

Plan de la présentation

- Les actions utilisateurs
- Les familles d'événements en Java
- Les écouteurs d'événements
- Aide Sun pour la gestion d'événements "[awt](#)" et "[swing](#)" de Java

Actions possibles sur une IHM

- Un utilisateur agit sur une IHM en utilisant le clavier ou la souris
- Le programmeur peut agir directement sur une IHM
- Chaque action donne lieu à la création d' "événements" par la machine virtuelle Java

Actions utilisateurs sur une IHM

- Dans le langage Java, ces événements ont été classés par familles.
- Ces familles ont été créées en fonction de deux critères :
 - Le type des composants : fenêtre, bouton, ascenseur, ...
 - Le type du périphérique : souris ou clavier, ...

La famille d'événements Souris (awt)

- On presse sur le bouton de souris
- On relache sur le bouton de souris
- On clique sur la souris
- Le curseur de la souris quitte un composant
- Le curseur de la souris entre dans un composant

La famille d'événements de déplacement de souris (awt)

- La souris est déplacée avec un bouton de la souris pressé
- La souris est déplacée sans aucun bouton de la souris pressé

La famille relative à une action sur un composant

- Il n'y a qu'un seul événement dans cette famille : "Action sur le composant"
- Elle ne concerne que les composants :
 - **Button** : on veut déclencher l'action associé au bouton grâce à un **click bouton**
 - **TextField** : on veut valider le champ de saisie à l'aide de la **touche entrée**
 - **MenuItem** : on veut déclencher l'action associé à l'item grâce à un **click bouton**

La famille des événements d'une fenêtre

- La fenêtre devient active : elle gagne le focus
- La fenêtre devient inactive : elle perd le focus
- La fenêtre est ouverte pour la première fois
- La fenêtre est iconifiée
- La fenêtre est désiconifiée
- L'utilisateur clique sur la croix de fermeture de la fenêtre
- La fenêtre est dissociée de la tâche graphique

La famille des événements d'un conteneur

- On ajoute un composant dans le conteneur
- On retire un composant du conteneur

La famille des événements du clavier

- On presse sur une touche du clavier
- On relache une touche du clavier
- On tape sur une touche du clavier

Interfaces de traitements d'événements

- Chacune de ces familles est associée dans Java à une interface qui a autant de prototypages que d'événements dans la famille
- Chaque prototypage correspond à la signature du futur traitement à effectuer quand l'événement associé se produit

L'interface `MouseListener`

void **mouseClicked**(MouseEvent e)

void **mouseEntered**(MouseEvent e)

void **mouseExited**(MouseEvent e)

void **mousePressed**(MouseEvent e)

void **mouseReleased**(MouseEvent e)

L'interface `MouseEventListener`

void **mouseDragged**(MouseEvent e)

void **mouseMoved**(MouseEvent e)

L'interface ActionListener

void **actionPerformed**(ActionEvent e)

L'interface WindowListener

void **windowActivated**(WindowEvent e)

void **windowClosed**(WindowEvent e)

void **windowClosing**(WindowEvent e)

void **windowDeactivated**(WindowEvent e)

void **windowDeiconified**(WindowEvent e)

void **windowIconified**(WindowEvent e)

void **windowOpened**(WindowEvent e)

L'interface ContainerListener

void **componentAdded**(ContainerEvent e)

void **componentRemoved**(ContainerEvent e)

L'interface `KeyListener`

void **keyPressed**(KeyEvent e)

void **keyReleased**(KeyEvent e)

void **keyTyped**(KeyEvent e)

Principe de gestion d'événements

Le programmeur doit d'abord pour chaque composant graphique de l'IHM :

- 1) recenser les événements à traiter
- 2) trouver pour chaque événement, l'interface de gestion d'événements associée

Principe de gestion d'événements

Puis le programmeur doit :

- 1) créer des classes "écouteurs" qui implémentent les différentes interfaces de gestion d'événements
- 2) Lors de la fabrication de l'IHM, associer aux composants concernés des objets "écouteurs"

Principe de gestion d'événements

Puis le programmeur doit :

- 1) créer des classes "écouteurs" qui implémentent les différentes interfaces de gestion d'événements
- 2) Lors de la fabrication de l'IHM, associer aux composants concernés des objets "écouteurs"

Exemple de gestion d'événements



Avant avoir
cliqué sur
Bouton 1



Après avoir
cliqué sur
Bouton 1

Exemple de gestion d'événements

- Le programmeur :
 - a créé une classe écouteur qui réagit à un click bouton sur un bouton
 - a alloué deux objets de cette classe
 - a associé le premier écouteur au premier bouton et le deuxième écouteur au deuxième bouton

Exemple de gestion d'événements

```
class EcouteurBouton implements ActionListener {  
  
    private JLabel jlb;  
  
    public EcouteurBouton(JLabel par_jlb) {  
        jlb = par_jlb;  
    }  
  
    public void actionPerformed(ActionEvent e) {  
        JButton jb = (JButton) e.getSource();  
  
        jlb.setText("click sur "+jb.getText());  
    }  
}
```

Exemple de gestion d'événements

```
public class Ex1Evt extends JFrame {
    JButton jb_1, jb_2;
    JLabel jlb_1;

    public Ex1Evt() {
        super("Gestion de bouton");
        this.setLayout(new
        GridLayout(3,1));
        this.add(jb_1 = new
        JButton("Bouton1"));
        this.add(jb_2 = new
        JButton("Bouton2"));
        this.add(jlb_1 = new
        JLabel("",JLabel.CENTER));
    }
}
```

```
EcouteurBouton eb = new
EcouteurBouton(jlb_1);
    jb_1.addActionListener(eb);
    jb_2.addActionListener(eb);
    this.pack();
    this.setVisible(true);
}

public static void main(String []
args) {
    Ex1Evt ee = new Ex1Evt();
}
}
```

La gestion d'évènement avec Java

