Institut des Sciences Appliquées et Économiques
ISAE – Cnam Liban
Centre du Liban Associé au CNAM de Paris

Date: 11/ 7/ 2013 – Durée : 3h00

Final – 2ème semestre
2012 - 2013

Sujet coordonné par :
Abdallah EL Asmar

Proposé pour les centres d'enseignement de :
Beyrouth – Baalbek – Tripoli – Bickfaya -

Documents autorisés

Corrigé - Examen Final de
Programmation Avancée – NFP121

Exercice 1 (9½ pts)

a) **(4½ pts)**

```
public class TacheElementaire implements Tache {
    private String nom;    private int cout;
    public TacheElementaire(String nom, int cout) {
       this.nom = nom;    this.cout = cout;
    }
    public String getNom() {     return nom;
    }
    public int getCout() {     return cout;
    }
    public String toString(){    return "[" + nom + ", " + cout +"]";
     }
  }

import java.util.*;
public class TacheComplexe implements Tache {
    private Collection<Tache> sousTaches;
    private String nom;
    public TacheComplexe(String nom) {   this.nom = nom;
       sousTaches = new ArrayList<Tache>();
    }
    public void ajouter(Tache tache) {    sousTaches.add(tache);
    }
     public void supprimer(Tache tache) {  sousTaches.remove(tache);
     }
    public String getNom() {  return nom;
    }
    public int getCout() {   int result = 0;
       for (Tache t : sousTaches) {     result += t.getCout();   }
    return result;
    }
    public String toString() {   String S=nom + ": [";
       for (Tache t : sousTaches) {     S += t.toString() + ",";   }
       S += "]";
       return S;
    }
}
```

b) **(2½ pt)**

Design Pattern: Decorator

```
public abstract class DecorateurTache implements Tache {

   protected Tache tache;
   public DecorateurTache(Tache tache) { this.tache = tache;
    }
   publc int getCout () {       return tache.getCout();
   }
   public String getNom() {      return tache.getNom();
   }
}

public class TacheTechnique extends DecorateurTache {
  private String technique;
  public TacheTechnique (Tache tache){      super(tache);
  }
  public void setTechnique(String technique) {
     this.technique = technique;
  }

  public String toString() {
     return tache.toString() +" : " + technique;
  }
}
```

De la même manière, on peut définir la classe TacheDuree

c) **(2½ pt)** On applique le pattern **Bridge**

```
  public abstract class CategoriesTache {

     protected Tache tache ;
     public CategoriesTache(Tache tache) { this.tache = tache;
      }
      public abstract int calculCout();

  }

  public class Programmation extends CategoriesTache {
      private final int coefficient = 2;
      public Programmation(Tache tache) {  super(tache);
      }
      public int calculCout(){
         return tache.getCout()* coefficient;
      }
  }
```

De la même manière, on peut définir la classe **Analyse**

Exercice 2 (3 pts)

```
public static void trouver(String nomClasse) {
    Class objetClass = null;
    try {
     objetClass = Class.forName(nomClasse);
    } catch (ClassNotFoundException e ) {
          System.out.println(e);
     }
    Method [] listMethodes = objetClass.getMethods();
    Class<?> [] listParam;
    for (Method m: listMethodes) {
       listParam = m.getParameterTypes();
       if ( (Modifier.isPublic(m.getModifiers()))
          && (listParam.length == 1) && (listParam[0] == int.class )
          && (m.getReturnType() == int.class))
          {
             System.out.println (m.getName());
          }
     }
  }
```

Exercice 3 (7½ pts)

a) (3½ pts)

```
import javax.swing.*;
import java.awt.*;
public class ChatSwing extends JFrame {
  private JTextArea messages =
                          new JTextArea(12, 20);
  private JLabel nom = new JLabel();
  private JTextField texte =
                           new JTextField(15);
  private JButton bOK = new JButton("OK");
  private JPanel p1 = new JPanel();
  private JPanel p2 = new JPanel();
  public ChatSwing(String pseudo) {
     super("Chat de " + pseudo);
     nom.setText(pseudo);
        setDefaultCloseOperation
  (JFrame.EXIT_ON_CLOSE);
  Container c = this.getContentPane();
  c.setLayout(new BorderLayout());
  p1.setLayout( new FlowLayout());
  p2.setLayout(new FlowLayout());
  p1.add(messages);
  c.add (p1, BorderLayout.NORTH);
   p2.add(nom);
   p2.add(texte);
   p2.add(bOK);
   c.add (p2, BorderLayout.SOUTH);
   this.pack();
   this.setVisible(true);
    }
  }
```

b) **(4 pts)**

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import java.util.*;
public class ChatSwing extends JFrame  implements Observer{
   private JTextArea messages = new JTextArea(12, 20);
   private JLabel nom = new JLabel();
   private JTextField texte = new JTextField(15);
   private JButton bOK = new JButton("OK");
   private JPanel p1 = new JPanel();
   private JPanel p2 = new JPanel();
   private Modele modele;
   public ChatSwing(String pseudo, Modele modeleText) {
       super("Chat de " + pseudo);
       modele = modeleText;
       nom.setText(pseudo);
       setDefaultCloseOperation (JFrame.EXIT_ON_CLOSE);
       Container c = this.getContentPane();
       c.setLayout(new BorderLayout());
       p1.setLayout( new FlowLayout());
       p2.setLayout(new FlowLayout());
       p1.add(messages);        c.add (p1, BorderLayout.NORTH);
       p2.add(nom);        p2.add(texte);
       bOK.addActionListener(new ActionListener(){
         public void actionPerformed (ActionEvent event){
            String msg = messages.getText() +nom.getText()+ ">" + texte.getText()+"\n";
            modele.setMessage(msg);
            texte.setText("");
           }
       });
       p2.add(bOK);        c.add (p2, BorderLayout.SOUTH);
       modele.addObserver(this);
       this.pack();        this.setVisible(true);
  }
  public void update(Observable t, Object o)
       { messages.setText(modele.getMessage());
   }
}

Import java.util.*;
public class Modele extends Observable {
   private String message;
    public String getMessage(){return message;}
   public void setMessage(String message){
       this.message = message;
           setChanged();
           notifyObservers();
   }
}
```