Méthodes intelligentes

Introduction

La plupart des programmeurs débutants n'exploitent pas les méthodes à leur juste valeur. J'ai moi même mis longtemps avant de créer des méthode que j'appelle « intelligentes ». Pour moi une méthode intelligente est une méthode qui admet ou renvoie un argument. Ce type de méthode est bien sûr expliqué dans le chapitre sur les méthodes, mais d'une façon un peu trop théorique. Dans ce chapitre, je développe l'implémentation de la classe **Convert** du package **com.jgflsoft.utils**, et ajoute un exemple de retour de donnée.
La classe Convert sert à la conversion de certains types de base. Ses méthodes seront très utiles aux utilisateurs du Pascal/Delphi : on y trouve StrToInt() et IntToStr() ; méthodes que cherchent désespérément certains programmeurs expérimentés en PascalObjet et novices en Java.

Argument « d'entrée »

Certaines méthodes nécessitent un argument pour fonctionner. Cet argument se place lors de l'appel de la méthode entre ses parenthèses.
Exemple :

```
JLabel label = new JLabel() ;
String texte = new String("C'est le texte !!");
Label.setText(texte) ;
```

La méthode setText() admet un argument de type String.

Argument « de retour »

Certaines méthodes renvoient un argument après exécution. Vous devez avoir l'habitude des méthodes déclarées avec le mot clé void. Ces méthodes ne renvoient rien, d'où le ce mot clé signifiant « vide ».
Note : en Java le mot clé void a été seulement conservé pour les types de retour. Il était également utilisé si aucun argument « d'entrée » n'était spécifié en C++ : *void func(void) ;*
Une certain nombre de programmeurs débutants en Java (plus particulièrement ceux habitués à Delphi) essaient de convertir en String un int, pour l'afficher. Ils utilisent en vain la méthode **toString()**.
(Il suffit en fait de taper une instruction du type **System.out.println("" + i);** avec **int i = 5;** par exemple)
Cette méthode toString() renvoie la valeur String d'un objet :

```
JFileChooser jfc = new JFileChooser() ;
// quelques instructions sans importance pour ce chapitre
File fichier = new File(jfc.getSelectedFile().toString());
```

Est communiqué comme argument « s'entrée » à notre nouveau fichier le nom du fichier sélectionné dans le JFileChooser, qui m'est autre que l'argument de « retour » de la méthode **toString()**.

On déclare un méthode avec le type d'argument qu'elle retourne :
Exemple pour un retour de String : **public String retour() { }**
Exemple pour un retour de Fichier : **public File retour() { }**
ETC…

A la fin de la méthode concernée, on doit trouver un instruction du type :
return elementARetourner ;

Convert.java

```java
/**
 * Titre : Convert<p>
 * Description : Convertir les types de base de Java (sur le modèle du Pascal)<p>
 * Copyright : Copyright (c) Guillaume Florimond (2000)<p>
 * Société : JGFL<p>
 * @author Guillaume Florimond
 * @version 1.0
 */
package com.jgflsoft.utils;

public class Convert {

  public Convert() {
  }

  // De String vers Integer
  public static int strToInt(String s) {
  Integer ger = new Integer(s);
  int i = ger.intValue();
  return i;
  }

  //De INT vers String
  public static String intToStr(int i) {
  String texte = new String();
  texte = texte.valueOf(i);
  return texte;
  }

  // De INT vers long
  public static long intToLong(int i) {
  Integer ger = new Integer(i);
  long lg = ger.longValue();
  return lg;
  }
}
```