Cours JAVA - FMS

Initiation au langage JAVA

Par

Stéphane MASCARON
Architecte Logiciels Libres
http://mascaron.net

Cours JAVA - FMS

- Historique du langage
  - On peut faire remonter la naissance de Java à 1991.
    - Les ingénieurs de chez SUN ont cherché à concevoir un langage applicable à de petits appareils électriques
    - Reprenant le concept de machine virtuelle déjà exploité auparavant par le Pascal UCSD
    - Le programme source est d'abord traduit en « byte code » et non en code machine. Plus portable.

Cours JAVA - FMS

- En fait, ce projet de langage pour code embarqué n'a pas abouti.
  - Mais ces concepts ont été repris en 1995 dans la réalisation du logiciel HotJava, un navigateur Web écrit par SUN en Java.
  - Il était capable d'exécuter des applets écrits précisément en byte codes.
  - Les autres navigateurs Web ont suivi.
  - Cela a contribué à l'essor du langage.

Cours JAVA - FMS

- Les versions ce sont succédées après 1995, faut un dico ! :
  - 1.01 et 1.02 en 1996
  - 1.1 en 98
  - 1.2 (rebaptisée Java 2) en 1999
  - 1.3 en 2000,
  - 1.4 en 2002,
  - 5.0 en 2004 (toujours appelées Java 2)

NB : Ainsi on dit : J2SE 1.4 (Java 2 Standard Edition 1.4) basée sur le JDK 1.4 (Java Development Kit 1.4), plus récemment du J2SE5.0 (JDK 5.0) ou encore de Java 5

  - Les dernières versions depuis JSE 6 (le 2 a disparu !), ou plus simplement Java 6, puis Java 7 et Java 8 qui est sorti mi-mars 2014
  - Enfin le 21 Septembre 2017 la sortie de Java 9 (JSE 9)

Cours JAVA - FMS

- Les Bases du langage :
  - Premier exemple de programme Java :

```
public class PremProg {
    public static void main (String args[]) {
        System.out.println ("Mon premier programme Java") ;
    }
}
```

  - Utiliser un éditeur de code pour taper ce petit programme nous permettant de tester notre environnement Java.
  - Ensuite dans un terminal compilez via javac :

```
$ javac PremProg.java
$ ls -l
-rw-rw-r-- 1 stephane stephane  436 sept.   3 01:12 PremProg.class
-rw-rw-r-- 1 stephane stephane  137 sept.   3 01:12 PremProg.java
```

Cours JAVA - FMS

- Structure générale d'un programme Java :

```
public class <NomClasse> {
    ...
}
```

- Elle correspond théoriquement à la définition d'une classe nommée « NomClasse »
- Elle est suivie d'un bloc, c'est-à-dire d'instructions délimitées par des accolades { et } qui définissent le contenu de cette classe.
- La seule définition ici est celle d'une "méthode" particulière nommée « main »

Cours JAVA - FMS

- Informations complémentaires
  - Notre programme comporte ici une seule instruction :
    - `System.out.println(''Mon premier programme Java'') ;`

      Ce code permet décrire sur la sortie standard via la fonction « println »
    - Nous verrons lors de l'étude des Classes les mots clés réservés « statitc », « void ».

Cours JAVA - FMS

- Strcuture d'une application Java
  - Une application Java sera un ensemble de classes dont au moins une contiendra la méthode :

    « `public static void main(String args[]){}` »
  - Cette fonction instaciera une classe que l'on pourra comparer à un contrôleur.
  - Ce contrôleur instaciera l'interface utilisateur
  - L'interface détectera les actions utilisateur pour charger les classe contenant les traitements.

Cours JAVA - FMS

- Les caractéristiques de Java
  - Java est interprété : nous l'avons vu, à la compilation c'est un pseudo code appelé « byte codes » qui est généré d'où le slogan : « Write Once, Run Anywhere »
  - Java est portable : Le « byte codes » reste indépendant de la machine sur laquelle il s'exécute. Il faut juste une JVM.
  - Java est orienté Objet : Chaque fichier source contient la définition d'une ou plusieurs classes qui sont utilisées les unes avec les autres pour former une application.

Cours JAVA - FMS

- Les caractéristiques de Java (suite)
  - Java est simple : Le choix des auteurs a été d'abandonner les éléments mal exploités des autres langages (pointeurs, l'héritage multiple et la surcharge des opérateurs).
  - Java est fortement typé : toutes les variables sont typées et il n'existe pas de conversion automatique qui risquerait une perte de données. Si une telle conversion doit être réalisée, le développeur doit obligatoirement utiliser un cast ou une méthode statique fournie en standard pour la réaliser.

Cours JAVA - FMS

- Les caractéristiques de Java (suite)
  - Java assure la gestion de la mémoire : L'allocation de la mémoire pour un objet est automatique à sa création et Java récupère automatiquement la mémoire inutilisée grâce au garbage collector qui restitue les zones de mémoire laissées libres suite à la destruction des objets.
  - Java est sûr : Un programme Java planté ne menace pas le système d'exploitation. Il ne peut pas y avoir d'accès direct à la mémoire. (Applet sécurisée)

Cours JAVA - FMS

- Les caractéristiques de Java (suite)
  - Java est économe : le pseudo code a une taille relativement petite car les bibliothèques de classes requises ne sont liées qu'à l'exécution.
  - Java est multitâche : il permet l'utilisation de threads qui sont des unités d'exécutions isolées. La JVM, elle même, utilise plusieurs threads.
  - Java est intrinsèquement réseau : il possède un ensemble de classe dans le package « java.net »

Cours JAVA - FMS

- Les règles syntaxique de base
  - Java est sensible à la casse ;
  - Les blocs de code sont encadrés par des accolades ;
  - Chaque instruction se termine par un caractère ';' (point virgule) ;
  - Une instruction peut tenir sur plusieurs lignes ;
  - L'indentation est ignorée du compilateur mais elle permet une meilleure compréhension du code

Cours JAVA - FMS

- Les mots réservés du langage Java

abstract	continue	for	new	switch
assert	default	goto	package	synchronized
boolean	do	do	private	this
break	double	implements	protected	throw
byte	else	import	public	throws
case	enum (Java 5)	instanceof	return	transient
catch	extends	int	short	try
char	final	interface	static	void
class	finally	long	strictfp	volatile
const*	float	native	super	while

Cours JAVA - FMS

- Les identificateurs
  - Chaque objet, classe, programme ou variable est associé à un nom : l'identificateur qui peut se composer de tous les caractères alphanumériques et des caractères _ et $. Le premier caractère doit être une lettre, le caractère de soulignement ou le signe dollar.
  - Un identificateur ne peut pas appartenir à la liste des mots réservés du langage Java ni correspondre aux littéraux true, false et null.

Cours JAVA - FMS

- Les commentaires
  - Ils ne sont pas pris en compte par le compilateur donc ils ne sont pas inclus dans le pseudo code. Ils ne se terminent pas par un caractère ";".
  - Il existe trois types de commentaire en Java :

Type de commentaires	Exemple
Commentaire abrégé	`// commentaire sur une seule ligne` `int N=1 ; // initialisation compteur`
Commentaire multiligne	`/* commentaire ligne 1` `commentaire ligne 2 */`

Commentaire de documentation automatique (javadoc)	`/** commentaire de la méthode`
	`* @param val la valeur à traiter`
	`* @since 1.0`
	`* @return la valeur de retour`
	`* @deprecated Utiliser la nouvelle méthode XXX`
	`*/`

Cours JAVA - FMS

- La déclaration et l'utilisation de variables
  - La déclaration de variables : Une variable possède un nom, un type et une valeur. La déclaration d'une variable doit donc contenir deux choses : un nom et le type de données qu'elle peut contenir.
  - Une variable est utilisable dans le bloc où elle est définie.
  - La déclaration d'une variable permet de réserver la mémoire pour en stocker la valeur.
  - Il est possible de définir plusieurs variables de même type en séparant chacune d'elles par une virgule.
  - Le type d'une variable peut être :
    - soit un type élémentaire dit aussi type primitif :
      ```
      int note ;
      ```
    - soit une classe
      ```
      String nom;
      ```

Cours JAVA - FMS

- La déclaration et l'utilisation de variables (suite)
  - Pour les objets, il est nécessaire en plus de la déclaration de la variable de créer un objet avant de pouvoir l'utiliser.
    - Il faut réserver de la mémoire pour la création d'un objet avec l'instruction new.
    - La libération de la mémoire se fait automatiquement grâce au garbage collector.

Exemple

```
MaClasse instance ; // déclaration de l'objet
Instance = new MaClasse() ; // création de l'objet,
réservation en mémoire
//ou
MaClasse instance = new MaClasse(); // déclaration et création
```

Cours JAVA - FMS

- La déclaration et l'utilisation de variables (suite)

Exemple

```
int[] nombres = new int[10] ; // déclaration tableau

int i=3, j=4 ;  // déclaration et affectation sur une ligne.
```

- Les tableau se déclare avec les crochets après le type et dans le constructeur du type prédéfinie ou de la classe.
- Il est possible de déclarer et d'affecter les variables en série sur une ligne.

Cours JAVA - FMS

- Les types élémentaires
  - Les types élémentaires ont une taille identique quelque soit la plate-forme d'exécution :

Type	Désignation	Longueur	Valeurs	Commentaire
boolean	Valeur logique	1 bit	true ou false	Pas de conversion
byte	octet signé	8 bits	-128 à 127	
short	entier court signé	16 bits	-32768 à 32767	
char	caractère Unicode	16 bits	\u0000 à \uFFFF	Entouré de cotes simples dans du code
int	entier signé	32 bits	-2147483648 à 2147483647	
float	nombre à virgule flottante simple précision	32 bits	1.401e-045 à 3.40282e+038	
double	Nombre à virgule flottante double précision	64 bits	2.22507e-308 à 1.79769e+308	
long	entier long	64 bits	-9223372036854775808 à 9223372036854775807	

Cours JAVA - FMS

- Le format des types élémentaires
  - Le format des nombres entiers : Il existe plusieurs formats pour les nombres entiers : les types **byte**, **short**, **int** et **long** peuvent être codés en décimal, hexadécimal ou octal.
    - Pour un nombre hexadécimal, il suffit de préfixer sa valeur par **0x**.
    - Pour un nombre octal, le nombre doit commencer par un **zéro**.
    - Le suffixe **l** ou **L** permet de spécifier que c'est un entier long.

Cours JAVA - FMS

- Le format des types élémentaires (suite)
  - Le format des nombres décimaux : Il existe plusieurs formats pour les nombres décimaux. Les types **float** et **double** stockent des nombres flottants : pour être reconnus comme tels ils doivent posséder soit : un point, un exposant ou l'un des suffixes f, F, d, D.
  - Il est possible de préciser des nombre qui n'ont pas de partie entière ou pas de partie décimale.

Exemple

```
float pi = 3.141f; double valeur = 3d;
float flottant1 = +.1f , flottant2 = 1e10f;
System.out.println("flottant1 : " + flottant1);
System.out.println("flottant2 : " + (long) flottant2);
```

Cours JAVA - FMS

- Le format des types élémentaires (suite)
  - Le format des caractères :
    - Un caractère est codé sur 16 bits car il est conforme à la norme Unicode.
    - Il doit être entouré par des apostrophes (‘c’)
    - Une valeur de type **char** peut être considérée comme un entier non négatif de 0 à 65535

Exemple

```
char code = 'D';
int index = code - 'A';
System.out.println("index = " + index);  // affiche 3
```

Cours JAVA - FMS

- L'initialisation des variables

Exemple

```
int nombre; // déclaration
nombre = 100; //initialisation
//Ou sur la même ligne
int nombre = 100; //déclaration et initialisation
```

  - En Java, toute variable appartenant à un objet (définie comme étant un attribut de l'objet) est initialisée avec une valeur par défaut en accord avec son type au moment de la création.
  - Cette initialisation ne s'applique pas aux variables locales des méthodes de la classe.

Cours JAVA - FMS

- L'initialisation des variables (suite)
  - Les valeurs par défaut lors de l'initialisation automatique des variables d'instances sont :

Type	Valeur par défaut
boolean	false
byte, short, int, long	0
float, double	0.0
char	\u000
classe	null

  - Remarque : Dans une applet, il est préférable de faire les déclarations et initialisations dans la méthode init().

Cours JAVA - FMS

- L'affectation
  - le signe = est l'opérateur d'affectation et s'utilise avec une expression de la forme :

    `variable = expression.`

  - L'opération d'affectation est associative de droite à gauche : il renvoie la valeur affectée ce qui permet d'écrire : **`x = y = z = 0 ;`**
  - Il existe des opérateurs qui permettent de simplifier l'écriture d'une opération d'affectation associée à un opérateur mathématique : (cf. tableau slide suivante)

Cours JAVA - FMS

- Opérateurs simplificateurs

Opérateur	Exemple	Signification
=	a = 10	Équival. à : a = 10
+=	a += 10	Équival. à : a = a + 10
-=	a -= 10	Équival. à : a = a - 10
*=	a *= 10	Équival. à : a = a * 10
/=	a /= 10	Équival. à : a = a / 10
%=	a %= 10	Équival. à : a = a % 10 (reste de la division entière)
^=	a ^= 10	Équival. à : a = a ^ 10
<<=	a <<= 10	a = a << 10 a est complété par des zéros à droite
>>=	a >>= 10	a = a >> 10 a est complété par des zéros à gauche
>>>=	a >>>= 10	a = a >>> 10 décalage à gauche non signé

Cours JAVA - FMS

- Opérateurs simplificateurs (suite)
  - Remarque :
    - Attention : Lors d'une opération sur des opérandes de types différents, le compilateur détermine le type du **résultat en prenant le type le plus précis des opérandes**. Par exemple,
      - une multiplication d'une variable de type float avec une variable de type double donne un résultat de type double.
      - Lors d'une opération entre un opérande entier et un flottant, le résultat est du type de l'opérande flottant.

Cours JAVA - FMS

- Les comparaisons

Opérateur	Exemple	Signification
>	a > 10	strictement supérieur
<	a < 10	strictement inférieur
>=	a >= 10	supérieur ou égal
<=	a <= 10	inférieur ou égal
==	a == 10	Egalité
!=	a != 10	diffèrent de
&	a & b	ET binaire
^	a ^ b	OU exclusif binaire
|	a | b	OU binaire
&&	a && b	ET logique (expression booléenne)
||	a || b	OU logique (expression booléenne)
? :	a ? b : c	opérateur conditionnel retourne b ou c en fct de a

Cours JAVA - FMS

- Priorité des opérateurs
  - Les opérateurs sont exécutés dans l'ordre suivant à l'intérieur d'une expression qui est analysée de gauche à droite, l'usage des parenthèses permet de modifier cet ordre de priorité :
    - incréments et décréments
    - multiplication, division et reste de division (modulo)
    - addition et soustraction
    - comparaison
    - le signe = d'affectation d'une valeur à une variable

Cours JAVA - FMS

- Les opérations arithmétiques
  - Les opérateurs arithmétiques se notent + (addition), - (soustraction), * (multiplication), / (division) et % (reste de la division). Ils peuvent se combiner à l'opérateur d'affectation comme on vient de la voir.
  - L'arithmétique entière : Pour les types numériques entiers, Java met en oeuvre une sorte de mécanisme de conversion implicite vers le type int appelé **promotion entière**.

Exemple

```
short x= 5 , y = 15;
x = x + y ; //erreur à la compilation
Incompatible type for =. Explicit cast needed to convert int to short.
x = x + y ; //erreur à la compilation
^
1 error
```

Cours JAVA - FMS

- Les opérations arithmétiques (suite)
  - Dans l'exemple précédent : Les opérandes et le résultat de l'opération sont convertis en type **int**. Le résultat est affecté dans un type **short** : il y a donc risque de perte d'informations et donc une erreur est émise à la compilation.
  - Cette promotion évite un débordement de capacité sans que le programmeur soit pleinement conscient du risque.
  - il est nécessaire, pour régler le problème, d'utiliser une conversion explicite ou cast.

Exemple

```
short x= 5 , y = 15;
x = (short) (x + y) ; //cast le type int en short
```

Cours JAVA - FMS

- L'incrémentation et la décrémentation
  - Les opérateurs d'incrémentation et de décrémentation sont :

    `n++ ; ++n ; n-- ; --n`

  - Si l'opérateur est placé avant la variable (préfixé), la modification de la valeur est immédiate
  - sinon la modification n'a lieu qu'à l'issue de l'exécution de la ligne d'instruction (postfixé)

Exemple

```
System.out.println(x++); // est équivalent à
System.out.println(x); x = x + 1;

System.out.println(++x); // est équivalent à
x = x + 1; System.out.println(x);
```

Cours JAVA - FMS

- Les structures de contrôles
  - Comme la quasi-totalité des langages de développement orientés objets, Java propose un ensemble d'instructions qui permettent d'organiser et de structurer les traitements.
  - L'usage de ces instructions est similaire à celui rencontré avec leur équivalent dans d'autres langages.

Cours JAVA - FMS

- Les structures de contrôles (suite)
  - Les boucles
    - While :

Exemple

```
while ( boolean )
{
... // code à exécuter dans la boucle
}
```

Le code est exécuté tant que le booléen est vrai. Si avant l'instruction while, le booléen est faux, alors le code de la boucle ne sera jamais exécuté

Ne pas mettre de « ; » après la condition sinon le corps de la boucle ne sera jamais exécuté

Cours JAVA - FMS

- Les structures de contrôles (suite)
  - Les boucles
    - do … while() :

Exemple

```
do {
...
} while ( boolean );
```

Cette boucle est au moins exécutée une fois quelle que soit la valeur du booléen;

Cours JAVA - FMS

- Les structures de contrôles (suite)
  - Les boucles
    - **for** (initialisation; condition; modification)

Exemple

```
for (i = 0 ; i < 10; i++ ) { .…}

for (int i = 0 ; i < 10; i++ ) { .…}

for ( ; ; ) { ... } // boucle infinie
```

L'initialisation, la condition et la modification de l'index sont optionnelles.

- Dans l'initialisation, on peut déclarer une variable qui servira d'index et qui sera dans ce cas locale à la boucle.

Cours JAVA - FMS

- Les structures de contrôles (suite)
  - Les boucles
    - **for** (initialisation; condition; modification)

Il est possible d'inclure plusieurs traitements dans l'initialisation et la modification de la boucle : chacun des traitements doit être séparé par une virgule.

Exemple

```
for (i = 0 , j = 0 ; i * j < 1000;i++ , j+= 2) { ....}
```

La condition peut ne pas porter sur l'index de la boucle.

Cours JAVA - FMS

- Les branchements conditionnels
  - If (boolean) { ...} else { } :

Structure

```
if (boolean) {
...
} else if (boolean) {...} else {...}
```

  - Switch : Avant Java 7, on ne peut utiliser switch qu'avec des types primitifs de 32 bits max.

Structure

```
switch (expression) {
    case constante1 :
      instr11;
      Instr12;
      break;
    ...
    default : ...
}
```

Cours JAVA - FMS

- Les branchements conditionnels (suite)
  - Switch (suite)

A partir de Java 7, il est possible d'utiliser un type String avec l'instruction switch.

Si une instruction case ne contient pas de break alors les traitements associés au case suivant sont exécutés.

Il est possible d'imbriquer des switch (ouch!)

Cours JAVA - FMS

- Les branchements conditionnels (suite)
  - L'opérateur ternaire :

(condition) ? valeur-vrai : valeur-faux

Exemple

```
if (niveau == 5) total = 10;
else total = 5 ;

// equivalent à :
total = (niveau == 5) ? 10 : 5;

// dans une sortie standard
System.out.println((sexe == " H ") ? " Mr " : " Mme ");
```

Cours JAVA - FMS

- Les débranchements :
  - break : permet de quitter immédiatement une boucle ou un branchement. Utilisable dans tous les contrôles de flot
  - continue : s'utilise dans une boucle pour passer directement à l'itération suivante

break et **continue** peuvent s'exécuter avec des blocs nommés. Il est possible de préciser une **étiquette** pour indiquer le point de retour lors de la fin du traitement déclenché par le break.

Une étiquette est un nom suivi du caractère deux-points qui définit le début d'une instruction.

Cours JAVA - FMS

- Les tableaux :
  - Ce sont des objets : ils sont donc dérivés de la classe Object. Il est possible d'utiliser les méthodes héritées telles que equals() ou getClass().
  - Le premier élément d'un tableau possède l'indice 0.
  - La déclaration des tableaux :
    - Java permet de placer les crochets après ou avant le nom du tableau dans la déclaration.

Exemple

```
int tableau[] = new int[50]; // déclaration et allocation
OU int[] tableau = new int[50];

OU int tab[]; // déclaration
tab = new int[50]; //allocation
```

Cours JAVA - FMS

- Les tableaux (suite) :
  - Java ne supporte pas directement les tableaux à plusieurs dimensions : il faut déclarer un tableau de tableau.

Exemple

```
float[][] tableau = new float[10][10];
```

  - La taille des tableaux de la seconde dimension peut ne pas être identique pour chaque occurrence.

Exemple

```
int dim1[][] = new int[3][];
dim1[0] = new int[4];
dim1[1] = new int[9];
dim1[2] = new int[2];
```

  - Chaque élément du tableau est initialisé selon son type par l'instruction new : 0 pour les numériques, '\0' pour les caractères, false pour les booléens et null pour les chaines de caractères et les autres objets.

Cours JAVA - FMS

- Les tableaux (suite) :
  - L'initialisation explicite d'un tableau :

Exemple

```
int tableau[5] = {10,20,30,40,50};
int tableau[3][2] = {{5,1},{6,2},{7,3}};
```

  - La taille du tableau n'est pas obligatoire si le tableau est initialisé à sa création :

Exemple

```
int tableau[] = {10,20,30,40,50};
```

  - Le nombre d'éléments de chaque ligne peut ne pas être identique :

Exemple

```
int[][] tabEntiers = {{1,2,3,4,5,6},{1,2,3,4},{1,2,3,4,5,6,7,8,9}};
```

Cours JAVA - FMS

- Le parcours d'un tableau

Exemple

```
for (int i = 0; i < tableau.length ; i ++) { ... }
```

  - La variable length retourne le nombre d'éléments du tableau.
  - Pour passer un tableau à une méthode, il suffit de déclarer les paramètres dans l'en-tête de la méthode

Exemple

```
public void printArray(String texte[]){ ...}
```

  - Les tableaux sont toujours transmis par référence puisque ce sont des objets.
  - Un accès a un élément d'un tableau qui dépasse sa capacité, lève une exception du type **java.lang.arrayIndexOutOfBoundsException**.

Cours JAVA - FMS

- Les conversions de types
  - Lors de la déclaration, il est possible d'utiliser un cast :

Exemple

```
int entier = 5;
float flottant = (float) entier;
```

  - La conversion peut entrainer une perte d'informations.
  - Il n'existe pas en Java de fonction pour convertir : les conversions de type se font par des méthodes.
  - La bibliothèque de classes API fournit une série de classes dites « **classes enveloppes** » qui contiennent des méthodes de manipulation et de conversion de types élémentaires.

Cours JAVA - FMS

- Liste des classes enveloppes

Classe	Rôle
String	Pour les chaînes de caractères Unicode
Integer	Pour les valeurs entières (integer)
Long	Pour les entiers longs signés (long)
Float	Pour les nombres à virgule flottante (float)
Double	pour les nombres à virgule flottante en double précision (double)

- Les classes portent le même nom que le type élémentaire sur lequel elles reposent avec la première lettre en majuscule.

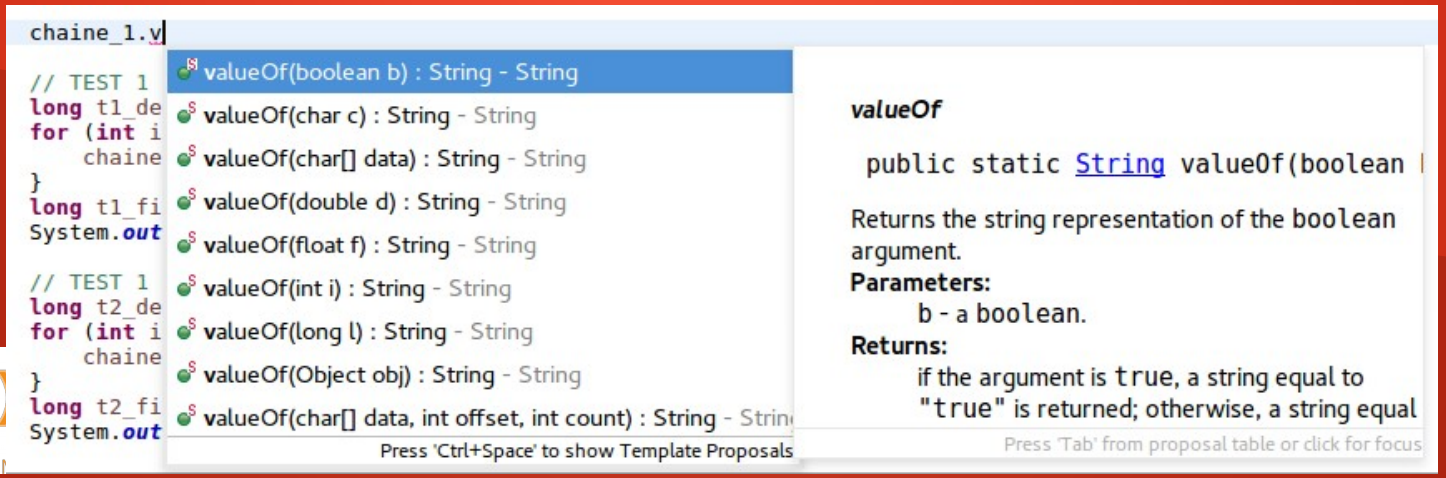
Cours JAVA - FMS

- Liste des classes enveloppes (suite)
  - Ces classes contiennent généralement plusieurs constructeurs. Pour y accéder, il faut les instancier puisque ce sont des objets.

Exemple

```
String chaine_1 ;
chaine_1 = new String("test");
```

  - L'obje chaine_1 permet d'accéder aux méthodes de la classe **java.lang.String**



Cours JAVA - FMS

- Exercices de conversion :
  - Ecrire une programme Java dans une méthode main qui réalise les conversions suivantes :

    1) Un entier **int** en chaîne **String**

    2) Une chaîne **String** en entier **int**

    3) Un entier **int** en entier **long**

  25 minutes, écrire le code dans un éditeur de texte de base (gedit, ...)

Cours JAVA - FMS

- Correction exercices de conversion

Exercice 1 : Un entier **int** en chaîne **String**

```
int i = 10;
String montexte = new String();
Montexte = montexte.valueOf(i);
```

Exercice 2 : Une chaîne **String** en entier **int**

```
String montexte = new String("10");
Integer monnombre = new Integer(montexte);
int i = monnombre.intValue(); //conversion d'Integer en int
```

Exercice 3 : Un entier **int** en entier **long**

```
int i=10;
Integer monnombre = new Integer(i);
long j = monnombre.longValue();
```

Cours JAVA - FMS

- Exercice :
  - Ecrire un programme Java nommé « Moyenne » en utilisant comme modèle la classe « PremProg » et sa méthode « main » écrite en début de cours pour calculer la moyenne de notes passez en paramètre lors de l'exécution de la classe.
    - A savoir : les paramètres séparés par un « espace » après le nom de la classe lors de l'appel de la machine virtuelle sont récupérés dans le tableau « args » paramètre de la fonction « main » de la classe.
    - Exemple d'exécution :

```
$ java Moyenne 4 12 14 6 10
La moyenne des notes est : 9.2
```

Cours JAVA - FMS

Exemple de correction exercice calcul de la Moyenne de notes

```
public class Moyenne {
    public static void main(String[] args) {
        float note = 0.0f, somme=0.0f, moyenne=0.0f;
        int nbNote = 0;
        nbNote = args.length;
        if (nbNote == 0) {
            System.out.println("Vous devez passer les notes en paramètre
                                séparées par des espaces");
            System.exit(0);
        }
        else {
            for (int i=0; i< nbNote; i++) {
                Float laNote = new Float(args[i]);
                note = laNote.floatValue();
                if (note < 0  || note > 20) {
                    System.out.println("Les notes doivent être comprises
                                        entre 0 et 20");
                    System.exit(0);
                }
                else somme += note;
            }
        }
        moyenne = somme / nbNote;
        System.out.println("La moyenne des notes est : " + moyenne);
    }
}
```

Cours JAVA - FMS

- La manipulation des chaînes de caractères
  - La définition d'un caractère se fait grâce au type char :

Exemple

```
char touche = '%';
```

  - La définition d'une chaîne se fait grâce à l'objet String :

Exemple

```
String texte = " bonjour ";
```

  - Les variables de type String sont des objets. Partout où des constantes chaînes de caractères figurent entre guillemets, le compilateur Java génère un objet de type String avec le contenu spécifié. Il est donc possible d'écrire :

Exemple

```
String texte = " Java Java Java ".replace('a','o');
```

Cours JAVA - FMS

- La manipulation des chaînes de caractères (suite)
  - Les chaînes de caractères ne sont pas des tableaux : il faut utiliser les méthodes de la classe String d'un objet instancié pour effectuer des manipulations.
  - Il est impossible de modifier le contenu d'un objet de type String. Cependant, il est possible d'utiliser les méthodes de la classe String pour effectuer une modification qui va créer une nouvelle chaîne de caractères.

Cours JAVA - FMS

- Il suffit alors d'affecter le résultat de ces méthodes à la variable pour qu'elle pointe sur la nouvelle chaîne de caractères contenant les modifications.

Exemple

```
String texte = " Java Java Java " ;
texte = texte.replace('a','o');
```

- Java ne fonctionne pas avec le jeu de caractères ASCII ou ANSI, mais avec Unicode (Universal Code). Ceci concerne les types char et les chaînes de caractères. Le jeu de caractères Unicode code un caractère sur plusieurs octets.
- Les caractères 0 à 255 correspondent exactement au jeu de caractères ASCII étendu.

Cours JAVA - FMS

- Les caractères spéciaux dans les chaines
  - Avec le caractère d'échappement \. Le tableau suivant recense ces principaux caractères.

Caractères séciaux	Affichage
\’	‘ Apostrophe
\’’	‘’ Guillemet
\\	\ Antislash
\t	Tabulation
\b	Retour arrière (backspace)
\r	Retour chariot
\f	Saut de page (form feed)
\n	Saut de ligne (new line)
\0ddd	Caractère ASCII ddd (octal)
\xdd	Caractère ASCII dd (hexadécimal)
\udddd	Caractère Unicode dddd (hexadécimal)

Cours JAVA - FMS

- L'addition de chaînes de caractères
  - Java admet l'opérateur « + » comme opérateur de concaténation de chaînes de caractères.
  - L'opérateur « + » permet de concaténer plusieurs chaînes de caractères. Il est possible d'utiliser l'opérateur « += » .

Exemple

```
String texte = " ";
texte += " Hello "; texte += " World3 ";
System.out.println(" texte : " + texte) ;
```

  - Cet opérateur sert aussi à concaténer des chaînes avec tous les types de bases. La variable ou constante est alors convertie en chaîne et ajoutée à la précédente. Il faut au moins une chaîne dans l'expression.

Cours JAVA - FMS

- La comparaison de deux chaines
  - Il est nécessaire d'utiliser la méthode « equals ()» pour comparer 2 chaînes de caractères.

Exemple

```
String texte1 = " texte 1 ";
String texte2 = " texte 2 ";
if ( texte1.equals(texte2) ) { … } else { … }
```

  - Optimisation sur la concaténation,
    - La concaténation avec le signe « + » sur les objets de type **String** n'est pas optimisée. Il est conseillé d'utiliser la classe **StringBuffer** qui permet de réaliser des concaténations plus rapides.
    - Voici, sur le slide suivant, un code à taper dans un éditeur puis compilez-le, et exécutez le programme.

Cours JAVA - FMS

- TP : Optimisation de la concaténation

Exemple : à écrire dans une méthode main (String [] args)

```
// démonstration de non performance de la concaténation de la classe String
public static void main (String[] args) {
  String chaineAppended = "JBKJ ZEhfizu flIUHFBLZEfLGFLsgf SLDIGFBhfFBIhfLZFG
  LJHSDbc lzgczgf jsbczigf iezugf :sjb lzfg zeufg zgf IUGF :ZGF";
  String chaine_1 = "";
  StringBuffer chaine_2 = new StringBuffer("");
  // TEST 1 BOUCLE DE CONCATÉNATION sur String
  long t1_debut = System.currentTimeMillis();
  for (int i = 0; i < 10000; i++) { chaine_1 += chaineAppended; }
  long t1_fin = System.currentTimeMillis();
  System.out.println("concaténation String: " + (t1_fin - t1_debut) + " ms");
  // TEST 2 BOUCLE DE CONCATÉNATION sur StringBuffer
  long t2_debut = System.currentTimeMillis();
  for (int i = 0; i < 10000; i++) { chaine_2.append(chaineAppended); }
  long t2_fin = System.currentTimeMillis();
  System.out.println("concaténation StringBuffer:"+(t2_fin - t2_debut)+" ms");
}
```

Exécution :

```
temps de traitement concaténation String: 7882 ms
temps de traitement concaténation StringBuffer: 1 ms
```

Cours JAVA - FMS

- La détermination de la longueur d'une chaine
  - La méthode length() permet de déterminer la longueur d'une chaine.

Exemple

```
String texte = " texte ";
int longueur = texte.length();
```

- La modification de la casse d'une chaine
  - Les méthodes Java **toUpperCase**() et **toLowerCase**() permettent respectivement d'obtenir une chaîne tout en majuscules ou tout en minuscules.

Exemple

```
String texte = " texte ";
String textemaj = texte.toUpperCase();
```

Cours JAVA - FMS

- Ressources complémentaires
  - Support de cours sur Java :
    - Développons en Java de Jean-Michel Doudoux

http://jmdoudoux.fr/java/dej/dej_2_10.pdf

    - Pensez en Java de Bruce Eckel (Version française

http://bruce-eckel.developpez.com/livres/java/traduction/tij2/

    - Programmation Java pour les enfants, les parents et les grands-parents.

https://java.developpez.com/livres-collaboratifs/javaenfants/
Ce livre est ancien, mais il permet de comprendre les concepts de base et propose des exercices d'apprentissage pour débutant.