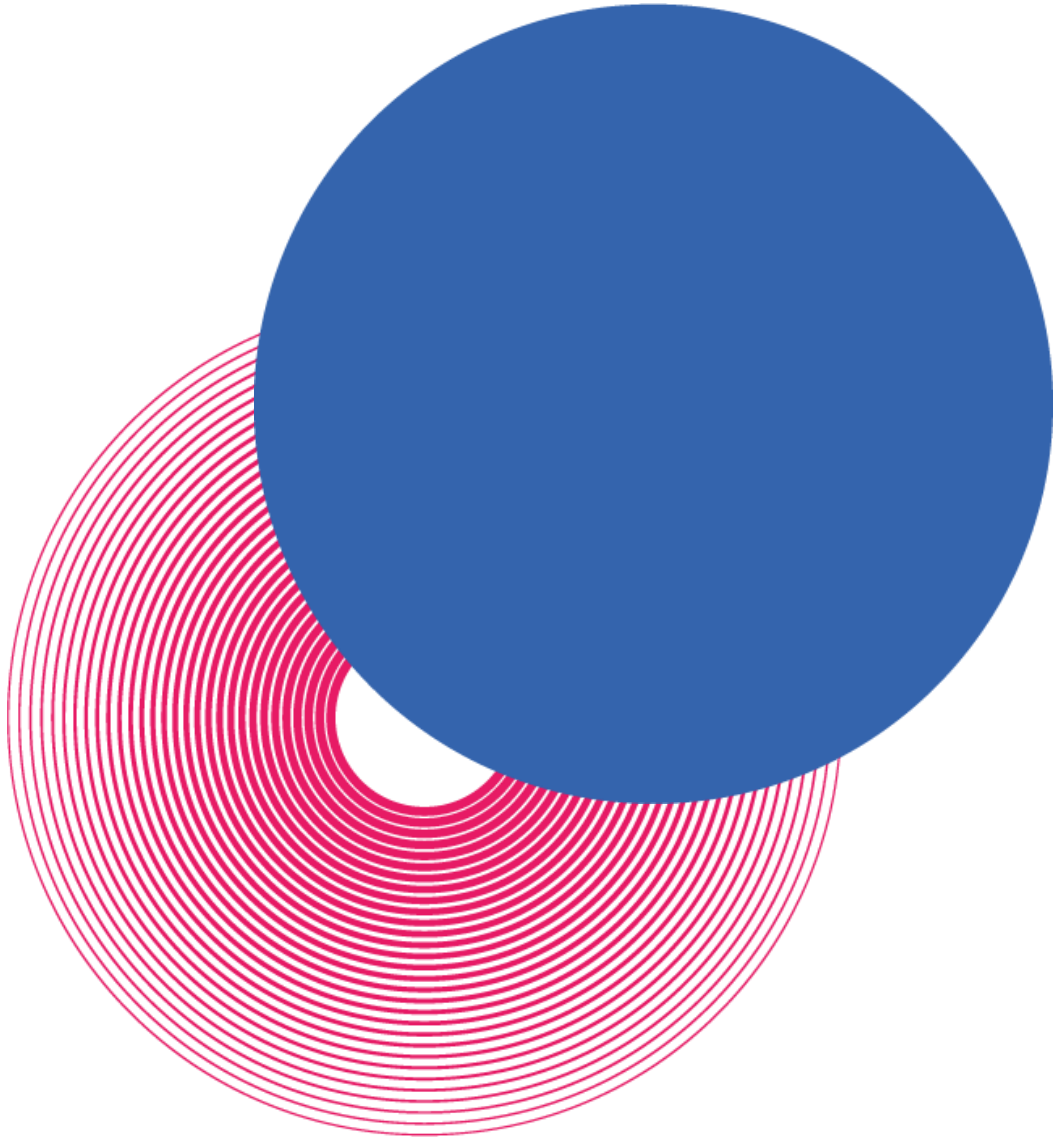
Entrada por consola

Para usar la clase Scanner se debe incluir al inicio del archivo la siguiente línea:

```
import java.util.Scanner;
```

Al finalizar su uso es una buena práctica cerrar el flujo usando el método close.

```
sc.close( );
```



¡A practicar!