Java - Introduction

3 octobre 2004

Table des matières

1 Place de Java dans le monde informatique

Java est soutenu par Sun Microsystems : http ://java.sun.com/.

1.1 Types d'applications

applications stand alone : indépendante.

applets (ou appliquette) : application insérée dans un environnement Web.

serveurs web et applications n-tiers : partie présentation (Servlets, JSP), partie métier (Java Beans), liaison avec des bases de données.

schéma d'application N-Tiers La figure 1 page 2 montre le schéma de principe d'une application N-tiers, architecture typique des serveurs Web applicatifs.

1.2 Raison d'être de Java

La conception du langage Java a été faite en respectant un certain nombre de principes.

fiabilité : les programmes d'entreprise, de e-commerce, de banques, ne peuvent pas se permettre d'être instables,

portabilité : entre différents systèmes d'exploitation - mais aussi entre différents serveurs Java,

lisibilité : deux développeurs doivent aboutir au même résultat quand ils répondent aux mêmes besoins - ce qui est loin d'etre le cas dans les langages comme C,

orientation Objet : puissance de calcul, facilité à modéliser le monde réel,

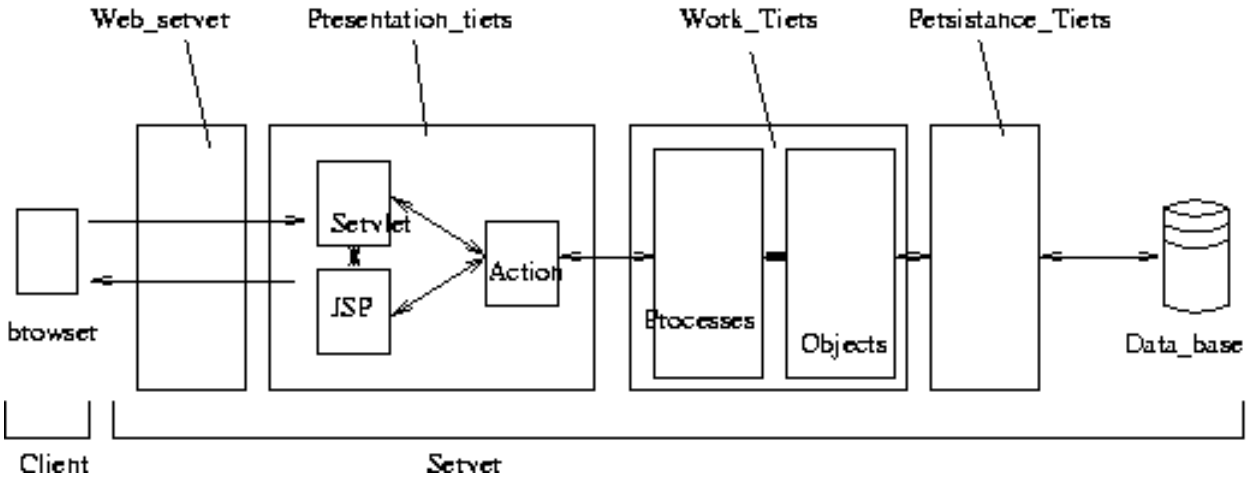

FIG. 1 – Schéma d'une application n-tiers

rapidité de la programmation : réutilisabilité des modules, aides à la conception (UML, Design Pattern),

sécurité : 'Bac à sable' java, qui isole les programmes Java du reste du système.

schema d'application sur JVM (ou IDE) sur OS La figure 2 page 2 présente les couches logicielles mises en oeuvre dans l'utilisation de Java. On remarque en particulier la présence d'une machine virtuelle qui isole les programmes du Système d'Exploitation.

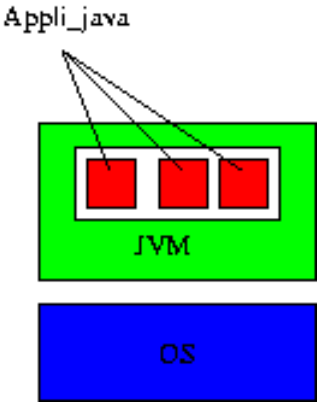

FIG. 2 – Schéma de l'architecture Java : OS, JVM et application

Petit glossaire Java :

API : Application Programming Interface, Package (ensemble de classes) qui permet de réaliser une tâche particulière,

IDE : Integrated Developpment Environnement (Environement de développement intégré),

JVM : Java Virtual Machine, émulation d'une machine réelle pour l'envirronement Java,

Package : regrouppement de classes,

UML : Unified Modelling Language, Langage spécifique de modélisation objet (utilisable pour tous les langages objets, et en partie pour les langages non objet).

2 Premier Programme

2.1 Version basique

Le programme HelloWorld.java, dans sa version basique, montre l'implémentation minimale d'un programme en Java.

2.1.1 Code du programme HelloWorld.java

```
public class HelloWorld{

  public static void main(String[] args){
      System.out.println("Hello World !");
    }

}

/* take care of the name of the file being the same as the one of the class
 * take care of the parameter of the 'main' method
 */
```

2.1.2 Description

Les Classes :

```
public class HelloWorld{
  ...
}
```

- Les Classes : un fichier dans un programme java est une classe indépendante.
  Une classe permet d'instancier un type d'objet particulier.

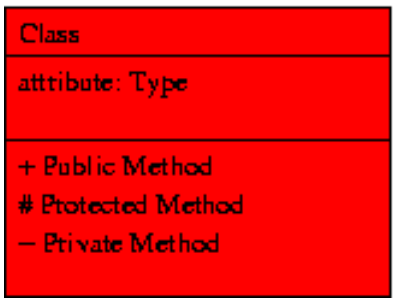

FIG. 3 – Classe Java représentée en UML

La méthode 'main' :

```
  public static void main(String[] args){
      System.out.println("Hello World !");
    }
```

– La méthode 'main' : si la classe est exécutée directement, c'est cette méthode qui est lancée.
– Elle est de type 'public static'.
– Elle retourne 'void'.
– Elle peut accepter des arguments.
– Dans la plupart des cas, une classe est appelée par une autre (création d'objets). Il n'y a donc pas de méthode 'main'.

Les appels de méthodes
– Pour appeler une méthode d'un autre classe, il faut soit importer cette classe, soit donner la référence exhaustive de la méthode.
– Exemple : import java.io ;
– Exemple : System.out.println("Text") ;
– Les méthodes d'une classe peuvent être spécifiées public (accès par toutes les classes), protected, private (accès d'un objet à lui-même).

Le tableau suivant montre les différents types d'accès possibles aux méthodes en fonction de la spécification.

Specifier	class	sub-class	package	world
private	X			
protected	X	X	X	
public	X	X	X	X
package	X		X	

Qu'est ce qu'un objet ? C'est une instance d'une classe, c'est à dire un élément qui a sa vie propre, et dont les méthodes et variables sont définis selon la classe.

Comment faire ?
– pour exécuter un programme : java maClasse.class
– pour compiler un programme : javac maClasse.java

2.2 Avec l'introduction des concepts objets

Le programme HelloWorld2.java, dans sa version avancée, montre l'introduction des concepts objets dans le programme minimal du paragraphe précédent, en conservant les mêmes fonctionnalités.

2.2.1 Code du programme HelloWorld2.java

```
class HelloWorld2 {
    protected String msg;


    public HelloWorld2(String message) {
this.msg = message;
    }
```

```
    public void act(){
System.out.println(msg);
}

    public static void main(String[] args){
HelloWorld2 HW = new HelloWorld2("HelloWorld !");
HW.act();
    }
}
```

2.2.2 Description

Les variables

```
protected String msg;
```

- Souvent, elles sont private : seul un objet peut accéder à ses propres variables, par le biais des méthodes getter/setter.
- ce sont des types basiques (int, double), ou des objets.

Les méthodes

```
    public void act(){
System.out.println(msg);
}
```

- Elles sont publiques si d'autres objets peuvent y accéder, protected ou private sinon.
- Elles contiennent 0, 1 ou plusieurs paramètres, donc le type doit être indiqué
- Elles retournent des données, ou non. Si oui, leur type est indiqué. Sinon, 'void' est retourné.

Le constructeur

```
    public HelloWorld2(String message) {
this.msg = message;
    }
```

- Il permet de créer un nouvel objet
- Il réalise l'initialisation des variables
- On peut accèder aux méthodes par le biais de 'this', si l'objet appelle une méthode qu'il contient lui-même,
- On peut accèder aux méthodes par le biais d'un nouvel objet si l'appel de la méthode doit être indépendant de l'objet dans lequel on se trouve.

2.3 Avec documentation (javadoc)

Javadoc Cette API permet d'éditer automatiquement des pages de documentation complètes sous forme de page web (HTML).

Le programme HelloWorld_doc.java, dans sa version documentée, montre la documentation type d'un programme Java.

2.3.1 Code de HelloWorld_doc.java

```
/**
 * Presentation of the class <br> you can write down HTML code
 * required fields :
 * @author pparrend
 * @version 1.2
 * @since version 1.2
 */

public class HelloWorld_doc {
    protected String msg;

    /**
     * What the method does : constructor
     * @param message the parameter of the method, here a <code>String</code>
     * @see no related classes
     */
    public HelloWorld_doc(String message) {
this.msg = message;
    }

    /**
     * This method prints the message known as <code>msg</code> on the standard output
     *
     * @param void
     * @return void
     */
    public void act(){
System.out.println(msg);
}

    public static void main(String[] args){
HelloWorld_doc HW = new HelloWorld_doc("HelloWorld !");
HW.act();
    }
}
```

2.3.2 Description

La classe : on indique ce qu'elle représente, et les fonctionnalitßes qu'elle implémente.

Paramètres :

- @author, l'auteur (requis)
- @version, le numéro de version (requis),
- @exception, @throws, type d'exception susceptibles d'être levé
- @see, les classes auquelles la classe fait référence.
- @deprecated, pour les API ayant été remplacé (doit contenir la référence de la nouvelle API)
- @since, numéro de version du produit pour lequel la classe a été ajoutée
- {@link}, lien vers la documentation d'une autre API

– @serial, @serialField, @SerialData.

Les méthodes : on indique ce qu'elles effectuent

Paramètres :

– @param, les paramètres.
– @return, le type de donnée retourné
– @exception, @throws, type d'exception susceptibles d'être levé
– @see, les classes auquelles la méthode fait référence.
– @since, numéro de version du produit pour lequel la méthode a été ajoutée
– @serial, @serialField, @SerialData.

Le code : on utilise du code HTML (ex :
, etc...)

<**code**> : en particulier, on utilise le mot clé <¡ code>...</code>. pour les éléments suivants : mots clés java, nom de package, nom de classe , nom de méthodes, nom d'interface, nom de champs, nom d'argument, exemple de code.

Recommendation :

– Etre bref,
– Utiliser la troisième personne du singulier (pas la deuxième).

Tous les détails sur Javadoc : http ://java.sun.com/j2se/javadoc/writingdoccomments/

2.4 L'applet

Les applets sont des classes java qui génèrent des éléments (textes, graphiques, etc.) insérés dans des pages web.
Elles ont un accès limité en lecture et en écriture sur le système de fichier, car elles peuvent venir d'un serveur qui ne soit pas de confiance (untrusted).

2.4.1 Code de l'applet HelloWorld.java

```
import java.applet.Applet;
import java.awt.Graphics;

public class HelloWorld extends Applet {
    public void paint(Graphics g) {
        g.drawString("Hello world!", 50, 25);
    }
}
```

2.4.2 Description

– Méthode paint : tout applet doit contenir une des trois méthodes init, start ou paint,
– méthode drawString : de la classe java.awt.Graphic,
– héritage : public class HelloWorld extends Applet.
HelloWorld hérite de la classe Applet :

1. Toutes les variables et méthodes de Applet sont disponibles dans un objet HelloWorld,
2. des extensions sont définies : nouvelles variables et/ou nouvelles méthodes.

2.4.3 La page web associée

Le code de la page web associée, HelloWorld.html, est :

```
<HTML>
<HEAD>
<TITLE> A Simple Program </TITLE>
</HEAD>
<BODY>

Here is the output of my program:
<APPLET CODE="HelloWorld.class" WIDTH=150 HEIGHT=25>
</APPLET>
</BODY>
</HTML>
```

L'exemple de l'applet est issu des tutoriaux disponibles en ligne sur http ://java.sun.com/docs/books/tutorial/index.html.