## 1. Les commentaires

Il existe trois types de commentaire en Java :

| Type de commentaires | Exemple |
|---|---|
| commentaire abrégé | `// commentaire sur une seule ligne` |
| commentaire multiligne | `/* debut du commentaire`<br>`fin du commentaire */` |
| commentaire de documentation automatique | `/**`<br>`* commentaires générés`<br>`* automatiquement pour le code`<br>`* exemple commentaires pour une`<br>`* methode:`<br>`* @param val la valeur à traiter`<br>`* @return la valeur de retour`<br>`*/` |

**Remarques :**

- Ils ne sont pas pris en compte par le compilateur donc ils ne sont pas inclus dans le bytecode.
- Ils ne se terminent pas par un caractère ";".

## 2. Les types primitifs

Les types primitifs commencent tous par une minuscule.

| Type | Désignation | Longueur | Valeurs | Commentaires |
|---|---|---|---|---|
| boolean | valeur logique : true ou false | 1 bit | true ou false | pas de conversion possible vers un autre type |
| 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 Java |
| int | entier signé | 32 bits | -2147483648 à 2147483647 | |
| float | virgule flottante simple précision (IEEE754) | 32 bits | 1.401e-045 à 3.40282e+038 | |
| double | virgule flottante double précision (IEEE754) | 64 bits | 2.22507e-308 à 1.79769e+308 | |
| long | entier long | 64 bits | -9223372036854775808 à 9223372036854775807 | |

**Remarque :**

- Les types primitifs existent sous plusieurs formats

## 3. Initialisation des variables

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 :**

- 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.

## 4. Opération d'affectation

Il existe plusieurs opérateurs qui permettent de simplifier l'écriture d'une opération d'affectation associée à un opérateur mathématique comme suit :

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

## 5. Les comparaisons

Java propose des opérateurs pour toutes les comparaisons comme indiqué ci-dessous :

| 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 | différent de |
| & | a & b | ET binaire |
| ^ | a ^ b | OU exclusif binaire |
| \| | a \| b | OU binaire |
| && | a && b | ET logique (pour expressions booléennes) : l'évaluation de l'expression cesse dès qu'elle devient fausse |
| \|\| | a \|\| b | OU logique (pour expressions booléennes) : l'évaluation de l'expression cesse dès qu'elle devient vraie |
| ? : | a ? b : c | opérateur conditionnel : renvoie la valeur b ou c selon l'évaluation de l'expression a (si a alors b sinon c) ; b et c doivent retourner le même type |

## 6. La priorité des opérateurs

Java définit les priorités dans les opérateurs comme suit (du plus prioritaire au moins prioritaire)

| | |
|---|---|
| les parenthèses | ( ) |
| les opérateurs d'incrémentation | ++<br>-- |
| les opérateurs de multiplication, division et modulo | *<br>/<br>% |
| les opérateurs d'addition et soustraction | +<br>- |
| les opérateurs de décalage | <<<br>>> |
| les opérateurs de comparaison | <<br>><br><=<br>>= |
| les opérateurs d'égalité | ==<br>!= |
| l'opérateur OU exclusif | ^ |
| l'opérateur ET | & |
| l'opérateur OU | \| |
| l'opérateur ET logique | && |
| l'opérateur OU logique | \|\| |
| les opérateurs d'assignement | =<br>+=<br>-= |

## 7. Les caractères spéciaux dans les chaines

Dans une chaîne de caractères, plusieurs caractères particuliers doivent être utilisés avec le caractère d'échappement \. Le tableau ci-dessous recense les principaux caractères :

| Caractères spéciaux | Affichage |
|---|---|
| \' | Apostrophe |
| \" | Guillemet |
| \\ | Antislash |
| \t | Tabulation |
| \b | Retour arrière (backspace) |
| \r | Retour chariot |
| \f | Saut de page (form feed) |
| \n | Saut de ligne (newline) |
| \0ddd | Caractère ASCII ddd (octal) |
| \xdd | Caractère ASCII dd (hexadécimal) |
| \udddd | Caractère Unicode dddd (hexadécimal) |

## 8. Quelques méthodes utilitaires de la classe String

Le tableau suivant dresse quelques méthodes utiles fournies par la classe String. Reportez-vous à la documentation de la classe pour consulter la liste complète des méthodes.

| Méthodes la classe String | Rôle |
|---|---|
| charAt(int) | renvoie le nième caractère de la chaîne |
| compareTo(String) | compare la chaîne avec l'argument |
| concat(String) | ajoute l'argument à la chaîne et renvoie la nouvelle chaîne |
| endsWith(String) | vérifie si la chaîne se termine par l'argument |
| equalsIgnoreCase(String) | compare la chaîne sans tenir compte de la casse |
| indexOf(String) | renvoie la première position de l'argument dans la chaîne ou null |
| lastIndexOf(String) | renvoie la dernière position de l'argument dans la chaîne ou null |
| length() | renvoie la longueur de la chaîne |
| replace(char,char) | renvoie la chaîne en remplaçant les occurrences du caractère fourni |
| startsWith(String int) | vérifie si la chaîne commence par la sous chaîne |
| substring(int,int) | renvoie une partie de la chaine |
| toLowerCase() | renvoie la chaîne en minuscule |
| toUpperCase() | renvoie la chaîne en majuscule |
| trim() | enlève les caractères non significatifs de la chaîne |

## 9. Modificateurs d'accès (autorisations)

| Élément | Autorisations |
|---|---|
| Variable | Lecture et écriture |
| Méthode | Appel de la méthode |
| Classe | Instanciation d'objets de cette classe et accès aux variables et méthodes de classe |