Techniques des Logiciels Interactifs

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

consignes :
Le barème est **indicatif**. La qualité de la **présentation**, de l'**expression**, de l'**orthographe** sera prise en compte dans la notation de **manière significative**.
Si le sujet présente des ambiguïtés, précisez vos choix. Il sera tenu compte de vos hypothèses. Lorsqu'il est demandé de produire du code, ne vous focalisez pas sur la correction de la syntaxe.

Généralités (6 points)

Question 1

a) La loi de Moore (les capacités des ordinateurs doublent à intervalles de temps constants) a des implications pour l'Interaction Homme-Machine. Expliquez en quoi elle amène à reconsidérer les priorités lors du développement d'une application interactive.
b) Selon J.-D. Fekete [1996], le noyau fonctionnel doit fournir les trois services suivants : notification, prévention des erreurs, et annulation. Expliquez pourquoi en donnant un exemple d'interaction impossible ou difficile à réaliser pour chacun de ces services lorsqu'il est absent.
c) Dans les années 80, B. Shneiderman définissait la "manipulation directe" comme l'utilisation de métaphores du monde réel au sein de l'interface, en particulier le fait que les objets d'intérêt aient des représentations continues, et que l'utilisateur agisse dessus par le biais d'actions physiques, ces opérations devant être rapides, incrémentales et réversibles. En quoi les services du noyau fonctionnel énumérés par Fekete favorise la manipulation directe ?

Question 2

Vous faites partie d'une équipe chargée de développer une nouvelle interface pour un logiciel existant. Dans le code du noyau fonctionnel, vous découvrez le passage suivant :

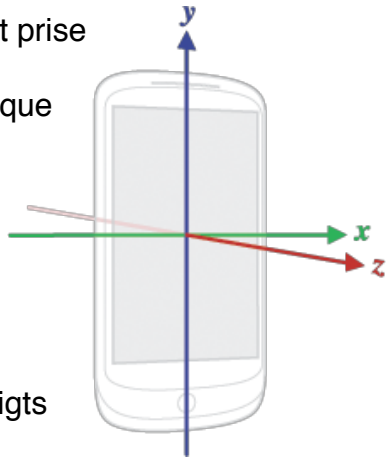
```
package nf;

class UIUpdater {
   ...
   public void loop() {
      while(not ended) {
         // met à jour l'affichage de la valeur mesurée par le capteur
         ui.sensor_label.setText(sensor.getValue().toString());
         // attend 1/10 de seconde avant de remesurer
         Thread.sleep(100);
      }
   }
   ...
}
```

a) Quels sont les problèmes que vous identifiez dans ce code ?
b) Quelles modifications allez-vous suggérer à l'équipe chargée du noyau fonctionnel ?

Programmation par événements (8 points)

Question 3

Un des problèmes des dispositifs mobiles à écran tactile (tablettes, téléphones, etc.) est l'absence du mode «survol» du pointeur : alors qu'avec la souris on peut positionner le curseur sur un objet sans forcement enfoncer de bouton, le tactile ne permet pas de survoler un objet sans contact avec l'écran. Cette limitation fait qu'il n'existe pas pour les mouvements avec le tactile de distinction équivalente à celle qui existe entre les mouvements avec bouton enfoncé *(drag)* et sans bouton enfoncé *(motion)* pour la souris. Pour pallier ce manque la technique TouchOver qui utilise l'inclinaison du dispositif (angle de rotation autour de l'axe des X lorsqu'il est en orientation portrait ; cf. schéma) pour simuler un bouton :



- quand le doigt est posé sur l'écran, l'inclinaison du mobile est prise comme référence ;
- lorsque le doigt bouge, l'application se comporte comme lorsque la souris bouge sans bouton enfoncé ;
- lorsque l'inclinaison dépasse de plus de 10 degrés l'inclinaison initiale, l'application se comporte comme si le bouton gauche venait d'être enfoncé ;
- lorsque l'inclinaison revient à une valeur inférieure à celle mesurée initialement, l'application se comporte comme si le bouton venait d'être relâché ; et
- entre ces deux derniers évènements, les mouvements de doigts sont interprétés comme un déplacement de souris, bouton gauche enfoncé (e.g. un *drag*).

a) En considérant les évènements : ↓ (doigt qui arrive en contact de l'écran) ; ↑ (doigt qui repart) ; M (doigt qui bouge sur l'écran) ; et RX (rotation du mobile autour de son axe des X), dessinez la machine à états qui déclenche les actions suivantes selon la spécification informelle donnée ci-dessus : `move(Point p)` ; `drag(Point p)` ; `press(Point p)` ; et `release(Point p)` (resp. curseur qui bouge sans/avec bouton enfoncé ; bouton gauche enfoncé/relâché).

Les évènements ↓, ↑ et M ont une méthode `Point getPoint()` qui donne la position du doigt sur l'écran et l'évènement RX une méthode `int getPitch()` qui donne l'angle du mobile autour de l'axe des X en degrés.

b) Réalisez en java cette machine à états, en considérant que les évènements des doigts et de l'inclinaison sont gérés grâce aux interfaces `FingerListener` et `OrientationListener` décrites ci-dessous (elles disposent chacune d'un adaptateur associé `XxxxAdapter` pour `XxxxListener`) :

```
public interface FingerListener {
      public void fingerPressed(FingerEvent e);
      public void fingerReleased(FingerEvent e);
      public void fingerMoved(FingerEvent e);
}
public interface OrientationListener {
      public void orientationChanged(OrientationEvent e);
}
```

Les `FingerEvent` ont une méthode `Point getPoint()` et les `OrientationEvent` ont une méthode `int getPitch()`.

Composants interactifs (4 points)

Question 4

Les figures ci-dessous donnent la fenêtre des contacts de l'application Skype sous Mac OS X avant et après un redimensionnement. Sachant que la largeur de la colonne de gauche peut être modifiée par l'utilisateur, donnez l'arbres des widgets 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 fils.
Certains aspects ne peuvent être rendus simplement avec SWING : les boutons collés dans la barre de boutons pourront être considérés comme un seul bouton.
Ne détaillez pas le contenu de la table des contacts.