La structure fondamentale du langage

La syntaxe du langage Java est à l'origine très inspirée du C et du C++. Ces deux langages de programmation ont également servi de base au C#. Donc si vous connaissez C, C++ ou C# vous retrouverez en Java des structures de langage qui vous sont familières.

Les instructions

Java est un langage de programmation impératif. Cela signifie qu'un programme Java se compose d'instructions (**statements**) décrivant les opérations que la machine doit exécuter. En Java une instruction est délimitée par un **point-virgule**.

```
double i = 0.0;
i = Math.sqrt(16);
```

Les blocs de code

Java permet de structurer le code en bloc. Un bloc est délimité par des **accolades**. Un bloc permet d'isoler par exemple le code conditionnel à la suite d'un **if** mais il est également possible de créer des blocs anonymes.

```
double i = 0.0;
i = Math.sqrt(16);

if (i > 1) {
  i =  -i;
}

{
  double j = i * 2;
}
```

Un bloc de code n'a pas besoin de se terminer par un point-virgule. Certains outils émettent un avertissement si vous le faites.

Les commentaires

Un commentaire sur une ligne commence par **//** et continue jusqu'à la fin de la ligne :

```
// ceci est un commentaire
double i = 0.0; // ceci est également un commentaire
```

Un commentaire sur plusieurs lignes commence par **/*** et se termine par ***/** :

```
/* ceci est un commentaire
   sur plusieurs lignes */
double i = 0.0;
```

Il existe un type spécial de commentaires utilisé par l'utilitaire javadoc. Ces commentaires servent à générer la documentation au format HTML de son code. Ces commentaires, appelés **commentaires javadoc**, commencent par **/**** :

```
/**
* Une classe d'exemple.
*
* Cette classe ne fait rien. Elle sert juste à donner un exemple de
* commentaire javadoc.
*
* @author David Gayerie
* @version 1.0
*/
public class MaClasse {

}
```

Le formatage du code

Le compilateur Java n'impose pas de formatage particulier du code. Dans la mesure où une instruction se termine par un point-virgule et que les blocs sont délimités par des accolades, il est possible de présenter du code de façon différente. Ainsi, le code suivant :

```
double i = 0.0;
i = Math.sqrt(16);
```

```
if (i > 1) {
  i = -i;
}
```

est strictement identique pour le compilateur à celui-ci :

```
double i=0.0;i=Math.sqrt(16);if(i>1){i=-i;}
```

Cependant, le code source est très souvent relu par les développeurs, il faut donc en assurer la meilleure lisibilité. Les développeurs Java utilisent une convention de formatage qu'il **faut** respecter. Des outils comme Eclipse permettent d'ailleurs de reformater le code (sous Eclipse avec le raccourci clavier *MAJ + CTRL + F*). Rappelez-vous des conventions suivantes :

```
// On revient à la ligne après une accolade (mais pas avant)
if (i > 0) {
  // ...
}

// On revient systématiquement à la ligne après un point virgule
// (sauf) dans le cas de l'instruction for
int j = 10;
for (int i = 0; i < 10; ++i) {
  j = j + i;
}

// Dans un bloc de code, on utilise une tabulation ou des espaces
// pour mettre en valeur le bloc

if (i > 0) {
  if (i % 2 == 0) {
    // ...
  } else {
    // ...
  }
}

// On sépare les variables des opérateurs par des espaces
i = i + 10; // plutôt que i=i+10
```

Les conventions de nommage

Chaque langage de programmation et chaque communauté de développeurs définissent des conventions sur la façon de nommer les identifiants dans un programme. Comme pour le formatage de code, cela n'a pas d'impact sur le compilateur mais permet de garantir une bonne lisibilité et donc une bonne compréhension de son code par ses pairs. Les développeurs Java sont particulièrement attachés au respect des conventions de nommage.

Convention de nommage

Type	Convention	Exemple
Packages	Un nom de package s'écrit toujours en minuscule. L'utilisation d'un _ est tolérée pour représenter une séparation.	java.utils com.company.extra_utils
Classes et interfaces	Le nom des classes et des interfaces ne doivent pas être des verbes. La première lettre de chaque mot doit être en majuscule (écriture dromadaire).	MyClass SuppressionClientOperateur
Annotations	La première lettre de chaque mot doit être une majuscule (écriture dromadaire). Il est toléré d'écrire des sigles intégralement en majuscules.	@InjectIn @EJB
Méthodes	Le nom d'une méthode est le plus souvent un verbe. La première lettre doit être en minuscule et les mots sont séparés par l'utilisation d'une majuscule (écriture dromadaire).	run() runFast() getWidthInPixels()
Variables	La première lettre doit être en minuscule et les mots sont séparés par l'utilisation d'une majuscule (écriture dromadaire). Même si cela est autorisé par le compilateur, le nom d'une variable ne doit pas commencer par _ ou $. En Java, les développeurs n'ont pas pour habitude d'utiliser une convention de nom pour différencier les variables locales des paramètres ou même des attributs d'une classe. Le nom des variables doit être explicite sans utiliser d'abréviation. Pour les variables « jetables », l'utilisation d'une lettre est d'usage (par exemple i, j ou k)	widthInPixels clientsInscrits total

Type	Convention	Exemple
Constantes	Le nom d’une constante s’écrit intégralement en lettres majuscules et les mots sont séparés par _.	LARGEUR_MAX INSCRIPTIONS_PAR_ANNEE

Les mots-clés

Comme pour la plupart des langages de programmation, il n’est pas possible d’utiliser comme nom dans un programme un mot-clé du langage. La liste des mots-clés en Java est :

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

> **goto** et **const** sont des mots-clés réservés mais qui n’ont pas de signification dans le langage Java.

Il existe également des mots réservés qui ne sont pas strictement des mots-clés du langage :

```
true   false   null
```

Revision #2
Created 10 February 2025 11:41:12 by Nicolas
Updated 10 February 2025 11:50:01 by Nicolas