Eléments de base d'un programme en Java

I – Préliminaire propre à JBuilder

L'outils de développement Java que nous utilisons est JBuilder 9 sous Linux. JBuilder nous permet de :

- **Éditer** les programmes Java (c'est à dire écrire et/ou modifier le code source)
- **Compiler** les programme (c'est-à-dire convertir le fichier source en un fichier exécutable)
- **Exécuter** les programmes, une fois qu'ils sont compilés

Avant d'écrire un programme en Java dans JBuilder, il faut créer ou ouvrir un projet. Le projet a pour but de définir le répertoire de travail dans lequel sera enregistré tous les fichiers :

- Les fichiers source, avec l'extension **.java**
- Et les fichiers exécutables, avec l'extension **.class**

Vous pouvez créer un seul projet une bonne fois pour toutes, et ouvrir toujours ce même projet, même si vous voulez écrire et tester plusieurs programmes différents : vous n'êtes pas obligé de créer un nouveau projet pour chaque nouveau programme.

Procédure à suivre dans JBuider 9 pour créer un nouveau projet :

- Menu **Fichier** + **Nouveau projet…**
- Une boîte de dialogue ***« Expert projet »*** s'ouvre sur l'écran.
- Dans le champ **Nom**, entrez le nom du projet
- Dans le champ **Répertoire**, entrez le répertoire du projet
- Puis cliquez sur le bouton **Terminer**

Une fois votre projet créé, vous pouvez y ajouter un ou plusieurs programme source en Java. Pour créer un nouveau fichier source :

- Menu **Fichier** + **Nouveau fichier…**
- Une boîte de dialogue ***« Création d'un nouveau fichier »*** s'ouvre sur l'écran.
- Dans le champ **Nom**, entrez le nom du fichier
- Dans le champ **Type**, vérifiez que le type de fichier est **java**
- Dans le champ **Répertoire** est inscrit le répertoire du projet (il n'y a rien à modifier)
- Cochez la case ***« Ajouter le fichier enregistré au projet »***
- Puis cliquez sur le bouton **OK**

Vous disposez alors d'un nouveau fichier, vierge pour l'instant, pour écrire votre programme en Java. Vous pouvez ainsi ouvrir plusieurs nouveaux fichiers, et donc écrire plusieurs programmes à la fois, dans le même projet.

II – Structure de base d'un programme en Java

En Java, un programme est appelé une ***classe***. La structure de base d'un programme est la suivante :

```
class toto {
        public static void main(String[ ] args) {

        }
}
```

La première ligne, avec le mot clé **class**, marque le début du programme (ou de la classe, c'est pareil) et indique que le programme s'appelle toto.

Le nom du fichier source, avec l'extension .java, dans lequel est enregistré le programme doit toujours être le même que celui de la classe.

Cela signifie que, pour fonctionner, le programme ci-dessus doit être enregistré dans un fichier **toto.java**.

Affichage d'un message à l'écran

Pour afficher un message à l'écran, on utilise la fonction **System.out.print**. Par exemple, le programme suivant affiche ***Bonjour !*** à l'écran :

```
class toto {
        public static void main(String[ ] args) {
                System.out.print("Bonjour !");
        }
}
```

Pour exécuter ce programme dans JBuilder 9, il faut cliquer droit sur l'onglet portant le nom du programme (sous la barre d'icônes en haut de l'écran), puis cliquer sur ***« ▶ Exécuter en utilisant les valeurs par défaut »*** : le résultat du programme est alors affiché dans la partie basse de l'écran de JBuilder.

Remarques :

- En Java, les chaînes de caractères sont placées entre doubles côtes (touche 3 du clavier en minuscule)
- En Java, chaque ligne de programme se termine par un point-virgule
- En Java, il y a une différence entre les majuscules et les minuscules : dans le programme ci-dessus, les lettres S des mots String et System sont en majuscules, et toutes les autres sont en minuscules. Si cette casse n'était pas respectée, des erreurs serait générées lors de l'exécutiuon.

Utilisation des variables

Le programme suivant déclare une variable nommée nombre (ligne 3), lui donne la valeur 2004 (ligne 4), puis l'affiche à l'écran (ligne 6) :

```
1  class toto {
2          public static void main(String[ ] args) {
3                  int nombre;
4                  nombre = 2004;
5                  System.out.print("La variable nombre vaut : ");
6                  System.out.print(nombre);
7          }
8  }
```

En Java, avant d'utiliser une variable, il faut la déclarer : c'est ce qui est fait à la ligne 3 du programme ci-dessus. Déclarer une variable consiste à définir le type de la variable. Par exemple, le type **int** utilisé précédemment indique que la variable **nombre** ne pourra contenir que des nombres entiers. Dans un programme, la déclaration des variables se fait toujours au début du programme (après la 2ème accolade ouvrante).

Les commentaires dans un programme

Afin de rendre le code source de nos programmes plus lisible, il est possible d'y ajouter des commentaires. Les commentaires sont des notes personnelles écrites par le programmeur (donc par vous) et qui sont totalement ignorées par le compilateur lors de l'exécution du programme. Dans un programme Java, un bloc de commentaire commence par les caractères **/*** et de termine par les caractères ***/** : il est alors affiché en vert dans JBuilder 9. Par exemple, une fois commenté, le programme précédent est plus agréable à lire :

```
class toto {       /* nom du programme */
        public static void main(String[ ] args) {

                int nombre;        /* déclaration des variables */

                nombre = 2004;        /* on donne une valeur à la variable nombre */

                System.out.print("La variable nombre vaut : ");

                /* affichage de la valeur de la variable nombre : */
                System.out.print(nombre);
        }
}
```

III – Les différents types de variables en Java

En Java, il existe 9 types de variables différents, qui se répartissent en 4 catégories :

- 4 types « nombres entiers »
- 2 types « nombres réels »
- 2 types « caractères »
- 1 type « logique »

Les 4 types « nombres entiers »

Comme le montre le tableau ci-contre, les 4 types de nombres entiers sont des entiers relatifs : il peuvent contenir aussi bien des nombres positifs que négatifs.

type	valeur la plus petite	valeur la plus grande
byte	– 128	127
short	– 32 768	32 767
int	– 2 147 483 648	2 147 483 647
long	– 9 223 372 036 854 775 808	9 223 372 036 854 775 807

Les 2 types « nombres réels »

Le type **float** est le type standard pour les nombres réels : il est généralement suffisant et précis pour la plupart des applications.

type	valeur la plus petite non nulle	valeur la plus grande
float	1.4 e – 45	3.4 e +38
double	5 e – 324	1.7 e + 308

Le type **double**, correspondant aux réels à haute précision, supporte la notation scientifique (utilisée aussi dans les calculatrices) dans laquelle la lettre **e** signifie ***« 10 exposant »***. Par exemple, en notation scientifique, un million s'écrit **1e6** : cette écriture correspond au nombre réel 1 000 000 (*1 fois 10 exposant 6*).

Les 2 types « caractères »

En Java, les caractères se placent entre simples côtes (touche 4 du clavier en minuscule) et les chaînes de caractères se placent entre doubles côtes (touche 3 du clavier). Le type **S**tring, correspondant aux chaînes de caractères, commence par un **S majuscule**.

type	peut contenir
char	un seul caractère
String	une chaîne de caractères

Exemple d'utilisation :

```
char c;
String s;
c='A';
s='' Bonjour, ceci est une chaîne de caractères de type String '' ;
```

Le type « logique »

Le type **boolean** est un type logique ne pouvant prendre que 2 valeurs possibles : **true** (correspondant à **vrai** ou encore à un **1 logique**) et **false** (correspondant à **faux** ou encore à un **0 logique**). Ce type permet de mémoriser des résultats de conditions logiques issues de l'algèbre de Boole.

type	valeur possible	signification
boolean	true	vrai
	false	faux

Exemple d'utilisation :

```
boolean b;          /* déclare la variable b de type boolean */
b=8>12;             /* donne à b la valeur de la condition 8>12 */
System.out.print(b); /* affiche false car la condition testée est fausse */
```

Comme Java sait manipuler des variables logiques, ce langage possède aussi les opérateurs logiques de base :

Le ET s'écrit avec 2 signes **&** (touches Alt Gr + 1), le OU s'écrit avec 2 signes **|** (touche Alt Gr + 6). Exemples de conditions logiques utilisables en Java :

```
b= ! (8>12) && (7<23);      /* 8 pas supérieur à 12 et 7 inférieur à 23 */
b=(a>10) || (c<9) || (d<3); /* a>10 ou c<9 ou d<3 */
```

opérateur	fonction logique
&&	ET
||	OU
!	NON

IV – Saisie d'une chaîne de caractère au clavier

Contrairement à l'affichage, la saisie au clavier n'est pas très directe en Java. En effet, pour saisir une chaîne de caractères au clavier, nous avons besoins d'insérer dans notre programme les 3 lignes de code suivantes :

```
BufferedReader clavier=new BufferedReader(new InputStreamReader(System.in)) ;
try { s=clavier.readLine(); }
catch(IOException e) { }
```

Ces 3 lignes permettent de donner à la variable **s** la valeur de la chaîne de caractères tapée au clavier. La variable **s** doit avoir été déclarée de type **String** au début du programme.
Une fois tapées, ces 3 lignes pourront bien sûr être réutilisées par un simple copier/coller, sans forcément les réécrire chaque fois qu'on aura besoin de saisir un message au clavier.

Par exemple, le programme ci-dessous demande à l'utilisateur d'entrer son nom au clavier, puis affiche ensuite le nom à l'écran :

```
import java.io.*;

class toto {
        public static void main(String[] args) {
        String nom;

        System.out.print("Entrez votre nom puis appuyez sur la touche Entrée : ");

        BufferedReader clavier = new BufferedReader(new InputStreamReader(System.in)) ;
        try { nom = clavier.readLine(); }
        catch(IOException e) { }

        System.out.print("Votre nom est " + nom);
        }
}
```

Remarques :

- la première ligne du programme (**import** java.io.*;) autorise le programme à utiliser la bibliothèque java.io. Cette ligne est obligatoire pour saisir une chaîne de caractères au clavier.
- Comme le montre la dernière ligne, pour afficher plusieurs chaînes de caractères à la suite avec une seule instruction **System.out.print**, il faut séparer les chaînes par le signe **+**.

Comment saisir un nombre au clavier ?

Pour saisir un nombre au clavier, le principe consiste à saisir une chaîne de caractères ne contenant que des chiffres, puis à convertir la chaînes de caractères en nombre réel **Float** (avec un **F majuscule** cette fois !). Si on désire un nombre entier, il faut alors convertir à nouveau le **Float** en un entier. Par exemple, dans le programme ci-contre, la chaîne de caractère s qui vaut "56" au début est convertie en un entier n de valeur numérique 56.

```
String s;
int n;
Float reel;
s = "56";
reel = Float.valueOf(s);
n = reel.intValue();
```

V – Exercices d'application

Exercices 1 : Réalisez un programme qui invite l'utilisateur à entrer son NOM, son Prénom, et son age, puis qui affiche ces 3 renseignements à l'écran.

Exercices 2 : Réalisez une « calculatrice » qui demande 4 nombres un par un, puis qui affiche leur somme.

Exercices 3 : Réalisez un convertisseur de Francs en Euros, qui demande un prix en Francs puis qui affiche l'équivalent en Euros. Pour convertir un **Float** F en un **float** f, on pourra utiliser la ligne `f=F.floatValue();`

Exercices 4 : Réalisez un convertisseur d'Euros en Francs, qui demande un prix en Euros puis qui affiche à l'écran l'équivalent en Francs.