

---

# Techniques des Logiciels Interactifs

durée : 1h30

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

- Comment distinguer ce qui fait partie du noyau fonctionnel de ce qui fait partie de l'interface d'une application ?
- Expliquez pourquoi il faut séparer le code du noyau fonctionnel de celui de l'interface.
- Expliquez pourquoi le mécanisme des *callbacks* (fonctions de rappel) est important dans cette séparation.
- Selon Fekete [1996], le noyau fonctionnel doit fournir les trois services suivants : notification, prévention des erreurs, et annulation. Dans une application de client mail, donnez un exemple d'interaction rendue possible pour chacun de ces services.

---

## Programmation par événements (9 points)

---

### Question 2

On veut combiner des interactions qui existent dans le cadre contraint de l'interaction sur écran tactile pour lesquels on n'a pas de boutons gauche et droit, et en se concentrant sur de l'interaction à un doigt. On a donc que 3 types d'évènements : l'arrivée du doigt au contact de l'écran (↓), son déplacement (D), et le départ du doigt (↑).

Dans un navigateur web, on souhaite combiner les trois interactions suivantes :

- le clic (↓ suivi de ↑) doit permettre l'ouverture des liens (géré par une fonction `open_link(Point p)` donnée).
- le drag (↓ suivi de un ou plusieurs D, puis un↑) doit permettre de faire défiler la page verticalement (géré par une fonction `scroll_page(int dY)` donnée).
- l'appui long (↓ maintenu immobile pendant au moins 500ms) ouvre un menu contextuel (géré par une fonction `open_menu(Point p)` donnée). Une fois qu'il est ouvert, le déplacement du doigt (D) met à jour la commande sélectionnée du menu (géré par une fonction `highlight_menu(Point p)` donnée) et le relâché (↑) active cette commande sélectionnée (géré par une fonction `select_menu(Point p)` donnée). Si on relâche le doigt avant l'ouverture du menu, il ne se passe rien.

a) **Dessinez les 3 machines à états qui spécifient ces interactions.**

On considère ici les trois interactions prises séparément. On supposera que les évènements souris (↓, D et ↑) ont une méthode `getPoint()` qui retourne un `Point` avec la position du curseur, et des méthodes `getX()` et `getY()` qui en retournent les coordonnées séparément si besoin. Les instances de `Point` ont également des méthodes `getX()` et `getY()`.

Pour gérer le délais de l'appui long, on peut appeler la méthode `timer.restart()` pour programmer un évènement qui arrivera 500ms plus tard. 500 millisecondes après que cette méthode soit appelée, votre machine à états recevra un évènement qu'on notera (T).

**b) Dessinez une machine à états qui combine ces trois interactions.**

Combinez ces trois machines à états pour que les trois interactions puissent cohabiter. Pour que le clic ne soit pas confondu avec un drag sans déplacement, on débutera un drag seulement si le curseur a parcouru une distance de plus de 5 pixels par rapport au point où le doigt est arrivé en contact avec l'écran.

Pour calculer cette distance, les instances de `Point` ont une méthode `distance(Point p)` qui donne leur distance à un autre point.

**c) Réalisez en java cette machine à états.**

Pour cela, **vous est donnée une classe `Canvas`** qui utilisera votre classe `Interaction`. C'est la classe `Interaction` vous devez réaliser.

```
package exam.ui;

import javax.swing.JComponent;
import java.awt.Point;

public class Canvas extends JComponent {
    Interaction interaction;
    public Canvas() {
        interaction = new Interaction(this);
    }

    public void open_link(Point start) { ... }
    public void scroll_page(int dy) { ... }

    public void open_menu(Point start) { ... }
    public void highlight_menu(Point p) { ... }
    public void select_menu(Point end) { ... }
}
```

**Est également fournie une classe `Timer500`** gérant le timer dont il vous faudra utiliser une instance qui peut être réemployée à chaque fois que nécessaire :

```
import javax.swing.Timer;
public class Timer500 extends Timer {
    public Timer500() { ... }
    public void addActionListener(ActionListener listener) { ... }
    public void restart() { ... }
}
```

La méthode `addActionListener` permet de s'abonner pour recevoir un évènement (de type `ActionEvent`) quand le timer expirera. Il faut pour cela implémenter l'interface `ActionListener`, qui est donnée par Java :

```
package java.awt.event;

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

On considèrera que les doigts envoient des évènements `MouseEvent` standards utilisés par Java pour la souris. pour écouter la souris on doit implémenter les interfaces suivantes (l'interface `MouseListener` qui réunit ces deux interfaces est également disponible) :

```
public interface MouseListener {
    public void mousePressed(MouseEvent e);
    public void mouseReleased(MouseEvent e);
    ...
}

public interface MouseMotionListener {
    public void mouseDragged(MouseEvent e);
    ...
}
```

**Votre classe `Interaction` pourra commencer ainsi :**

```
package exam.ui;

public class Interaction ... {
    protected Canvas canvas;
    protected Timer500 timer = new Timer500();

    public Interaction(Canvas canvas) {
        this.canvas = canvas;
        ...
    }
    ...
}
```

## Composants interactifs (5 points)

### Question 3

Les figures ci-dessous donnent la fenêtre principale de l'application Preview 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.

