Programmation Objet – Initiation à Java

Letícia SEIXAS PEREIRA

Cours 01 – Introduction au langage Java / Éléments de syntaxe

14/10/2018

Introduction / Syntaxe

1) Introduction au langage Java

2) Eléments de syntaxe

Introduction au langage Java

Qu'est ce que « Java »?

- Java est un langage de programmation orienté objet développé par *Sun Microsystems* (aujourd'hui racheté par Oracle);
- Java est un langage de programmation interprété **et** compilé.

Introduction au langage Java

Quels types d'application pour « Java »?

- On peut faire de nombreuses sortes de programmes avec Java:
  - des applications, sous forme de fenêtre ou de console;
  - des applets, qui sont des programmes Java incorporés à des pages web;
  - des applications pour appareils mobiles;
  - …

Introduction au langage Java

Les différentes versions de Java:

JDK 1.0	1996
JDK 1.1	1997
J2SE 1.2	1998
J2SE 1.3	2000
J2SE 1.4	2002
J2SE 5.0	2004
Java SE 6	2006
Java SE 7	2011
Java SE 8	2014
Java SE 9	2017
Java SE 10	2018
Java SE 11	2018
Java SE 12	Mars/2019
Java SE 13	**Septembre/2019**

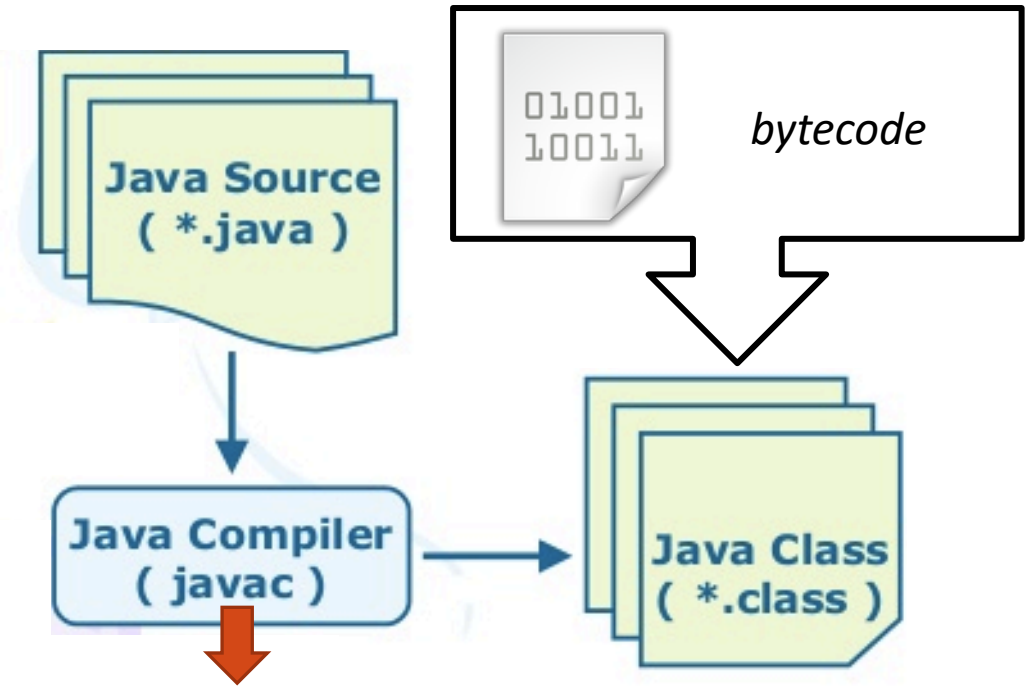
Introduction au langage Java

Comment cela fonctionne t'il?


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

Introduction au langage Java

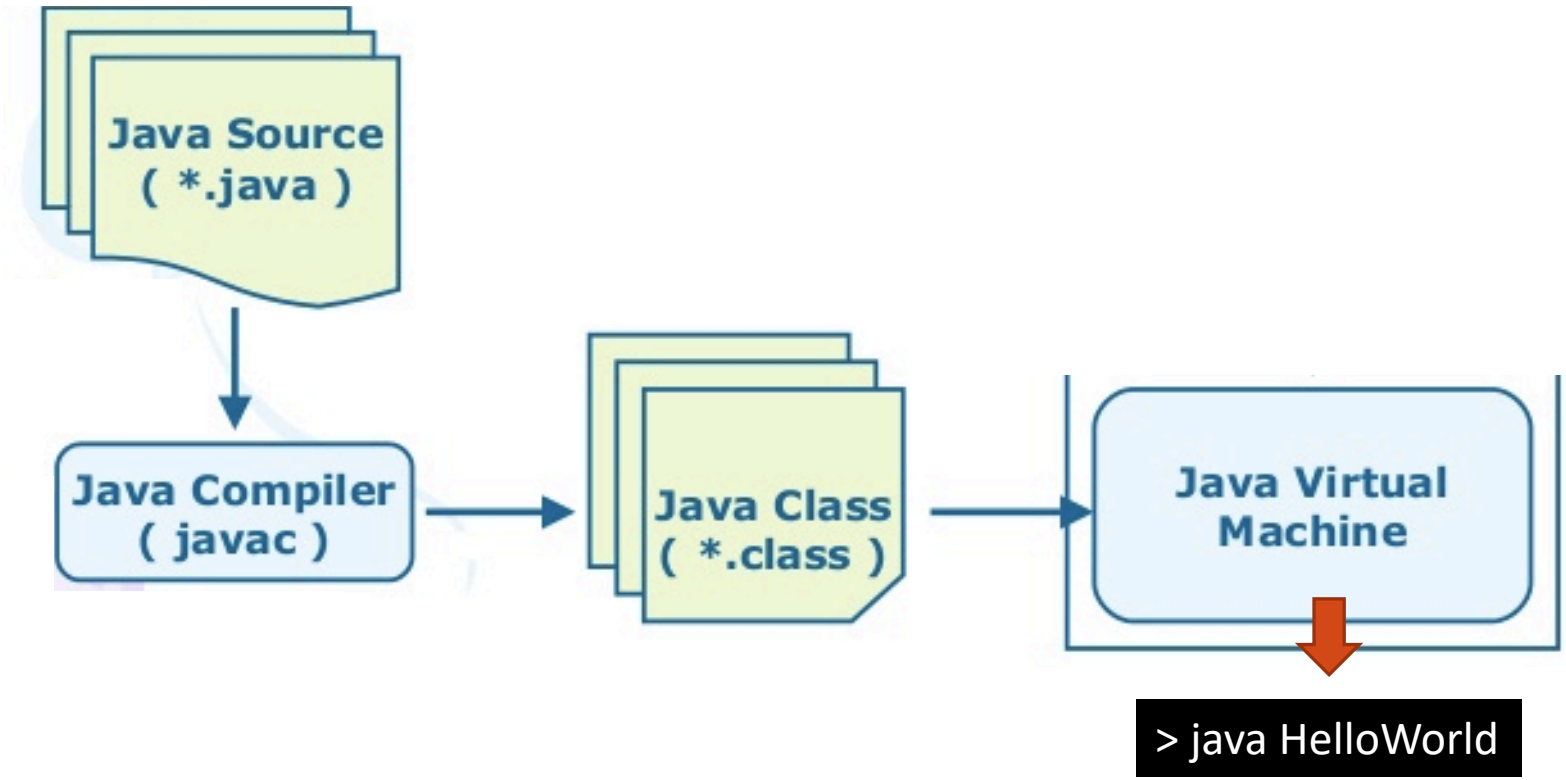
Comment cela fonctionne t'il?




```
> javac HelloWorld.java
```

Introduction au langage Java

Comment cela fonctionne t'il?



Le Java : JVM, JRE, JDK

Le Java : JVM, JRE, JDK

JVM: Java Virtual Machine (Machine virtuelle Java)

- Permet d'interpréter et d'exécuter du code JAVA.

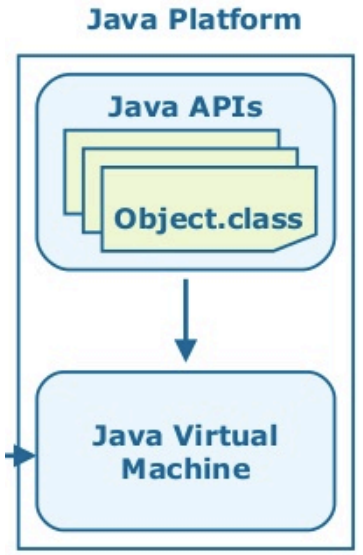
Le Java : JVM, JRE, JDK

JVM: Java Virtual Machine (Machine virtuelle Java)

- Permet d'interpréter et d'exécuter du code JAVA.

JRE: Java Runtime Environment (Environnement d'exécution JAVA)

- Un environnement d'exécution Java, formé par la JVM et les bibliothèques, tout ce dont vous avez besoin pour exécuter une application Java.



Le Java : JVM, JRE, JDK

JVM: Java Virtual Machine (Machine virtuelle Java)

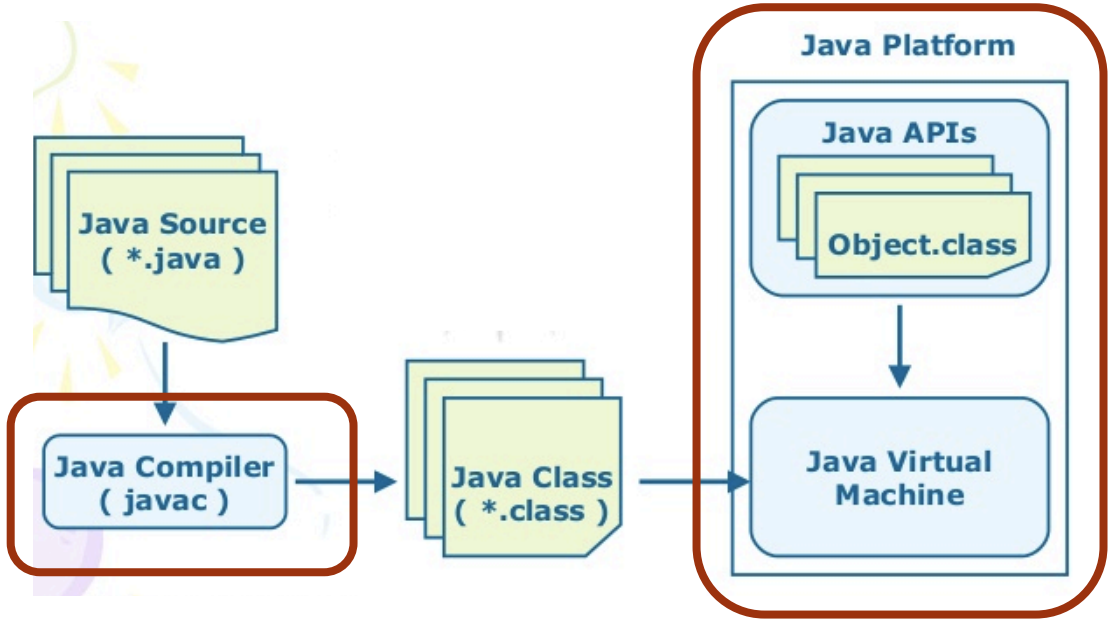
- Permet d'interpréter et d'exécuter du code JAVA.

JRE: Java Runtime Environment (Environnement d'exécution JAVA)

- Un environnement d'exécution Java, formé par la JVM et les bibliothèques, tout ce dont vous avez besoin pour exécuter une application Java.

JDK: Java Development Kit (Kit de développement Java)

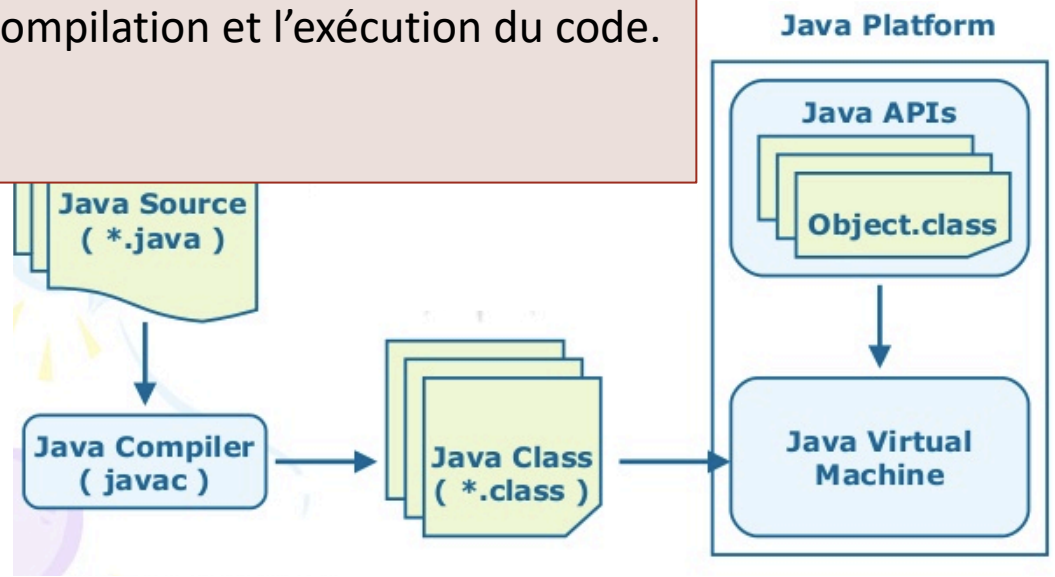
- Contient en plus d'un JRE, un compilateur java (javac).



Environnement de développement

Notepad++, gedit, Sublime Text : éditeurs pour écrire du code Java.

La ligne de commande (i.e. la console) pour la compilation et l'exécution du code. (***javac*** et ***java***)



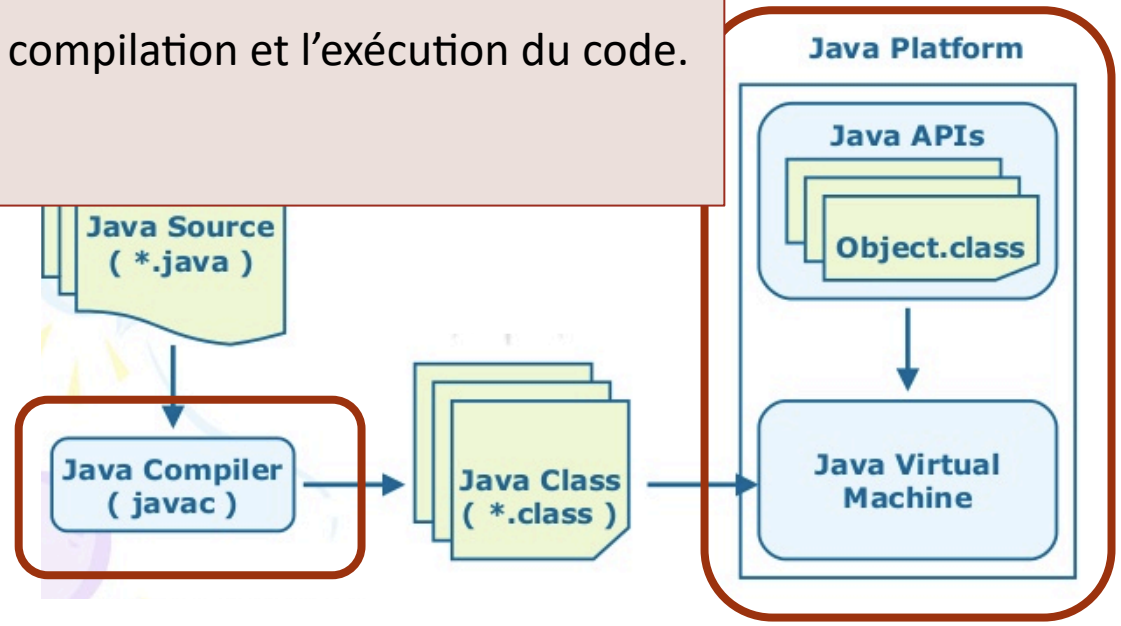
Environnement de développement

Notepad++, gedit, Sublime Text : éditeurs pour écrire du code Java.

La ligne de commande (i.e. la console) pour la compilation et l'exécution du code. (***javac*** et ***java***)

+ JDK

https://www.oracle.com/technetwork/java/javase/downloads/jdk13-downloads-5672538.html



Exercice d'application : Exemple de compilation

- Il faut savoir qu'en Java:
  - Le code source s'écrit dans une structure qu'on appelle classe;
  - Un fichier source a pour extension **.java**;
  - Un fichier source doit avoir le même nom que la classe qu'il contient;
  - Tous les programmes Java sont composés d'au moins une classe qui doit contenir une méthode appelée **main** (ce sera le point de démarrage de notre programme).
  - Toute instruction se termine par un **point-virgule « ; »**.

Exercice d'application : Exemple de compilation

- Soit le code suivant:

```
public class Bonjour {

    public static void main(String[]args) {

        System.out.println("Bonjour le monde!");

    }

}
```

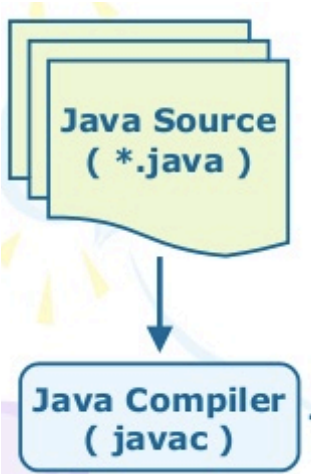
Exercice d’application : Exemple de compilation

- Soit le code suivant:

Bonjour.java

```
public class Bonjour {

    public static void main(String[]args) {

        System.out.println(“Bonjour le monde!”);

    }

}
```

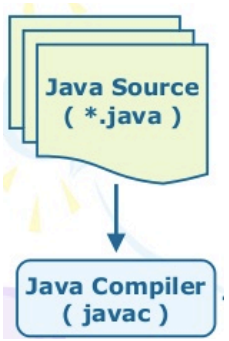
Exercice d’application : Exemple de compilation



Exercice d'application : Exemple de compilation

- Pour compiler:
  - Ouvrez une console;
  - Positionnez-vous dans le répertoire contenant le fichier source;
  - Tapez:

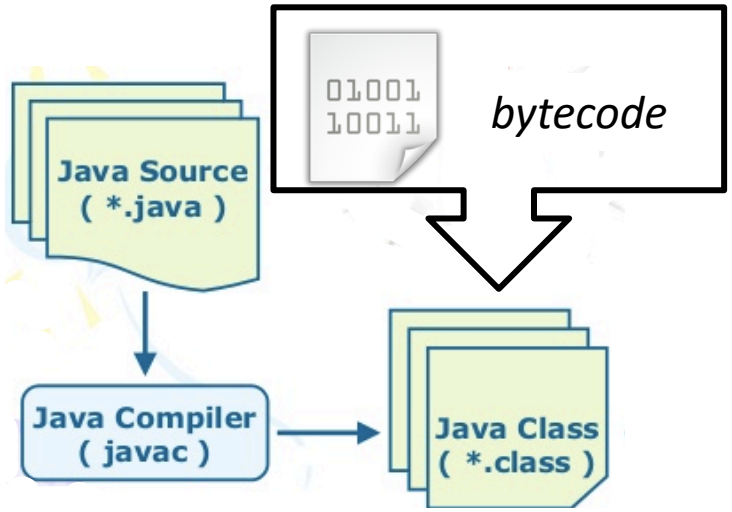
```
> javac Bonjour.java
```



Exercice d'application : Exemple de compilation

- Pour compiler:
  - Ouvrez une console;
  - Positionnez-vous dans le répertoire contenant le fichier source;
  - Tapez:

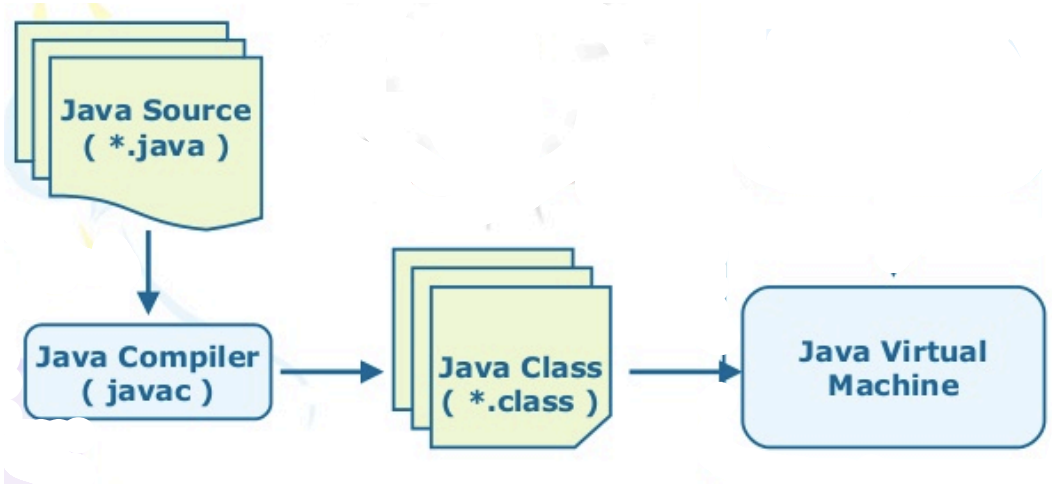
```
> javac Bonjour.java
```



Exercice d’application : Exemple de compilation

- Pour éxecuter:

```
> java Bonjour
Bonjour le monde!
```



Éléments de syntaxe

Éléments de syntaxe

Conventions de codage

- Un nom de classe ou interface doit commencer par une majuscule:
  - Ex: classe **P**oint, class **M**aster**H**andi, class **B**onjour
- Un nom d'attribut doit commencer par une minuscule:
  - Ex: **l**oop**C**ounter, x, y, **a**ge
- Une constante s'écrit en majuscules:
  - Ex: PI, ERROR_MESSAGE

- JAVA est sensible à la casse.
- Ne pas utiliser les noms réservés comme noms de variables ou noms de classes.

Éléments de syntaxe

Liste de mots réservés (*Keywords*)

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

Littéraux :

true, false, et null

Éléments de syntaxe

Commentaires

- Les commentaires améliorent la lisibilité et la compréhension du programme:
  - Les commentaires ne sont pas considérées par l'ordinateur comme des instruction à exécuter

```
// Un commentaire sur une seule ligne

/*    Un commentaire
     qui s'étend
     sur plusieurs lignes*/

/**
  * commentaire de la methode – commentaire de documentation automatique
  * @param val la valeur à traiter
  * @since 1.0
  * @return la valeur de retour
  * @deprecated Utiliser la nouvelle methode XXX
  */
```

Éléments de syntaxe

Les identificateurs en Java

- Un identificateur est une suite de caractères parmi:
  - les lettres (minuscule ou majuscule);
  - les chiffres;
  - le « blanc souligné » (_);
  - …

```
int x;

int y = 2;

int age = 15;

int qtEtudiants;

int qtEtudiantsM1;
```

- Un identificateur ne peut pas commencer par un chiffre (après la premier caractère, on peut mettre autant de chiffre qu'on veut);

Éléments de syntaxe

Type de données

Types de données

- Java est un langage *typé;*

Types de données

- Java est un langage *typé;*
- Pour chaque type:
  - Un code spécifique
  - Un ensemble d'opération – en fonction du type de variable
  - Un intervalle de valeurs possibles – en fonction du codage utilisé

Éléments de syntaxe

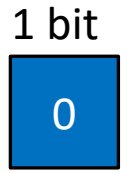
Type de données

- Une variable est un élément qui stocke des informations de toute sorte en mémoire;
- Les types de variables en Java sont répartis en deux catégories:
  - des variables de type simple ou « primitif »;
  - des variables de type structurés.
- Déclaration d'une variable:
  - `<type de la variable> <nom de la variable>`

```
int x;

int y = 2;

int age = 15;

int qtEtudiants;

int qtEtudiantsM1;
```

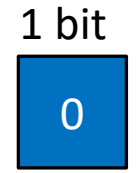
Bits et octets

- Bit – chiffre binaire
  - L'unité la plus simples utilisé par l'ordinateur;
  - Il ne peut être qu'un 0 ou un 1 (faux ou vrai).

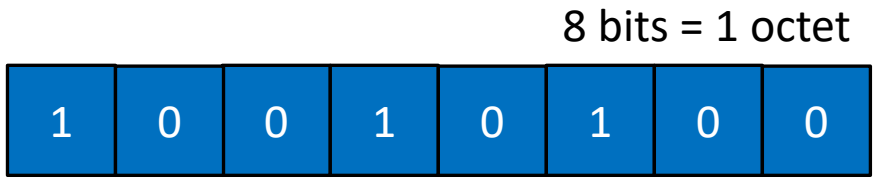


Bits et octets

- Bit – chiffre binaire
  - L'unité la plus simples utilisé par l'ordinateur;
  - Il ne peut être qu'un 0 ou un 1 (faux ou vrai).



- Octet – regroupement de 8 bits
  - 00000000 ou encore 11111111 – et toutes les valeurs intermédiaires entre ces deux extrêmes (10100110, 11001011…);
  - Il peut prendre 2^8 valeurs différentes (256 = [-128,127]).



Types primitifs

```
char monChar;

int monEntier;

double nbEtoiles;
```

Types primitifs - Booléen

	type	défaut	bits
Booléens	`boolean`	false	1

Opérateurs		
&& (et)	|| (ou)	! (non)

```
boolean present = true;
```

Types primitifs - Numériques entiers

	type	défaut	bits
Octets	`byte`	0	8 bits (1 octet)
Entiers courts	`short`	0	16 bits (2 octets)
Entiers	`int`	0	32 bits (4 octets)
Entiers longs	`long`	0	64 bits (8 octets)
Opérateurs arithmétiques			
- Opposé	+ Addition	- Soustraction	
* Multiplication	/ Division	% Module	
Opérateurs relationnels			
< Inférieur	<= Inférieur ou égal	== Égal	
!= Différent	> Supérieur	>= Supérieur ou égal	

```
byte unOctet; int populationFrance; long nbEtoiles;
```

Types primitifs - Numériques réels

	type	défaut	bits
Double	`double`	0	64 bits (8 octets)
Simple	`float`	0	32 bits (4 octets)
Opérateurs arithmétiques			
- Opposé	+ Addition	- Soustraction	
* Multiplication	/ Division	% Module	
Opérateurs relationnels			
< Inférieur	<= Inférieur ou égal	== Égal	
!= Différent	> Supérieur	>= Supérieur ou égal	

```
float distance = 123.45; double surface = 1e-4 (1*10^-4 = 0.0001);
```

Types primitifs - Caractère

	type	**défaut**	**bits**
Caractères	char	0	16 bits (2 octets)

Opérateurs relationnels		
< Inférieur	<= Inférieur ou égal	== Égal
!= Différent	> Supérieur	>= Supérieur ou égal

```
char caractere =‘i’; char exemple2 = ‘2’;
```

Exercices d’application

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

1. **Ecrivez un programme Java pour imprimer la somme de deux nombres.**

   Données de test: 74 + 36

   Exemple sortie:

   110

2. **Ecrivez un programme Java pour diviser deux nombres et imprimer à l'écran.**

   Données de test : 50/3

   Exemple sortie:

   16

Tableaux

- Un tableau est un structure de données qui permet le stockage d'un nombre fini de valeurs de même type.
- Déclaration:

```
<type> [] <identificateur>;
<type> <identificateur> [];
```

```
int [] vecteur;
int vecteur [];
```

Tableaux

- Un tableau est un structure de données qui permet le stockage d'un nombre fini de valeurs de même type.
- Déclaration:
  - Crée une référence sur un tableau de type "double" .

```
<type> [] <identificateur>;          double [] notes;
<type> <identificateur> [];          double notes [];
```

Tableaux

- Création (Instancier le tableau):

```
<identificateur> = new <type> [<taille>];
<type> <identificateur> = {<val1, <val2>…};
```

Tableaux

- Création (Instancier le tableau):
  - Allocation d'autant d'espace mémoire consécutives qu'il est indiqué entre les [];

```
<identificateur> = new <type> [<taille>];
<type> <identificateur> = {<val1, <val2>…};
```

Tableaux

- Création (Instancier le tableau):
  - Allocation d'autant d'espace mémoire consécutives qu'il est indiqué entre les [];

```
<identificateur> = new <type> [<taille>];
<type> <identificateur> = {<val1, <val2>…};
```

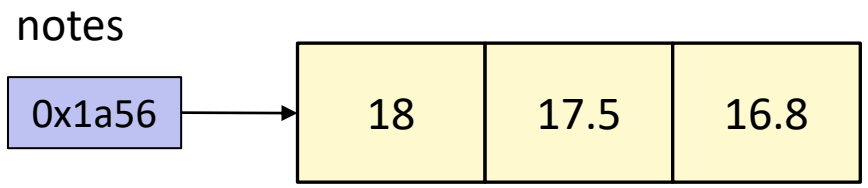
```
double notes [];
notes = new double[3];
notes[0] = 18;
notes[1] = 17.5;
notes[2] = 16.8;
```

Tableaux – Types primitif

```
double[] notes = {18, 17.5, 16.8};
```

Tableaux – Types primitif

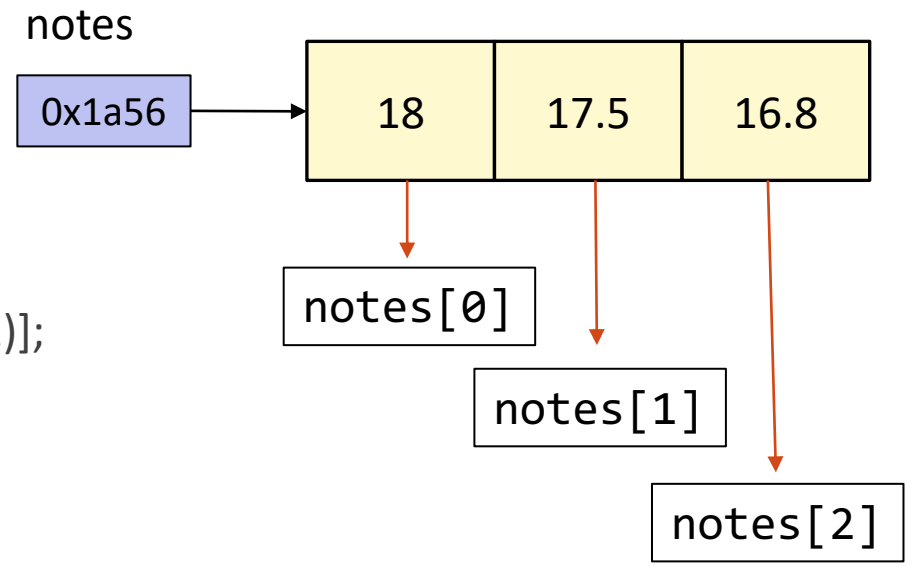
```
double[] notes = {18, 17.5, 16.8};
```



Tableaux – Manipulation et accès

```
double[] notes = {18, 17.5, 16.8};
```

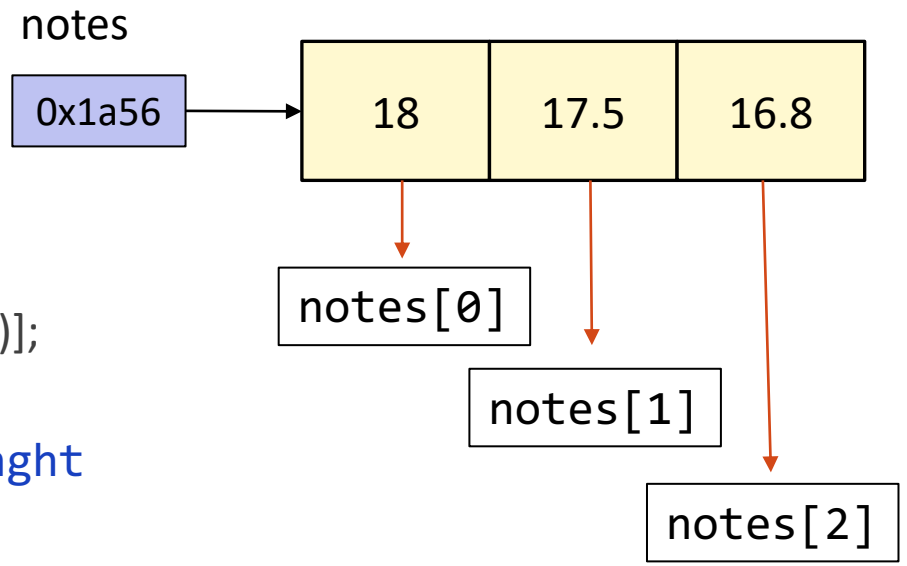
- Chaque case constituant un élément du tableau peut être manipuler comme tout autre variable:
  - Attribuer un valeur dans une case du tableau;
  - Afficher un élément du tableau;
  - Utiliser un élément du tableau dans les opérations.
- La numérotation des indices est toujours de [0 à (<taille>-1)];



Tableaux – Manipulation et accès

```
double[] notes = {18, 17.5, 16.8};
```

- Chaque case constituant un élément du tableau peut être manipuler comme tout autre variable:
  - Attribuer un valeur dans une case du tableau;
  - Afficher un élément du tableau;
  - Utiliser un élément du tableau dans les opérations.
- La numérotation des indices est toujours de [0 à (<taille>-1)];
- Pour savoir la taille d'un tableau il y l'attribut *readonly* `lenght`



Chaînes de caractères

- Une variable chaîne de caractères et une suite de caractères dénotée entre guillemets:

```
System.out.println(“Fait-il beau?”);
```

```
String question = “Fait-il beau?”;
System.out.println(question);
```

Chaînes de caractères

- La classe `String`
  - Est une classe prédéfinie de Java;

Chaînes de caractères

- La classe `String`
  - Est une classe prédéfinie de Java;
  - Elle contient plusieurs méthodes pour l'utilisation des chaînes de caractères;

Chaînes de caractères

- La classe `String`
  - Est une classe prédéfinie de Java;
  - Elle contient plusieurs méthodes pour l'utilisation des chaînes de caractères;
  - Les objets de type `String` sont des chaînes constantes:
    - Une fois créés, elles ne peuvent plus être modifiées

Chaînes de caractères

Quelques méthodes de la classe `String`

- `length()`: Retourne la taille de la chaîne;

```
String question = "Fait-il beau?";
System.out.println(question.length());
```

https://docs.oracle.com/en/java/javase/13/docs/api/java.base/java/lang/String.html

Chaînes de caractères

Quelques méthodes de la classe `String`

- `length()`: Retourne la taille de la chaîne;

```
String question = "Fait-il beau?";
System.out.println(question.length());
```

- `substring()`: Extrait une sous-chaîne d'un mot;

https://docs.oracle.com/en/java/javase/13/docs/api/java.base/java/lang/String.html

Chaînes de caractères

Quelques méthodes de la classe `String`

- `length()`: Retourne la taille de la chaîne;

```
String question = "Fait-il beau?";
System.out.println(question.length());
```

- `substring()`: Extrait une sous-chaîne d'un mot;

- `charAt(int i)`: Retourne placé à la position spécifiée;

https://docs.oracle.com/en/java/javase/13/docs/api/java.base/java/lang/String.html