


# Java



JAVA és un llenguatge de programació orientat a objectes, multiplataforma.

Es pot considerar un llenguatge compilat i interpretat.

**Origen:** Empresa Sun Microsystem (James Gosling i Patrick Naughton), comprat posteriorment per Oracle (gestor de base de dades).

## Característiques:

- **Senzill** (característiques més enutjoses d'altres llenguatges, i en general ocupa poc espai)
- **Orientat a objectes** (un dels que més)
- **Distribuït** (s'adapta bé a l'hora de programar en xarxa)
- **Segur** (un dels més segurs, no 100%)
- Neutre respecte arquitectura (**multiplataforma**)
- **Compilat i interpretat**
- **Multifil** (es poden programar diversos processos en el mateix programa)



# Llenguatge compilat i no interpretat

```
/* Programa: Suma y multiplicación de dos números (Solución) */

#include <conio.h>
#include <stdio.h>

int main()
{
    int n1, n2, producto, suma;

    printf( "\n  Introduzca primer numero (entero): " );
    scanf( "%d", &n1 );
    printf( "\n  Introduzca segundo numero (entero): " );
    scanf( "%d", &n2 );

    suma = n1 + n2;
    producto = n1 * n2;

    printf( "\n  La suma es: %d", suma );
    printf( "\n\n  La multiplicación es: %d", producto );

    getch(); /* Pausa */

    return 0;
}
```

CODI FONT

COMPILAR  
(pas de codi font a  
codi màquina, per  
exemple  
.exe en windows)



0 0 0 0	1 1 1 0
0 0 0 0	0 0 0 0
0 0 1 1	1 1 1 0
0 1 1 0	0 1 0 0
0 0 0 0	1 1 0 0
1 0 0 1	0 0 0 1
1 1 0 0	0 0 1 0
0 0 0 0	0 0 1 0
0 0 1 0	0 0 0 0

CODI MÀQUINA

PER DIFERENTS PLATAFORMES:

- Windows
- Linux
- Mac

S'han de crear versions diferents per a poder executar-ho en cadascuna de les plataformes

JRE:

Java Runtime Environment (Entorn d'execució de Java).

També conegut com **JVM** (Màquina virtual de Java, java virtual machine)

Principal característica de Java : multiplataforma

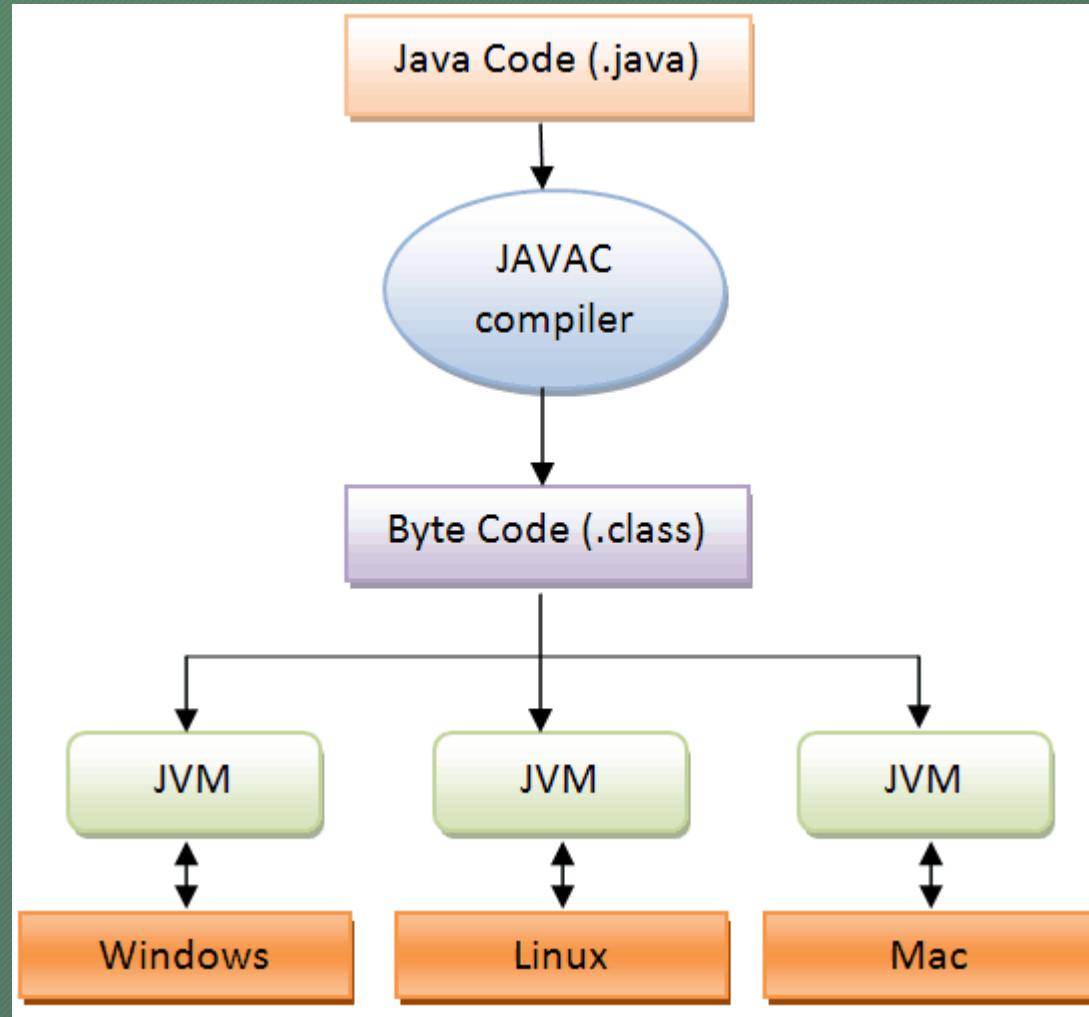
Que permet que inicialment un arxiu java sigui compilat i després interpretat per la JVM.

JRE

# Java es **multiplataforma**:

El compilador de java és javac.

Se li afegeix un nivell entre el programa executable i el codi màquina.



JVM de Java per a casa plataforma de hardware específic

En **compilar** obtenim un arxiu intermedi de bytecodes , codi intermedi entre el codi font i el codi màquina.

El codi per a la JVM és de tipus .class

JVM **interpreta** l'arxiu de bytecodes (.class)

**Write once,  
run every  
where**



# IDE

Entorn de desenvolupament integrat.

- Aplicació de software que facilita als programadors a desenvolupar codi de software de manera eficient.
- Combina eines de desenvolupador comuns en una única interfície gràfica d'usuari.

IDE més emprats per programar en Java:

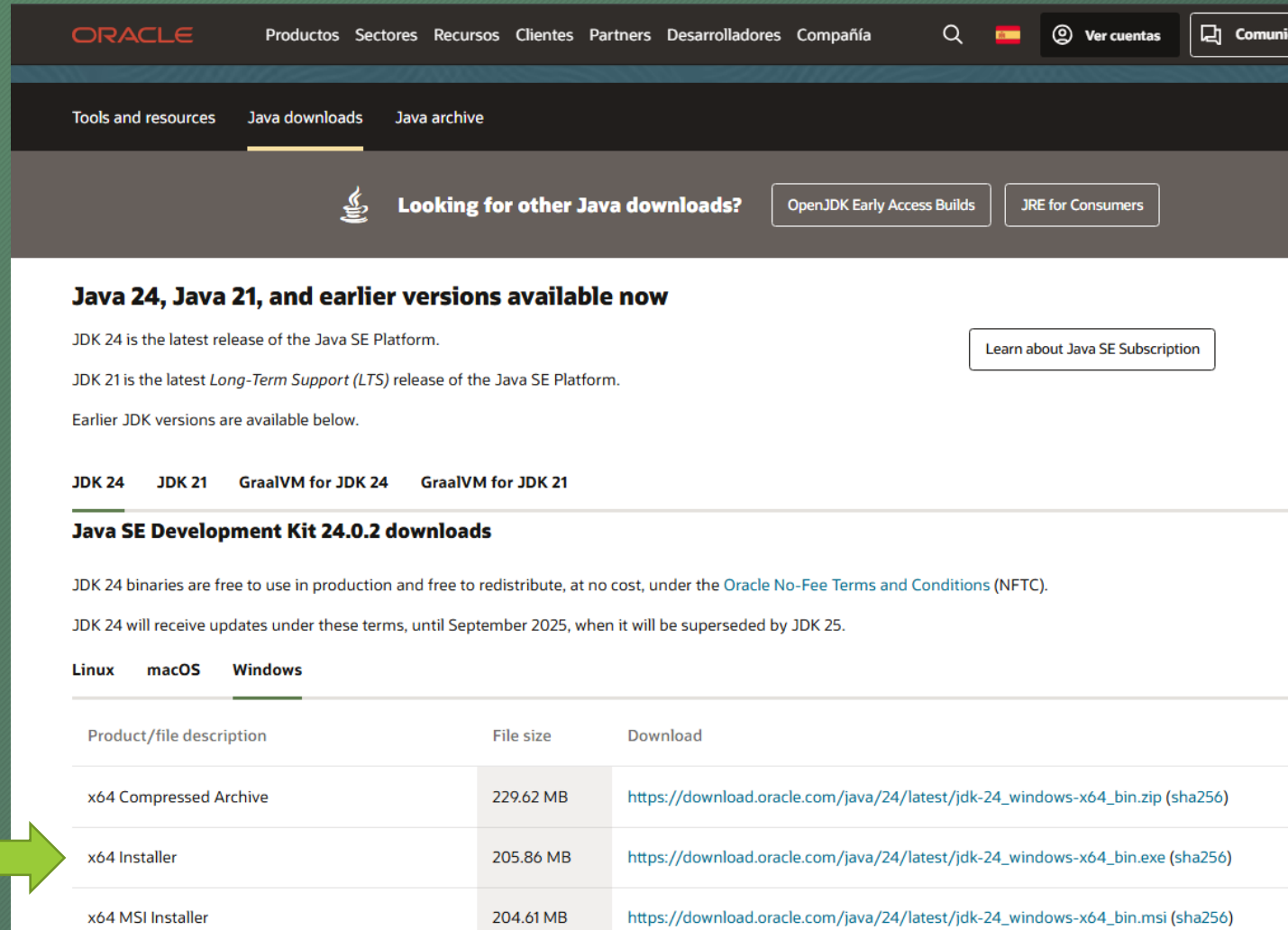
Eclipse, NetBeans, IntelliJ Idea, BlueJ...

Eclipse



# Instal·lacions

Instal·lar JAVA: <https://www.oracle.com/es/java/technologies/downloads/#jdk24-windows>



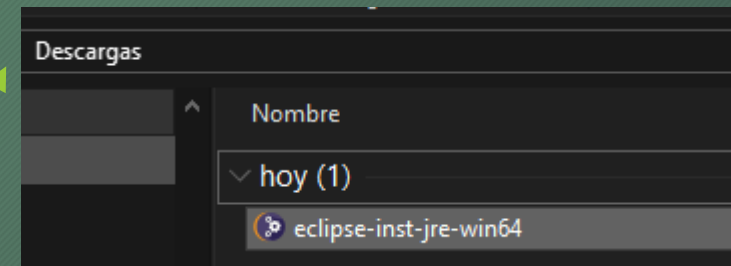
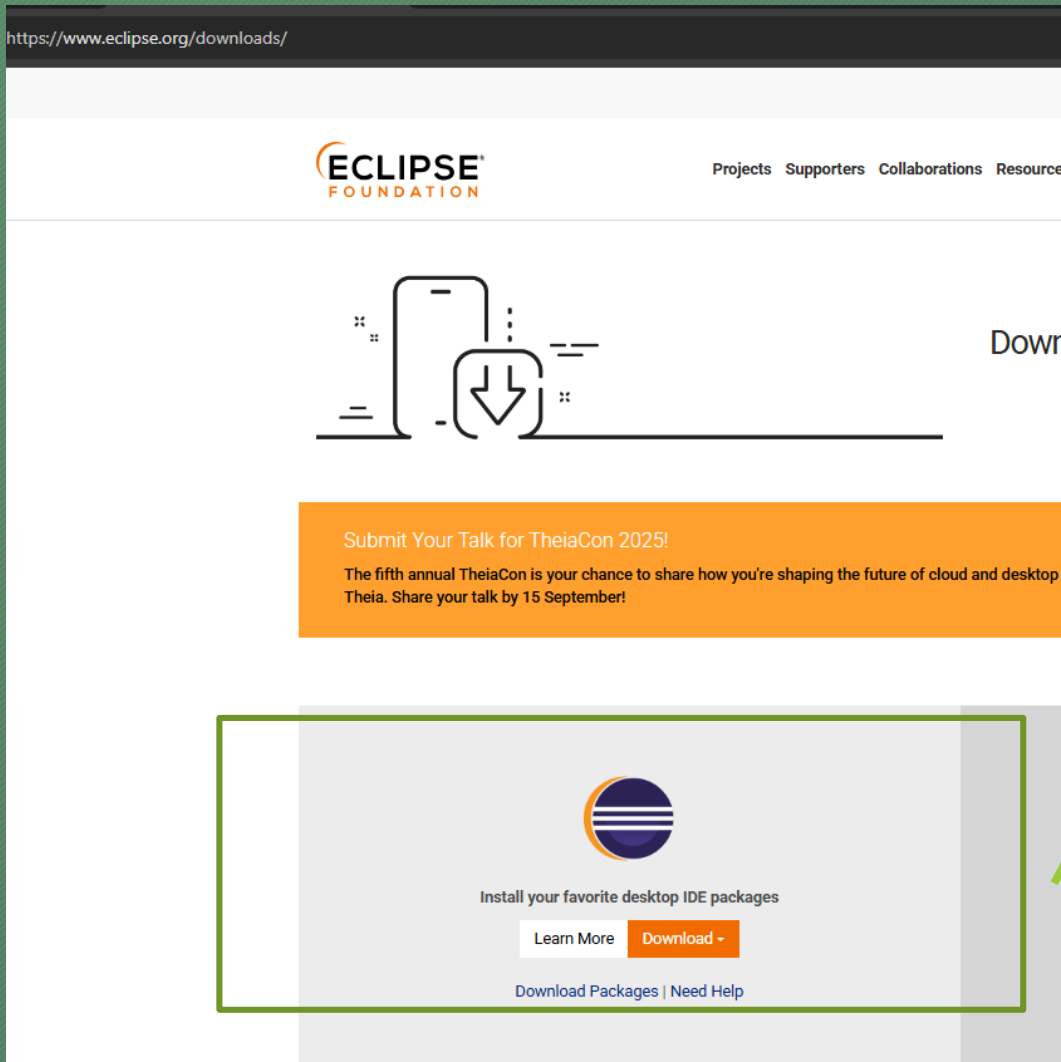
The screenshot shows the Oracle Java Downloads page for JDK 24 on Windows. The page includes a navigation bar with the Oracle logo and links to Products, Sectors, Resources, Clients, Partners, Developers, and Company. Below the navigation bar, there are tabs for Tools and resources, Java downloads (selected), and Java archive. A section titled "Looking for other Java downloads?" includes links to OpenJDK Early Access Builds and JRE for Consumers. The main content area features a heading "Java 24, Java 21, and earlier versions available now" and a button to "Learn about Java SE Subscription". It also includes a section for "Java SE Development Kit 24.0.2 downloads" with a table of download links for Linux, macOS, and Windows. A green arrow points to the "x64 Installer" row in the Windows section.

Linux	macOS	Windows
<b>Java SE Development Kit 24.0.2 downloads</b>		
JDK 24 binaries are free to use in production and free to redistribute, at no cost, under the <a href="#">Oracle No-Fee Terms and Conditions</a> (NFTC).		
JDK 24 will receive updates under these terms, until September 2025, when it will be superseded by JDK 25.		
Linux	macOS	Windows
Product/file description	File size	Download
x64 Compressed Archive	229.62 MB	<a href="https://download.oracle.com/java/24/latest/jdk-24_windows-x64_bin.zip">https://download.oracle.com/java/24/latest/jdk-24_windows-x64_bin.zip</a> (sha256)
x64 Installer	205.86 MB	<a href="https://download.oracle.com/java/24/latest/jdk-24_windows-x64_bin.exe">https://download.oracle.com/java/24/latest/jdk-24_windows-x64_bin.exe</a> (sha256)
x64 MSI Installer	204.61 MB	<a href="https://download.oracle.com/java/24/latest/jdk-24_windows-x64_bin.msi">https://download.oracle.com/java/24/latest/jdk-24_windows-x64_bin.msi</a> (sha256)

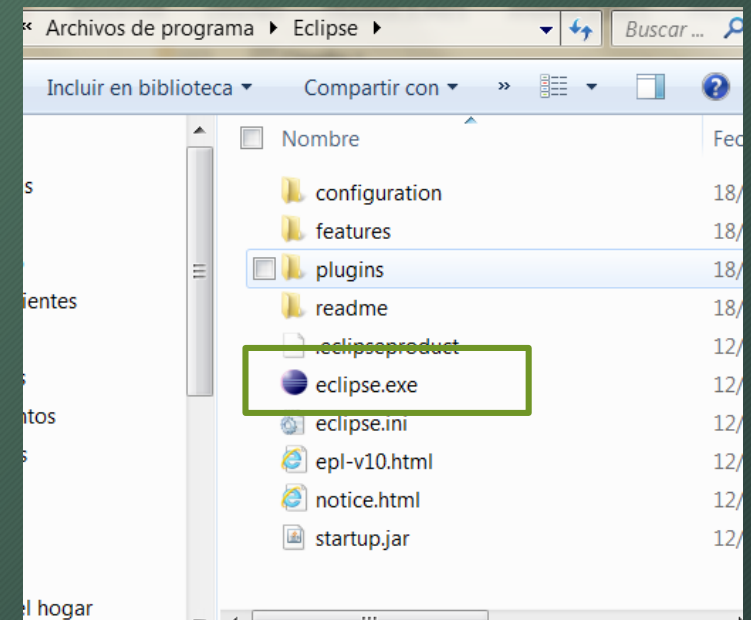
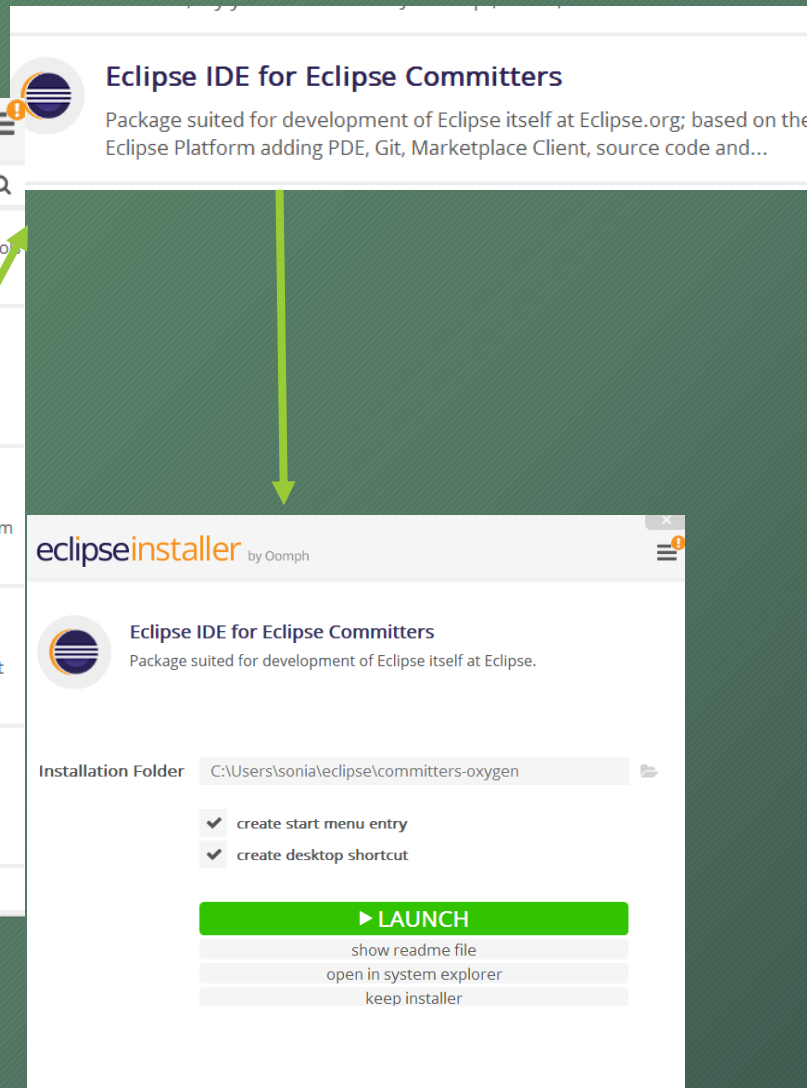
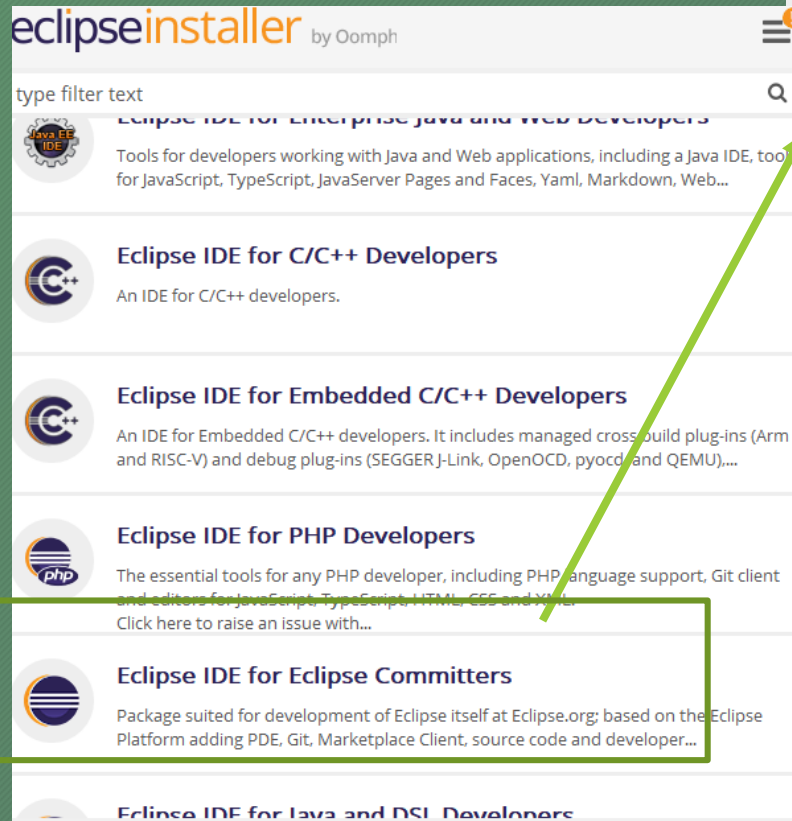
# Instal·lacions

Instal·lar Eclipse: <https://www.eclipse.org/downloads/>

Descomprimir o instal·lar en la unitat Principal.



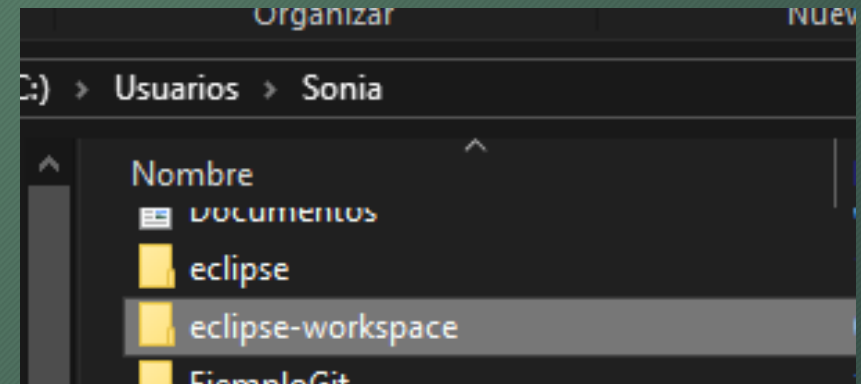
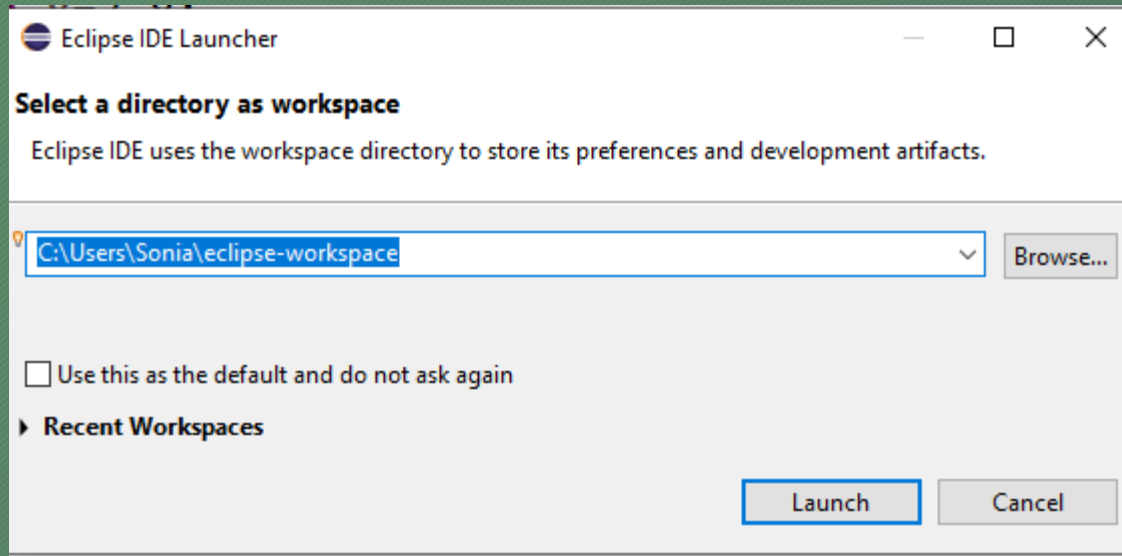
Com instal·lar eclipse: <https://jarroba.com/instalar-bien-eclipse-un-ide-de-muchos/>  
Des del propi instal·lador es pot escollir després qualsevol altre entorn.



Com instal·lar eclipse: <https://jarroba.com/instalar-bien-eclipse-un-ide-de-muchos/>  
Des del propi instal·lador es pot escollir després qualsevol altre entorn.



Es pot definir la ubicació de l'espai de treball (workspace)





# Eclipse IDE for Eclipse Committers

## Package Description

Package suited for development of Eclipse itself at Eclipse.org; based on the Eclipse Platform adding PDE, Git, Marketplace Client, source code and developer documentation.

Click [here](#) to raise an issue with the Eclipse Platform.

Click [here](#) to open a bug report with the Eclipse Git team provider.

This package includes:

- Git integration for Eclipse
  - Eclipse Java Development Tools
  - Maven Integration for Eclipse
  - Eclipse Plug-in Development Environment
- Detailed features list

<https://www.eclipse.org/downloads/packages/release/2023-03/r/eclipse-ide-eclipse-committers>

Millors IDE per programar :<https://www.diarlu.com/mejores-ide-programar-java/>

## Tipus programa Java

- Aplicacions de consola (Shell)
- Aplicacions de propòsit en general
- Miniaplicacions (programes que s'executen en el navegador)



## Nomenclatura :

<https://jcodepoint.com/java/convencion-de-nombres-en-java/>

<https://www.discoduroderoer.es/convencion-de-nombres-en-java/>

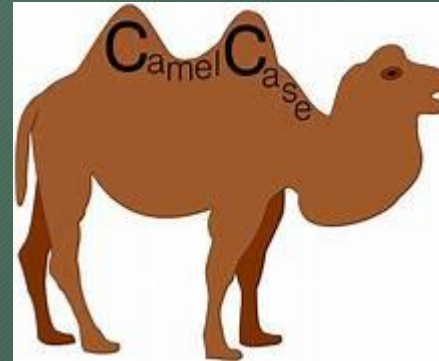
## Tipus de notació:

<https://www.neoguias.com/tipos-notacion-nombres/>

<https://www.aluracursos.com/blog/convenciones-de-nomenclatura-camel-pascal-kebab-snake-case>

Existeixen dos tipus de camelCase:

- **UpperCamelCase** (més conegut com PascalCase), quan la primera lletra de cadascuna de les paraules és majúscula. Exemple: ExempleDeUpperCamelCase.
- **lowerCamelCase** (o simplement camelCase), igual que l'anterior amb l'excepció que la primera lletra és minúscula. Exemple: exempleDeLowerCamelCase.





## En Java:

- **Paquets** : noms de paquets amb un nom de domini en minúscules, seguit de components addicionals separats per punts. **exemple.paquet.nom**
- **Classes**: substantius , noms de classes descriptius i significatius i seguir el format **PascalCase**. Per una paraula : Cotxe per més d'una: ExempleClass
- **Interfícies**: descriptius i utilitzar adjectius o substantius per a indicar el seu propòsit, es recomana seguir el format **PascalCase** on la primera lletra de cada paraula està en majúscula. ExempleInterficiNom
- **Variables**: **camelCase**, per exemple :userNom. No poden començar per números. Si és una paraula int numero, si són dos també es poden posar int numEntero o int num\_entero.
- **Mètodes**: utilitzant verbs o frases verbals que descriguin amb precisió l'acció que realitzen dins del codi, per exemple: calcularPreuTotal. **camelCase**.
- **Constants**: es declaren usant lletres majúscules i guions baixos. Final int MAX\_INTENTS=3
- **Nom projecte**: **camelCase**, per exemple: primerProjecte



En Java, cada programa comença amb una **classe principal**.

La declaració d'aquesta classe és essencial, és la classe que actua com el punt d'inici de l'execució d'un programa.

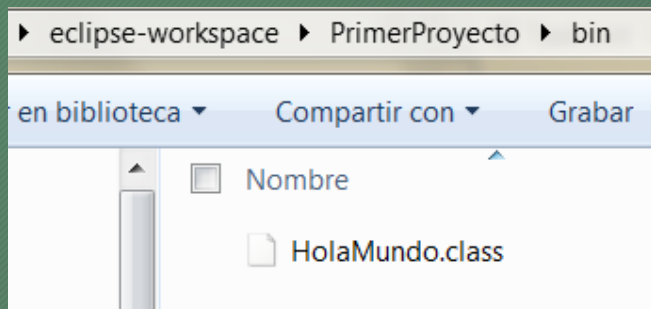
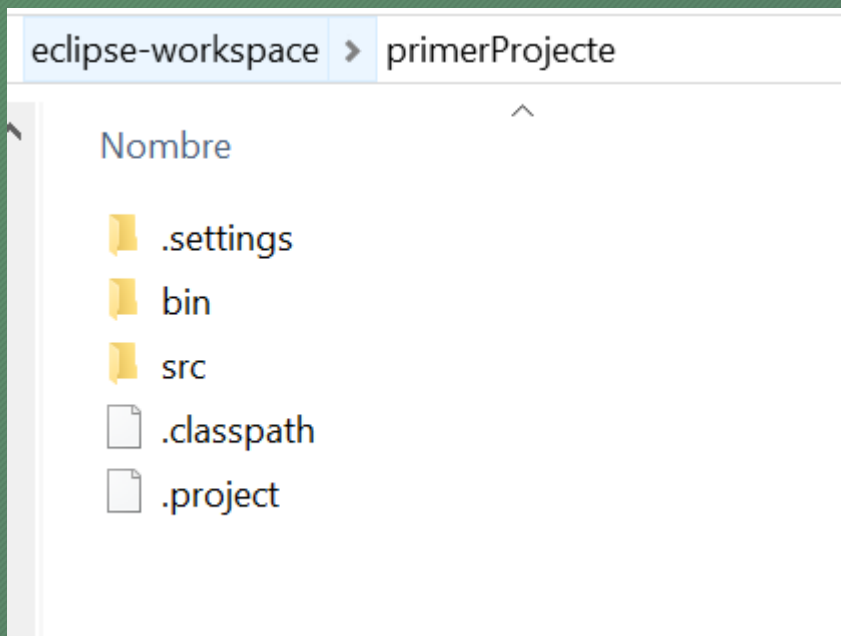
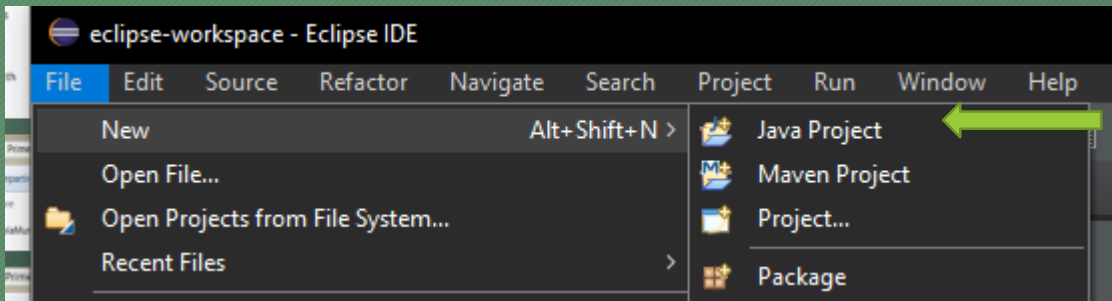
Aquesta classe ha de tenir el mètode main:

```
public static void main(String[] args) {  
    // Código del programa  
}
```

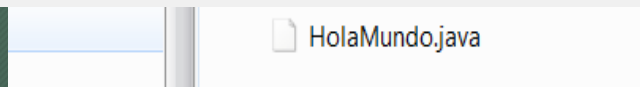
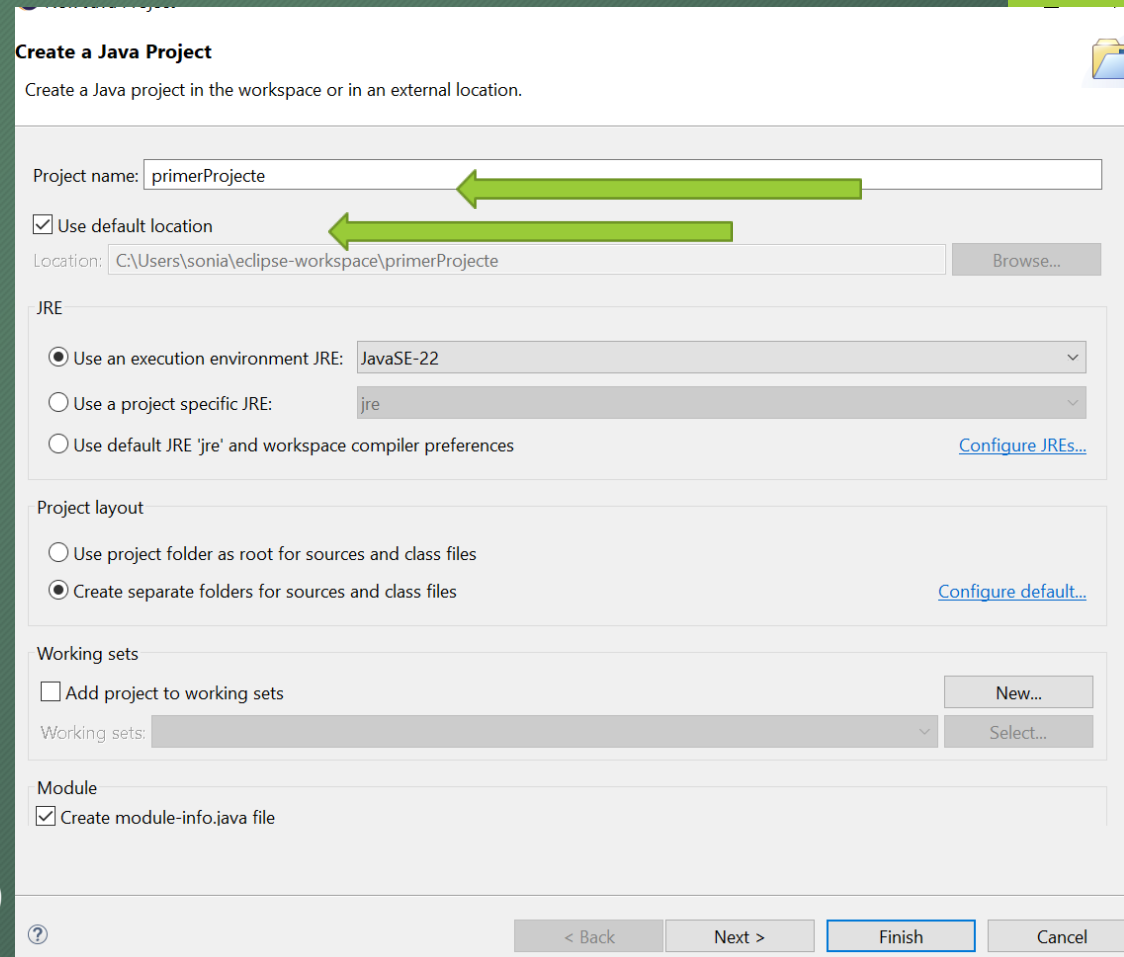
Mètode main, public i static.  
Visible i sense necessitat de crear una  
Instància.

```
*/  
public class HolaMundo {  
    public static void main(String[] args) {  
  
    }  
}
```

Classe principal, HolaMundo



Classes pròpies i  
predefinides (api)

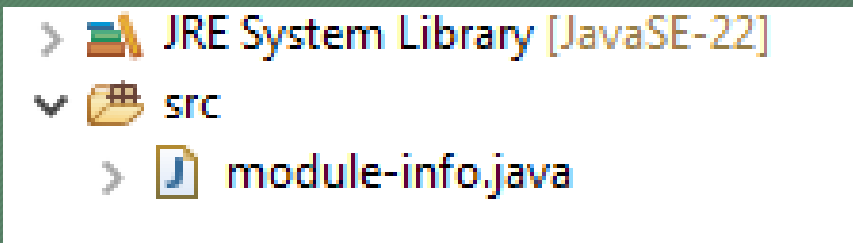
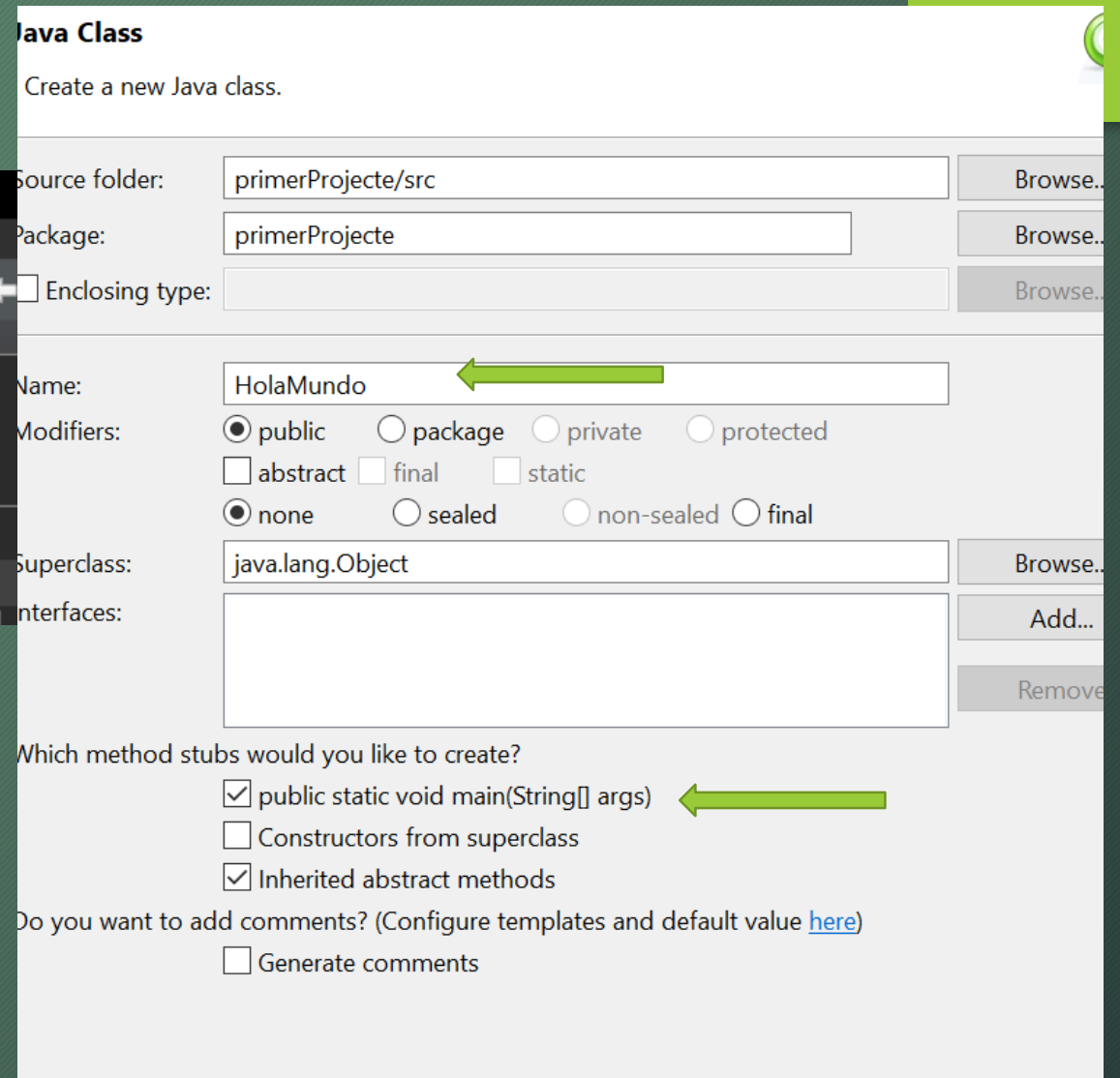
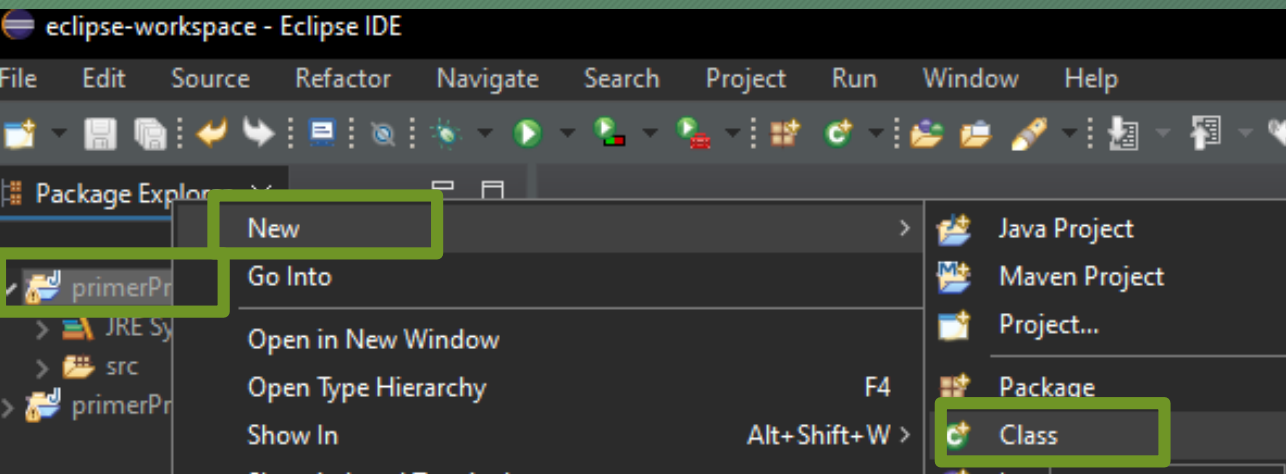


# Aplicacions de consola

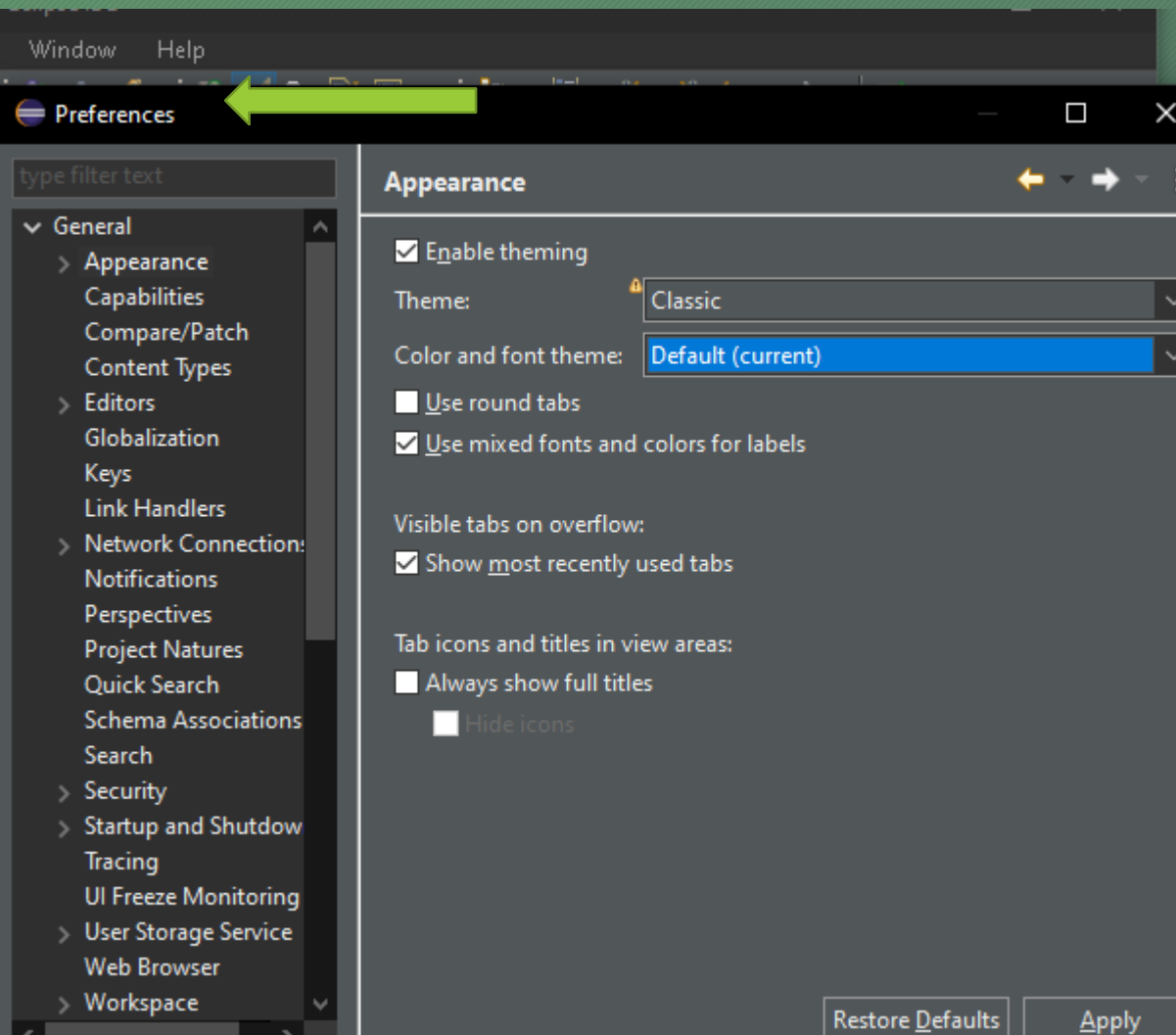
Tot projecte ha de tenir almenys una classe, anem a crear una classe principal.

Estructura mínima de una classe (main)

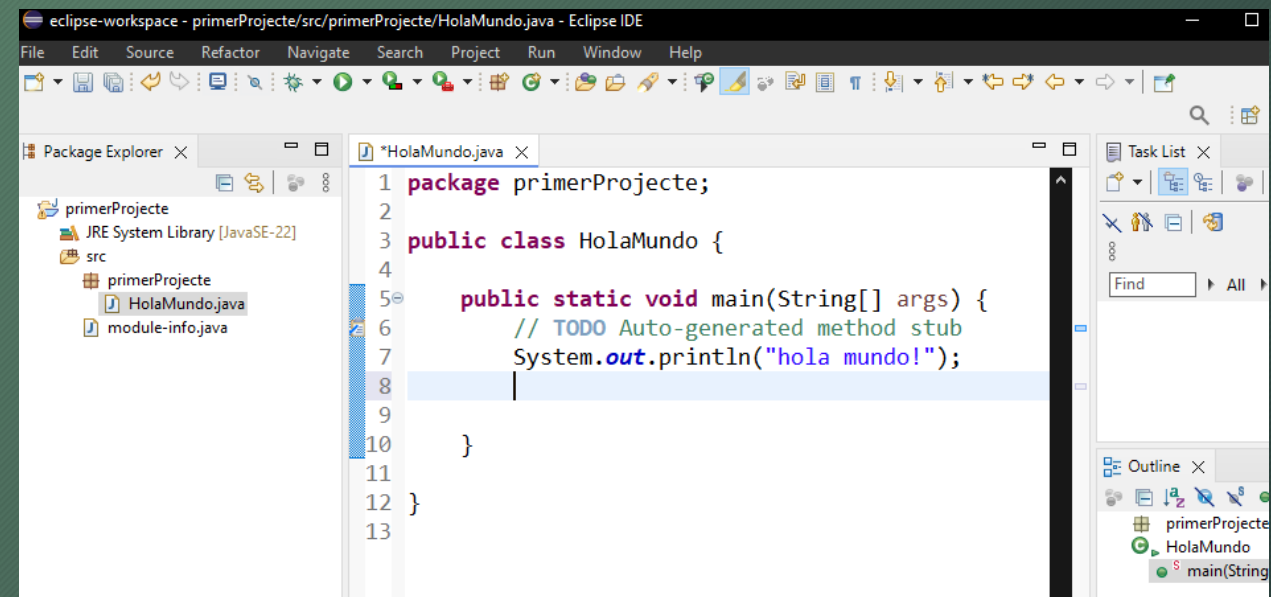
consola

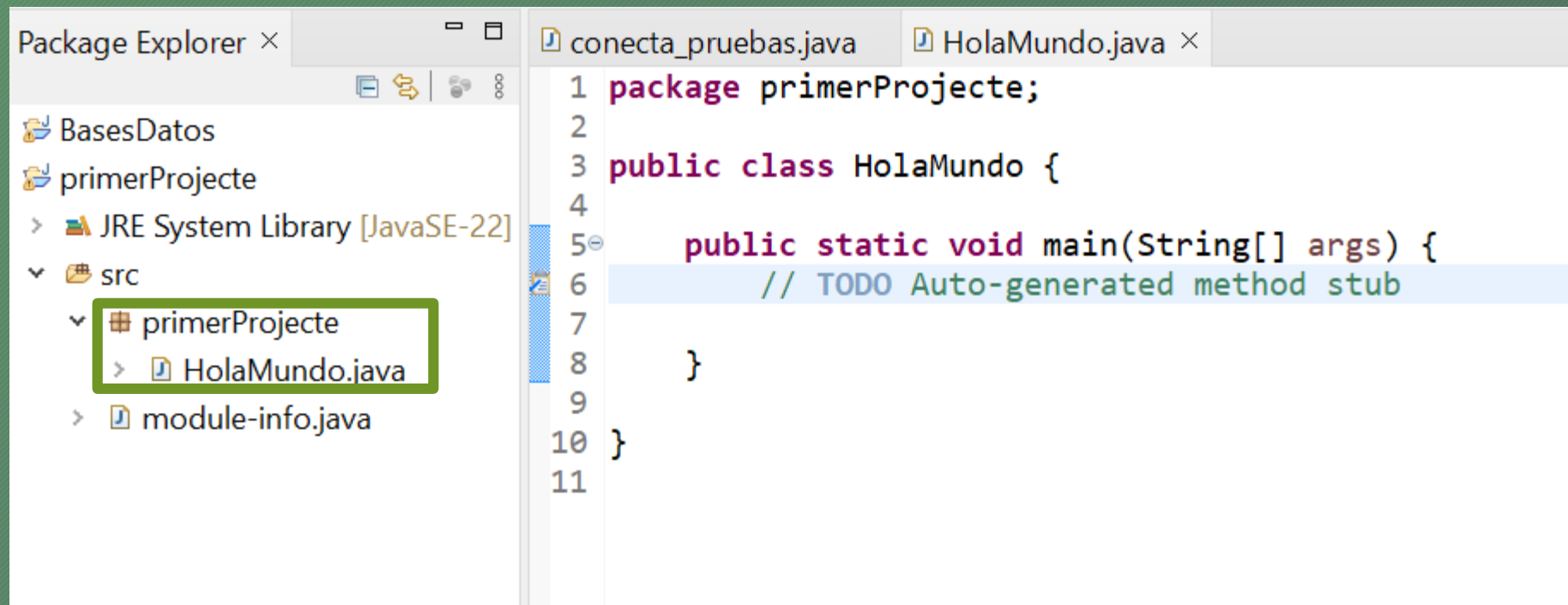




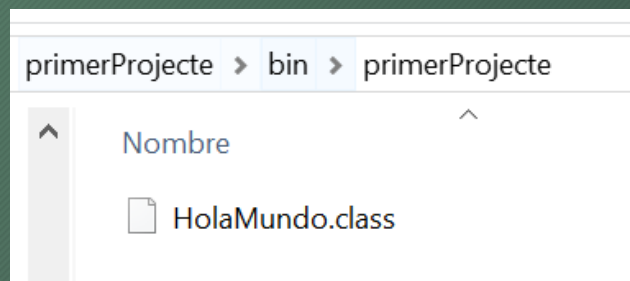
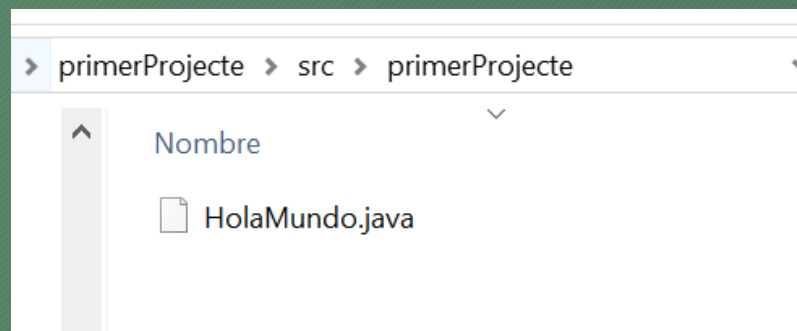


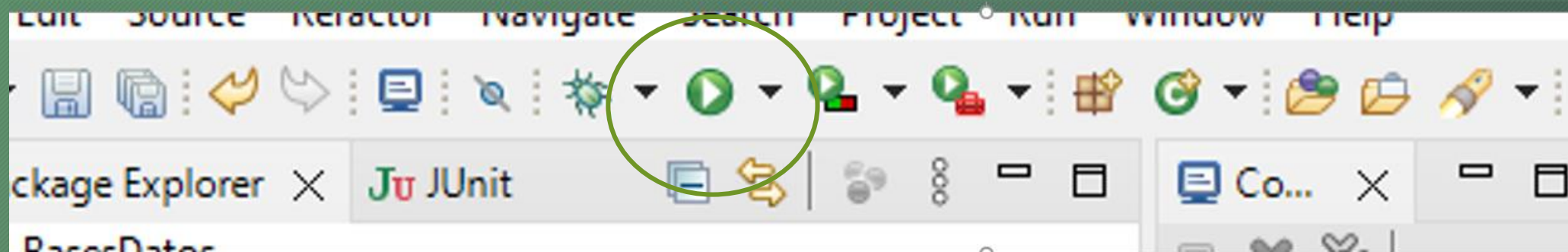
Windows/Preferences/General/Appearance/Theme



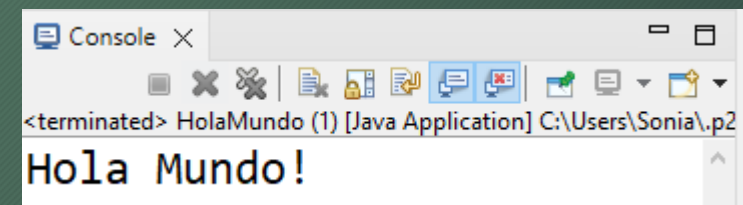


consola





```
package primerProyecto;  
  
public class HolaMundo {  
  
    public static void main(String[] args) {  
        // TODO Auto-generated method stub  
        System.out.println("Hola Mundo!");  
    }  
}
```





```
6  
7 int edad=40;
```

Warning

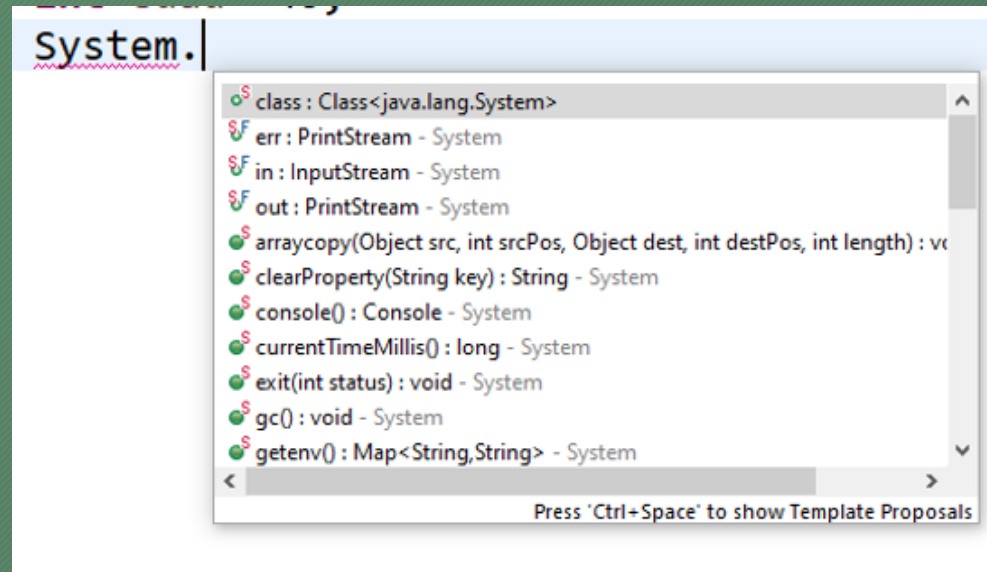
```
7 int edad=42;  
8 Sytem.out.println("Mi edad es: "+edad);  
9
```

Error

consola

```
1 package segundoProyecto;  
2  
3 public class HolaMundo {  
4  
5     public static void main(String[] args) {  
6         // TODO Auto-generated method stub  
7         int edad =40;  
8         Sytem.out.println("Mi edad es :"+edad);  
9  
10  
11     }  
12  
13 }  
14
```

## Intellisense



consola

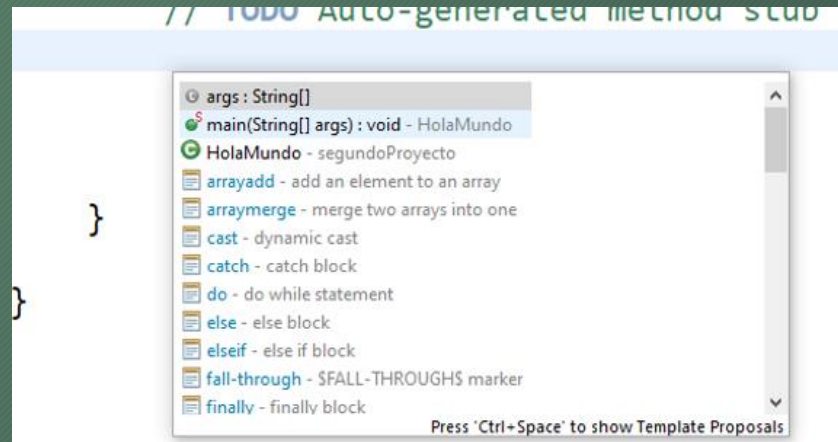
## Abreviacions

syso o sysout, seguit de Ctrl + Barra  
espai (space)

```
System.out.println("...");
```

Template proposals (Propuestas de  
plantilla)

Ctrl + Space



Paraules reservades i  
variables definides

# Java ( Sintaxi)



## Variables en Java

sintaxi

**Espai** (memòria de l'ordinador) on s'emmagatzemarà un valor que pot anar canviant durant l'execució del programa.

**Declarar una variable:** `int contador;`  
(reservem un espai en la RAM de l'ordinador,  
un `long` reserva un espai major que el `int` per exemple)

**Inicialitzar la variable:** `contador=0;` (s'emmagatzema el valor en aquest espai de la RAM) o tot en una sola línia: `int comptador=0;`

Java és molt estricta amb les variables, s'han de declarar i s'han d'inicialitzar.

`int, double, char, boolean, etc.`

# Emmagatzemar dades: Tipus primitius de dades

sintaxi

	NOMBRE	TIPO	OCUPA	RANGO APROXIMADO
<b>TIPOS PRIMITIVOS</b> (sin métodos; no son objetos; no necesitan una invocación para ser creados)	byte	Entero	1 byte	-128 a 127
	short	Entero	2 bytes	-32768 a 32767
	int	Entero	4 bytes	$2 \cdot 10^9$
	long	Entero	8 bytes	Muy grande
	float	Decimal simple	4 bytes	Muy grande
	double	Decimal doble	8 bytes	Muy grande
	char	Carácter simple	2 bytes	---
	boolean	Valor true o false	1 byte	---

# Emmagatzemar dades: Tipus primitius de dades

- Enters

NOMBRE	TIPO	OCUPA	RANGO APROXIMADO
byte	Entero	1 byte	-128 a 127
short	Entero	2 bytes	-32768 a 32767
int	Entero	4 bytes	$2 \cdot 10^9$
long	Entero	8 bytes	Muy grande

sintaxi

Sufix L

Espai reservat en el moment de declarar les variables



Exemples:    `int` des de -2,147,483,648    fins    2,147,486,647

`Long 500000L;`

long	8	-9,223,372,036,854,775,808 a 9,223,372,036,854,775,807
int	4	-2,147,483,648 a 2,147,483,647
short	2	-32,768 to 32,767
byte	1	-128 to 127



# Emmagatzemar dades: Tipus primitius de dades

sintaxi

- Coma Flotant (decimals)

Tipos	Tamaño en bytes	Rango
float	4	3.4E-38..3.4E+38 y -3.4E-38..-3.4E+38
double	8	1.7E-308..3.4E+308 y -1.7E-308..-1.7E+308

De 6 a 7 xifres decimals significatives.  
Sufix f o F.

15 xifres  
decimals significatives

A Java, per defecte, els decimals són de tipus double pel que s'ha d'agregar la lletra f al final de la declaració literal per a indicar que és un número float.

# Emmagatzemar dades: Tipus primitius de dades

sintaxi

- **Char**

Representar caràcters.  
Entre cometes simples. 'a'

- **Booleans (algebra de Boole)**

True o false

char	Caràcter simple	2 bytes
boolean	Valor true o false	1 byte

El tipus de dada **String** és una classe que representa cadenes de caràcters i s'utilitza àmpliament en aplicacions per a emmagatzemar i manipular text. A Java, una cadena de text es defineix utilitzant cometes dobles (" ").

```
/* Ejemplo - aprenderaprogramar.com */  
Precio = 42; // Entero tipo int. Un número sin punto decimal se interpreta normalmente como int.  
importe_acumulado = 210; // Entero tipo int  
profesor = "Ernesto Juárez Pérez"; // Tipo String  
aula = "A-44"; // Tipo String  
capacidad = 1500; // Entero tipo int  
funciona = true; // Tipo boolean  
esVisible = false; // Tipo boolean  
diametro = 34.25f; // Tipo float. Una f o F final indica que es float.  
peso = 88.77; // Tipo double. Un número con punto decimal se interpreta normalmente como double.  
edad = 19; // Entero tipo short  
masa = 178823411L; // Entero tipo long. Una l o L final indica que es long.  
letra1 = 'h'; // Tipo char (carácter). Se escribe entre comillas simples.
```



# Operadors

sintaxi

- Aritmètics
- D'assignació
- Incrementals
- Lògics
- Relacionals
- Condicional (ternari)

# Operadors Aritmètics

sintaxi

- + Suma. Els operands poden ser sencers o reals
- - Resta. Els operands poden ser sencers o reals
- \* Multiplicació. Els operands poden ser sencers o reals
- / Divisió. Els operands poden ser sencers o reals. Si tots dos són sencers el resultat és sencer. En qualsevol altre cas el resultat és real.
- % Resta de la divisió. Els operands poden ser de tipus sencer o real

## Nom de classe: OperadorsAritmetics

### Exercici

1. Definició de dues variables senceres amb valor 10 i 3 respectivament
2. Definició de dues variables de tipus double amb valors 12.5 i 2.0 respectivament.
3. Definició de dues variables tipus char amb valors P i T respectivament.
4. Comprovar la sortida que obtenim de sumar les diferents combinacions de variables.



```
HolaMundo.java  Variables.java  *OperadoresAritmeticos.java ✕
1
2
3 public class OperadoresAritmeticos {
4     public static void main(String[] args) {
5
6         int a = 10, b = 3;
7         double v1 = 12.5, v2 = 2.0;
8         char c1='P', c2='T';
9
10        System.out.print(a+b);
11        //probar diferentes posibilidades con las variables
12    }
13
14 }
15
```

```
public static void main(String[] args) {
    // TODO Auto-generated method stub
    int a=10,b=3;
    float v2=12.5f;
    int z=a+v2;
    System.out.println("hola mundo!");
}
```

Type mismatch: cannot convert from float to int

2 quick fixes available:

- Add cast to 'int'
- Change type of 'z' to 'float'

Press 'F2' for focus

Sol

```
public static void main(String[] args) {
    // TODO Auto-generated method stub
    int a=10,b=3;
    float v2=12.5f;
    float z=a+v2;
    System.out.println("hola mundo!");
}
```

En les operacions en les quals apareixen operands de distint tipus, java converteix els valors al tipus de dada de major precisió de totes les dades que intervenen.

cannot convert from float to int  
( però un int sí es pot convertir a un float)

La conversió de tipus és quan s'assigna un valor d'un tipus de dades primitiu a un altre tipus.

- **Widening Casting** (automatically) - converting a smaller type to a larger type size

`byte` -> `short` -> `char` -> `int` -> `long` -> `float` -> `double`

- **Narrowing Casting** (manually) - converting a larger type to a smaller size type

`double` -> `float` -> `long` -> `int` -> `char` -> `short` -> `byte`

pot ser **implícit** (automàtic i sense pèrdua d'informació, com de `int` a `double`) o **explícit** (manual, posant el tipus entre parèntesi (tipus)) per a convertir de major a menor grandària, com de `double` a `int`, la qual cosa pot implicar pèrdua de precisió.

Conversions que són segures per no suposar pèrdua d'informació.

TIPO ORIGEN	TIPO DESTINO
byte	double, float, long, int, char, short
short	double, float, long, int
char	double, float, long, int
int	double, float, long
long	double, float
float	Double



### Nom de la classe: Operacions

1. Donades les variables

```
int a = 10, b = 3;
```

2. Realitzar l'operació necessària per a obtenir la sortida :  
La suma dels números és: 13

3. Veure què passa si posem la següent instrucció:

```
System.out.print("la suma de "+a+" i "+b+" és:"+suma);
```

```
public class Operaciones {  
  
    public static void main(String[] args) {  
        // TODO Auto-generated method stub  
        int a = 10, b = 3;  
        int suma=a+b;  
        System.out.print("la suma de "+a+" y "+b+" es:"+suma);  
    }  
}
```



Concatenació

En la mateixa classe anterior:

- Fer la resta
- La multiplicació
- La divisió

Exercici

Observació:

Veure què succeeix si es canvia de int a float en la definició de variables però no en les variables suma, resta, multi i divi.

Com podríem obtenir la sortida :

La suma és:13.0 la resta és: 7.0, la divi és: 3.3333333 i la multi és: 30.0



```
2 public class Operaciones {  
3  
4     public static void main(String[] args) {  
5         // TODO Auto-generated method stub  
6         float a=10f ,b=3f;  
7         float suma=a+b;  
8         float resta=a-b;  
9         float multi=a*b;  
0         float divi=a/b;  
1         System.out.print("La suma de es:"+suma+  
2             ",la resta es: "+resta+",la divi es: "+divi+" y la multi es: "+multi);  
3  
4     }  
5  
6 }  
7
```

# Constants en Java

Sintaxi

Valor que no varia

Insertar constants: final double pulg=2.55;

```
1
2 public class Apies {
3     public static void main(String[] args) {
4         // TODO Auto-generated method stub
5         final double a=0.0328084;
6         a=5;
7     }
```

Classe : Apies.

Posar codi necessari per a realitzar el pas de cm a peus.

Tenint en compte la relació  $1 \text{ cm} = 0,0328084 \text{ peus}$

Sortida desitjada:

La mida de 10.0 cm és: 0.328084000000000004 peus



Sol

```
1
2 public class Apies {
3     public static void main(String[] args) {
4         // TODO Auto-generated method stub
5         final double apies=0.0328084;
6         double medidaEnCm=10;
7         double result=medidaEnCm*apias;
8         System.out.print("La medida de "+medidaEnCm+" cm es: "+result+" pies");
9
10    }
11 }
12
```

PrimerProyecto

- JRE System Library [JavaSE-1.8]
- src
  - (default package)
    - Apies.java
    - HolaMundo.java
    - Operaciones.java
    - OperadoresAritmeticos.java
    - Variables.java

- **Narrowing Casting** (manually) - converting a larger type to a smaller size type

`double` -> `float` -> `long` -> `int` -> `char` -> `short` -> `byte`

Col·locant el tipus de dada desitjada entre parèntesi () abans de la variable o valor.

```
double miDecimal = 9.78;  
int miEntero = (int) miDecimal; // El 0.78 es perd  
System.out.println(miEntero); // Sortida: 9
```

# Conversions de tipus amb mètodes 'parse'

El mètode parse() analitza una cadena de text per a extreure informació i convertir-la en un objecte de la classe que anomena, com per exemple per convertir un String a un tipus numèric .

S'utilitza en especial per a treballar amb dades d'entrada que sovint venen com a text, com a números ingressats per un usuari en una interfície, i transformar-los en el format intern de Java que el programa pot processar i manipular.

```
String cadena="100";
int numEntero = Integer.parseInt(cadena);
System.out.println(numEntero); // Sortida: 100 número
double doble = Double.parseDouble("900");
System.out.println(doble); // Sortida: 900.00 número
float importe = Float.parseFloat("900.5");
System.out.println(importe); // Sortida: 900.5 número
boolean bol = Boolean.parseBoolean("false");
System.out.println(bol); // Sortida: false booleano
```



# Operadors d'Assignació:

Sintaxi

Assignació simple '='

Assignació composta

```
num1 += num2;    // igual que num1 = num1 + num2;  
num1 -= num2;    // igual que: num1 = num1 - num2  
num1 *= num2;    // igual que: num1 = num1 * num2  
num1 /= num2;    // igual que: num1 = num1 / num2  
num1 %= num2;    // igual que: num1 = num1 % num2
```

```
num1=4;  
num2=3;
```

```
num1 += num2;  
num1 -= num2;  
num1 *= num2;  
num1 /= num2;  
num1 %= num2;
```

# Operadors d' Increment

sintaxi

Operadors que permeten incrementar o decrementar les variables en una unitat. Es poden usar davant i darrere de la variable . El seu comportament varia si està en un conjunt d'operacions.

‘ ++’      ‘ --’

```
int a=6;
```

```
a++;
```



```
int num1 = 4; int num2 = 3;  
int davant = ++num1 * num2;
```

Primer increment, ++num1, num1 val 5 i després es calcula num1\*num2:  $5*3 = 15$

-----

```
int num1 = 4; int num2 = 3;  
int darrera = num1++ * num2;
```

Primer es calcula el valor de darrera: num1\*num2:  $4*3 = 12$ ; i després es calcula l'increment, num1++, num1 és igual a 5;

# Exercicis Operadors: OperadorsIncrementals.doc

Comprovar el codi anterior (davant,darrera) en una classe de java anomenada OperadorsIncrementals

Afegint també, per exemple, l'apartat 1 del Exercici anterior



```
public class OperadoresIncrementales {  
  
    public static void main(String[] args) {  
        // TODO Auto-generated method stub  
        int num1 = 4;  
        int num2 = 3;  
        int delante = ++num1 * num2;  
  
        System.out.println(delante);  
        num1 = 4;  
        num2 = 3;  
        int darrera = num1++ * num2;  
  
        System.out.println(darrera);  
        int x; int y;  
        x = 5;  
        y = ++x;  
        y = x++;  
        System.out.println("el valor de y es: "+y+" el valor de x es: "+x);  
        x = 8;  
        y = x--;  
        y = --x;  
        System.out.println("el valor de y es: "+y+" el valor de x es: "+x);  
    }  
}
```

# Operadors Relacionals

Permeten comparar variables segons relació d'igualtat/desigualtat o relació major/menor. El resultat de la comparació és un valor booleà.

'>': Major que

'<': Menor que

'==': Iguals (valor, forçant el tipus)

'!=': Distints

'>=': Major o igual que

'<=': Menor o igual que

'===': Del mateix tipus i amb el mateix valor

# Operadors Lògics

&&	Operador and (y)
	Operador or (o)
!	Operador not (no)

Suma lògica (&&)

true; si les dues ho són

Producte lògic(||)

true; si una de les dues opcions ho són

Negació (!)

retorna el contrari



# Operadors Lògics

A	B	A OR B
F	F	F
F	V	V
V	F	V
V	V	V

A	B	A AND B
F	F	F
F	V	F
V	F	F
V	V	V

A	NOT A
F	V
V	F

F: Falso

V: Verdadero

## Condicional (ternari)

expresió1 ? expresió2 : expresió3

Retorna l'un o l'altre valor en funció de la condició

Z=(15>=2)? 2:3

.....

```
int i = 10, j;
```

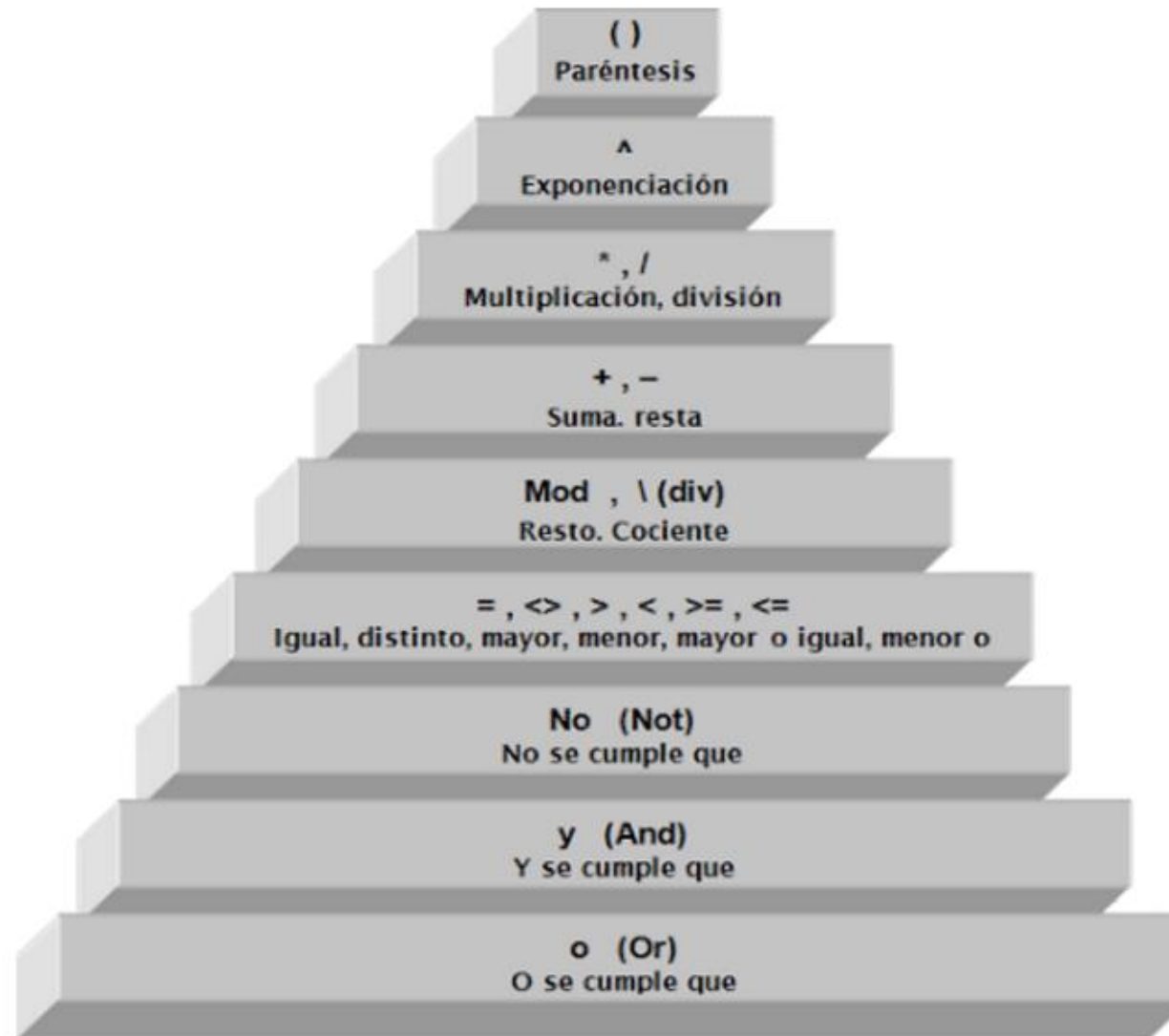
```
j = (i < 0) ? 0 : 100;
```

# Exercicis Operadors:

## OperadorsLogics\_Condicionals.doc



# Ordre de prioritat



# Format de sortida (printf)

```
1
2 public class FormatoSalidaValores {
3
4     public static void main(String[] args) {
5         // TODO Auto-generated method stub
6         double y=1520.50;
7         double result=y/3;
8         System.out.println(result);
9         System.out.printf("%1.2f", result);
10    }
11
12 }
13
```

Problems @ Javadoc Declaration Search Console

<terminated> FormatoSalidaValores [Java Application] C:\Progra

506.8333333333333

506,83

Sortida de dades amb format  
...printf()  
1.2 dos decimals  
1.5 cinc decimals

```
public static void main(String[] args) {
    // TODO Auto-generated method stub
    float a=12.23456765f;
    int numero = 42;
    System.out.printf("El número es %d\n", numero);
    System.out.println(a);
    System.out.printf("%.2f\n", a); // %n salt de línia
    System.out.printf("%1.2f", a);
}
```

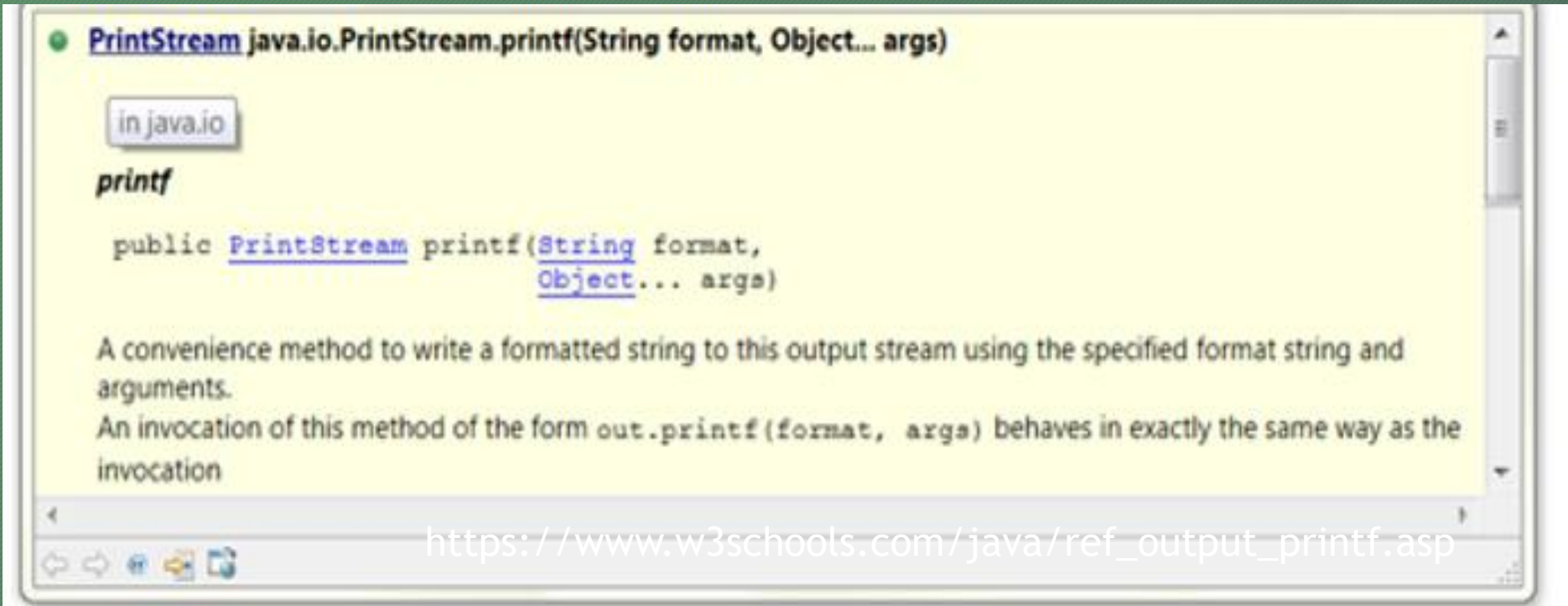
<terminated> Apies [Java Application] C:\

El número es 42

12.234568

12,23

12,23|



La sintaxis básica de printf en Java es:

```
System.out.printf(formato, argumento1, argumento2, ...);
```

- **formato:** Es una cadena de formato que incluye texto literal y especificadores de formato.
- **argumentos:** Son los valores que se insertarán en los lugares indicados por los especificadores de formato.

[https://www.w3schools.com/java/ref\\_output\\_printf.asp](https://www.w3schools.com/java/ref_output_printf.asp)



# Especificadors de Format Comuns

- 1) %d: Per **sencers** (int).
- 2) %f: Per números **decimals** (float, double).
- 3) %s: Per **cadena de text** (String).
- 4) %c: Per **caràcters** individuals.
- 5) %b: Per valors **booleans**.

```
// TODO Auto-generated method stub
int numero = 42;
System.out.printf("El número es %d\n", numero);
// Imprime: El número es 42

boolean b = true;
System.out.printf("El valor es %b\n", b);
// Imprime: El valor es true

double d = 3.14159;
System.out.printf("El valor de PI es %.2f\n", d);
// Imprime: El valor de PI es 3.14

String nombre = "Juan";
System.out.printf("Hola, %s\n", nombre);
// Imprime: Hola, Juan

char inicial = 'J';
System.out.printf("La inicial es %c\n", inicial);
// Imprime: La inicial es J
```

# Paraules reservades

Les paraules reservades a Java, o \*keywords, són termes amb un significat predefinit que no poden usar-se com a identificadors (noms de variables, classes, mètodes, etc.) en el teu codi. Aquestes paraules són la base de la sintaxi del llenguatge i són utilitzades pel compilador per a interpretar l'estructura del programa. Java té un conjunt de paraules reservades, algunes de les quals inclouen abstract, class, if, for, while, i void, entre altres.

Llistat paraules reservades :

[https://codigofacilito.com/articulos/palabras\\_reservadas\\_java](https://codigofacilito.com/articulos/palabras_reservadas_java)

# Comentaris

```
// Comentari una línia  
  
/*  
 * comentari més d'una línia  
 *  
 *  
 *  
 */
```

Ctrl + / : comentar i descomentar



# Caràcters especials

A Java, els caràcters especials poden referir-se tant als caràcters literals com a les **seqüències de sortida** (com a `\n` per a nova línia o `\t` per a tabulador i la `\` per no compilar el caràcter següent i escriure'l literal).

Per a treballar amb ells, es pot usar el tipus de dada `char` o la classe `String`, per exemple.

```
¡Hola, "Mundo"!
¡Hola,
Mundo!
¡Hola,  Mundo!
```

```
2
3 public class HolaMundo {
4
5     public static void main(String[] args) {
6         // TODO Auto-generated method stub
7         String saludo1 = "¡Hola, \"Mundo\"!";
8         System.out.println(saludo1);
9         String saludo2 = "¡Hola, \nMundo!";
10        System.out.println(saludo2);
11        String saludo3 = "¡Hola, \tMundo!";
12        System.out.println(saludo3);
13
14    }
```

A part dels caràcters especials de sortida tenim també :

- Caràcters Literals

Símbols que no són lletres ni dígit, com @, #, !, o l'ús de caràcters no llatins com ñ o emojis.

- Metacaracteres en Expressions Regulars

Símbols amb significat especial en les expressions regulars, com ., \*, +, [ ], etc..