


ANNEXE 6 - Ce qu'il faut savoir sur les JCheckBox et les JRadioButton

Les cases à cocher (JCheckBox) 

1) Lorsqu'un usager clique sur un JCheckBox, quel type d'objet événementiel sera généré ?

A) un WindowEvent B) un ItemEvent C) un ItemListener D) un CaretEvent

2) Cet objet pourra être reçu par quel type d'écouteur ?

A) un ActionListener B) un MouseListener C) un ItemListener D) un CheckListener

3) On inscrira les sources JCheckBox à cet écouteur avec la méthode : _____

4) Pour mettre en œuvre l'interface ItemListener, on doit coder quelle méthode :

5) Qu'est ce que ça signifie ?

6) important : un ActionEvent est également généré lorsqu'on clique sur un JCheckBox. Pourquoi ?

- A) car l'actionEvent correspond à l'événement principal qui peut se produire avec un JCheckBox, soit cliquer dessus
- B) car l'actionEvent est généré automatiquement sur toutes les composantes SWING
- C) car un JCheckBox est une sous-classe de JButton

*** On peut donc gérer un JCheckBox avec un écouteur qui met en œuvre un ActionListener (donc coder la méthode actionPerformed)

7) Pour savoir si un objet JCheckBox est sélectionné ou pas

- **À partir de la source :**

```
jCheckBox1.isSelected ()
```

- **À partir de l'objet événementiel ItemEvent :**

```
e.getStateChanged() == ItemEvent.SELECTED ou DESELECTED
```

~~~~~

**Les boutons radio ( JRadioButton )**

- sont habituellement placés en groupe, utilisés lorsque la situation impose qu'un seul des boutons puisse être sélectionné à la fois. ( la sélection d'un bouton impose la « désélection » de tous les autres. )
- cette relation logique est conservée à l'aide d'un objet `ButtonGroup` provenant du package `javax.swing`

**Exercice : placer deux JRadioButton sur un GUI vide à l'aide de la palette de composantes**

Code :

```
① JRadioButton homme = new JRadioButton ( "homme", true );
JRadioButton femme = new JRadioButton ( "femme", false);
```

```
② ButtonGroup groupe = new ButtonGroup();
```

dans la méthode `jbInit` , ajouter les boutons au groupe :

```
groupe.add(homme);
groupe.add(femme);
```

- ① La création des boutons dans le UIEditor utilisera le constructeur par défaut, vous devrez rajouter à la main les paramètres. Dans ce cas-ci, le bouton *homme* sera sélectionné par défaut. Vous pouvez également utiliser les méthodes `getText` et `setSelected(true)` pour parvenir au même résultat
- ② On crée un objet `ButtonGroup` avec le constructeur par défaut. En ajoutant ces boutons au groupe, cela permet qu'un seul `JRadioButton` soit sélectionné à la fois

**Gestion des événements**

- Lorsqu'un usager clique sur un `JRadioButton` faisant partie d'un `ButtonGroup`, deux objets `ItemEvent` sont générés, traitables par un écouteur `ItemListener` ( un objet d'une classe privée mettant en œuvre l'interface `ItemListener` )  
  
pourquoi deux `ItemEvents` ? un généré par la sélection d'un nouveau `JRadioButton`, l'autre généré par la désélection du précédent. ( 2 `radioButtons` changent d'état )
- À l'usage, il s'est avéré plus facile de traiter les événements sur les `JRadioButtons` avec de simples `ActionListeners`

**NB : Lorsque vous avez un GUI comprenant à la fois des `JButton` et des `JPanels` par exemple, vous pouvez implémenter les deux interfaces dans la même classe privée, tel :**

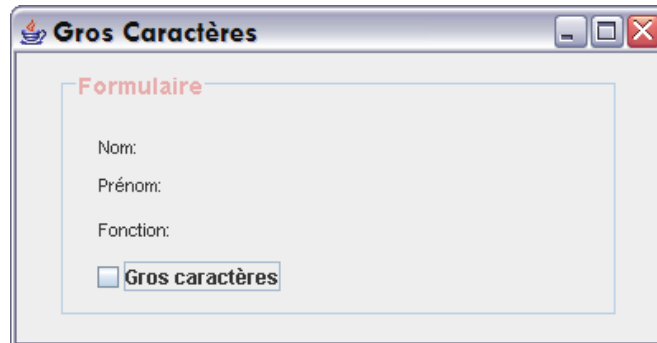
```
private class Gestion implements ActionListener,
MouseMotionListener
{
    public void actionPerformed ( ActionEvent e )
    {
    }
    public void mouseMoved ( MouseEvent i )
    {
    }
    public void mouseDragged ( MouseEvent i )
    {
    }
}
```

**EXERCICES :**

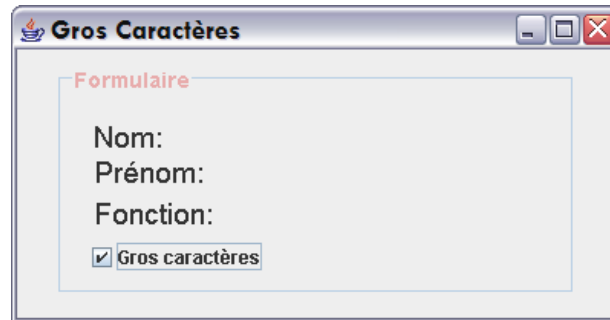
Codez les GUI suivants à l'aide de la méthode 3 ( en 3 étapes, **sans** la méthode automatisée `JDeveloper` )

- 8) ( `JRadioButton`, `ButtonGroup`, `ItemListener` ou `ActionListener` ) Réaliser un frame permettant de modifier l'alignement de texte dans un champ texte à l'aide de 3 objets `JRadioButton` ( gauche, centré, droit ).

- 9) ( `JCheckBox`, `JLabel`, classe `Font` ) Reproduisez l'interface graphique suivante ( `JLabels` : texte avec la police Arial, 11 pt ):



Faites en sorte que lorsqu'on sélectionne l'option « Gros caractères » le texte des `JLabels` est soudain l'apparence suivante :



et vice-versa lorsque l'option n'est plus sélectionnée

améliorations / difficultés :

- ♠ grossir le texte → utiliser un objet `Font`
- ♠ modifier tous les `JLabels` en même temps sans avoir à les nommer un à un → utiliser la méthode `getComponents` sur le `JPanel` avec une boucle
- ♠ changer la couleur du `TitledBorder` du `JPanel` → créer un objet `TitledBorder`