

Java

Pour l'Ide : IntelliJ, eclipse fondation

Mu SQL : on premise, avec wamp (xamp, namp,...), docker, le cloud.

Le JAVA est un langage de programmation orienté objet.

Qu'elle est sont but:

- optimiser du temps des cycles de développement (compilation, execution)
- Langages système simple, orienté et interprété
- Les applications produites sont portable (c'est à dire qu'elle sont cross-platform, t'en que l'os peut exécuter du java)
- Comparer au C++ la gestion de la mémoire et des erreurs est simplifiée
- Il est possible de créer des applications multi threadés (utiliser plusieurs processeur ' calculer')
- Les applications sont très robustes et sécurisées grâce à des vérifications du bytecode avant l'exécution.
- Le bytecode est un type de code de programme compilé et exécuté sur un système informatique appelé machine virtuelle.

La plateforme Java :

- Schéma simplifié
- JDK: Java Developpement Kit
- API
- JAVAC: le compilateur de java
- JRE (Java Runtime Environnement) :

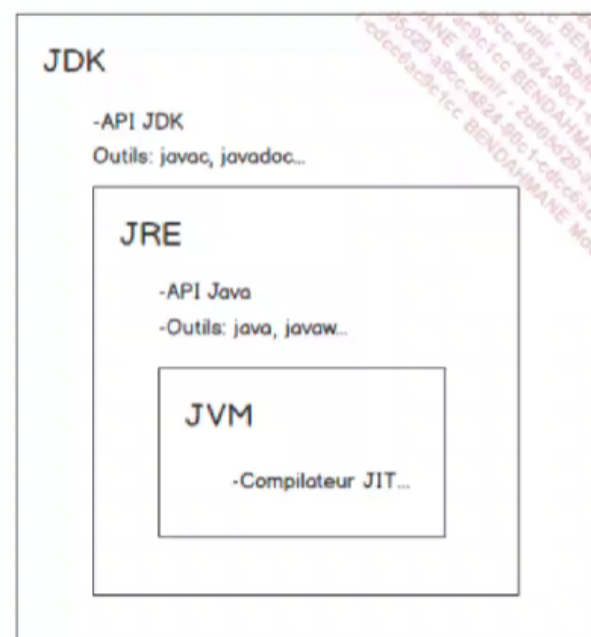
environnement d'exécution installer sur la machine

cliente . Il propose l'outil java qui permet d'exécuter

les programmes java.

- JVM (Java Virtual Machine) : permet l'exécution

du programme passe en paramètre de la commande Java .



Deux implémentations pour notre platform:

Oracle JDK (Support , mise a jour , ...)meilleur , open jdk.

Création d'une nouvelle variables d'environnement système JAVA_HOME avec
comme valeur le lien du jdk .

Ensuite il faut ajouter à la variable d'environnement path %JAVA_Home%\bin

test "java -version "

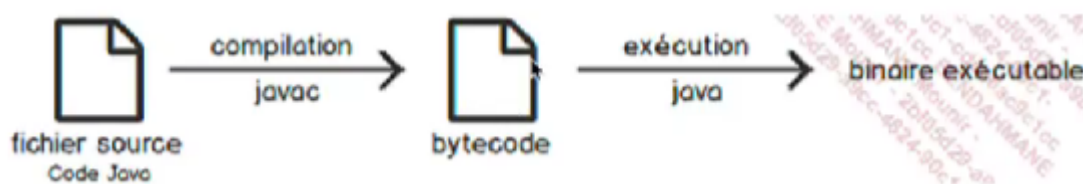
La premier programme :

Un programme java est une collection de fichier java destiné à fonctionner ensemble
public porter

static

void type de retour

Il faut plusieurs étapes pour l'exécution d'un programme en java :



- Compilation (transforme le code source en bytecode `langage intermédiaire`).
- Le bytecode n'est pas du binaire.
- C'est la JVM qui charge le bytecode et le compile à la volée (JIT: Just In Time) pour exécuter du code qui sera compréhensible par la machine hôte.*

Utiliser le compilateur / bytecode :

// Exécution d'un compilateur

`javac nom_de_votre_fichier_source`

//Code source en java

//exécution bytecode

`java nom_de_votre_fichier_bytecode`

Structure des programmes JAVA :

En Java, tout est objet

```
public class Nom {  
    // votre super code java  
    public static void main (String [] arg  
    {  
        // votre super code java  
    }  
    //votre super code java  
}
```

Les variables en java :

Instances de Classe: l'objet créé à partir d'une classe

Les catégories de variables:

- Les variables d'instance: elle existe que si une instance de classe est disponible. Chaque instance possède sa propre version de la variable.
- Les variables de classe: elles sont déclarée à l'intérieur d'une classe mais avec le mot clé `static`. La variable de classe est disponible directement depuis la classe et existe en un exemplaire unique. On peut avoir 50 instances d'une même classe, mais la valeur d'une variable de classe est unique.
public static String test = "Hello World ! " ; variable de class car class
- La variables Locales: sont déclarées à l'intérieur d'une méthode
Dans la méthode

La Nomenclature des variables:

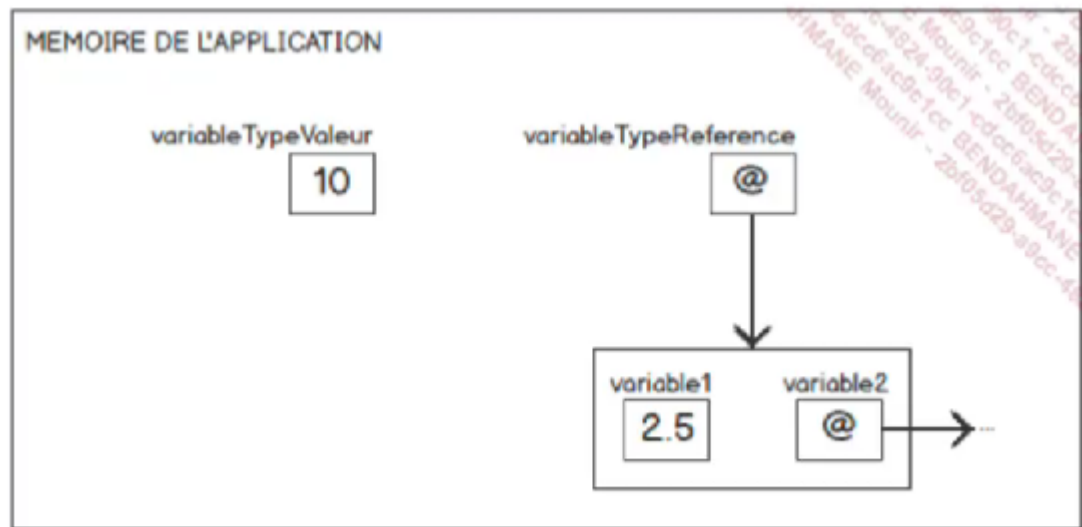
- Le nom d'une variable commence par une lettre
- On peut mettre lettre, chiffre et underscore
- Nombre illimité de caractère faire concie
- case Sensible / sensible à la case (majuscule non equal a minuscule)

a. Les types de variables

On distingue deux types de variables :

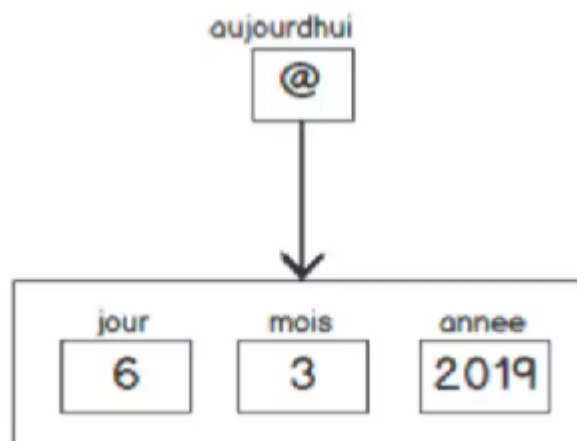
- Les types valeurs: celles qui stockent des valeurs

- Les types références: celles qui ne stockent pas réellement de valeur.

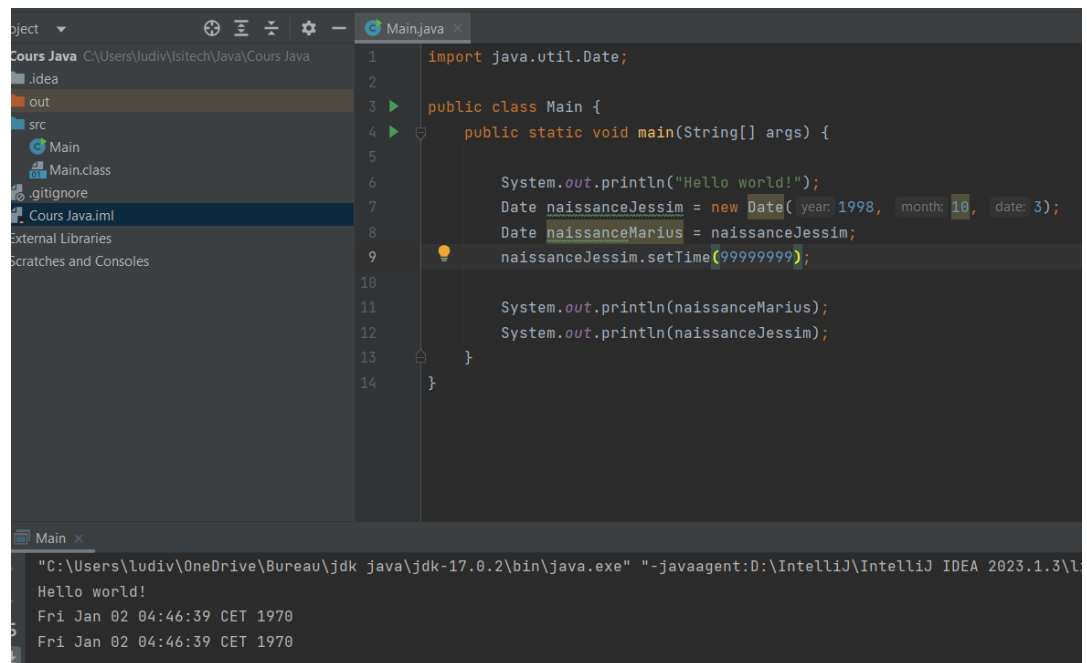


Les types valeurs désignent les types primitifs, on en compte 4 en java : entier, décimaux, les chars et les bools.

Les types références désignent un type plus complexe: un ensemble cohérent de variables. Ils contiennent une référence vers ces variables.



La vérification



```
1 import java.util.Date;
2
3 public class Main {
4     public static void main(String[] args) {
5
6         System.out.println("Hello world!");
7         Date naissanceJessim = new Date( year: 1998, month: 10, date: 3);
8         Date naissanceMarius = naissanceJessim;
9         naissanceJessim.setTime(99999999);
10
11         System.out.println(naissanceMarius);
12         System.out.println(naissanceJessim);
13     }
14 }
```

Terminal output:

```
"C:\Users\ludiv\OneDrive\Bureau\jdk java\jdk-17.0.2\bin\java.exe" "-javaagent:D:\IntelliJ\IntelliJ IDEA 2023.1.3\l
Hello world!
Fri Jan 02 04:46:39 CET 1970
Fri Jan 02 04:46:39 CET 1970
```

b. La déclaration

En java:

[modificateurs] type maVariable;

//ne pas oublier le point virgule

On peut declarer plusieurs variables du meme type sur la meme ligne :

type var1, var2, var3,var4;

int ageDeTom;

Date armistice, noel;

Pour initialiser une variable

int frenchWorldCupYear = 1998;

Les types d'entier

Types entiers signés			
Type	Valeur minimale	Valeur maximale	Espace de stockage
<code>byte</code>	-128	127	8 bits
<code>short</code>	-32768	32767	16 bits
<code>int</code>	-2147483648	2147483647	32 bits
<code>long</code>	-9223372036854775808	9223372036854775807	64 bits

Les types décimaux
pour un float mettre un f à la fin

Types numériques signés			
Type	Valeur minimale positive	Valeur maximale positive	Espace de stockage
<code>float</code>	1.4E-45	3.4028235E38	32 bits
<code>double</code>	4.9E-324	1.7976931348623157E308	64 bits

c. Les constantes

Il arrive que des variables ne doivent pas être modifiées au cours de l'exécution du programme: il convient alors de définir des constantes

Pour définir une constante on utilise le mot clé `final`

`final double PI = final double PI =`

`3,14159265358979323846264338327950288419716939937510582;`

L'initialisation de la constante est obligatoire lors de sa déclaration.

Souvent les constantes sont définies avec des membres statiques.

Par convention on les écrit en majuscules. Full majuscule

d. Les énumérations

Elles permettent de définir un ensemble de constante:

```

public enum Month{
    JANVIER,
    FEVRIER,
    MARS,
    AVRIL,
    MAI,
    JUIN,
    JUILLET,
    AOUT,
    SEPTEMBRE,
    OCTOBRE,
    NOVEMBRE,
    DECEMBRE
}

```

L'équivalent avec une classe:

```

public class Month
{
    public static final int JANVIER = 0;
}

```

La déclaration d'une enum peut être considéré comme une utilisation d'une "classe cachée"

Cette hérite de ' java.lang.Enum'

Créer un Dalton

Dalton temel = Dalton.AVERELL

e. Les tableaux

//declaration

int[] unTableau ;

//initialisé

//un tableau de taille 122 (122 valeur de 0 à 121)

unTableau = new int [122];

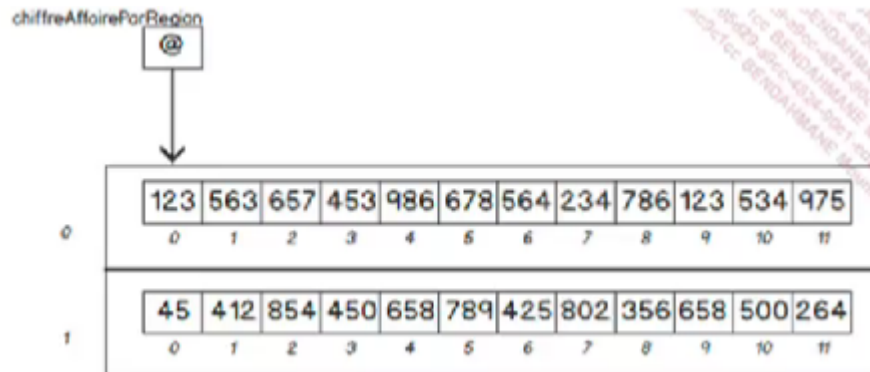
// autre maniere d'initialisé le tableau

//pas besoin de dire de taille et pas besoin de declaration

int [] autreSyntaxe = {3, 4, 5, 8};

//On peut acceder aux elements d'un tableau de cette maniere
 autreSyntaxe[4];
 //Si on tente d'acceder a un index inexistant vous allez obtenir une
 exception de type: 'ArrayIndexOutOfBoundsException'

Il est possible de travailler avec des tableaux a plusieurs dimensions:



La syntaxe pour récupérer des éléments dans un tableau a 2D est la suivante:

```
int elemTableau2D = tableau2D[0][1]
```

autre méthode de création:

```
int [][] matrice;
```

```
matrice = new int [2] []
```

```
matrice [0]= new int [4];
```

```
matrice [1] = new int [4];
```

//permet de definir un tableau 2D avec deux ligne et trois colonnes

```
int [][] encoreUneAutreSyntaxe = {{1,2,3}, {23,21,22}}
```

L' Énoncés :

Crée un tableaux contenant 10 String, et remplir ce tableau avec des adresse mail, exemple

"jpp@sfr.fr"

"tom@gmail.com"

"fred@sfr.fr"

"victor@sfr.fr"

"chris@sfr.fr"

"robert@orange.fr"

"paul@sfr.fr"

"lise@gmail.com"

["thierry@isitech.fr"](mailto:thierry@isitech.fr)

["marie@isitech.fr"](mailto:marie@isitech.fr)

Calculez le pourcentage de fournisseurs de services mail (pour une adresse @gmail.com le fournisseur est gmail).

docs string

<https://docs.oracle.com/en/java/javase/17/docs/api/java.base/java/lang/String.html>