

Initiation au langage Java

Patrick Cégielski
cegielski@u-pec.fr

Mai 2010

Pour Irène et Marie

Legal Notice

Copyright © 2002 Patrick Cégielski

Université Paris Est Créteil - IUT

Route forestière Hurtaut

F-77300 Fontainebleau

cegielski@u-pec.fr

Table des matières

1	Historique et mise en place de Java	1
1.1	Historique de Java	1
1.1.1	Le projet Oak	1
1.1.2	Changement de nom	2
1.1.3	Un ordinateur spécifique	2
1.1.4	La vidéo à la demande	2
1.1.5	Introduction dans Internet	2
1.1.6	Le kit de développement	2
1.1.7	Les environnements de développement intégrés	2
1.2	Applications, applets, scripts et beans Java	3
1.3	Mise en place du kit de développement	4
1.3.1	Mise en place pour Windows 95/98	4
1.3.2	Mise en place pour Linux	5
1.4	Visualiser les applets de démonstration	6
1.4.1	Visualiser à travers un navigateur	6
1.4.2	Visualiser à l'aide de l'inspecteur	6
1.5	Développer les applications et les applets	6
1.5.1	Un exemple d'application	6
1.5.2	Un exemple d'applet	7

Chapitre 1

Historique et mise en place de Java

Java est un langage de programmation proche du langage C++. Langage de programmation à part entière, il est surtout connu par son utilisation sur Internet car c'est le seul langage actuel à pouvoir s'adapter à des systèmes informatiques différents (sans avoir à adapter le programme).

1.1 Historique de Java

1.1.1 Le projet Oak

Le développement de Java a commencé en 1990 chez Sun Microsystems Inc. à Mountain View, en Californie. Une équipe dirigée par James GOSLING se consacrait alors au développement d'un nouveau langage de programmation destiné au pilotage des appareils électroménagers. Le projet, qui portait le nom de code **Oak** (chêne en anglais, car on voyait un chêne depuis les fenêtres du bureau où se développait le projet) avait pour but d'éliminer la compilation spécifique au système d'exploitation de l'appareil en question.

1.1.2 Changement de nom

Le nom ‘oak’ était déjà utilisé par un autre langage de programmation. Il fallait donc trouver un nouveau nom. L’équipe de développement ne parvenait pas à trouver un nom adéquat, elle s’accorda une pause café puis, de guerre lasse, donna au nouveau langage le nom **Java** qui veut dire « café » en argot américain (à comparer à « kawa » ou « jus » en français), d’après le nom d’une fève de café.

Remarquons au passage la tradition qui s’est établie, consistant à choisir un nom du champ linguistique lié au café pour ce qui gravite autour de Java (voir **Café** de Symantec, **beans**, fève [de café] en anglais,...).

1.1.3 Un ordinateur spécifique

L’équipe de James GOSLING présenta un prototype d’ordinateur portable appelé *7 (*Star Seven*). Le *7 se commandait par une interface graphique. De petites figurines animées – notamment *Tumbling Duke* qui est devenu la mascotte de Java – accompagnait l’utilisateur dans le maniement de l’appareil.

1.1.4 La vidéo à la demande

À cette époque le concept de vidéo à la demande (**VOD** pour *Video On Demand* en anglais) semblait prometteur avec le développement de réseau câblé de télévision. Java pouvait s’adapter aux plates-formes nécessairement multiples. Malheureusement le boom attendu n’eut lieu que plus tard.

1.1.5 Introduction dans Internet

La carrière de Java ne connaissait pas de succès. SUN se mit en quête d’un nouveau domaine d’application au printemps 1994. C’est à ce moment que le réseau Internet, qui n’était alors connu que des universités et des centres de recherche, connut un engouement grand public. SUN reconnut très vite les capacités de son projet à fonctionner dans des réseaux hétérogènes du type Internet.

SUN mit toute son énergie à préparer les premières applications écrites en Java. En mai 1995, soit à peine un an plus tard, la société présenta publiquement à San Francisco la première version bêta du navigateur **HotJava** entièrement développé en JAVA. Le succès fut considérable. En peu de temps divers éditeurs de logiciels annoncèrent qu’ils allaient supporter le standard Java.

La société *Netscape Communications* occupait alors une position dominante dans Internet avec son navigateur *Netscape Navigator*. Son président déclara que les futures versions du navigateur supporteraient Java.

1.1.6 Le kit de développement

L’équipe de développement, qui entre-temps travaillait à Palo Alto en Californie, redoubla d’efforts et, en un délai record d’un an, mit au point une première version bêta du kit de développement Java (**JDK** pour *Java Developer’s Kit*) présenté au printemps 1996. Il s’agit d’un compilateur en ligne de commande (et non avec interface graphique) parfaitement fonctionnel.

1.1.7 Les environnements de développement intégrés

Des éditeurs de logiciel ont rapidement conclu des accords de licence pour diffuser des environnements plus élaborés, par exemple Symantec pour **Café** ou Microsoft pour **Visual J++**,

abandonnés depuis.

Le grand problème est que les applications Java sont à peu près vingt fois plus lentes que les programmes C++. De grandes sociétés travaillent à la mise au point d'accélérateurs de code pour Java.

1.2 Applications, applets, scripts et beans Java

L'engouement pour Java a fait que, contrairement aux autres langages de programmation pour lesquels il n'existe que des programmes d'une seule sorte, il existe plusieurs sortes de *programmes* :

- les **applications Java** sont les programmes au sens habituel, c'est-à-dire donnant lieu à un exécutable après compilation ;
- les **applets Java** (pour *APPL*ication *intern*ET) sont des programmes lancés à partir d'un fichier HTML (le langage de description de page d'Internet) par l'intermédiaire d'un navigateur, en fait pas n'importe quel navigateur puisqu'il faut que celui-ci interprète le programme, mais c'est le cas de tous les navigateurs actuels ;
- les **scripts Java** sont écrits en langage **JavaScript**, langage inspiré de Java mais largement indépendant, dont le code est directement inséré dans un fichier HTML, au départ uniquement pour le navigateur Netscape ;
- les **beans Java** sont des composants qu'il suffit de relier pour concevoir une application.

JavaScript est une extension des fonctions HTML à l'usage des non-programmeurs et est plus limité que Java. Les beans permettent une conception rapide. Les applications Java sont intéressantes mais l'exécutable demande un interpréteur, dépendant du système sur lequel on se trouve. Au départ ce sont surtout les applets qui ont connu un grand succès.

1.3 Mise en place du kit de développement

Le kit de développement peut se procurer soit par Internet sur le serveur de Sun :

`http://www.javasoft.com`

ou l'un de ses sites miroir, soit sur un CD-ROM le contenant.

Le kit de développement JDK comprend trois utilitaires essentiels : un compilateur Java (appelé **javac**), un interpréteur (appelé **java**) et un inspecteur d'applets (appelé **appletviewer**). Il comprend aussi des bibliothèques de classes Java et des exemples d'applets.

Le **compilateur** permet de traduire le programme source en un *pseudo-code* indépendant du système d'exploitation. L'**interpréteur**, dépendant du système d'exploitation, permet d'exécuter les applications Java. Nous avons déjà dit que les applets Java sont exécutées dans notre navigateur Internet préféré; cependant on peut vérifier son fonctionnement en dehors d'un tel navigateur avec l'**inspecteur** (ou **visionneur**) d'applets.

1.3.1 Mise en place pour Windows 95/98

Donnons un petit air désuet en décrivant ce qu'il fallait faire lors du premier JDK. C'est tout à fait analogue pour les versions actuelles; il est donc inutile de traiter la dernière version, de toute façon dépassée l'année suivante.

1.3.1.1 Récupération du kit

Le kit adapté à Windows 95/98 est :

`JDK-1.0-win32-x86.exe`

avec en fait une version plus récente que la version 1.0. La taille du fichier est d'environ 3 Mo pour la version 1.0.

1.3.1.2 Décompression du kit

Le fichier ainsi obtenu est un fichier d'archive auto-extractible. Il suffit de se placer à la racine du disque dur (en général **C :**), de copier le fichier à cet endroit et de le faire exécuter. Ceci crée une série de sous-répertoire sous le répertoire **\JAVA** dans lesquels les divers composants du kit sont installés. On peut alors sauvegarder le fichier originel ou le détruire.

1.3.1.3 Modifier le PATH

On doit rajouter une variable de PATH dans le fichier **AUTOEXEC.BAT** pour le répertoire d'exécutables de Java grâce à un éditeur de textes. On aura alors, par exemple, la ligne :

`PATH=C:\DOS;C:\WINDOWS;C:\JAVA\BIN`

1.3.1.4 Utilisation des fichiers compressés

Le kit JDK v1.0 permet l'utilisation de fichiers de classes compressés de type ZIP (ce qui permet de gagner de la place sur le disque dur). Un fichier appelé **CLASSES.ZIP** comprend toutes les classes Java standard. Il faut ajouter une variable **CLASSPATH** dans le fichier **AUTOEXEC.BAT** pour indiquer où il se trouve :

`SETCLASSPATH=. ;C:\JAVA\LIB\CLASSES.ZIP`

Il s'agit, dans les versions ultérieures, de `classes.jar`, compressé avec `jar`, le système d'archivage de Java.

1.3.1.5 Ajouter une variable **HOME** et un répertoire **.hotjava**

Plusieurs des outils du kit de développement cherchent un répertoire « **HOME** » dans lequel des informations sur la configuration sont sauvegardés. Il faut rajouter dans le fichier `AUTOEXEC.BAT` :

```
SETHOME=C:\JAVA
```

si `JAVA` est le répertoire pour Java.

Les informations sur la configuration sont placées dans le répertoire **.hotjava** (remarquez que le nom de ce répertoire commence par un point) qui doit être un sous-répertoire du répertoire indiqué par la variable **HOME**. Il faut donc créer un tel répertoire, par exemple sous DOS :

```
MKDIRC:\JAVA\.hotjava
```

1.3.2 Mise en place pour Linux

Cela dépend de la distribution Linux.

1.4 Visualiser les applets de démonstration

Le kit de développement est livré avec un certain nombre d'applets de démonstration. On peut, d'autre part, en récupérer par ailleurs. Voyons comment les visualiser (ou les exécuter, comme on veut).

1.4.1 Visualiser à travers un navigateur

Faites démarrer votre navigateur (Internet) préféré, par exemple *Microsoft Internet Explorer*. Dans le menu déroulant **F**ichier, choisir **O**uvrir... Dans la fenêtre « Ouverture » qui s'affiche alors, choisir **P**arcourir... Double-cliquez sur le dossier « java », puis sur le dossier « demo », puis sur le dossier de la démonstration qui vous intéresse, par exemple « Fractal ». Cliquez sur « example1 » (en fait « example1.html »), puis sur le bouton **O**uvrir et, enfin, dans la nouvelle fenêtre, sur le bouton **O**K. Une courbe de Von Koch se dessine petit à petit.

1.4.2 Visualiser à l'aide de l'inspecteur

Placez-vous dans la fenêtre de commandes DOS (sous Windows 95/98; surtout pas après redémarrage sous DOS, car JAVA ne fonctionne qu'en 32 bits). Placez-vous dans le répertoire adéquat :

```
CDC:\JAVA\DEMO\FRACTAL
```

puis faites exécuter l'applet (en dehors de tout navigateur) :

```
appletviewer example1.html
```

L'applet est exécutée dans une fenêtre dénommée « AppletViewer : CLSFractal.class ».

On remarquera l'icône avant le nom de la fenêtre : une tasse de café fumant ; il s'agit de celui du navigateur *HotJava* de Sun.

1.5 Développer les applications et les applets

1.5.1 Un exemple d'application

Écrivons une application permettant d'écrire « Ceci est une application » à l'écran.

1.5.1.1 Première étape : écriture du programme source

Écrivons le programme source suivant :

```
class Essai1
{
    public static void main(String args[])
    {
        System.out.println("Ceci est une application");
    }
}
```

à l'aide de votre éditeur de textes préféré, par exemple *Edit* du DOS, *WordPad* de Windows ou *emacs* sous Linux, en faisant attention aux minuscules et aux majuscules.

1.5.1.2 Deuxième étape : sauvegarde du programme source

Enregistrer le programme, nécessairement sous le nom *Essai1.java*. Le nom doit être celui de la classe que l'on définit et l'extension **java**.

1.5.1.3 Troisième étape : compilation du programme

Il faut compiler le programme :

```
javac Essai1.java
```

Ceci a pour effet de créer un fichier **Essai1.class**. Ce programme objet est en « byte-code » indépendant du système d'exploitation.

Javac est évidemment une contraction de *JAVA Compiler*.

1.5.1.4 Quatrième étape : interprétation du programme

Il faut maintenant faire exécuter le programme, en fait l'interpréter :

```
java Essai1
```

en faisant bien la différence entre minuscule et majuscule (même sous DOS) et en remarquant qu'il n'y a pas d'extension.

L'exécution a pour conséquence que, sur la ligne suivante est affiché :

```
Ceci est une application
```

puis on revient au prompteur.

1.5.2 Un exemple d'applet

Écrivons une applet permettant d'afficher « Ceci est une applet » à l'écran (en fait dans une fenêtre).

1.5.2.1 Première étape : écriture du programme source

Écrivons le programme source suivant :

```
import java.awt.*;
public class Essai2 extends java.applet.Applet
{
    public void init()
    {
        resize(100,150);
    }
    public void paint(Graphics g)
    {
        g.drawString("Ceci est une applet", 10, 10);
    }
}
```

à l'aide de votre éditeur de textes préféré, en faisant attention aux minuscules et aux majuscules.

1.5.2.2 Deuxième étape : sauvegarde du programme source

Enregistrer le programme, nécessairement sous le nom **Essai2.java**.

1.5.2.3 Troisième étape : compilation du programme

Il faut compiler le programme :

```
javac Essai2.java
```

Ceci a pour effet de créer un fichier **Essai2.class**. Ce programme objet est en « byte-code » indépendant du système d'exploitation.

1.5.2.4 Quatrième étape : incorporation de l'applet dans un document HTML

Il faut maintenant incorporer l'applet dans un document HTML. Ce fichier ne servant qu'à lancer l'applet, il peut être réduit au minimum :

```
<HTML>
<TITLE> Exemple d'applet </TITLE>
<applet code="Essai2.class" width=100 height=50></applet>
</HTML>
```

à conserver, par exemple, sous le nom **Essai2.html**.

On peut ne pas utiliser le couple de balises `<TITLE> </TITLE>` si l'on veut. Ici le nom du fichier peut être quelconque, bien qu'il est traditionnel d'utiliser l'extension `html` (ou `htm`).

1.5.2.5 Cinquième étape : exécution de l'applet

Pour faire exécuter l'applet il suffit de visionner le fichier HTML, soit avec l'inspecteur d'applet, soit avec un navigateur, comme indiqué ci-dessus. L'exécution a pour conséquence que, dans une fenêtre, est affiché :

Ceci est une applet.

Remarque.- Dans la version JDK 1.2, la classe `Applet` a été améliorée par la version `JApplet`. On peut donc utiliser le programme :

```
import java.awt.Graphics;
import javax.swing.JApplet

public class Essai3 extends JApplet
{
    public void init()
    {
        resize(100,150);
    }
    public void paint(Graphics g)
    {
        g.drawString("Ceci est une applet",10,10);
    }
}
```

mais il vaut mieux éviter d'utiliser ces applets, non implémentées dans tous les navigateurs.