Université Montpellier II
Conception et Programmation par objets (HLIN 505)

TD/TP - Événements

Exercice 1. Écoute des événements "souris".

(a) Écrire une application, composée d'une fenêtre cadre contenant un panneau, qui permet d'écouter certains événements en provenance de la souris, en respectant les consignes suivantes :

- La classe de la fenêtre hérite de la classe JFrame.
- Le panneau est directement une instance de JPanel.
- C'est la fenêtre qui écoute les événements souris.

Plus précisément, on s'intéresse aux actions suivantes :
- l'utilisateur a appuyé sur un bouton de la souris (action "mouse pressed")
- l'utilisateur a relâché un bouton de la souris (action "mouse released")
- l'utilisateur a cliqué sur un bouton de la souris, c'est-à-dire appuyé puis relâché un bouton de la souris, sans déplacer la souris (action "mouse clicked").

Lorsque l'utilisateur effectue l'une de ces trois actions, on veut voir affiché dans la fenêtre console le type d'action suivi des coordonnées de la souris sur le panneau lors de l'événement.

Les méthodes correspondantes sont les méthodes mousePressed, mouseReleased et mouseClicked de l'interface **java.awt.event.MouseListener**. Cette interface est munie de la classe adapteur **MouseAdapter**. Notez qu'il existe une autre interface permettant d'écouter des événements souris lorsque la souris se déplace : MouseMotionListener (nous l'utiliserons plus tard). De plus, Java 1.5 a introduit l'interface **MouseInputListener**, qui regroupe les méthodes des deux interfaces MouseListener et MouseMotionListener.

Un événement de type MouseEvent contient les coordonnées de la souris au moment de l'action (accesseurs getX() et getY()).

Vérifier qu'un événement correspondant à un clic n'est généré que si l'appui-relâchement d'un bouton s'est fait sans déplacer la souris (autrement dit, lorsque les points associés aux actions "mouse pressed" et "mouse released" sont différents, il n'y a pas d'action "mouse clicked").

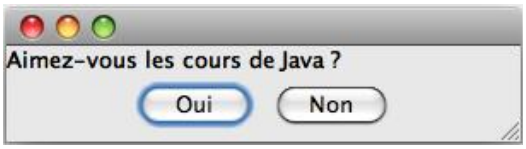
(b) Ajouter une écriture console qui indique le nombre de clics exécuté en un même point
(*ne faites pas de comparaisons de points*, l'événement généré par un clic contient cette information : il ne vous reste qu'à trouver la méthode appropriée dans la classe MouseEvent).

(c) Dans votre méthode main, créez plusieurs fenêtres (instances de la même classe). Constatez que ces fenêtres partagent la même fenêtre console.

(d) Lors d'une écriture console, on veut indiquer de quelle fenêtre provient l'écriture. Pour cela on ajoute :
- un attribut (d'instance) associant à chaque instance de fenêtre un numéro
- un attribut de classe (statique) dans la classe fenêtre, permet de compter le nombre de fenêtres créées. Le numéro associé à une fenêtre est son rang de création.

Exercice 2. Aimez-vous les cours de Java ?

On souhaite écrire une application qui affiche la fenêtre suivante :



Lorsque l'utilisateur clique sur le bouton « OUI » la boite de dialogue suivante s'affiche :

Par contre, on ne veut pas que l'utilisateur puisse cliquer sur le bouton « NON » (car nous préférons penser que, comme vous, il aime les cours de Java). Ainsi, dès que la souris arrive sur le bouton « NON », les deux boutons doivent inverser leurs places avant que l'utilisateur n'ait le temps de cliquer.

Indications :

- pour détecter l'arrivée de la souris sur un composant, utiliser la méthode `mouseEntered` de l'interface `MouseListener`.
- pour ouvrir une boite de dialogue, utiliser la méthode statique `showMessageDialog()` de la classe `JOptionPane`.
- pour inverser les boutons d'un panneau, il suffit de les enlever du panneau (méthode `remove(`*`composant`*`)`) puis de les rajouter (méthode `add(`*`composant`*`)`) dans l'ordre inverse. Il faut ensuite appeler la méthode `validate()` sur le panneau : cette méthode avertit le LayoutManager du panneau qu'il doit redisposer ses composants.

Exercice 3. Du texte transformé en majuscule dans un formulaire

Écrire une application avec une interface graphique composée de deux étiquettes (JLabel) et deux champs de texte (JTextField) permettant de saisir un nom et un prénom.
Cas 1 : Transformer, à la volée, en majuscules, les lettres saisies par l'utilisateur dans le champ « Nom ».
Indications :
- Utiliser ici un écouteur de type KeyListener.
- Implémenter la méthode keyTyped(KeyEvent e) de l'interface KeyListener (ou étendre la classe KeyAdapter).
- Pour récupérer le caractère de la touche sur laquelle l'utilisateur a appuyé, invoquer la méthode getKeyChar() de l'objet KeyEvent. Celle-ci retourne une valeur de type char.
- Pour empêcher l'affichage d'un caractère saisi par l'utilisateur, invoquer consume() de l'objet KeyEvent.

Cas 2 : Transformer en majuscules ces lettres lorsque l'utilisateur quitte le champ « Nom ».
Indication :

- Utiliser ici un écouteur de type FocusListener qui doit implémenter une des deux méthodes : focusGained(FocusEvent e) ou focusLost(FocusEvent e).

Exercice 4. Interdire le copier/coller lors d'une saisie de mot de passe

Reprendre l'interface graphique créée précédemment et ajouter deux champs de mot de passe (JPasswordField) étiquetés : « Nouveau mot de passe » et « Re-saisir Nouveau mot de passe »

Interdire à l'utilisateur de faire un copier-coller du nouveau mot de passe saisi (Ctrl-C puis Ctrl-V).
Indications :
- Utiliser un écouteur de type KeyListener et implémenter seulement les deux méthodes keyPressed(...) et keyReleased(...). Pas besoin d'implémenter la méthode keyTyped(...)
- En utilisant l'objet KeyEvent, qui est le paramètre reçu par les méthodes de l'écouteur, on peut obtenir le caractère de la touche (getKeyChar()), ou le code du caractère (getKeyCode()). Ce dernier représente un entier qui est égal à l'une des constantes de la classe KeyEvent (voir sa documentation). Par exemple, si l'utilisateur appuie sur la touche « *Control* », l'invocation de la méthode getKeyCode() sur l'objet KeyEvent renvoie une valeur égale à : KeyEvent.VK_CONTROL.
- Swing effectue réellement le copier-coller après que l'utilisateur ait appuyé sur la touche 'v' (au moment où la méthode keyPressed(...) s'exécute, c'est à dire au moment de l'appui sur la la touche 'v' et non après relâchement de cette touche)
- Swing interdit déjà le Ctrl-C sur un champ de type JPasswordField. Il suffit de gérer le cas du Ctrl-V sur les deux champs de mot de passe. L'objectif est d'empêcher un « coller » qui fait suite à un « copier » effectué à partir d'un autre champs de texte ou en dehors de l'interface graphique.

Émettre un « son système », lorsque l'utilisateur clique sur Ctrl-V sur les deux champs de mot de passe :
```
Toolkit tk = Toolkit.getDefaultToolkit();
tk.beep();
```

Rendre le code portable sur tous les OS (Window, Mac et Linux). Pour faire un copier-coller sous Mac OS, il faudra clicker sur Meta-C et Meta-V.

Indications :
- Le code de la touche Meta est : KeyEvent.VK_META
- Pour récupérer le nom de l'OS (sous la forme d'une chaine de caractères), il faudra écrire : System.getProperty("os.name")

Exercice 5. Afficher/Masquer un mot de passe

Écrire une application ayant une interface graphique pour entrer le nom d'utilisateur et le mot de passe.

Ajouter à cette interface graphique une case à cocher (JCheckBox) suivie de l'étiquette (JLabel) : « Afficher/masquer le mot de passe ».

Implémenter l'affichage et le masquage du mot de passe en associant, au composant « case à cocher », un écouteur d'événement de type MouseListener.

Indications :
- Si l'utilisateur clique sur la case à cocher pour afficher le mot de passe, créer un composant de type JTextField et le remplir avec le texte qui se trouve dans le composant JPasswordField. Supprimer

ensuite le composant JPasswordField et le remplacer par le composant JTextField créé précédemment. Faire l'inverse, si l'utilisateur souhaite masquer le mot de passe.
- Mettre le composant JPasswordField seul dans un panneau pour que son remplacement par le composant JTextField (ou l'inverse) se fasse facilement.
- Invoquer la méthode updateUI() sur le panneau sur lequel le remplacement a été effectué, pour que le rafraichissement de l'affichage de l'interface graphique soit effectif.

Exercice 6. Pas de labels pour les champs de texte

Écrire un formulaire de saisie d'informations concernant un utilisateur : nom, prénom et adresse email. Ne pas mettre des labels avant les champ de texte. Les remplacer par du texte écrit directement sur les champs.
Par exemple, ajouter le texte suivant sur le champ de saisie du prénom : « Votre prénom ici ». Celui-ci s'affichera lors de la première visualisation de la fenêtre.
Dès que l'utilisateur clique sur ce champ de texte, le texte « Votre prénom ici » doit disparaître. Faire la même chose pour les autres champs.

Exercice 7. Éditeur de carrés

Complétez l'éditeur de carrés commencé pendant le cours (partie 2).
Cet éditeur se compose d'une fenêtre intitulée "Editeur de carrés" de taille 500 par 500 au départ.
Lorsque vous pressez la souris dans cette fenêtre à un endroit vide (hors des carrés existants), un nouveau carré de dimension 25 par 25 est ajouté au point où vous avez cliqué.
Lorsque vous double-cliquez sur un carré, il doit s'effacer.
Lorsque vous pressez la souris sur un carré, celui-ci suit la souris et se déplace avec jusqu'au point où vous relâchez la souris.

Regardez pour cela dans l'API la classe **`MouseAdapter`** et l'interface **`MouseMotionListener`** qui seront spécialisées pour définir vos écouteurs de souris, ainsi que les méthodes disponibles dans **`MouseEvent`**.

Exercice 8. La calculatrice

Animez la mini-calculatrice que vous avez dessinée au TP précédent.