Examen (avec document)

Préambule : Répondre de manière concise et précise aux questions. Ne pas mettre de commentaires de documentation sauf s'ils sont nécessaires à la compréhension.

Les exercices sont relativement indépendants.

Barème indicatif :

exercice	1	2.1	2.2	2.3	3.1	3.2	3.3
points	4	3	2	4	3	2	2

L'objectif de ces exercices est de développer une application graphique qui réalise des vérifications sur des classes en signalant des erreurs et des avertissements et produit un rapport XML.

La figure 1 présente l'interface utilisateur (IHM) de l'application. La partie supérieure est constituée d'une ligne de saisie permettant de renseigner le nom de la classe à analyser. Juste en dessous des boutons donnent accès aux différentes vérifications qui peuvent être réalisées. Au centre de la fenêtre, une zone d'affichage permet de visualiser les résultats d'une vérification sur la classe dont le nom est renseigné. Enfin, dans la partie inférieure de la fenêtre, deux boutons permettent respectivement de quitter l'application et de produire un rapport de tous les résultats obtenus au format XML dans un fichier nommé "rapport.xml". Le listing 1 est un exemple d'un tel rapport. Deux classes ont été vérifiées, java.util.Date et java.awt.Point, pour signaler les attributs non privés (erreur) et la présence d'un constructeur par défaut (avertissement).

FIGURE 1 – IHM de l'application

Exercice 1 : Les vérifications

Chaque vérification faite sur une classe sera écrite dans une classe Java dont la spécification est donnée par l'interface Verification (listing 2). La classe Erreur est donnée au listing 3.

1.1 *Pas d'attribut privé.* Écrire une première vérification dans une classe appelée AttributsPrives qui signale une erreur pour chaque attribut déclaré non privé dans la classe, sauf s'il est déclaré final. On rappelle que la classe Modifier propose des méthodes telles que isPrivate,

Listing 1 – Exemple de rapport

```xml
<?xml version="1.0" encoding="ISO-8859-1" ?>
<rapport>
  <classe name="java.awt.Point">
    <check type="AttributsPrives">
      <error>L'attribut public int java.awt.Point.x devrait être privé</error>
      <error>L'attribut public int java.awt.Point.y devrait être privé</error>
    </check>
    <check type="ConstructeurDefaut">
      <warning>Éviter de définir un constructeur par défaut</warning>
    </check>
  </classe>
  <classe name="java.util.Date">
    <check type="AttributsPrives" />
    <check type="ConstructeurDefaut">
      <warning>Éviter de définir un constructeur par défaut</warning>
    </check>
  </classe>
</rapport>
```

Listing 2 – L'interface Verification

```java
import java.util.Collection;
public interface Verification {
    /** Évaluer cette propriété sur la classe.
      * @param c la classe à vérifier
      * @return les erreurs détectées
      */
    Collection<Erreur> erreursDetectee(Class<?> c);
}
```

Listing 3 – La classe Erreur

```java
public class Erreur {
    private boolean erreur; // warning si false
    private String message; // explication de l'erreur

    public Erreur(String message) { this(message, true); }
    public Erreur(String message, boolean erreur) {
        this.message = message;
        this.erreur = erreur;
    }
    public boolean estErreur() { return this.erreur; }
    public boolean estWarning() { return ! this.erreur; }
    public String getMessage() { return this.message; }

    @Override public String toString() {
        String nature = this.estWarning() ? "warning" : "error";
        return "[" + nature + "] " + this.getMessage();
}   }
```

isFinal, etc. qui prennent en paramètre un entier correspondant au résultat de getModifiers() appliquée sur l'objet représentant un attribut.

1.2 *Pas de constructeur sans paramètre.* Écrire une vérification (classe ConstructeurDefaut) qui signale un avertissement si un constructeur sans paramètre est défini sur la classe.

Exercice 2 : Interface utilisateur

Le listing 4 est le squelette de la classe Main correspondant à l'interface utilisateur de la figure 1.

2.1 Compléter le code pour obtenir l'interface utilisateur de la figure 1.

2.2 Rendre actif le bouton Quitter.

2.3 Rendre actifs les boutons correspondant à la vérification d'une propriété. Si une erreur se produit lors de l'exécution de la vérification (par exemple parce que la vérification n'existe pas, comme pour Autre), un message d'erreur sera affiché dans la zone centrale de la fenêtre, à la place du résultat normal.

Exercice 3 : Génération du rapport

On s'intéresse maintenant à la génération du rapport XML.

3.1 Compléter l'application pour conserver pour chaque classe et pour chaque vérification le résultat obtenu. On précisera le type de structure de données à utiliser et les modifications à apporter au code déjà écrit.

3.2 Rendre actif le bouton Rapport pour qu'il provoque la création d'un fichier XML contenant les résulats calculés. Un exemple est donné au listing 1.

3.3 Donner une DTD possible pour les rapports XML.

Listing 4 – Le squelette de la classe Main

```
1  import java.awt.*;
2  import java.awt.event.*;
3  import javax.swing.*;
4  import java.io.*;
5  import org.jdom.*;
6  import org.jdom.output.*;
7
8  public class Main {
9      private JTextField nomClasse = new JTextField(20);
10     private JTextArea texte = new JTextArea(10, 20);
11
12     public Main(String[] noms) {
13         final JFrame fenetre = new JFrame("Vérificateur");
14         this.texte.setEditable(false);
15
16
17         fenetre.pack();
18         fenetre.setVisible(true);
19     }
20
21
22     public void ecrire(Document document, OutputStream out) {
23         try {
24             XMLOutputter sortie = new XMLOutputter(Format.getPrettyFormat());
25             sortie.output(document, out);
26         } catch (java.io.IOException e) {
27             throw new RuntimeException("Erreur␣sur␣écriture␣:", e);
28         }
29     }
30
31     public static void main(final String[] args) {
32         EventQueue.invokeLater(new Runnable() {
33             public void run() {
34                 if (args.length != 0) {
35                     new Main(args);
36                 } else {
37                     new Main(new String[] { "AttributsPrives",
38                             "ConstructeurDefaut", "Autre" });
39                 }
40             }
41         });
42     }
43
44 }
```