Techniques des Logiciels Interactifs

durée : **3h**
documents autorisés : **1 feuille A4 recto-verso manuscrite de votre main**

Commencez par lire l'intégralité du sujet en détail.
Le barème est **indicatif**. La qualité de la **présentation**, de l'**expression**, ainsi que la **précision** et la **concision** seront prises en compte dans la notation de **manière significative**. Si le sujet présente des ambiguïtés, précisez vos choix. Lorsqu'il est demandé de produire du code, ne vous focalisez pas sur la correction de la syntaxe.

Généralités (6 points)

Question 1

Le code d'une application interactive doit être organisé, à minima, en deux modules.
a) Comment se nomment ces deux parties ?
b) Donner deux raisons qui justifient cette organisation.
c) Quelle(s) dépendance(s) peu(ven)t exister entre ces deux modules ?

Question 2

Dans le projet sur lequel votre équipe travaille apparaissent les codes suivants :

```
package downloader.fc;

public class Downloader extends ... { [...]
      public Downloader(String uri) { [...] }
      public String download() throws InterruptedException {
            [...]
            while(true) {
                  // download the file chunk by chunk
                  [...]
                  setProgress(100*size/content_length);
            }
            return filename;
      }
}
```

```
package downloader.ui;
import downloader.fc.Downloader;

class Main extends ... {
   [...]
   public Main(String title) {
      [...]
      JTextField url_entry = new JTextField();
      JButton add_button = new JButton("add");
      add_button.addActionListener(new ActionListener() {
         public void actionPerformed(ActionEvent e) {
            Downloader downloader = new Downloader(url_entry.getText());
            downloader.download();
         }
      });
   }
[...]
}
```

a) Quand on clique sur le bouton `add_button`, l'interface graphique se fige complètement pendant un long moment et le système indique que l'application ne répond plus. Comment cela s'explique-t-il ?
b) Comment modifier ce code pour résoudre ce problème ?
c) Quel(s) nouveau(x) problème(s) cette solution pose-t-elle ?

Programmation par événements (9 points)

Question 3

Vous devez reprendre et améliorer le code qui permet de contrôler un lecteur multimédia depuis les boutons d'un casque audio. Le volume est controlé par deux boutons (un pour monter et l'autre pour baisser le niveau sonore). Ils fonctionnent tous les deux de la même façon, par exemple pour le bouton qui permet d'augmenter le volume :

- quand on appuie et qu'on relâche le bouton en moins de 500 ms, le volume est monté d'un cran ;
- quand on maintient le bouton enfoncé plus de 500 ms, le volume augmente d'un cran toutes les 500 ms jusqu'à ce qu'on relâche le bouton.

Le projet comporte le code suivant qui implémente ce comportement :

```
class VolumeController implements ButtonListener implements ActionListener {
   Timer timer = new Timer(500, this);
   boolean repeating = false;

   public void buttonPressed(ButtonEvent e) {
      repeating = false;
      timer.restart();
   }

   public void buttonReleased(ButtonEvent e)    {
      if(!repeating) {
         volume_up();
      }
      timer.stop();
   }

   public void actionPerformed(ActionEvent e) {
      repeating = true;
      volume_up();
   }
}
```

Ce code utilise un minuteur fournit par la classe `Timer`. Elle notifie les `ActionListener` qui lui sont abonnés (ici le `VolumeController` lui-même car il est passé en argument du constructeur, c.f. deuxième ligne du code). Une notification (`ActionEvent` reçu dans `actionPerformed`) est envoyée périodiquement (ici toutes les 500 ms, premier argument du constructeur) dès qu'on appelle `timer.restart()`, et ce jusqu'à ce qu'on appelle `timer.stop()`.

On admet que les interfaces suivantes sont fournies par Java :

```
public interface ButtonListener {
      public void buttonPressed(ButtonEvent e);
      public void buttonReleased(ButtonEvent e);
}
```

```
public interface ActionListener {
      public void actionPerformed(MouseEvent e);
}
```

La fonction `volume_up()` est fournie par une classe englobante qui fait l'interface avec le noyau fonctionnel dont on admettra l'existence.

a) Dessinez la machine à état qui modélise ce comportement. On notera les 3 types d'évènements ainsi : enfoncement du bouton (↓), relâchement du bouton (↑) et déclenchement du minuteur (T).
b) Réimplémentez cette machine à états en utilisant la méthodologie à base d'énumération vue en cours.

Un autre bouton sur le casque permet de contrôler la lecture des morceaux de musique. Il fonctionne de la manière suivante :

- si on appuie et qu'on relâche le bouton, la lecture commence ou s'arrête (en alternance, géré par la fonction `play_pause()`); sauf
- si on appuie et qu'on relâche le bouton 2 fois rapidement (*i.e.* si le deuxième appui a lieu moins de 500 ms après le premier), on passe au morceau suivant grâce à la fonction `next()` ; sauf
- si on appuie et qu'on relâche le bouton 3 fois rapidement (*i.e.* si en plus le troisième appui a lieu moins de 500 ms après le second), on revient début du morceau ou au morceau précédent grâce à la fonction `prev()`.

c) Dessinez la machine à état qui modélise juste lecture/pause, puis une seconde machine qui ajoute le passage au morceau suivant, puis enfin une troisième machine qui ajoute le retour au début.

Avec ce même bouton, on peut également avancer et reculer dans le morceau en cours de lecture en faisant un double (pour avancer) ou triple (pour reculer) appui qu'on maintient enfoncé à la fin. Pendant que le bouton est maintenu enfoncé, la lecture avance ou recule de 10 s (grâce à `skip(10)` ou `skip(-10)`) toutes les 500 ms.

d) Dessinez la machine à état qui combine ces interactions avec les 3 autres vues ci-dessus.

Composants interactifs (5 points)

Question 5

Les figures ci-dessous donnent la fenêtre principale de l'application iReal Pro sous MacOS avant et après un redimensionnement. Sachant que la largeur de la colonne de gauche peut être modifiée par l'utilisateur, donnez l'arbre des composants que vous utiliseriez pour coder cette interface avec Java/SWING en spécifiant si nécessaire pour les conteneurs les gestionnaires de géométrie utilisés et la localisation de leurs enfants. Ne détaillez pas le contenu de la vue centrale ni celui de la liste à gauche. Les éléments de contrôle sous la partition sont interactifs à l'exception de l'indication de temps 00:00. Le premier et le dernier ouvrent une liste qui permet de choisir le style d'accompagnement et la tonalité du morceau.