



INITIATION A JAVA

Installation

- 1) Installation de la dernière version du JDK (JDK 22) en vous rendant sur la page

<https://www.oracle.com/fr/java/technologies/downloads/#jdk22-windows> puis

x64 Installer	164.35 MB	https://download.oracle.com/java/22/latest/jdk-22_windows-x64_bin.exe (sha256)
---------------	-----------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------

2) Installer Eclipse

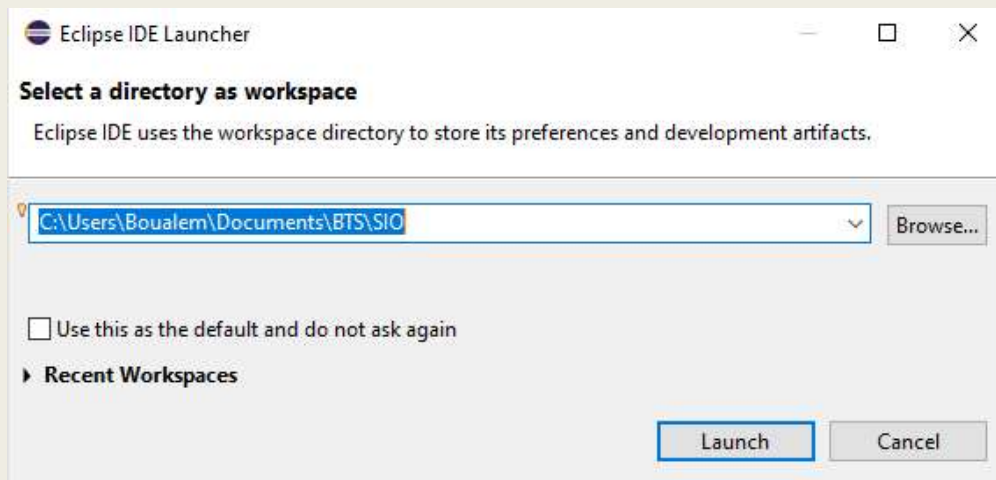
- Rendez-vous sur le site www.eclipse.org
- Choisir Downloads



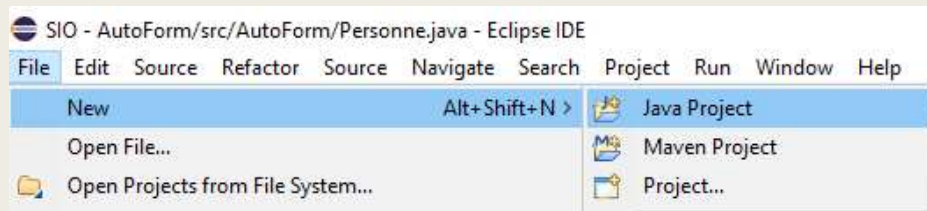
- Pour installer Eclipse, choisir l'option **Eclipse IDE JAVA for Developers**

3) Votre premier programme JAVA

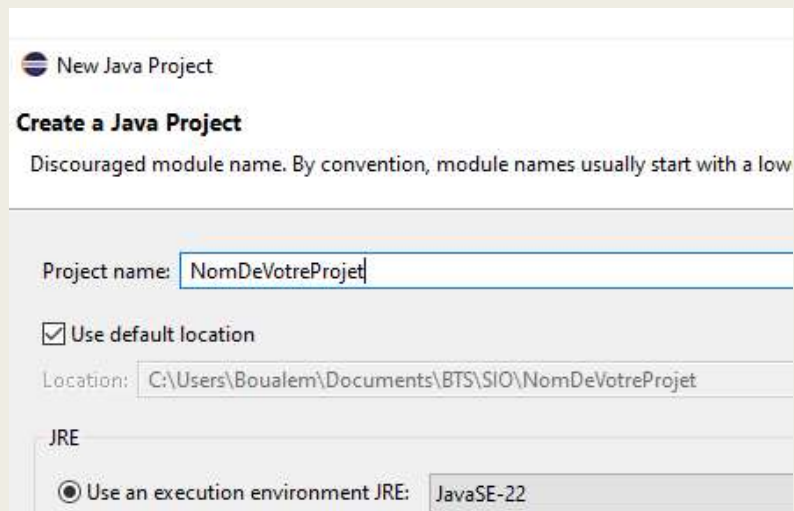
- Ouvrir Eclipse
- Choisir l'espace de travail (workspace)



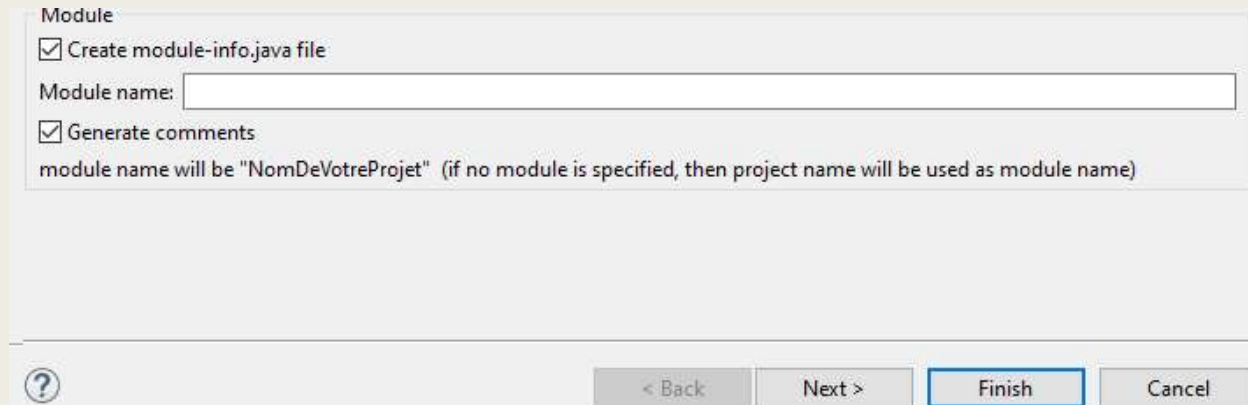
- Créer un nouveau projet :



- Renseigner le nom de votre projet :



- En bas de la fenêtre, Cliquer sur **Finish** pour finir :



Module

☒ Create module-info.java file

Module name:

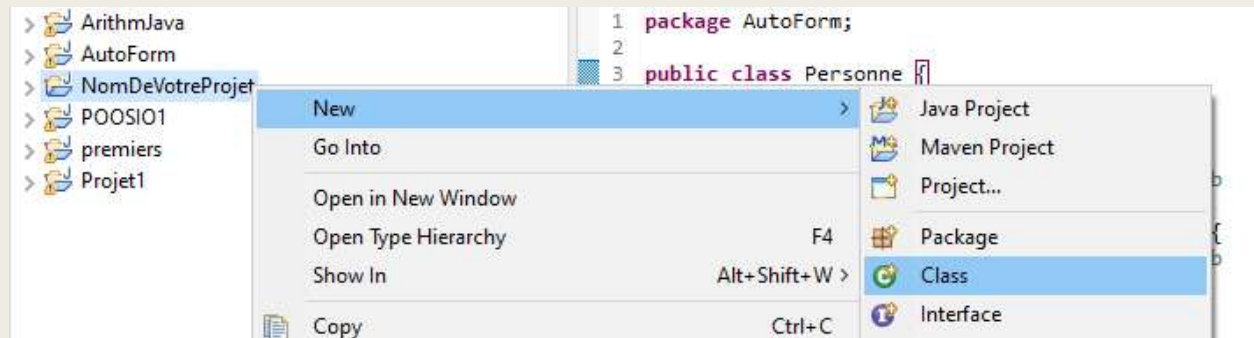
☒ Generate comments

module name will be "NomDeVotreProjet" (if no module is specified, then project name will be used as module name)

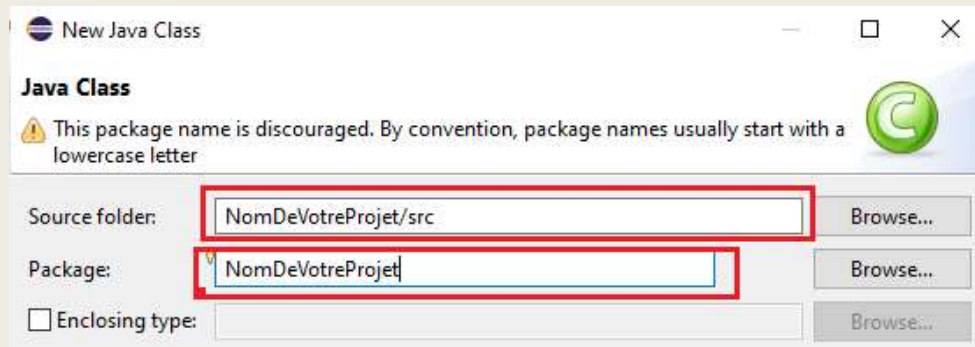
? < Back Next > Finish Cancel

- On crée ensuite une classe Main (seule une classe Main peut être compilée !)
 - Se positionner sur le nom du projet créé puis faire un clic droit ⇒ New ⇒ Class

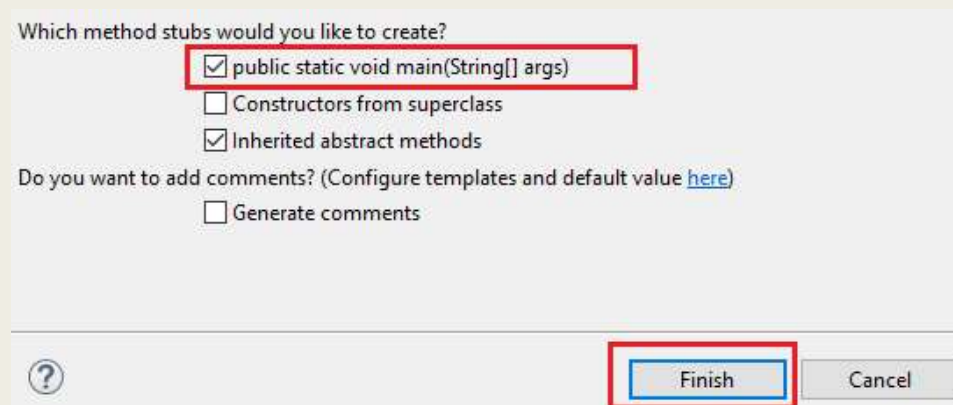
On obtient :



- Renseigner un nom de classe
- Cocher "**public static void main**"
- Cliquer sur finish



- Puis en bas de la fenêtre :



■ Syntaxe du langage Java

1) Les commentaires

Un **commentaire** est un texte ajouté au code source d'un **programme** servant à décrire le code source, et à faciliter sa compréhension. En JAVA,

- La séquence `//` permet d'insérer un commentaire sur une seule ligne. Le commentaire se termine donc obligatoirement à la fin de la ligne
- Pour insérer un commentaire sur plusieurs lignes, on l'intercale entre les séquences `/*` et `*/`. Ainsi, on écrira : `/* votre commentaire`

`.....*/`

-

2) Instructions, blocs et blancs

- ❑ Les instructions Java se terminent par un ; (point virgule)
- ❑ Les blocs sont délimités par :

{ pour le début de bloc

} pour la fin du bloc

Un bloc permet de définir un regroupement d'instructions. La définition d'une classe ou d'une méthode se fait dans un bloc.

- ❑ Les espaces, tabulations, sauts de ligne sont autorisés. Cela permet de présenter un code plus lisible.

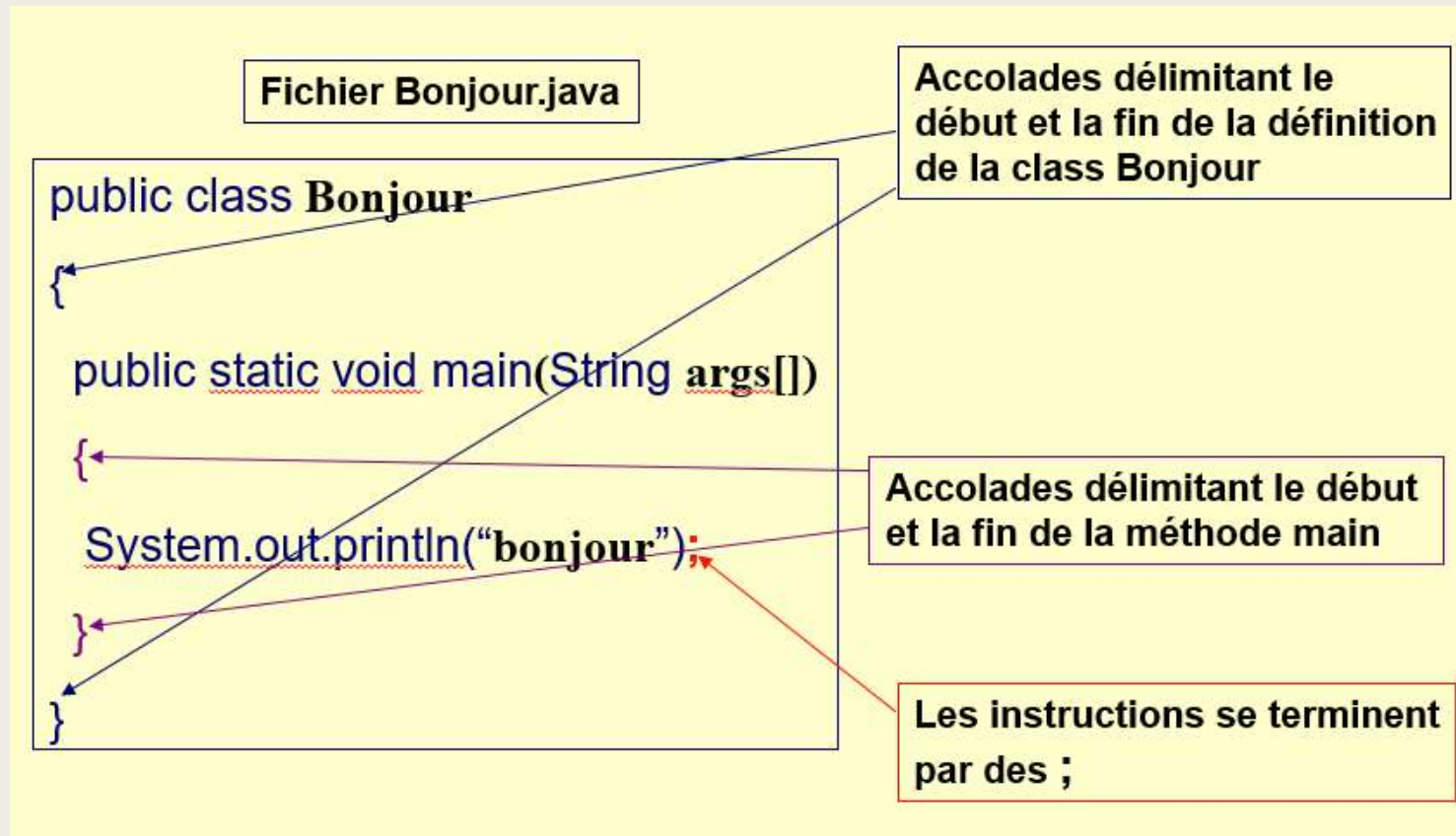
3) Point d'entrée d'un programme Java

- Pour pouvoir faire un programme exécutable il faut toujours une classe qui contienne une méthode particulière, la méthode « main »
 - *c'est le point d'entrée dans le programme : le microprocesseur sait qu'il va commencer à exécuter les instructions à partir de cet endroit*

```
2
3 public class Bonjour
4 { // Accolade débutant la classe Bonjour
5
6     public static void main(String[] args)
7
8     { /*Accolade débutant la méthode main. Notre programme
9       ne contient pour l'instant qu'une seule instruction */
10
11         System.out.println("Bonjour");
12
13     } // Accolade fermant la méthode main
14
15 } // Accolade fermant la classe Bonjour
16 }
```

La classe est l'unité de base de nos programmes. On la définit à l'aide du mot clé **class**

Exemple (1)



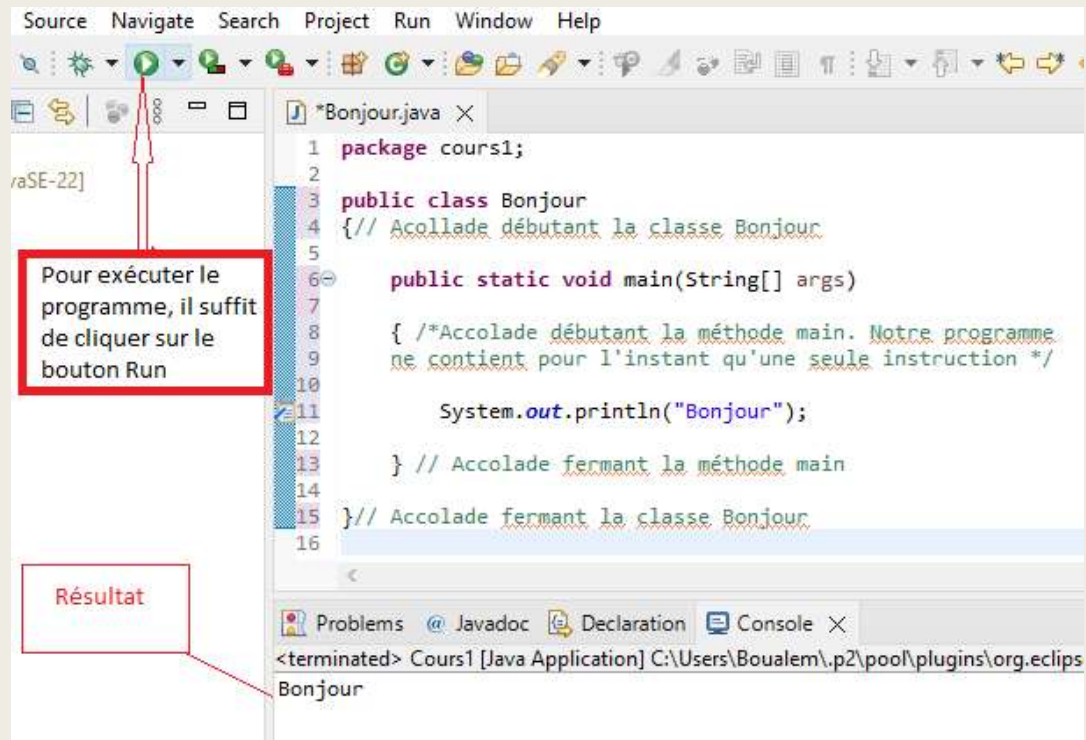
(2)

Fichier Bonjour.java

```
public class Bonjour
{
    public static void main(String args[])
    {
        System.out.println("bonjour");
    }
}
```

Une méthode peut recevoir des paramètres. Ici la méthode main reçoit le paramètre args qui est un tableau de chaîne de caractères.

4) Compilation et exécution



5) Identificateurs et conventions de nommage

- ❑ On a besoin de nommer les classes, les variables, les constantes, etc. ; on parle d'identificateur.
- ❑ Les identificateurs commencent par une lettre, _ ou \$

Attention : Java distingue les majuscules des minuscules

- ❑ Conventions sur les identificateurs :
 - *Si plusieurs mots sont accolés, mettre une majuscule à chacun des mots sauf le premier.*
 - ❑ exemple : uneVariableEntiere
 - *Le nom d'une classe ou d'une interface commence toujours par une **Majuscule**.*
 - ❑ exemples : MaClasse, UneJolieFenetre

- Conventions sur les identificateurs :

- *La première lettre est minuscule pour les méthodes, les attributs et les variables*

- exemples : setLongueur, i, uneFenetre

- *Les constantes sont entièrement en majuscules*

- exemple : LONGUEUR_MAX

Mots réservés

- Les mots réservés du langage sont nécessaires à la sémantique. Ce sont des :
- spécificateurs de type : int, float, char, ..., ...
- opérateurs symboliques : if, else, while, switch, ...
-
- **Vous ne pouvez pas** utiliser ces mots comme identificateurs de variables, de constantes ou de méthodes. En voici la liste :

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

6) Les types en JAVA

- Il existe 3 catégories de types en JAVA :

- ☐ Les types de base dits primitifs :

- un type **booléen** pour représenter les variables ne pouvant prendre que 2 valeurs (vrai et faux, 0 ou 1, etc.) : **boolean** avec les valeurs associées **true** et **false**
 - un type pour représenter les caractères : **char**
 - quatre types pour représenter les entiers de tailles diverses : **byte**, **short**, **int** et **long**
 - deux types pour représenter les réels : **float** et **double**

- ☐ Les types Objet

- ☐ Le type tableau

A) Les Types entiers

Type	Taille (en bits)	Valeur minimale	Valeur maximale
byte	8	$-2^7 = -128$	$2^7 - 1 = 127$
short	16	$-2^{15} = -32768$	$2^{15} - 1 = 32767$
int	32	$-2^{31} = -2147483648$	$2^{31} - 1 = 2147483647$
long	64	$-2^{63} = -9223372036854775808$	$2^{63} - 1 = 9223372036854775807$

Opérations sur les entiers

- opérateurs arithmétiques $+$, $-$, $*$
 - $/$: *division entière si les 2 arguments sont des entiers*
 - $\%$: *reste de la division entière*
 - exemples :
 - $15 / 4$ donne 3
 - $15 \% 2$ donne 1
- les opérateurs d'incrémentation $++$ et de décrémentation $--$
permettent d'ajouter ou de retrancher 1 à une variable.

- Exemple :
- `Int n=10;`
- `n++;` // que vaut n maintenant ?
- L'instruction « `n++;` » équivaut à « `n=n+1;` » et L'instruction « `n--;` » équivaut à « `n=n-1;` »
- L'instruction « `7++;` » est illégale !

B) Les réels

Type	Taille (en bits)	Chiffres significatifs	Valeur minimale	Valeur maximale
float	32	7	1.4239846E-45	3.40282347E38
double	64	15	4.9406564584124654E-234	1.797693134862316E308

Opérations sur les réels

□ Les opérateurs

- *opérateurs classiques +, -, *, /*
- *attention pour la division :*
 - *15 / 4 donne 3 division entière*
 - *11.0 / 4 donne 2.75*
(si l'un des termes de la division est un réel, la division retournera un réel).
- *puissance : utilisation de la méthode pow de la classe Math.*
 - *double y = Math.pow(x, a) équivalent à x^a , x et a étant de type double*

C) Type Booléen

- Les booléens
 - boolean
 - contient soit vrai (true) soit faux (false)
- Les opérateurs logiques de comparaisons
 - Egalité : opérateur ==
 - Différence : opérateur !=
 - supérieur et inférieur strictement à : opérateurs > et <
 - supérieur et inférieur ou égal : opérateurs >= et <=

Opérations sur les booléen

■ Opérateurs logiques (les plus utilisés) :

- La conjonction : *et logique* se note **&&**
- La disjonction inclusive : *ou logique* se note **||**
- La négation : *non logique* se note **!**

– **Tables de vérité des opérateurs ET et OU (inclusif)**

a	b	a b (a ou b)	a & b (a et b)
false	false	false	false
false	true	true	false
true	false	true	false
true	true	true	true

Exemples

- Donner la valeur de vérité de x dans les cas suivants :

a) $x=(4==5-1)$; b) $x=(5!=4)$; c) $x=(5>5)$; d) $x=(5<=5)$;

- ■ *si a et b sont 2 variables booléennes. Que vaut la variable c ?*

boolean a,b, c;

a= **true**;

b= **false**;

c= (a **&&** b);

c= (a **||** b);

c= **!(a && b)**;

c=**!a**;

D) Les caractères

- Notation :

- *char* : contient une seule lettre
- le type *char* désigne des caractères en représentation Unicode
 - Codage sur 2 octets contrairement à ASCII/ANSI codé sur 1 octet. Le codage ASCII/ANSI est un sous-ensemble d'Unicode
 - Notation hexadécimale des caractères Unicode de ' \u0000 ' à ' \uFFFF '.
 - Plus d'information sur Unicode à : www.unicode.org

Exemple

□ Notation en java

char *a,b,c;* // *a,b et c sont des variables du type char*

a='a'; // *a contient la lettre 'a'*

b= '\u0022' // *b contient le caractère guillemet : "*

c=97; // *x contient le caractère de rang 97 : 'a'*

