Semaine 10/07/2023

lundi 10 juillet 2023

09:14

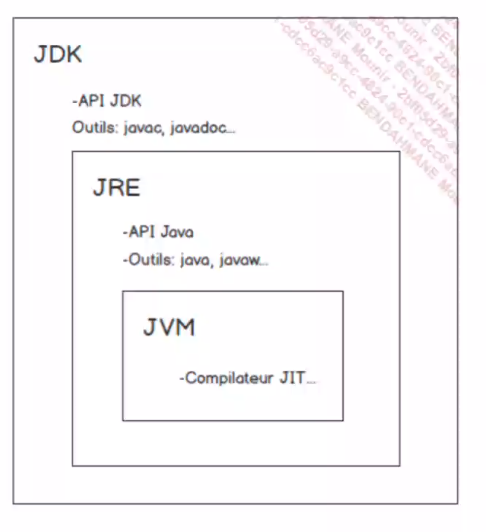
JAVA :

* **Lundi :**

1. **Cours**

* + Java = langage system simple , orienté objet et interprété
  + Optimisation du tps des cycles de dev ( compilation et exécution )
  + Les applications produits sont portables ( disponible dans plus platform)
  + Par rapport au c++ gestion de la mémoire et des erreurs sont simplifier
  + Il est possible de créer des applis multi threadés
  + Un presseur/ trusteur => A voir
  + Les applications sont très robustes et sécurisées grâce à des vérifications du bytecode avant exécution
  + Bytecode : est le résultat de la compilation d'un programme dont le [code source](https://www.techno-science.net/glossaire-definition/Code-source.html) est en Java par le [compilateur](https://www.techno-science.net/glossaire-definition/Compilateur.html) Java. Ce *bytecode* peut être exécuté sous de nombreux systèmes d'exploitation par l'[interpréteur](https://www.techno-science.net/definition/5322.html) Java ou la [machine virtuelle](https://www.techno-science.net/glossaire-definition/Machine-virtuelle.html) Java.

1. **La plateforme JAVA**

* + 

* + JDK : JAVA dev kit
  + API :
  + JavaC : compilateur JAVA
  + JRE : ( java runtime env ) l'environnement d'exécution installer sur la machine cliente. l'outil java c'est l'exécutable qui vas exécuter l'outil ( les programmes java )
  + JVM : (java virtuel machine ) permet l'exécution du programme passe en paramètre de la commande 'java'

1. **La machine virtuelle Java**

* + Elle est mise à disposition pour de multiples OS

* + **Ses rôles sont multiples :** 
    1. Elle charge des classes (et leur bytecode ) a l'instanciation ( la création d'objet à partir d'une classe ) d'une classe c'est elle qui charge le bytecode à interpréter
    2. Elle gère la mémoire : les pointeur et les références
    3. elle gère la libération de la mémoire : 'Gabage collector'
    4. Elle gère aussi la sécurité
    5. Elle gère l'interfaçage avec du code natif

1. **L'API JAVA**

* + Ensemble de la logiciels pré conçu pour fournir un grand nombre de fonctionnalités
  + L'API JAVA 17
  + 21 modules
  + 220 packages (l'équivalent des bibliothèques de classes en C# ou des librairies en langage C )
  + Plus de 4000 classes et interfaces mises à disposition
  + API accès aux données
  + Gérer les réseau (http, https)
  + Fonctionné avec XML

1. **Deux implémentations pour une plateforme**

* + Oracle JDK
  + OpenJDK

* + Pour Windows , Il est recommandé d'ajouter 'JAVA\_HOME' en tant que variable d'environnement :

C:\Users\Sara AWAD\Desktop\cours.informatique\JAVA\jdk-17.0.2

Ensuite vous modifiez la variable d'environnement PATH avec l'entrées :

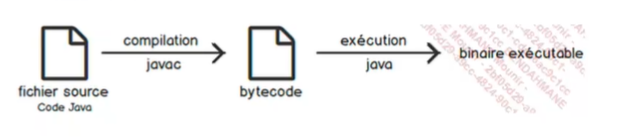
…..:%JAVA\_HOME%\bin

1. **MYSQL**

* + 'on premise'
  + Avec wamp (mamp, xamp)
  + Avec Docker
  + Dans le cloud

1. **Vote 1ere programme en JAVA**

* + Un programme java est une collection de fichier java destinés à fonctionner ensemble



* + Void qui return rien
  + Public
  + Static
  + **Il faut plusieurs étapes pour l'exécution d'un programme java :**

* + Compilation : code source en java => 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

* + **Pour utiliser le compilateur :**

Javac nom-de-vote-fichier-source

* + **Pour exécuter le bytecode :**

Java nom-de-vote-fichier-bytecode

1. **Structure des programmes JAVA:**

* + En java, tout est objet :

Public class Nom

{

//votre super code java

Public static void main(string[] args)

{

//votre code

}

//votre code java

}

1. **Les variable en JAVA:**

* + Il faut déclarer une variable avent l'appeler
  + Les catégories de variable :
    1. **Les variables d'instance :**( instance de classe : l'objet créé à partir d'une classe )
       - n'existe que si une instance de classe est disponible. Chaque instance possède sa propre version de la variable
    2. **Les variable de classe :** elle sont aussi déclarées à l'intérieur d'une classe mais avec le mot clé 'static' . La variable de la 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
    3. **Les variables locales :** sont déclarées à l'intérieur d'une méthode

Public class Main

{

//main méthode, static

Public static void main(String[]args)

{

System.*out*.println("Helloworld!");

}

}

1. **La nomenclature :**

* + Le nom d'une variable commence par une lettre
  + La façon d'écrire la nomenclateur d'une variable (lettre, chiffre et underscore)
  + Nombre illimité de caractères
  + Sensible à la casse
  + Keywords
  + Camel case : le nom commence par une minuscule et chaque nouveau mot commence par une maj

1. **Les types des variables :**

* + **On distingue 2 types des variables :**

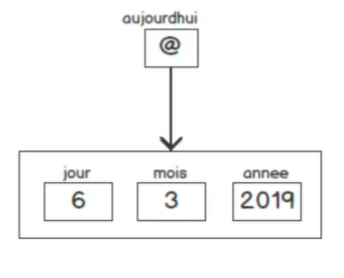
* + Les types valeurs : celles qui stockent des valeur
  + Les types références : celles qui ne stockent pas réellement de valeur

Une image contenant texte, capture d’écran, diagramme, ligne

Description générée automatiquement

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

Les types références désignent un type plus complexe : un ensemble cohérent de variables



1. **La déclaration :**

* + En java :

[modificateurs] type maVariable;

* + On peut déclarer plusieurs variables du même type sur la même ligne

Type var1, ver2, var3………;

* + Exemple:

Int ageDeTom;

Date armistice, noel;

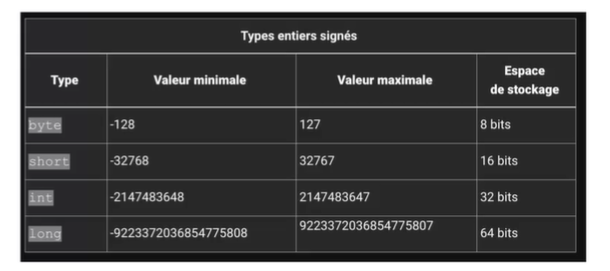
* + Pour initialiser une variable :

Int franchWorldCupYear = 1989;

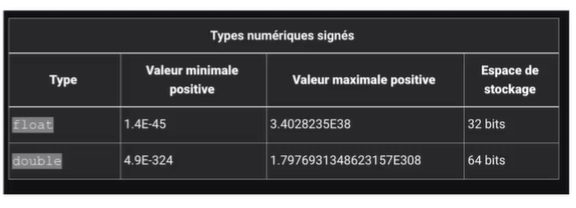
Une image contenant texte, capture d’écran, logiciel, Système d’exploitation

Description générée automatiquement

* + Pour les types d'entier :



* + Pour types décimaux :



1. **Les constante:**

* + Est une valeurs ou chaine de caractère ne doivent pas être modifier 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 = 3.145856465984;

* + l'initialisation de la constante est obligatoire lors de sa déclaration.
  + Souvent les constantes sont définies avec membres statiques (pas besoin d'être instancier pour être utiliser )

Par convention on les écrit en majuscule

* + Les énumération : elles permettent de definir un ensemble de constantes :

Public enum Month

{

JANVIER,

FEVREIER,

MARS,

AVRIL,

MAI,

JUIN,

JUILLET

}

* + l'équivalent avec une classe :

Public class Month

{

Public static final int JANVIER = 0;

}

* + La déclaration d'une enum peut être considère comme l'utilisation d'une "classe cache" . Cette classe hérite de java.lang.Enum

1. **Les tableaux :**

* + Java:

Int[] unTableau;

//Préciser la taille de tableau

unTableau = new int[122];

// avec cette syntaxe pas besoin de présicer la taille c'est automatique

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

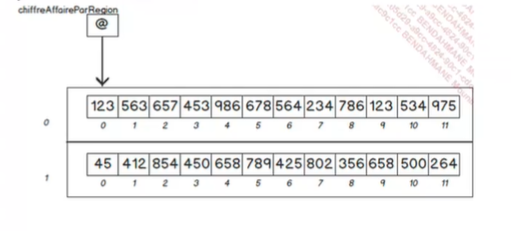
On peut accéder aux éléments d'un tableau de cette manière

autreSyntaxe[4];

* + Si vous tentez d'accéder à un index inexistent vous allez obtenir une exception de type :

'ArrayIndexOutOfboundException'

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



* + La syntaxe pour récupérer des éléments dans un tableau à 2D est la suivante :

Int elemTableau2D = tableau2D [0][1];

* + Autre methode de creation :

Int[][] matrice;

matrice = new int[2][];

matrice[0] = new int[4];

matrice[1] = new int[4];

// permet de définir

un tableau 2D avec deux lignes et trois colonnes

int [][] encoreUneAutreSyntaxe = {{1, 3, 4}, {23, 21, 22}};

1. **Exo :**

* + Créez un tableaux contenant 1@ String , et remplir ce tableau avec des adresses 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”;

“marie@isitech.fr”;

Calculez le pourcentage de fournisseurs de services mail.

(pour une adresse @gmail.com le fournisseur est gmail).

* + Consultez la javadoc :

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