Examen de programmation par objets

Janvier 2002

Licence informatique - UE2241MBE

Responsable J.Ferber

Tous documents manuels (cours, polys) autorisés. Livres non autorisés Durée : 2 h

Exercice 1 : Questions de cours

Répondez en une ou deux lignes maximum aux questions suivantes:

1. Quelle est la différence entre les mots clés **'extends'** et **'implements'** de Java. Dans quel cas utilise-t-on l'un ou l'autre?
2. Que signifie l'instruction **instanceof** en Java ?
3. Que signifie le mot clé **`protected`** en Java et où le rencontre-t-on ?
4. Soit le programme suivant (On suppose que la classe **Truc** a été définie par ailleurs):

```
Truc[] t = new Truc[4] ;
T[0]=new Truc(); T[1]=new Truc(); T[2]=new Truc();
for (int i=0 ;i<t.length ;i++){
      System.out.println(t.toString()) ;
}
```

a) Quelles sont les erreurs éventuelles qui seraient émises lors de la compilation ou de l'exécution de ce programme ?

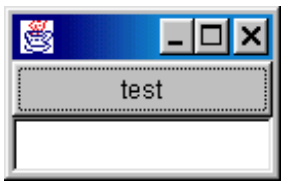
b) Quels sont les messages envoyés et les types des exceptions levées (à peu près, on ne vous demande pas les noms au caractère près évidemment).

Exercice 2: Gestionnaire d'événements

On désire réaliser un système permettant d'ajouter dynamiquement des "listener" à un bouton particulier **`b`**.

```
public class Truc extends Frame {

  Button b;
  TextField tf;

  TextField getTextField(){return tf;}

  void addButtonListener(ActionListener a){
     b.addActionListener(a);
  }
```

```
  public Truc(){
     setLayout(new GridLayout(2,1));
     b = new Button("test");
     tf = new TextField(20);
     add(b);
     add(tf);
  }

  public static void main(String[] args){
     Truc t1 = new Truc();
     t1.addButtonListener(new PrintListener());
     t1.addButtonListener(new FieldListener(t1.getTextField()));
     t1.pack();
     t1.show();
     }

}
```



Le lancement du programme affiche la fenêtre suivante:

Question 1: Ecrire la classe **`PrintListener`** qui imprime dans le flux de sortie (via la commande **`System.out.println(...)`**) l'étiquette du bouton lorsqu'on appuie sur ce bouton. Dans notre cas, cela imprimera **`'test'`**.

Question 2: Ecrire la classe **`FieldListener`** qui affiche dans le **`TextField`** qui lui est passé en argument lors de sa construction, le nombre de fois que l'on a cliqué dans ce bouton. Par exemple, si l'on a cliqué deux fois sur le bouton, le textfield affichera le nombre 2. On rappelle que la méthode **`setText(String s)`** d'un **`Textfield`** permet de placer un texte dans un **`Textfield`**.

Exercice 3 : Conception de programme

On désire réaliser un programme permettant de gérer une petite bibliothèque municipale. Pour cela on a analysé qu'on avait besoin d'une classe **Bibliotheque**, d'une classe **Adherent** et d'un ensemble de classes de **Document** permettant de répondre aux critères suivants:

- Les adhérents ont un prénom (chaîne de caractère) et un nom (chaîne aussi).
- la bibliothèque comprend un ensemble de documents.
- Ces documents sont soit des journaux, soit des volumes. Les volumes sont soit des dictionnaires soit des livres, soit des BD.
- Les documents sont caractérisés par un titre (chaîne de caractères) et un numéro (un entier) les identifiant. Les volumes ont en plus un auteur (chaîne). Les BD ont en plus un nom de dessinateur (chaîne), les journaux ont un titre et une périodicité (chaîne).
- Seuls les livres sont empruntables.
- Une bibliothèque connaît la liste de ses adhérents
- Les adhérents peuvent emprunter des livres (et uniquement des livres) et on doit pouvoir savoir à tout moment quels sont les livres empruntés par quels adhérents et quels sont les livres qu'un adhérent a emprunté.

Question 1: Faire le diagramme de classe UML permettant de modéliser cette bibliothèque.

Question 2: Donnez en Java la structure des classes correspondantes (note: n'écrivez pas trop gros pour que cela soit lisible rapidement SVP).

On veut ajouter un ensemble de méthodes permettant de gérer cette bibliothèque.

Question 3: Donnez le code des méthodes permettant de gérer l'emprunt et le rendu d'un livre. Dans la classe **Adherent**, ces méthodes sont définies ainsi:

```
dans Adherent
    void emprunter(Livre l){}
    void rendre(Livre l){}
```

Note: vous pouvez être amené à ajouter d'autres méthodes (dans Adherent ou d'autres classes) pour pouvoir décrire ces méthodes. Donnez alors le code de ces méthodes.

Question 4: Dans la classe Bibliothèque, donnez le code de la méthode

```
Document[] chercherDocument(String titre)
```

qui retourne, sous la forme d'un tableau de document, tous les documents dont le titre contient la chaîne "clef". La taille du tableau de documents correspond au nombre de documents trouvés.

(note: on rappelle que la méthode **indexOf(String str)** retourne l'index de la première de la sous-chaîne **str** occurrence dans la chaîne courante, et –1 si **str** n'est pas une sous-chaîne de la chaîne courante.

Question 5: Donnez l'ensemble des méthodes permettant à la bibliothèque de lister l'ensemble de ses documents avec leurs caractéristiques (c'est à dire d'imprimer dans le flux de sortie les attributs des documents) et, lorsqu'il s'agit d'un livre, de lister ses adhérents éventuels.

Pour lancer ce listing il suffira d'envoyer un message **listerDocuments()** à la bibliothèque:

```
Bibliotheque bib;
...

bib.listerDocuments();
```