Structures de données

Table des matières

La classe Vector

La classe Vector du package java.util permet de stocker des objets dans un tableau dont la taille évolue avec les besoins.

Utilisation

Constructeurs

Parmi les constructeurs disponibles, l'un n'attend aucun paramètre, l'autre attend une taille initiale en paramètre. On construira donc un objet vect de la classe Vector en écrivant simplement:

```
Vector vect=new Vector();
```

ou

```
Vector vect=new Vector(n);
```

pour prévoir une capacité initiale de n.

Ajouter un objet

On ajoute un objet à la fin du Vector en utilisant la méthode addElement. Par exemple, pour ajouter l'objet o, on écrira :

```
vect.addElement(o);
```

Le premier élément ajouté a l'indice 0, le suivant l'indice 1, etc... La méthode **size()** renvoie le nombre d'éléments contenus dans le vecteur qui sera aussi l'indice du prochain objet ajouté.

Lire un objet

On retrouve un objet à partir de son indice en utilisant la méthode elementAt. Par exemple on obtient l'objet d'indice n en écrivant :

```
o=vect.elementAt(n);
```

Attention, la méthode elementAt renvoie par défaut des objets de type Object, un transtypage est souvent nécessaire. Par exemple, si l'élément d'indice 2 est de type String, il faudra écrire :

```
Color c=(Color)vect.elementAt(2);
```

Autres méthodes

La classe Vector contient de nombreuses autres méthodes qui rendent son utilisation très aisée. Notons :

- contains(Object) : indique si l'objet est contenu dans le Vector.
- copyInto(Object[]) : copie les éléments dans un tableau classique.

- firstElement() : renvoie le premier élément.
- indexOf(Object) : renvoie l'indice de l'objet
- insertElementAt(Object, int) : insère l'objet à l'indice indiqué
- isEmpty() : indique si le Vector est vide
- lastElement() : renvoie le dernier élément
- removeAllElements() : vide le Vector
- removeElementAt(int) : retire l'objet dont l'indice est donné
- setElementAt(Object, int) : place l'objet à l'indice donné
- size() : renvoie le nombre d'éléments

L'applet glossaire

Nous allons construire une applet qui joue le rôle d'un glossaire. L'ensemble des mots du glossaire et de leurs définitions seront tirés d'un fichier et rangés dans un objet de type Vector.

Format du fichier de données

Le fichier de données (en Annexe) est un fichier texte dans lequel chaque ligne contient un mot et sa définition séparés par le signe =.

Déclarations

On importe les packages nécessaires, on implémente l'interface ItemListener pour les choix dans la liste des mots, on prépare les composants et un Vector glo qui contiendra toutes les lignes du fichier de données.

```
import java.applet.*;
import java.awt.*;
import java.awt.event.*;
import java.io.*;
import java.net.*;
import java.util.*;

public class glossaire extends Applet
 implements ItemListener {

 //fichier de données
 String nomFichier="glossaire.txt";
 //TextArea avec WordWrap
 TextArea zoneTexte=new TextArea
  ("",10,30,TextArea.SCROLLBARS_VERTICAL_ONLY);
 //Liste des mots du glossaire
 List liste=new List();
 //Structure de stockage des données
 Vector glo=new Vector();
```

Méthode init

On place les composants et on récupère le nom du fichier de données.

```
 public void init() {
  //positionnement des composants
  setLayout(new BorderLayout(4,4));
  add("Center",zoneTexte);
  zoneTexte.setEditable(false);
```

```
 zoneTexte.setBackground(Color.white);
 add("West",liste);
 //propriétés de la liste
 liste.addItemListener(this);
 liste.setMultipleMode(false);
 liste.setBackground(Color.white);


}
```

Lecture du fichier de données

On peut lire le fichier de données dans la méthode start.

```
public void start() {
 //lecture du fichier
 try {
  URL url=new URL(getCodeBase(),nomFichier);
  BufferedReader br=new BufferedReader
     (new InputStreamReader(url.openStream()));
  String ligne;
  while (null!=(ligne = br.readLine()))
   if (ligne.indexOf('=')>0) glo.addElement(ligne);
  br.close();
 } catch (Exception e) {}

 //remplissage de la liste

 for (int i=0; i<glo.size(); i++)
      liste.addItem(nom(i));
      //sélection initiale
      liste.select(0);
      affiche(def(0));
      liste.requestFocus();
}
```

On a utilisé les méthodes nom et affiche qui permettent d'extraire un mot du glossaire et d'afficher sa définition. Il faut écrire ces méthodes.

Méthodes d'utilisation du glossaire

La méthode nom(int n) renvoie le mot d'indice n, la méthode affiche(int n) affiche sa définition. On sépare les deux parties en utilisant la position du signe =.

```
String nom(int n) {
//retrouver le n ième mot de la liste
 if (n>=0 && n<glo.size()) {
  String S=(String)glo.elementAt(n);
  int pos=S.indexOf('=');
  return S.substring(0,pos);
 }
 else return "";
}


void affiche(int n) {
//retrouver la n ième définition
 if (n>=0 && n<glo.size()) {
  String S=(String)glo.elementAt(n);
```

```
            int pos=S.indexOf('=');
            S=S.substring(pos+1,S.length());
            //affichage, les # sont remplacés par des retours chariots
            S=S.replace('#','\n');
            zoneTexte.setText(S);
           }
          }
```

Choix dans la liste

Il reste à écrire la méthode itemStateChanged de gestion des évènements produits par la liste.

```
public void itemStateChanged(ItemEvent e) {
 if (e.getStateChange()==ItemEvent.SELECTED) {
  affiche(liste.getSelectedIndex());
 }
}
```

La classe Hashtable

La classe Hashtable du package java.util permet de créer des collections d'objets associés à des noms, en quelque sorte des dictionnaire. Une même Hashtable peut contenir des objets de classe quelconque.

Utilisation

Constructeurs

Parmi les constructeurs disponibles, l'un n'attend aucun paramètre, l'autre attend une taille initiale en paramètre. On construira donc un objet dic de la classe Hashtable en écrivant simplement:

```
Hashtable dic=new Hashtable();
```

ou

```
Hashtable dic=new Hashtable(n);
```

pour prévoir une capacité initiale de n.

Ajouter un objet

On ajoute un objet o dans la Hashtable dic en l'associant au nom S avec l'instruction:

```
dic.put(S,o);
```

Lire un objet

On retrouve l'objet o associé au nom S avec l'instruction:

```
o=dic.get(S);
```

Attention la méthode get renvoie des objets généraux (classe Object), il est donc souvent nécessaire d'effectuer un transtypage pour retrouver le type qui a été stocké.

Par exemple, si on crée une Hashtable contenant des entiers associés à leurs noms, on obtiendra l'entier associé à la chaîne "deux" en écrivant :

```
Integer I=(Integer)dic.get("deux");
```

Exemple : noms de couleurs

Ecrivons une applet qui permet de choisir une couleur à partir d'une liste de noms.

Déclarations

On utilise un Canvas pour colorier, une liste de couleurs et une Hashtable qui associe les couleurs à leurs noms. On implémente l'interface ItemListener pour gérer les changements dans la liste de sélection.

```
import java.applet.*;
import java.awt.*;
import java.awt.event.*;
import java.util.*;

public class Couleurs extends Applet
 implements ItemListener {

 Canvas zoneDessin=new Canvas();
 List liste=new List();
 Hashtable dico=new Hashtable();
```

Méthode init

La méthode init place les composants et remplit le dictionnaire et la liste.

```
 public void init() {
  setLayout(new BorderLayout());
  add(BorderLayout.WEST,liste);
  add(BorderLayout.CENTER,zoneDessin);
  liste.addItemListener(this);
  //initialisation du dictionnaire et de la liste
  dico.put("blanc",Color.white);
  liste.add("blanc");
  dico.put("bleu",Color.blue);
  liste.add("bleu");
  dico.put("cyan",Color.cyan);
  liste.add("cyan");
  dico.put("gris",Color.gray);
  liste.add("gris");
  dico.put("gris clair",Color.lightGray);
  liste.add("gris clair");
  dico.put("gris foncé",Color.darkGray);
  liste.add("gris foncé");
  dico.put("jaune",Color.yellow);
  liste.add("jaune");
  dico.put("magenta",Color.magenta);
  liste.add("magenta");
```

```
 dico.put("noir",Color.black);
 liste.add("noir");
 dico.put("orange",Color.orange);
 liste.add("orange");
 dico.put("rose",Color.pink);
 liste.add("rose");
 dico.put("rouge",Color.red);
 liste.add("rouge");
 dico.put("vert",Color.green);
 liste.add("vert");
 liste.select(0);
 zoneDessin.setBackground(Color.white);
}
```

Lecture d'une couleur

La méthode lireCouleur retrouvera l'objet de type Color associé à un nom de couleur.

```
public Color lireCouleur(String nom) {
 return (Color)dico.get(nom);
}
```

Notons le transtypage nécessaire car la méthode get renvoie par défaut une variable de type Object.

Changements de couleurs

La méthode itemStateChanged exigée par l'interface ItemListener va permettre de réagir aux changements de couleurs.

```
public void itemStateChanged(ItemEvent e) {
 if (e.getStateChange()==ItemEvent.SELECTED) {
  Color col=lireCouleur(liste.getSelectedItem());
  zoneDessin.setBackground(col);
  zoneDessin.repaint();
 }
}
```

Le choix d'une nouvelle couleur dans la liste correspond à l'évènement de statut SELECTED.

Exercices

1. Français/Anglais
   Ecrire l'applet anglais qui donne la traduction d'un certain nombre de mots. Les mots traduits sont contenus dans un fichier de type texte.

2. Citations
   Ecrire l'applet Citations qui affiche des citations tirées d'un fichier texte.

Annexe :

page d'accueil=Une page d'accueil est constituée par le premier écran affiché par un serveur Web lorsque l'on se connecte dessus. En anglais : home page.

ActiveX=Standard établi par Microsoft pour réaliser des briques de base de programmes connus sous le nom d'objets.

adresse électronique=Code secret au moyen duquel l'Internet vous identifie et vous permet de recevoir du courrier électronique. Elle se présente généralement sous la forme utilisateur@site.pays où utilisateur représente votre nom d'utilisateur, site, le nom de la machine sur laquelle est ouvert votre compte utilisateur et pays, un code de trois lettres pour les américains, deux lettres pour le reste du monde. site peut lui-même être composé de plusieurs noms séparés par des points.

AOL=(America On Line) Fournisseur d'accès à valeur ajoutée bien implanté en France. Revendique le plus grand nombre d'abonnés dans le monde (plus de 4 millions). Les abonnés ont une adresse de la forme utilisateur@aol.com ou utilisateur@aol.fr en France.

anonymous=(anonymous FTP) Méthode d'utilisation du programme FTP lorsqu'on n'a pas de compte ouvert sur un site particulier. On se logge alors sous le nom d'utilisateur anonymous et on donne comme mot de passe son adresse électronique.

applet=Non, ce n'est pas une petite pomme mais un petit programme écrit en Java. On télécharge des applets sur le Web en même temps qu'une page Web, avec un navigateur. En principe, sécurisé. Pas toujours.

archive=Fichier contenant un groupe de fichiers généralement compressés pour occuper moins de place et être transmis en moins de temps. Pour restituer ces fichiers dans leur état d'origine, on doit utiliser le programme de décompression approprié. Sur les PC, on utilise couramment le format ZIP.

bauds=Terme technique caractérisant la vitesse de modulation d'un signal sur une voie de transmission. A ne pas confondre avec bps (voir ce sigle) qui caractérise le débit efficace de la voie. Une ligne à 2 400 bauds supporte facilement un débit de 28 800 bps, voire de 33 600 ou même plus si elle est de bonne qualité.

bps=(Bits par seconde) Unité de mesure du débit d'une voie de transmission et caractérisant ce qu'on appelle improprement la vitesse d'un modem. Ne pas confondre avec baud (voir ce mot).

client/serveur=(modèle)Type d'architecture dans laquelle le travail est divisé en deux groupes : celui qui fournit (le serveur) et celui qui reçoit (le client) des informations.

compression=Opération visant à réduire la taille d'un fichier ou d'un groupe de fichiers (une archive). S'effectue au moyen de logiciels particuliers tel que PKZIP dans le monde PC ou StuffIt, chez les adeptes du Macintosh.

domaine=Nom officiel d'un ordinateur sur l'Internet. C'est ce qui est écrit immédiatement à droite du caractère @. Dans jdupont@urec.fr, le nom du domaine est urec.fr.

DNS=(domain name server)Ordinateur situé sur l'Internet et qui a la charge de traduire les noms de domaines (dummies.com, inria.fr...) en adresses numériques de la forme 140.186.81.2 aussi appelées adresses IP. Parfois appelé serveur de noms.

GIF=(Graphics Interchange Format) Format d'image défini par CompuServe et maintenant très largement utilisé sur l'Internet et ailleurs. Par suite de problèmes de copyright, il risque d'être progressivement abandonné et remplacé par un nouveau format : PNG.