Introduction à Java – Jour 1

1. Pourquoi apprendre Java ?

Vous connaissez déjà HTML et CSS : ces langages servent à construire et styliser des pages web. Mais ils ne permettent pas de gérer la logique ou d'interagir en profondeur avec un ordinateur. Java, au contraire, est un langage de programmation complet qui permet de créer des applications de bureau, des sites web dynamiques, des serveurs, et même des applications mobiles.

Java est également utilisé dans beaucoup d'entreprises car il est stable, sécurisé et fonctionne sur presque toutes les plateformes. Grâce à la machine virtuelle Java (JVM), un même programme peut tourner sur Windows, macOS, Linux, etc. sans être réécrit.

En résumé : HTML/CSS = structure et style ; Java = logique et interaction.

2. Qu'est-ce que Java ?

Java est un langage orienté objet, ce qui signifie qu'il repose sur le concept de 'classe' et 'd'objet'. Une classe est comme un modèle (ou plan) qui décrit les caractéristiques et comportements d'un objet.

Contrairement à HTML,CSS ou JS qui est interprété directement dans le navigateur, Java doit être **compilé**. Cela veut dire que le code source (écrit par le développeur) est transformé en un code intermédiaire appelé **bytecode**, que seule la JVM (Java Virtual Machine) peut lire.

Le processus est le suivant :

1️⃣ Tu écris ton code dans un fichier `MonProgramme.java`.

2️⃣ Le compilateur (`javac`) traduit ce code en bytecode (`MonProgramme.class`).

3️⃣ La JVM exécute ce bytecode sur ton ordinateur.

Ce système rend Java très portable : le même code peut s'exécuter partout, tant qu'une JVM est installée.

3. Le JDK et l'IDE

Pour programmer en Java, tu as besoin d'un ensemble d'outils appelé **JDK (Java Development Kit)**. Il contient :

• Le compilateur `javac` : traduit ton code en bytecode.

• La JVM : exécute le bytecode.

• Le JRE (Java Runtime Environment) : fournit les bibliothèques nécessaires au fonctionnement.

Pour écrire ton code, tu peux utiliser un **IDE (Integrated Development Environment)** comme IntelliJ IDEA, Eclipse ou Visual Studio Code. Un IDE facilite la vie du programmeur : il colorie le code, aide à détecter les erreurs et permet d'exécuter le programme en un clic.

➡️ En résumé : JDK = outils pour Java, IDE = espace de travail pour programmer.

4. Premier programme : Hello World

Le premier programme traditionnel consiste à afficher un simple message dans la console. Voici l'exemple classique en Java :

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

Explication ligne par ligne :

• `public class HelloWorld` → on crée une classe nommée HelloWorld (le nom du fichier doit être identique).

• `public static void main(String[] args)` → la méthode principale, point d'entrée du programme.

• `System.out.println` → affiche un message à l'écran (println = print line).

Quand tu exécutes ce programme, la JVM affiche le texte dans la console. Cela montre que ton environnement Java est bien configuré !

5. Les variables et types simples

En Java, une **variable** est une zone de mémoire dans laquelle on stocke une donnée. Chaque variable a un **nom** et un **type**. Le type indique le genre de données que la variable peut contenir.

Exemples de types primitifs :

• `int` → nombres entiers (ex : 10, -5, 42)

• `double` → nombres décimaux (ex : 3.14, 19.99)

• `boolean` → vrai ou faux (true / false)

• `char` → un seul caractère (ex : 'A')

• `String` (non primitif) → une suite de caractères (ex : 'Bonjour')

Exemple de code :

```
int age = 20;
 String nom = "Alice";
 boolean etudiant = true;
 System.out.println("Nom : " + nom);
 System.out.println("Âge : " + age);
 System.out.println("Étudiant : " + etudiant);
```

👉 En Java, chaque variable doit être déclarée avec son type. Cela aide le programmeur à éviter les erreurs et rend le code plus clair.

6. Exercice pratique

Créer un programme `Bienvenue.java` qui demande le prénom de l'utilisateur et affiche :

« Bonjour [prénom], bienvenue dans le monde Java ! »

Voici une version possible :

```
import java.util.Scanner;

 public class Bienvenue {
        public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        System.out.print("Entrez votre prénom : ");
        String prenom = input.nextLine();
        System.out.println("Bonjour " + prenom + ", bienvenue dans le monde Java !");
        input.close();
        }
}
```

Ce petit programme introduit la **lecture d'entrée utilisateur** avec la classe `Scanner`.
C'est une base importante pour les prochains exercices.

7. Plus avance

Dans les prochains cours, on apprendra à :

• Utiliser les conditions (`if`, `else`) pour faire des choix.

• Répéter des actions avec des boucles (`for`, `while`).

• Créer et utiliser nos propres méthodes.

• Manipuler des objets et comprendre la programmation orientée objet (POO).