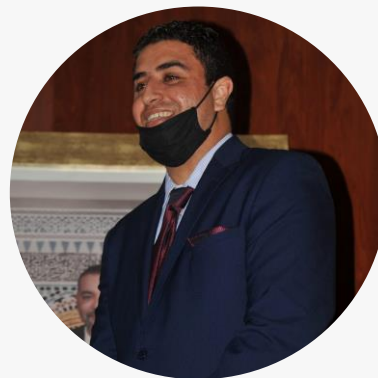


- Classes , objets, et méthodes
- Héritage
- Polymorphisme
- Tableaux et collections
- Gestion des exceptions
- Expression lambda
- L'API Stream
- Généricité
- Les entrées sorties
- Programmation multi-threads
- Programmation client serveur
- Accès aux bases de données.
- Interfaces graphiques avec Javafx

Introduction à la Programmation Orientée Objet Java



Abdelmajid BOUSSELHAM

Email: bousselham@enset-media.ac.ma

Researchgate : https://www.researchgate.net/profile/Abdelmajid_Bousselham2

Google Scholar: <https://scholar.google.com/citations?user=EZ7oxLMAAAAJ&hl=fr>

Scopus: <https://www.scopus.com/authid/detail.uri?authorId=8657730200>

Linkedin: <https://www.linkedin.com/in/abdelmajid-bousselham-ph-d-6729341b8/>

Langage de programmation Java

- **Java** est un langage de programmation orienté objet créé par **James Gosling**, employé de **Sun Microsystems**, présenté officiellement en **1995**.
- Java est utilisé dans plusieurs types d'applications telles que:
 - les applications mobiles (Android) ;
 - les applications Desktop ;
 - les applications Web ;
 - les applications d'entreprise et bien d'autres.
- Les applications **Java** sont compilées dans un code intermédiaire appelé **bytecode** qui peut s'exécuter sur n'importe quelle machine virtuelle Java.

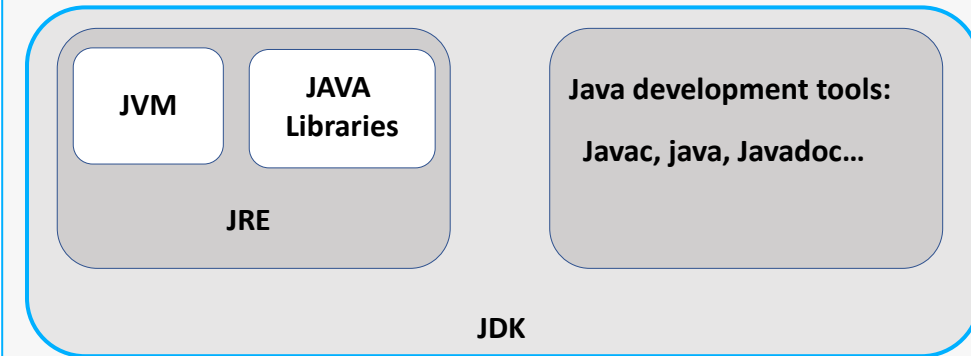


Langage de programmation Java

- **Java** est un langage de programmation orienté objet (classe, héritage, polymorphe...).
- Java est un langage **Multiplateforme**, son principe est: **Write Once Run Anywhere**.
- **Open source** : On peut récupérer le code source de java. Ce qui permet aux développeurs, en cas de besoin, de développer ou modifier des fonctionnalités de java.
- **Distribué** : Les programmes Java peuvent être facilement distribués sur un ou plusieurs systèmes connectés les uns aux autres via une connexion Internet en utilisant des middlewares come RMI.
- **Multithreading** : Il s'agit d'une fonctionnalité Java qui permet l'exécution simultanée de deux ou plusieurs parties d'un programme pour une utilisation maximale des processeurs.

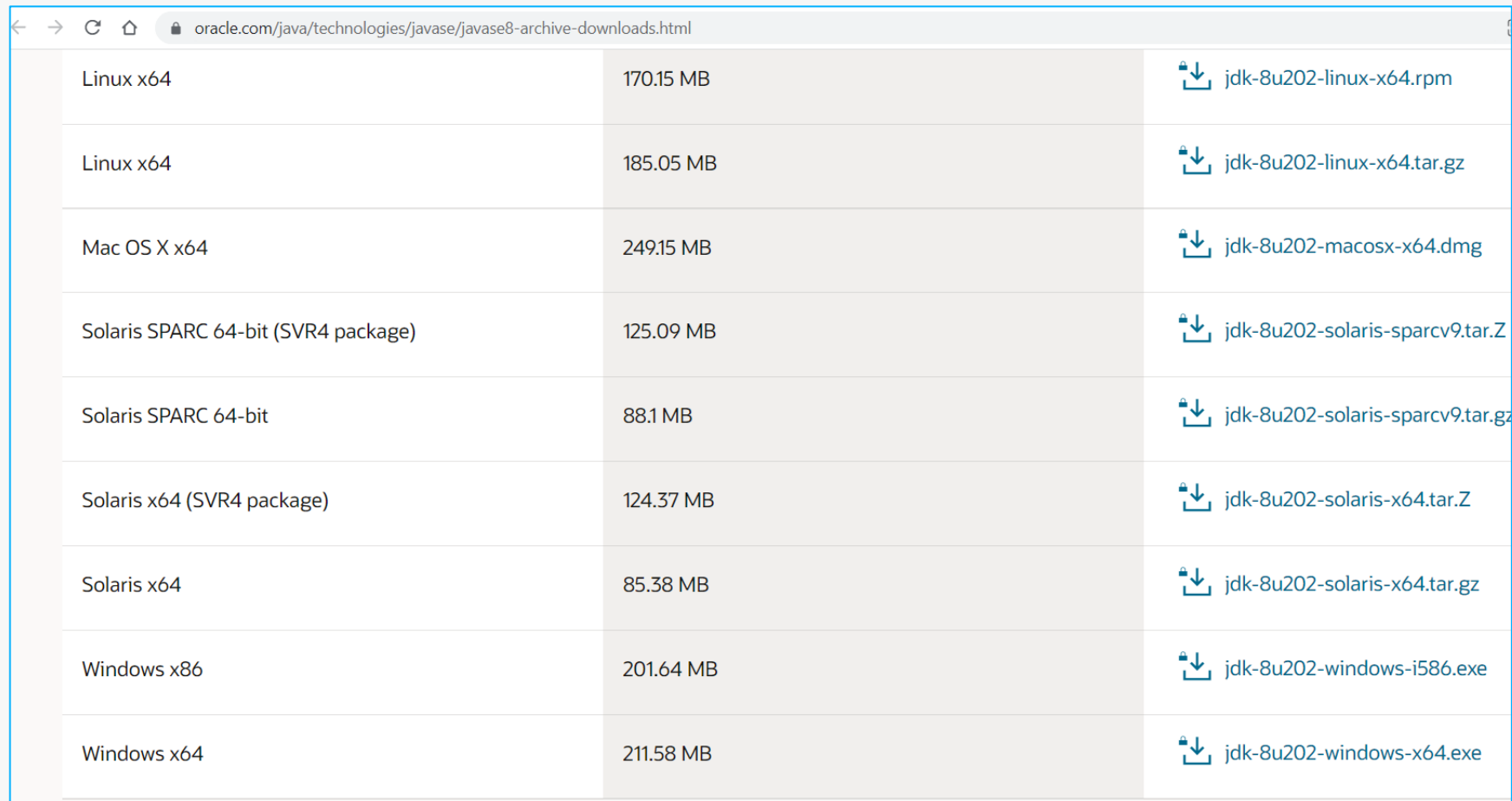
Le Kit de développement Java










- **JDK** (Java Development Kit): JDK est destiné aux développeurs de logiciels et comprend des outils de développement tels que le compilateur Javadoc, Java, Jar et un débogueur.
- **JRE** (Java Runtime Environment) : JRE contient les parties des bibliothèques Java requises pour exécuter des programmes Java et est destiné aux utilisateurs finaux. JRE peut être considéré comme un sous-ensemble de JDK.
- **JVM** (Java Virtual Machine) : Il s'agit d'une spécification qui fournit un environnement d'exécution dans lequel le bytecode Java peut être exécuté. Les JVM sont disponibles pour de nombreuses plates-formes matérielles et logicielles.



Configuration de l'environnement Java

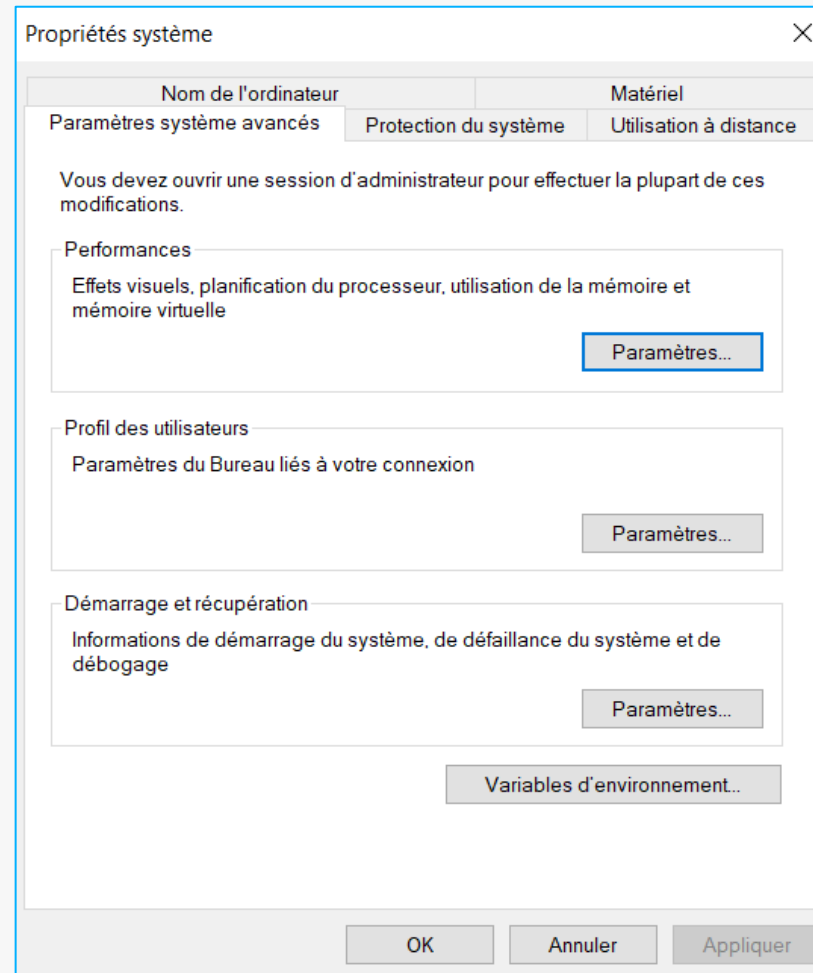
- **Étape 1 : Java 8 JDK** est disponible sur le lien suivant : <https://www.oracle.com/java/technologies/javase/javase8-archive-downloads.html> comme indiqué ci-dessous.

A screenshot of a web browser displaying the Oracle Java 8 archive downloads page. The browser's address bar shows the URL 'oracle.com/java/technologies/javase/javase8-archive-downloads.html'. Below the address bar is a table with nine rows, each representing a different operating system and architecture. Each row contains the OS name, the size of the download in MB, and a download link with a file icon and the filename. The rows are: Linux x64 (170.15 MB, jdk-8u202-linux-x64.rpm), Linux x64 (185.05 MB, jdk-8u202-linux-x64.tar.gz), Mac OS X x64 (249.15 MB, jdk-8u202-macosx-x64.dmg), Solaris SPARC 64-bit (SVR4 package) (125.09 MB, jdk-8u202-solaris-sparcv9.tar.Z), Solaris SPARC 64-bit (88.1 MB, jdk-8u202-solaris-sparcv9.tar.gz), Solaris x64 (SVR4 package) (124.37 MB, jdk-8u202-solaris-x64.tar.Z), Solaris x64 (85.38 MB, jdk-8u202-solaris-x64.tar.gz), Windows x86 (201.64 MB, jdk-8u202-windows-i586.exe), and Windows x64 (211.58 MB, jdk-8u202-windows-x64.exe).

Linux x64	170.15 MB	 jdk-8u202-linux-x64.rpm
Linux x64	185.05 MB	 jdk-8u202-linux-x64.tar.gz
Mac OS X x64	249.15 MB	 jdk-8u202-macosx-x64.dmg
Solaris SPARC 64-bit (SVR4 package)	125.09 MB	 jdk-8u202-solaris-sparcv9.tar.Z
Solaris SPARC 64-bit	88.1 MB	 jdk-8u202-solaris-sparcv9.tar.gz
Solaris x64 (SVR4 package)	124.37 MB	 jdk-8u202-solaris-x64.tar.Z
Solaris x64	85.38 MB	 jdk-8u202-solaris-x64.tar.gz
Windows x86	201.64 MB	 jdk-8u202-windows-i586.exe
Windows x64	211.58 MB	 jdk-8u202-windows-x64.exe

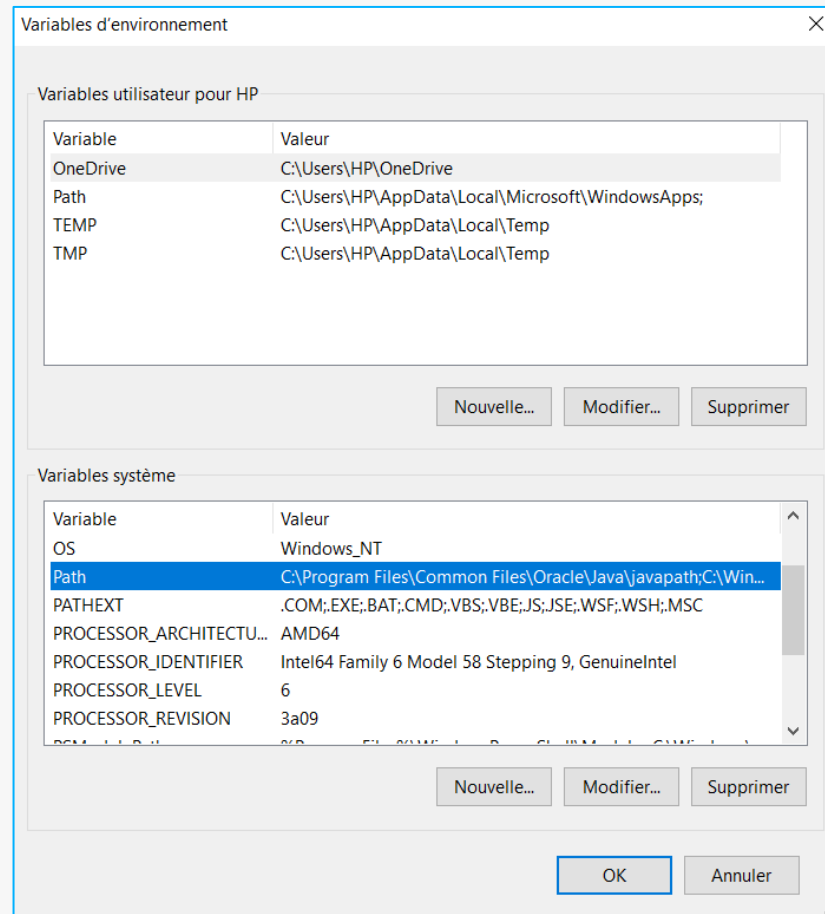
Configuration de l'environnement Java

- **Étape 2** : Allez dans Panneau de configuration -> Système et sécurité -> Système. Sous l'option Paramètres système avancés, cliquez sur Variables d'environnement comme indiqué ci-dessous.



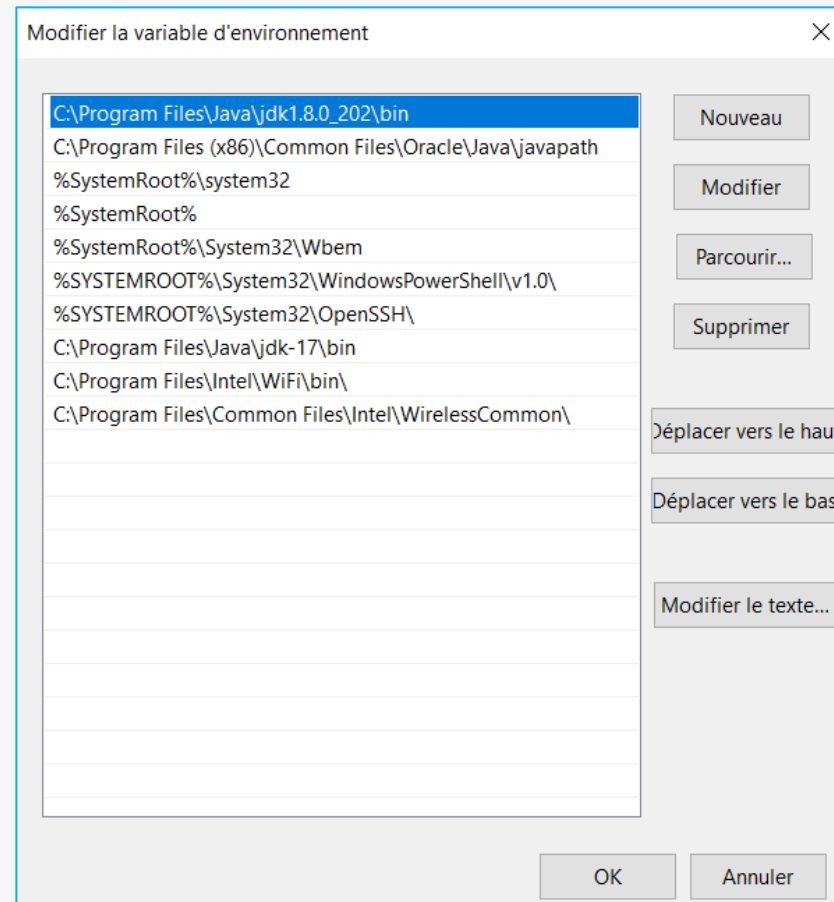
Configuration de l'environnement Java

- **Étape 3** : Maintenant, vous devez modifier la variable "path" sous Variables système afin qu'elle contienne également le chemin d'accès à l'environnement Java. Sélectionnez la variable path et cliquez sur le bouton Modifier comme indiqué ci-dessous.



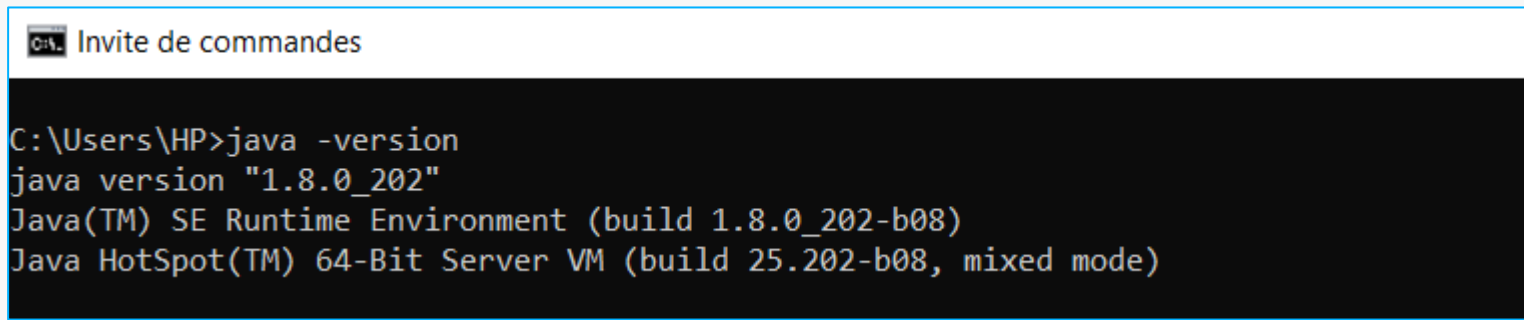
Configuration de l'environnement Java

- **Étape 4 :** Vous verrez une liste de différents chemins, cliquez sur le bouton Nouveau, puis ajoutez le chemin où Java est installé. Par défaut, Java est installé dans le dossier C:\Program Files\Java\jdk1.8.0_202\bin. Si vous avez installé Java à un autre emplacement, ajoutez ce chemin.



Configuration de l'environnement Java

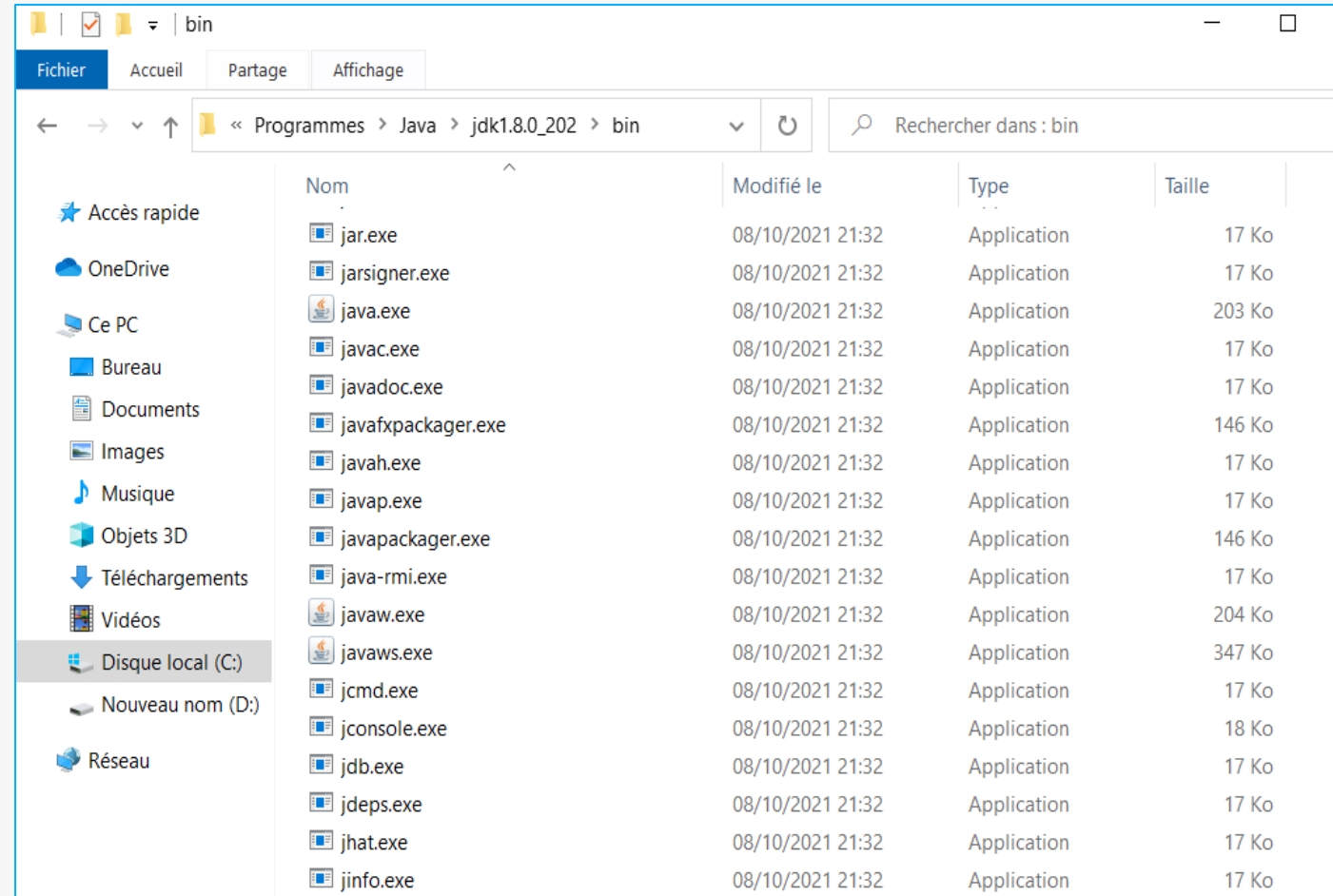
- **Étape 5** : Cliquez sur OK, enregistrez les paramètres, et vous avez terminé !! Maintenant, pour vérifier si l'installation est effectuée correctement, ouvrez l'invite de commande et tapez `java -version`. Vous verrez que Java est en cours d'exécution sur votre machine.



```
C:\Users\HP>java -version
java version "1.8.0_202"
Java(TM) SE Runtime Environment (build 1.8.0_202-b08)
Java HotSpot(TM) 64-Bit Server VM (build 25.202-b08, mixed mode)
```

Ce que contient le JDK

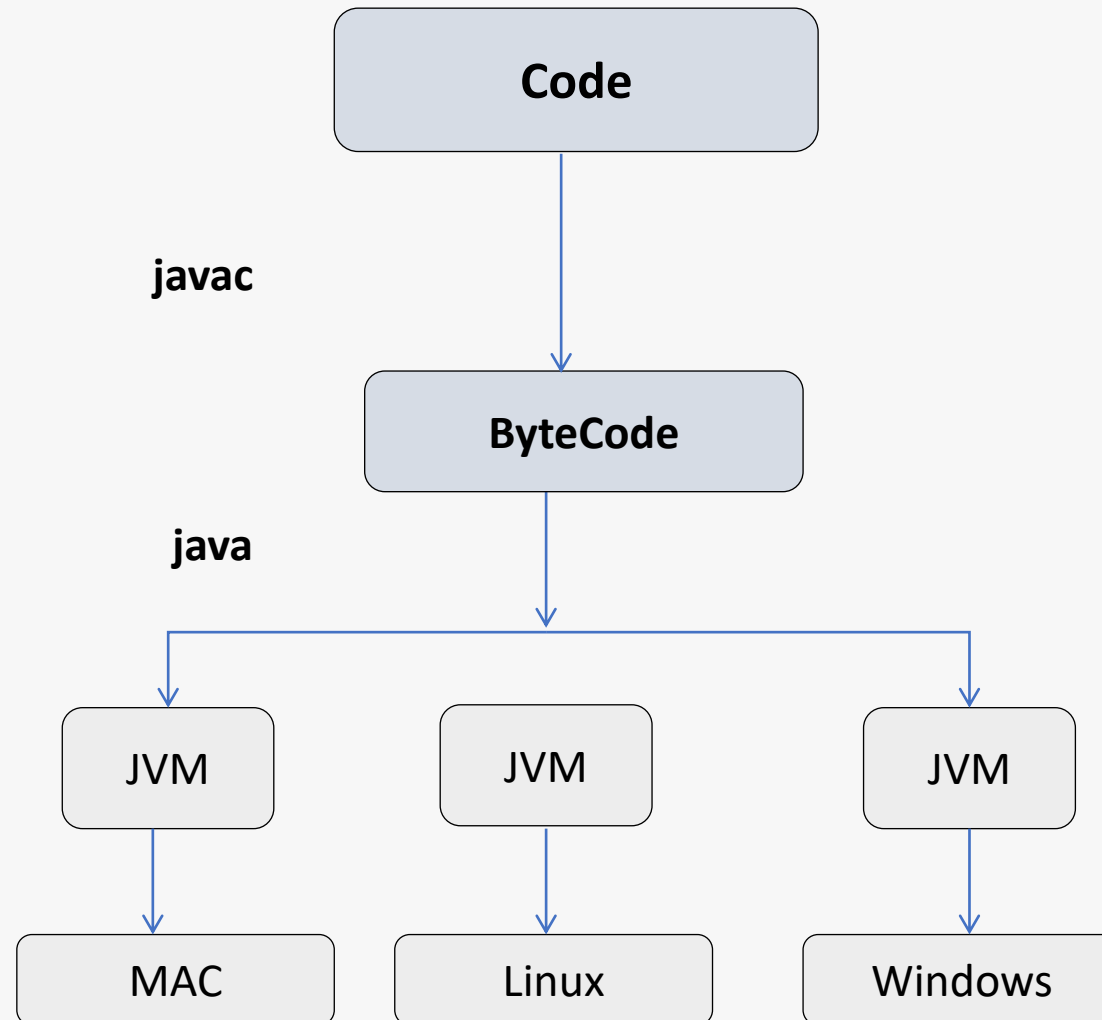
- Les programmes nécessaires au développement java sont placés dans le répertoire **C:\Program Files\Java\jdk1.8.0_202\bin** à savoir :
 - **javac.exe** : compilateur java.
 - **java.exe** : interpréteur du bytecode java.
 - **jdb.exe** : débogueur java.
 - **javadoc.exe** : générer la documentation de vos programmes java.
 - **jar.exe** : Permet de compresser les classes Java ainsi que tous les fichiers nécessaires à l'exécution d'un programme (graphiques, sons, etc.).
 - ...



The screenshot shows a Windows File Explorer window titled 'bin' with the address bar displaying the path: < < Programmes > Java > jdk1.8.0_202 > bin. The left sidebar shows the 'Disque local (C:)' selected. The main area displays a list of files with columns for 'Nom', 'Modifié le', 'Type', and 'Taille'. All files are of type 'Application' and were modified on '08/10/2021 21:32'.

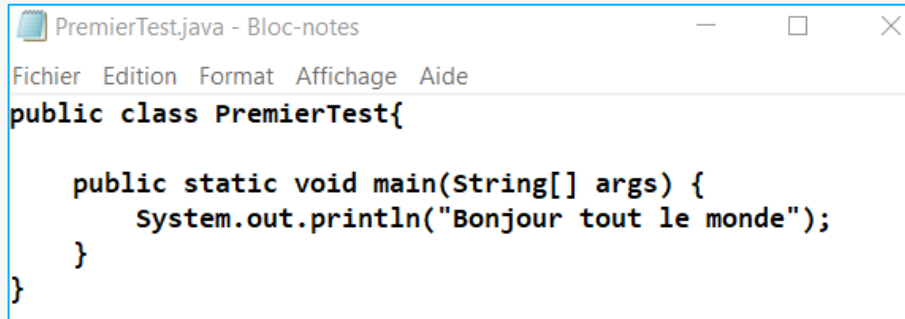
Nom	Modifié le	Type	Taille
jar.exe	08/10/2021 21:32	Application	17 Ko
jarsigner.exe	08/10/2021 21:32	Application	17 Ko
java.exe	08/10/2021 21:32	Application	203 Ko
javac.exe	08/10/2021 21:32	Application	17 Ko
javadoc.exe	08/10/2021 21:32	Application	17 Ko
javafxpackager.exe	08/10/2021 21:32	Application	146 Ko
javah.exe	08/10/2021 21:32	Application	17 Ko
javap.exe	08/10/2021 21:32	Application	17 Ko
javapackager.exe	08/10/2021 21:32	Application	146 Ko
java-rmi.exe	08/10/2021 21:32	Application	17 Ko
javaw.exe	08/10/2021 21:32	Application	204 Ko
javaws.exe	08/10/2021 21:32	Application	347 Ko
jcmd.exe	08/10/2021 21:32	Application	17 Ko
jconsole.exe	08/10/2021 21:32	Application	18 Ko
jdb.exe	08/10/2021 21:32	Application	17 Ko
jdeps.exe	08/10/2021 21:32	Application	17 Ko
jhat.exe	08/10/2021 21:32	Application	17 Ko
jinfo.exe	08/10/2021 21:32	Application	17 Ko

Le processus d'exécution d'un programme JAVA



Premier programme sans IDE

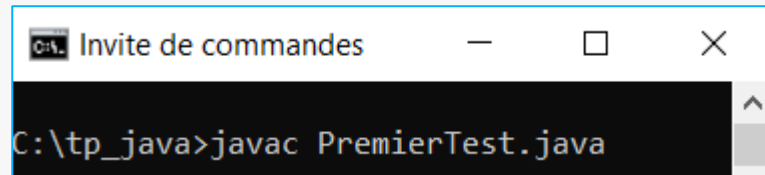
- Ecrire mon premier programme:



```
PremierTest.java - Bloc-notes
Fichier  Edition  Format  Affichage  Aide
public class PremierTest{

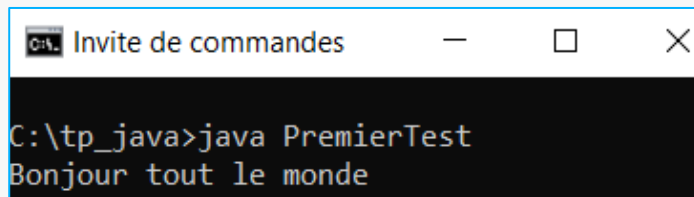
    public static void main(String[] args) {
        System.out.println("Bonjour tout le monde");
    }
}
```

- Compiler mon premier programme :



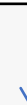
```
C:\tp_java>javac PremierTest.java
```

- Exécuter mon premier programme :



```
C:\tp_java>java PremierTest
Bonjour tout le monde
```

Création de **PremierTest.java**
Avec **Bloc-notes**
(Code source)



Compilation de Test.java
Avec **javac PremierTest.java**
Crée **PremierTest.class**
(ByteCode)



Exécution de PremierTest.class
Avec **java PremierTest**
Exécute le programme

Les fichiers .jar

- Un fichier jar (java archive) est une archive qui est utilisée pour envelopper des fichiers compilés .class, ainsi que toutes les ressources utilisées (images, audio, configuration, etc...).
- Un fichier .jar est basé sur le format Zip, on peut cependant renommer les fichiers .jar avec l'extension.zip et les manipuler avec les outils ZIP (7-Zip, WinZip, WinRAR, etc...).
- Le JDK fournit l'outil jar qui permet de créer ou extraire un fichier jar, visualiser son contenu, le modifier etc.
- Un fichier .jar peut être distribué et exécuté s'il contient au moins une classe avec une méthode main(). Ou peut être utilisé comme une bibliothèques qui sera utilisée par d'autres applications.
- Le fichier jar contient un fichier Manifest qui permet de préciser des informations d'exécution sur le fichier jar (classe principale de l'application, classpath, ...).

Les fichiers .jar

- Créer un fichier JAR

```
C:\java_projects>jar cvfe myApp.jar Application Application.class  
manifeste ajouté  
ajout : Application.class(entrée = 437) (sortie = 298)(compression : 31 %)
```

- Afficher le contenu du fichier jar

```
C:\java_projects>jar tf myApp.jar  
META-INF/  
META-INF/MANIFEST.MF  
Application.class
```

- Exécuter un fichier jar

```
C:\java_projects>java -jar myApp.jar  
Bonjour tout le monde
```

- Extraire un fichier jar

```
C:\java_projects>jar xf myApp.jar
```

- L'option **c** est utilisée pour créer un fichier JAR.
- L'option **f** permet de spécifier le nom de JAR à créer.
- L'option **t** affiche le contenu du fichier JAR.
- L'option **x** est pour extraire le fichier JAR.
- L'option **v** produit un affichage détaillé sur la console pendant la construction du fichier JAR. L'affichage indique le nom de chaque fichier lorsqu'il est ajouté au fichier JAR.
- l'option **m** utilisé pour inclure des informations de manifeste à partir d'un fichier manifeste existant.

Le classpath

- Le **classpath** en java permet de décrire au compilateur et à la JVM l'emplacement où sont disponibles les classes nécessaires pour la compilation et l'exécution d'une application.
- Si le classpath n'est pas défini par le programmeur, sa valeur par défaut sera "." (point), ce qui signifie que uniquement les classes du répertoire courant qui peuvent être chargées.
- Le **classpath** peut être défini avec trois méthodes :
 - à l'aide de la variable d'environnement **classpath**;
 - En utilisant l'option **-cp** ou **-classpath** dans la ligne de commandes;
 - ou de l'attribut **Class-Path** dans le fichier **Manifest.mf** à l'intérieur du fichier JAR en Java.

Outils de développement java

- Pour développer des application java, on peut utiliser un simple éditeur comme notepad de windows mais il est préférable d'utiliser un éditeur conçu pour la programmation java exemples: IntelliJ IDEA , eclipse,
- **IntelliJ IDEA** est l'un des meilleurs IDE pour le développement Java, et parmi les plus utilisés en entreprise.
- Autres IDE java :
 - eclipse (très utilisé).
 - NetBeans.
 - JDeveloper.
 - JCreator.

Instructions de base en java

Variables en Java

- Une variable est un nom donné à un emplacement mémoire. C'est l'unité de base de stockage dans un programme.
 - La valeur stockée dans une variable peut être modifiée pendant l'exécution du programme.
 - Une variable n'est qu'un nom donné à un emplacement mémoire, toutes les opérations effectuées sur la variable affectent cet emplacement mémoire.
 - En Java, toutes les variables doivent être déclarées avant utilisation..
- Déclaration d'une variable :

Diagram illustrating the declaration of a variable: `float note;`. The word `float` is labeled as the **Type** (indicated by an arrow). The word `note` is labeled as the **Nom** (indicated by an arrow).

- Initialiser les variables :

Diagram illustrating the initialization of a variable: `float note=18;`. The word `float` is labeled as the **Type** (indicated by an arrow). The word `note` is labeled as the **Nom** (indicated by an arrow). The value `18` is labeled as the **Valeur** (indicated by an arrow).

Variables en Java

- Une variable est un nom donné à un emplacement mémoire. C'est l'unité de base de stockage dans un programme.
 - Pour respecter la typologie de java, les nom des variables commencent toujours par un caractère en minuscule et pour indiquer un séparateur de mots, on utilise les majuscules. Exemples :

```
int nbPersonnes;
```

```
String nomPersonne;
```

- Valeurs par défaut des primitives :
 - Toutes les primitives de type numérique utilisées comme membres d'un objet sont initialisées à la valeur 0, Le type boolean est initialisé à la valeur false.

Types primitifs de données en Java

Type	Taille	Étendue
boolean	1 bit	true ou false
byte	8 bits	-128 à +127
char	16 bits	0 à 65 535
short	16 bits	-32 768 à +32 767
int	32 bits	-2 147 483 648 à + 2 147 483 647
long	64 bits	$-9,223 \times 10^{18}$ à $9,223 \times 10^{18}$
float	32 bits	$\pm 1.4 \times 10^{-45}$ à $\pm 3.4 \times 10^{38}$
double	64 bits	$\pm 4.9 \times 10^{-324}$ à $\pm 1.8 \times 10^{308}$
void	0 bit	-

Casting des données primitives

- Le casting (mot anglais qui signifie moulage), également appelé cast ou, parfois, transtypage, consiste à effectuer une conversion d'un type vers un autre type.
- Il y a deux types de casting :
 - **sur-casting** : convertit un type de données plus particulier vers un type de données plus général.

- Le **sur-casting** peut se faire implicitement ou explicitement.

```
int a=8;  
long b;  
b=a;    // Casting implicite  
b=(long)a; //Casting explicite
```

- **sous-casting** : convertit un type de données général vers un type de données plus particulier.
 - Le **sous-casting** ne peut se faire qu'explicitement.

```
float a=(float)7.5;  
double b=5;  
byte c=(byte)b;  
int d=6;  
byte e=(byte)d;
```

Les enveloppeurs (wrappers)

- Les types primitifs sont enveloppés dans des objets appelés enveloppeurs (wrappers). Les enveloppeurs sont des classes.
- Le tableau ci-dessous montre le type primitif et la classe wrapper équivalente :

Type de données primitif	La classe
byte	Byte
short	Short
int	Integer
long	Long
float	Float
double	Double
boolean	Boolean
char	Character

Les enveloppeurs (wrappers)

- Exemple :

```
public class Application {  
    public static void main(String args[]) {  
        int a1=5; // a1 est une primitive  
        Integer a2=3; // a2 est un objet  
        Integer a3=new Integer( value: 4); // a3 est un objet  
        System.out.println(a1);  
        System.out.println(a2.intValue());  
        System.out.println(a3.intValue());  
        // exemple pour les doubles  
        double b1=6.5; // b1 est une primitive  
        Double b2=4.6; // b2 est un objet  
        Double b3=new Double( value: 8.2); // b3 est un objet  
        System.out.println(b1);  
        System.out.println(b2.doubleValue());  
        System.out.println(b3.doubleValue());  
    }  
}
```

Résultat

5
3
4
6.5
4.6
8.2

Commentaires en Java

- En Java, il existe trois types de commentaires :

- Commentaires sur une seule ligne.

```
//ceci est un commentaire java
```

- Commentaires sur plusieurs lignes.

```
/* ceci  
   est un commentaire java sur plusieurs lignes  
*/
```

- Commentaires sur la documentation : permet de générer une page de documentation de référence, utilisée pour obtenir des informations sur les méthodes présentes, ses paramètres, etc.

```
/**début du commentaire  
 *  
 * Ceci est un commentaire de documentation  
 *  
 * Fin du commentaire */
```


Les opérateurs

- Opérateurs arithmétiques :

Opérateur	Description
+	opérateur d'addition
-	opérateur de soustraction
*	opérateur de multiplication
/	opérateur de division
%	opérateur de modulo (reste de la division entière)

Les opérateurs

- Opérateurs de comparaison

Opérateur	Nom	Description
==	opérateur d'égalité	renvoie true si le côté gauche est égal au côté droit.
!=	opérateur de différence	renvoie true si le côté gauche n'est pas égal au côté droit.
>	opérateur de supériorité stricte	renvoie true si le côté gauche est supérieur au côté droit.
> =	opérateur de supériorité	renvoie true si le côté gauche est supérieur ou égal au côté droit.
<	opérateur d'infériorité stricte	renvoie true si le côté gauche est inférieur au côté droit.
<=	opérateur d'infériorité	renvoie true si le côté gauche est inférieur ou égal au côté droit.

Les opérateurs

- Opérateurs logiques

Opérateur	Nom	Description
&&	ET logique	Renvoie True si les deux conditions sont vraies
	OU logique	Renvoie True si l'une des conditions est vraie
!	NON logique	Inverse l'état d'une variable booléenne (retourne la valeur true si la variable vaut false, false si elle vaut true)

Les opérateurs

- Opérateurs d'assignation

Opérateur	Description
=	affecte une valeur à la variable
+=	ajoute l'opérande gauche avec l'opérande droit, puis l'affecte à la variable de gauche.
-=	soustrait l'opérande gauche avec l'opérande droit, puis l'affecte à la variable de gauche.
*=	multiplier l'opérande gauche avec l'opérande droit, puis l'affecte à la variable de gauche.
/=	divise l'opérande gauche avec l'opérande droit, puis l'affecte à la variable de gauche.
%=	affecter le modulo de l'opérande gauche à l'opérande droit, puis l'affecte à la variable de gauche
++	utilisé pour incrémenter la valeur de 1
--	utilisé pour décrémenter la valeur de 1

Les opérateurs

- Opérateurs de manipulation de bits

Opérateur	Description
&	renvoie 1 si les deux bits de même poids sont à 1.
	renvoie 1 si l'un ou l'autre des deux bits de même poids est à 1 (ou les deux)
^	renvoie 1 si l'un des deux bits de même poids est à 1 (mais pas les deux)
~	Il s'agit d'un opérateur unaire qui renvoie la représentation du complément à un de la valeur d'entrée,
<<	décale les bits du nombre vers la gauche et remplit 0 sur les vides laissés en conséquence
>>	décale les bits du nombre vers la droite et remplit 0 sur les vides à gauche en conséquence

Les instructions conditionnelles

- Le test conditionnel: if

```
if (condition){  
    // Exécute ce bloc si la condition est vraie  
}
```

- Le test conditionnel: if (condition) { ... } else { ... }

```
If (condition) {  
    // Exécute ce bloc si la condition est vraie  
} else {  
    // Exécute ce bloc si la condition est fausse  
}
```

- Opérateur ternaire

```
(condition) ? instruction si vrai : instruction si faux
```

Les instructions conditionnelles

- Les structures de contrôle - switch

```
switch (expression) {  
    case Valeur1: Liste d'instructions break;  
    case Valeur2: Liste d'instructions break;  
    case Valeurs...: Liste d'instructions break;  
    default: Liste d'instructions break;  
}
```

- L'expression peut être de type byte, short, int char ou une énumération. À partir de JDK7, l'expression peut également être de type String.

Les traitements en boucle

- La boucle WHILE

```
while ( condition ) {  
    bloc d'instructions  
}
```

- La boucle – DO WHILE

```
do {  
    bloc d'instructions  
} while ( condition );
```

- La boucle FOR

```
for (initialisation; condition; incrémenter / décrémenter)  
{  
    bloc d'instructions  
}
```