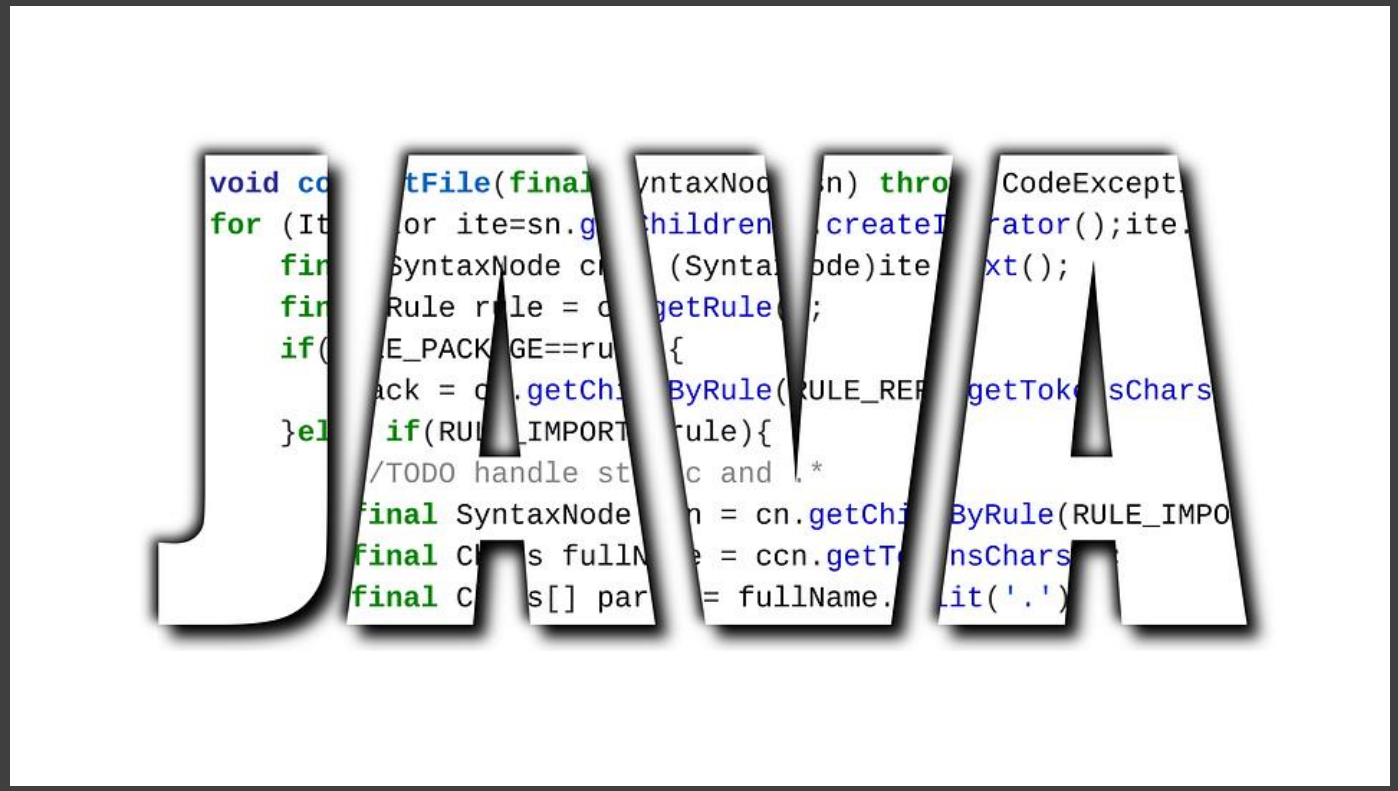
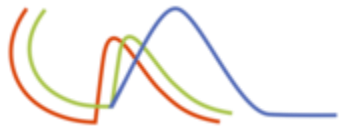


Tipos primitivos

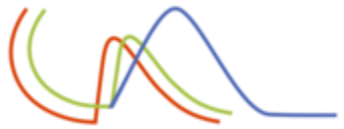
Tipos de datos

TIPO	DESCRIPCIÓN	TAMAÑO
`byte`	Entero de un byte	8 bits con signo en complemento a dos
`short`	Entero corto	16 bits con signo en complemento a dos
`int`	Entero	32 bits con signo en complemento a dos
`long`	Entero largo	64 bits con signo en complemento a dos
`float`	Nº en coma flotante de simple precisión	Nº de 32 bits IEEE 754
`double`	Nº en coma flotante de doble precisión	Nº de 64 bits IEEE 754
`char`	Un carácter Unicode	Carácter Unicode de 16 bits
`boolean`	Valor booleano (true, false)	8 bits



Valores por defecto

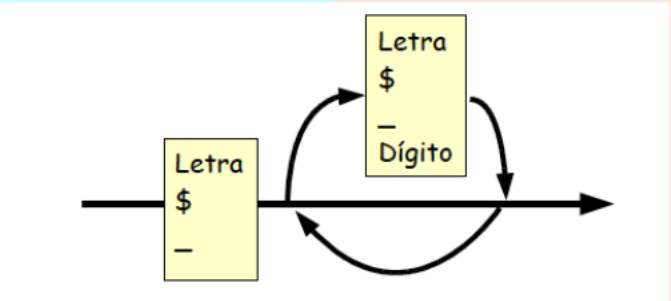
Valores por defecto: `boolean` a false, `char` a '\0' y el resto a cero. Los tipos referencia a `null`.



Identificadores

Utilizados para dar nombre a clases, métodos, campos, variables, constantes y etiquetas. Un identificador no puede coincidir con ninguna palabra reservada del lenguaje. Además, hay que tener en cuenta que Java es sensible a mayúsculas y minúsculas.

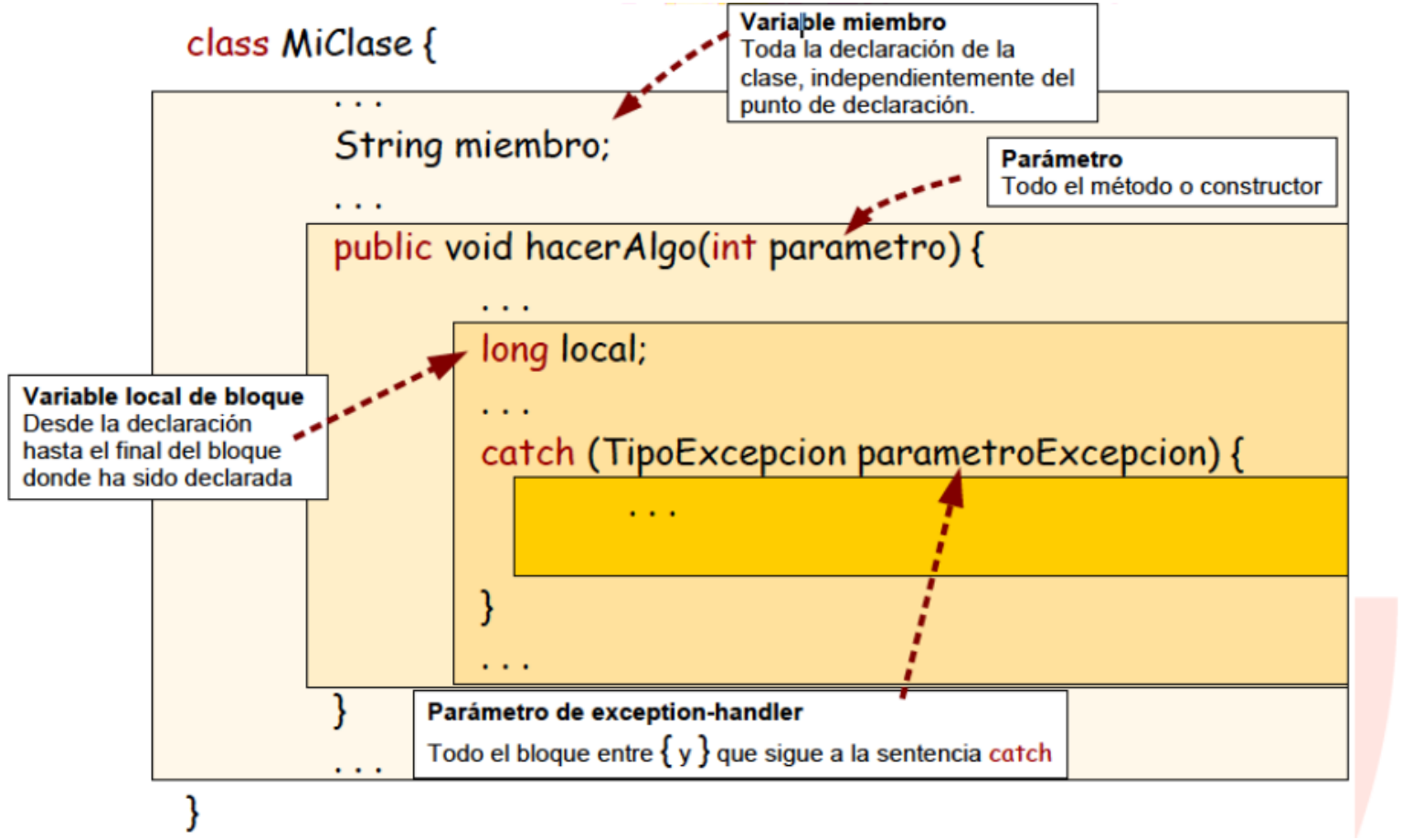
Un identificador solo puede comenzar por una letra, $ o _ y continuar con letras, $ o dígitos numéricos:



Para el nombrado de identificadores se sigue el siguiente convenio:

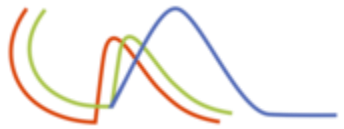
- **Variables**: las palabras en minúscula. Si el nombre es compuesto, la inicial de cada palabra a partir de la segunda en mayúsculas (Ej: unaVariable).
- **Clases**: todo en minúscula salvo la primera letra de cada palabra (Ej: MiClase).
- **Constantes**: las palabras en mayúsculas separadas cada una por el carácter subrayado (UNA_CONSTANTE).

Ámbito


class MiClase {
. . .
String miembro;
. . .
public void hacerAlgo(int parametro) {
. . .
long local;
. . .
catch (TipoExcepcion parametroExcepcion) {
. . .
}
. . .
}
. . .
}
Variable miembro
Toda la declaración de la clase, independientemente del punto de declaración.
Parámetro
Todo el método o constructor
Variable local de bloque
Desde la declaración hasta el final del bloque donde ha sido declarada
Parámetro de exception-handler
Todo el bloque entre { y } que sigue a la sentencia catch

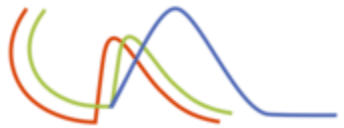
Operadores

TIPO	OPERADOR	SIGNIFICADO
Aritméticos	+	Adición (con String concatena las cadenas)
	-	sustracción
	*	multiplicación
	/	división
	%	resto de división entera
	Con los operadores anteriores se pueden combinar enteros, char y reales. char, byte y short se convierten a int antes de operar. Si los operandos de una expresión son de distinto tipo el inferior es convertido al tipo superior y el resultado es del tipo superior: int → long → float → double	



Precedencia de operadores

PRECEDENCIA OPERADORES	
Postfijos	[] . () expr++ expr--
Prefijos	++expr --expr ~ !
Creación o cast	new (tipo) expr
Aritméticos	* / %
	+ -
Desplazamiento	<< >> >>>
Relacionales	< <= >= > instanceof
Igualdad	== !=
AND bit a bit	&
XOR bit a bit	^
OR bit a bit	|
AND lógico	&&
OR lógico	||
Condicional	?:
Asignación	= += -= *= /= %= >>= <<= >>>= &= ^= |=

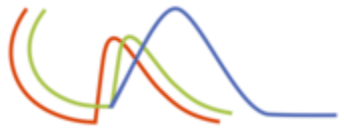


Paquetes más usados

PAQUETES MÁS USADOS	
java.io	Entrada/Salida

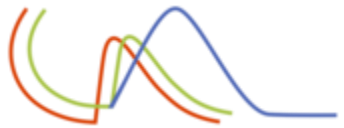
java.lang	Clases fundamentales: Object, tipos de datos (Integer, Double, String…), threads. Destacan:
	- **String**: cadena de caracteres
	- **StringBuffer**: secuencia de caracteres mutable y segura para subprocesos
	- **StringBuilder**: secuencia de caracteres mutable. Se recomienda su uso ya que es más eficiente
	- **Object**: clase raíz de la jerarquía. Métodos:
	  ○ clone: crea una copia de un objeto
	  ○ equals: indica si un objeto es igual a otro
	  ○ hashCode: devuelve un código hash para el objeto
	  ○ toString: cadena de caracteres que representa al objeto

java.math	Clases para tratamientos aritméticos
java.naming	API de JNDI
java.net	Aplicaciones de red
java.rmi	Interconexión aplicaciones protocolo RMI



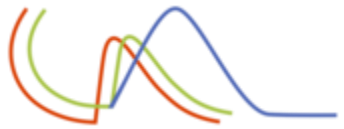
Paquetes más usados …

java.sql **javax.sql**	Manejo de fuentes de datos
java.util	Clases de utilidad: - **Collection** (interfaz): grupo de objetos - **Map** (interfaz): conjunto de pares clave-valor, no admite claves duplicadas - **HashMap**: implementa Map en tabla hash - **Set** (interfaz): colección que no contiene elementos duplicados - **HashSet**: implementación Set en tabla hash - **List** (interfaz): colección ordenada - **ArrayList**: array que implementa List - **Vector**: array - **Stack**: pila, hereda de vector - **Queue** (interfaz): cola - **Deque** (interfaz): cola que permite la inserción/borrado de elementos por los dos extremos

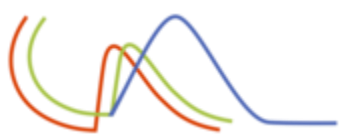


Paquetes más usados …

	- **Iterator** (interfaz): itera sobre una colección.
java.crypto	Operaciones criptográficas
javax.swing	Clases para interfaz gráfica de usuario
javax.xml	Tratamiento de documentos XML



https://docs.oracle.com/en/java/javase/21/docs/api/index.html



Gestión de la memoria de java