JAVA
FONDAMENTAUX

Présenté par :

Xavier TABUTEAU

Les variables

Une variable permet d'identifier une valeur avec un nom. Elle peut s'écrire qu'avec les caractères de l'alphabet en minuscule, majuscule, les chiffres et les caractères « _» et « $ ».
Elle ne peut pas commencer par un chiffre, ni avoir de caractères accentués ou spéciaux.
Par convention, on écrit les variables en camel case.

Remarque : par convention, les constantes s'écrient en majuscule uniquement.

Utilité des variables :

• La déclaration d'une variable sert à créer une référence dans un emplacement en mémoire.

• L' affectation d'une variable sert à lui associer une valeur.

Remarque : Les types de base ne sont pas des objets, ils n'ont aucune méthode.
Pour chaque type de base, il existe une classe correspondante riche en fonctionnalités.

Pour l'affectation d'une variable à un nombre, il est possible d'exprimer le nombre avec des « _ ».

Exemple :

```
int variable = 10_000;      // pour les grandes valeurs.
```

Ou

```
int variable = 01_02_03_04_05;          // pour un numéro de téléphone par exemple.
```

Les types de variables

- Les chaînes de caractères (String) : elles sont utilisées pour représenter du texte. Une chaîne commence et finie par un guillemet double. Depuis le JDK 15, il est possible d'écrire des String multi lignes. Cela s'appel un « TextBlock ». le TextBlock commence et fini par des triples guillemets double.

- Les chiffres (nombre entier, à virgule flottante, etc.) : ils sont utilisés surtout avec des opérateurs mathématiques.

- Les valeurs booléennes (Boolean) : elles sont des valeurs dichotomiques (soit vrai, soit faux).

- Les tableaux : ils sont utilisés pour créer des listes avec des indices qui permettent de récupérer la valeur associée.

- Les objets : ils sont des conteneurs qui peuvent inclure souvent tout type de données, y compris de sous-objets, des variables (propriétés / attributs), ou des fonctions (méthodes).

Les types de variables

Type	Taille	Description	Intervalle
char	2 octets	Une unité de code, suffisant à représenter un grand nombre de point de code, et même un caractère Unicode (UTF-16) 'b' '\u250C'	'\u0000' à '\uFFFF’
byte	1 octet	Un nombre entier de 8 bits signé (soit un octet)	-128 à 127
short	2 octets	Un nombre entier de 16 bits signé	-32 768 à 32 767
int	4 octets	Un nombre entier de 32 bits signé	−2 147 483 648 et +2 147 483 647
long	8 octets	Un nombre entier de 64 bits signé	−9 223 372 036 854 775 808 et +9 223 372 036 854 775 807
float	4 octets	Un nombre à virgule flottante de 32 bits signé (simple précision) 0.0F pas un nombre (Float.NaN)	de 2^{-149} (Float.MIN_VALUE) à $2^{128} - 2^{104}$ (Float.MAX_VALUE) −∞ (Float.NEGATIVE_INFINITY) +∞ (Float.POSITIVE_INFINITY)
double	8 octets	Un nombre à virgule flottante de 2^{-1074} de 64 bits signé (double précision) 0.0D pas un nombre (Double.NaN)	(Double.MIN_VALUE) $2^{1024} - 2^{971}$ Double.MAX_VALUE) −∞ (Double.NEGATIVE_INFINITY) +∞ (Double.POSITIVE_INFINITY)
Boolean	1 octet	Une valeur logique	false (faux) ou true (vrai)

« Boxing » et « Unboxing »

Boxing : Conversion automatique du type primitif en son équivalent objet.

Exemple :

```
int valeurPrimitive = 100;
Integer valeurObjet = valeurPrimitive;
```

Unboxing : Conversion automatique de l'objet en son type primitif.

Exemple :

```
Integer valeurNumerique = 100;
int valeur = valeurNumerique;
```

Conversions de types

Il est possible de convertir un type de variable en un autre.

Exemple de conversion de type entre chaîne de caractères et valeur numérique :

```
Double monDouble = 3.14159;
String pi = Double.toString(monDouble);


String valeurNumerique = "100";
Integer valeur = Integer.valueOf(valeurNumerique);
```

Remarque : il est possible de convertir temporairement une variable, on appel cela le casting.

Syntaxe :

```
float valFloat = (float)valInt;
```

Les Opérateurs

Les opérateurs permettent de manipuler ou comparer des valeurs, notamment des variables. Parmi les opérateurs utilisés fréquemment dans tous les langages de programmation vous trouverez :

- L'opérateur d'affectation
- Les opérateurs arithmétiques
- Les opérateurs binaire (comparaison)
- Les opérateurs logiques
- L'opérateur de négation

En Java il existe d'autres opérateurs très utilisés :

- Les opérateurs unitaires
- Les opérateurs avec assignation

Opérateur d'affectation

L'affectation d'une variable avec le signe « = »

Syntaxe :

« type » variable = valeur;

Remarque : Pour que l'affectation soit correcte il faut que les 2 opérandes soient de même type.

Exemple :

```
Integer variable1 = 100;
Integer variable2 = variable1;
```

Affectation multiple :

« type » var1, var2, …, varN = … = var2 = var1 = … ;

Opérateurs arithmétiques

Les opérateurs arithmétiques de base sont : +, -, *, / et % (modulo = reste de la division).

Remarque : l'opérateur « + » permet aussi de concaténer des chaînes de caractères.

Opérateurs unitaires (++ et --)

Les opérateurs unitaires permettent de raccourcir les expressions d'incrémentation et de décrémentation ainsi que faire des post ou pré affectations selon la position de l'opérateur unitaire par rapport à la variable concernée.

Opérateurs binaires (comparaison)

Egalité	→	==
Différent	→	!=
Inférieur	→	<
Supérieur	→	>
Inférieur ou égale	→	<=
Supérieur ou égale	→	>=

Les Variables

Opérateurs de logique

x & y → ET logique avec évaluation de tous les termes.
x | y → Ou logique avec évaluation de tous les termes.

x && y → ET logique. Evalue de gauche à droite et s'arrête des qu'un terme est faux.
x || y → OU logique. Evalue de gauche à droite et s'arrête des qu'un terme est vrai.

x ^ y → OU exclusif. Pour retourner vrai, il faut qu'un seul des termes soient vrai.

Opérateur de négation

!x → Négation du résultat d'une évaluation.

Opérateurs avec assignation

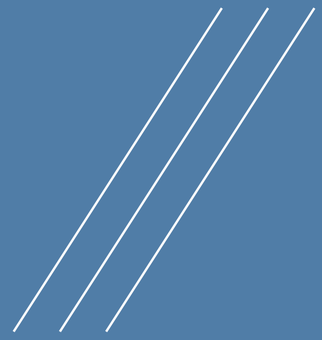
Cela permet de raccourcir une expression lorsque la variable ce trouve des 2 cotés de l'expression. Cela fonctionne avec tous les opérateurs de base.

Exemple :

i = i + 2; → i += 2;

Variables.java

JAVA
FONDAMENTAUX



Présenté par
Xavier TABUTEAU