JAV - TD 9
Les interfaces graphiques JAVA

AWT & SWING

- Première bibliothèque graphique JAVA: AWT
  - Package java.awt
  - Utilisation de code natif
  - Composants limités

- Nouvelle bibliothèque: SWING
  - Package javax.swing
  - Plus riche et plus personnalisable
  - Ne remplace pas AWT mais fournit des composants plus performants

Hiérarchie des composants

❖ **Les composants graphiques sont placés dans des conteneurs (Containers):**

- Les containers
  - JWindow
    - JFrame
    - JDialog
      - JFileDialog
  - JPanel
    - Applet
  - JTabbedPane
  - JScrollPane

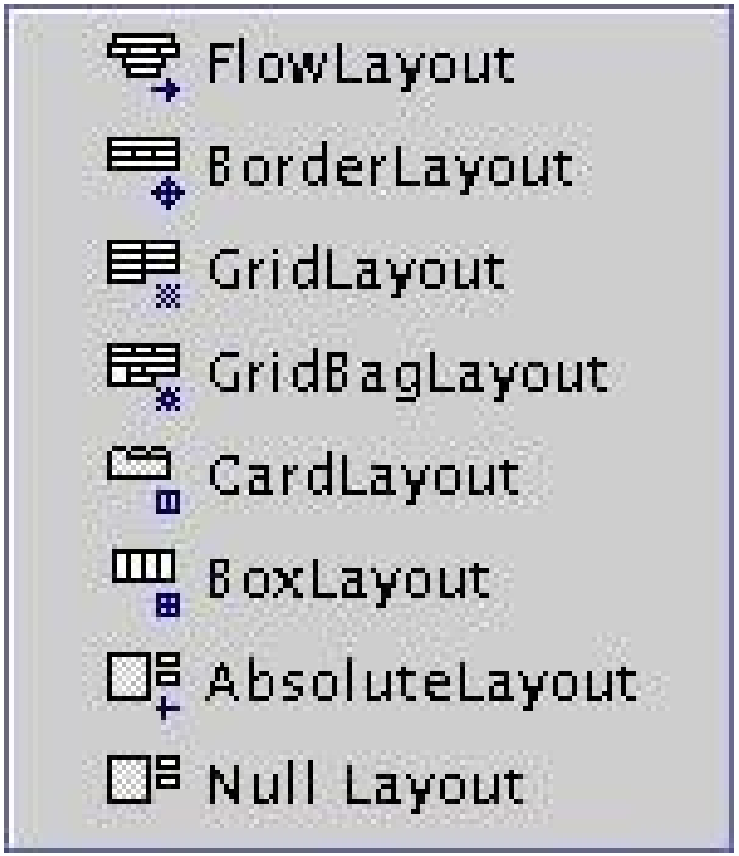
- Composants élémentaires
  - JLabel
  - JButton
  - JCheckbox
  - JRadioButton
  - JTextField
  - JTextArea
- Composants complexes
  - ButtonGroup
  - JComboBox
  - JList
  - JScrollBar
  - JMenuBar
  - JPopupMenu

Exercice

- Réaliser une classe Appli qui affiche une fenêtre (JFrame) contenant un JButton

Disposition des composants (1/2)

❖ Chaque conteneur utilise un gestionnaire de placement (Layout) pour la disposition des composants qu'il contient.



http://java.sun.com/docs/books/tutorial/uiswing/layout/visual.html

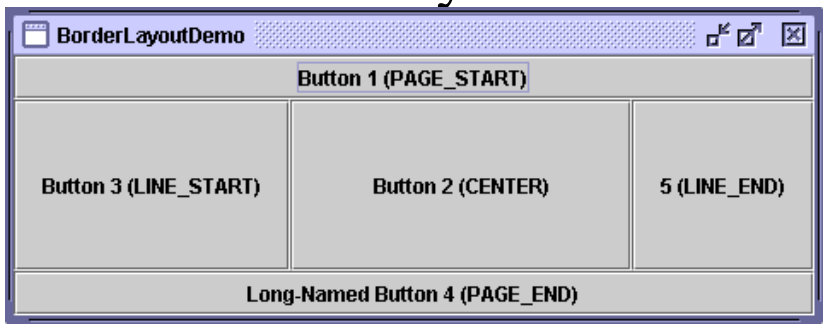
Disposition des composants (2/2)

- Exemples de dispositions

GridLayout

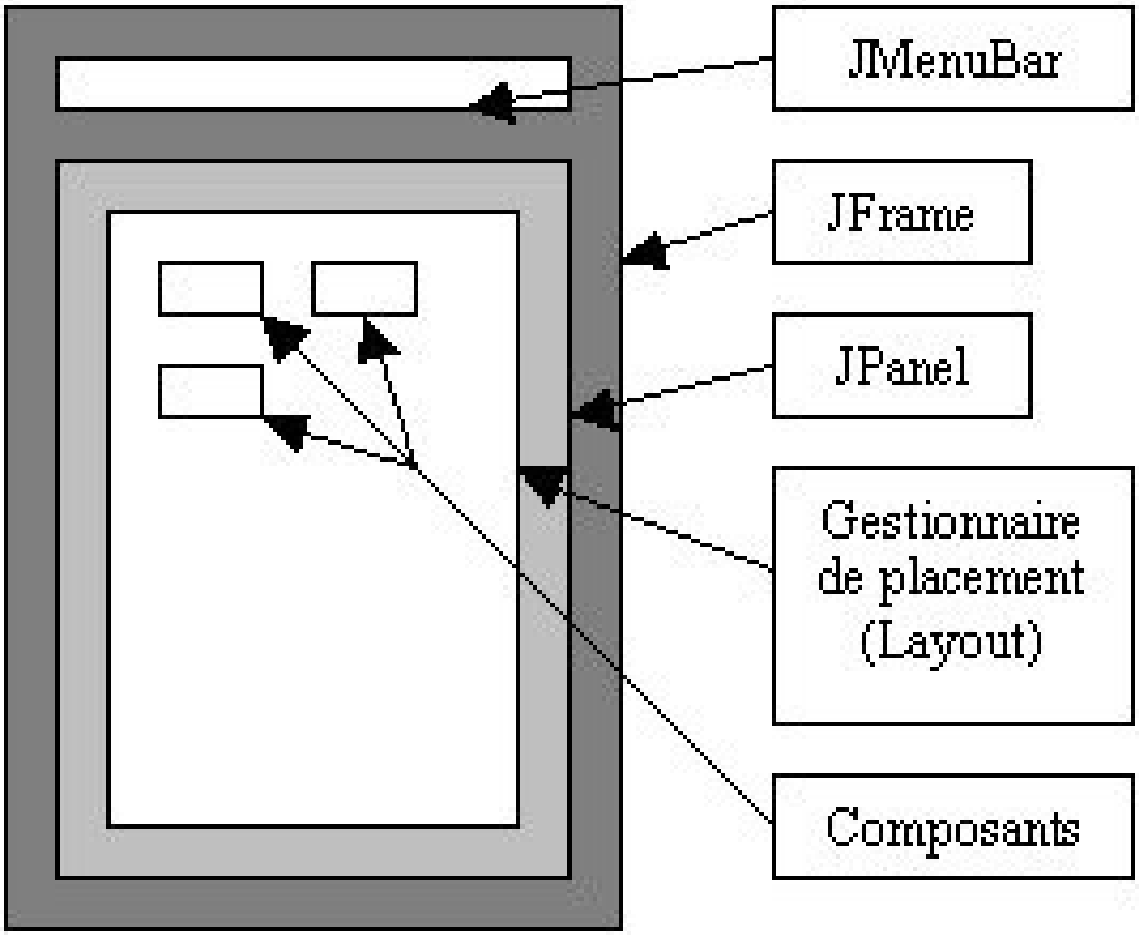

```
JFrame fenetre=new JFrame("GridLayoutDemo");
Container tmp = fenetre.getContentPane();
tmp.setLayout(new GridLayout(3,2));
tmp.add(new Button("Button 1"));
tmp.add(new Button("Button 2"));
tmp.add(new Button("Button 3"));
tmp.add(new Button("Long-Named Button 4 "));
tmp.add(new Button("5"));
```

BorderLayout




```
JFrame fenetre=new JFrame("BorderLayoutDemo");
Container tmp = fenetre.getContentPane();
tmp.setLayout(new BorderLayout());
tmp.add(new Button("Button 1(PAGE_START)",
BorderLayout.NORTH));
tmp.add(new Button("Button 3 (LINE_START)",
BorderLayout.WEST);
tmp.add(new Button("Button 2 (CENTER)")
BorderLayout.CENTER);
tmp.add(new Button("5 (LINE_END)")
BorderLayout.EAST);
tmp.add(new Button("Long-Named Button 4
(PAGE_END)") BorderLayout.SOUTH);
```

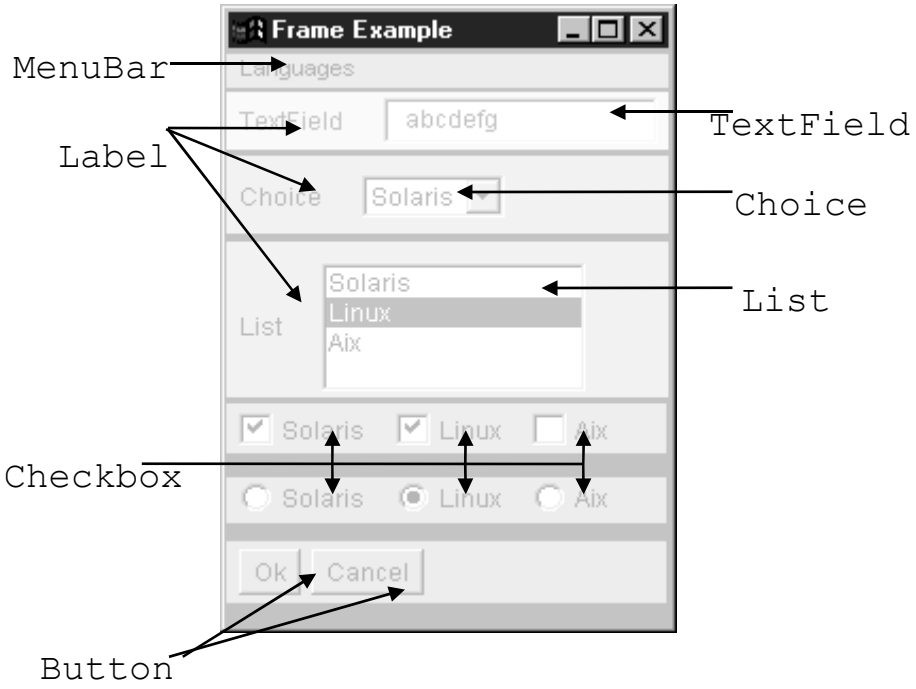
Composition d'une fenêtre JAVA

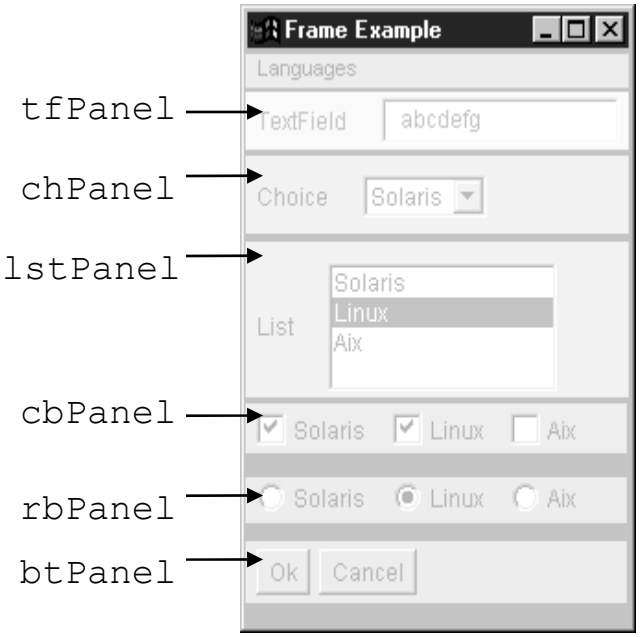


Exemple d'une fenêtre JAVA

```
// Création de la fenêtre
JFrame frame = new JFrame("ExempleSimple");
// Création du container
JPanel panel = new JPanel();
// Définition du gestionnaire de placement
panel.setLayout(new GridLayout(1,2))
// Création des composants
JLabel label = new JLabel("Entrer votre nom");
JTextField textField = new JTextField("toto");
// Ajout des composants au container
panel.add(label);
panel.add(textField);
// Ajout du container à la fenêtre
frame.getContentPane().add(panel); (*)
// Afficher la fenêtre
frame.pack();
frame.setVisible(true);
```

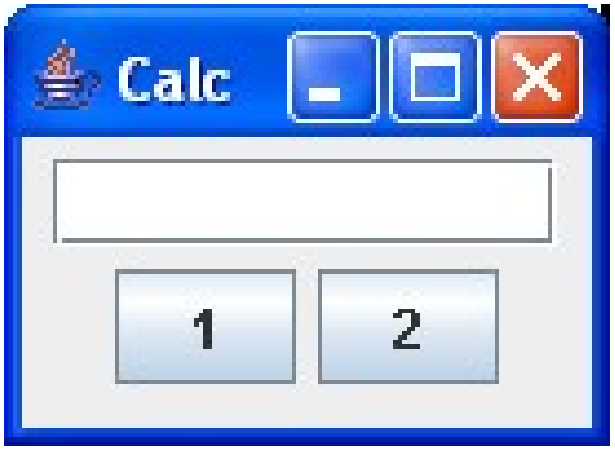
Exercice (à programmer)





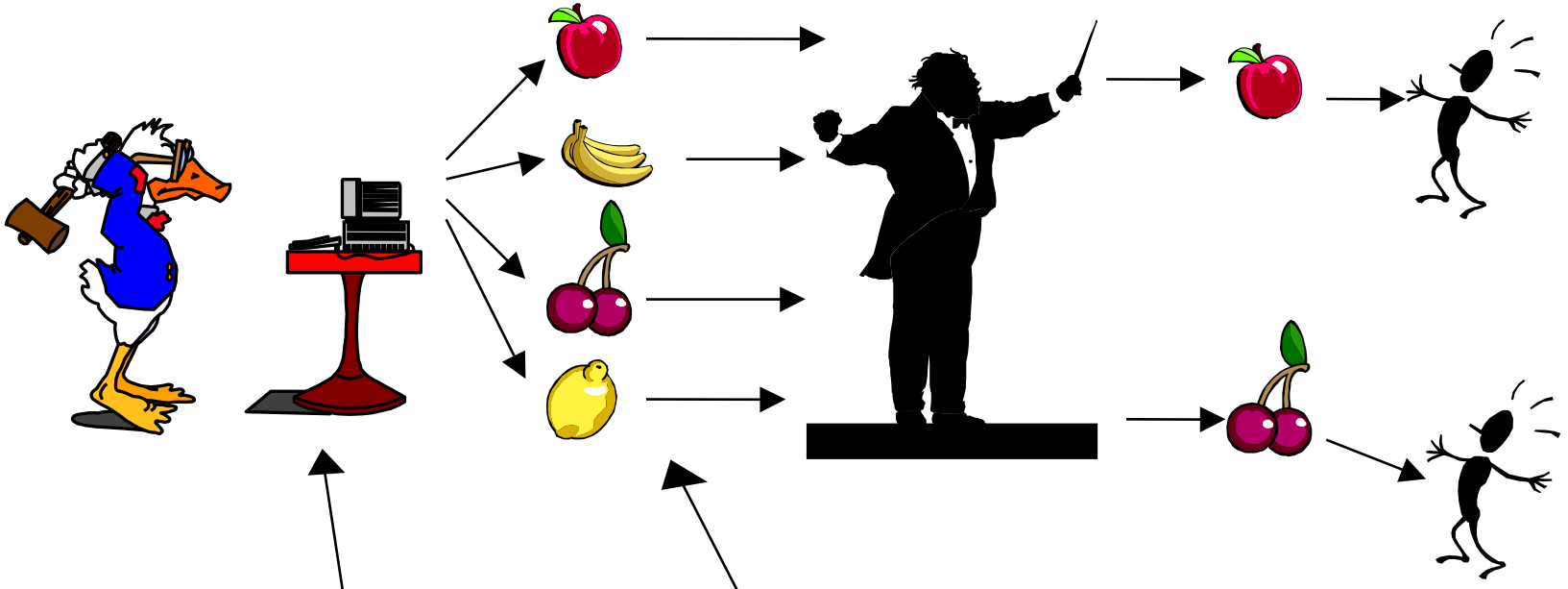
Exercice

- Réalisez l'interface suivante :



Fen : 150x10, JTextField 10 colonnes

Propagation des évènements



L ’utilisateur réalise une action

Source de l ’événement

JButton, JTextField…

Des événements sont alors générés

Événements

ActionEvent, MouseEvent…

La Machine Virtuelle reçoit tous les événements

Seuls les événements écoutés sont transmis

Écouteurs

ActionListener, MouseListener

Un exemple – 2 façons de faire

```
class FenListener implements ActionListener{
    JFrame fenetre = new JFrame();
    JButton jaune = new JButton("Jaune");
    public void actionPerformed(ActionEvent e){
      if (e.getSource()==jaune)
            System.out.println(“jaune ");
    }
    public static void main(String[] args)
      new BoutonAvecListener();
    }
    public FenListener(){
      jaune.addActionListener(this);
      fenetre.getContentPane().add(jaune);
      fenetre.pack();
      fenetre.show();
    }
}
```

```
class Fen {
    JFrame fenetre = new JFrame();
    JButton jaune = new JButton("Jaune");
    public static void main(String[] args)
      new BoutonAvecListener();
    }
    public FenAvecListener(){
      jaune.addActionListener(this);
      fenetre.getContentPane().add(jaune);
      fenetre.pack();
      fenetre.show();
    }
      public class Ecouteur implements ActionListener{
      public void actionPerformed(ActionEvent e){
      if (e.getSource()==jaune)
            System.out.println(“jaune ");
    }
}
```

Les acteurs

- Le composant
  - Indique les événements qu'il peut générer.
  - Button : MouseEvent, ActionEvent, ComponentEvent…
- L'événement
  - Indique l'action que l'utilisateur a générée.
  - Ex : MouseEvent
- Le listener
  - Il indique le traitement à faire sur une catégorie d'événements
  - MouseListener, ActionListener…

Exercice : Interaction 1

- Mettre en place les interactions :
  - 1 Déclarer le listener qui affiche bonjour sur la console à chaque clic sur un des deux boutons
  - 2 Abonner les boutons sur ce listener

* Listener : classe qui implante le Listener choisi
* Abonnement : Utilisation de la méthode addXXXListener(XXXListener objetListener) sur le composant qui doit générer l'événement

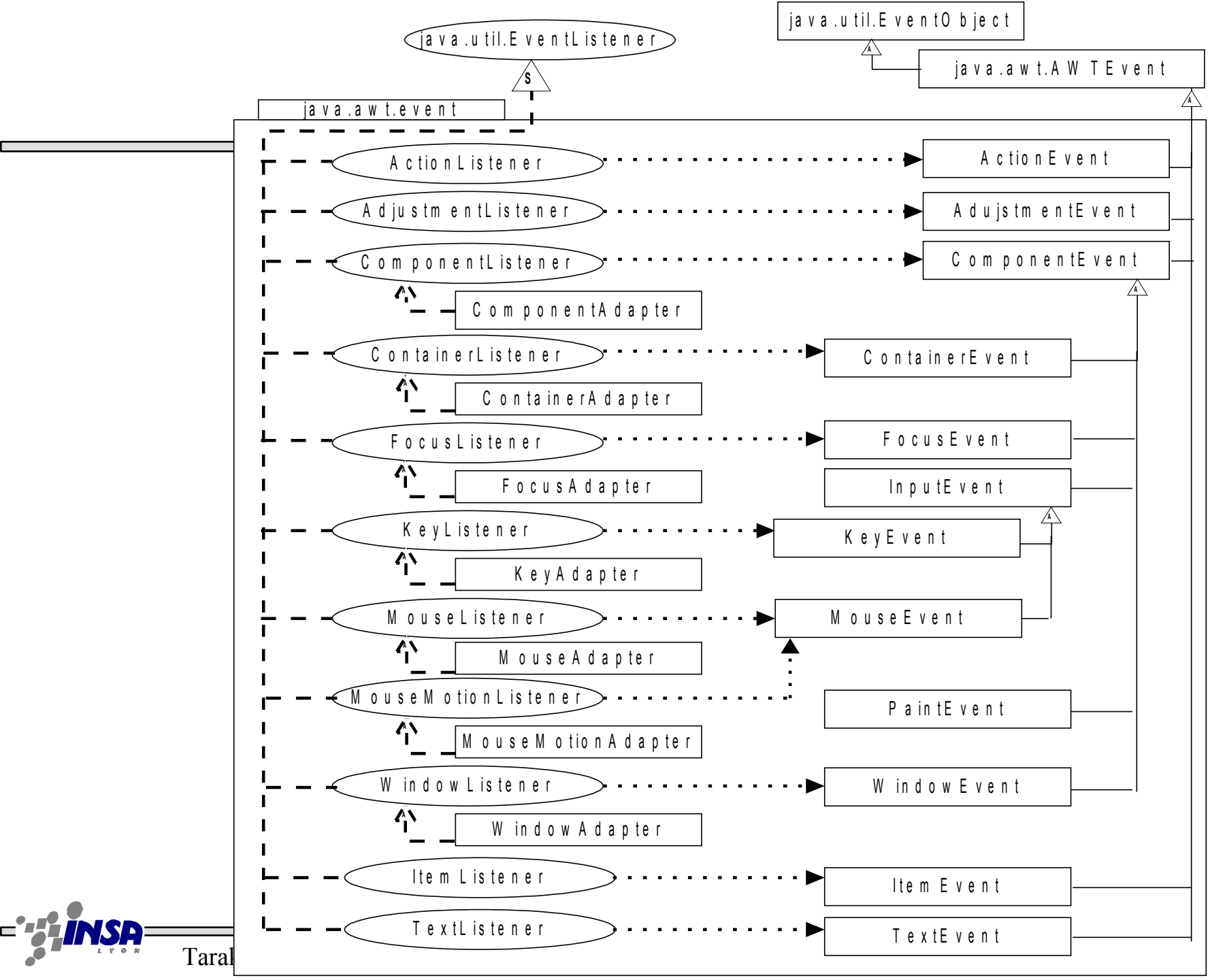
Exercice : Interaction 2

- Mettre en place les interactions :
  - 1 Déclarer le listener de traitement qui affiche la valeur du bouton cliqué dans le textfield
  - 2 Abonner les boutons sur ce listener

* Listener : classe qui implante le listener choisi
* Abonnement : Utilisation de la méthode addXXXListener(XXXListener objetListener) sur le composant qui doit générer l'événement

Les composants et leurs événements

- Tous les composants génèrent des événements
  - Car il dérivent de la classe Component qui génère des événements
- Tous les composants ne génèrent pas tous les événements
  - Un bouton ne génère pas d'événements de type text
- Il existe pour les composants élémentaires un événement de sémantique générale appelé ActionEvent, qui représente l'interaction standard avec l'utilisateur
  - Click sur bouton ==> ActionEvent
  - DoubleClick sur une liste ==> ActionEvent
  - Click sur un élément de liste ==> ActionEvent
  - <Return> à la fin d'une saisie dans un TextField ==> ActionEvent


java.util.EventObject
java.awt.AWTEvent
java.util.EventListener
java.awt.event
ActionListener
ActionEvent
AdjustmentListener
AdujstmentEvent
ComponentListener
ComponentEvent
ComponentAdapter
ContainerListener
ContainerEvent
ContainerAdapter
FocusListener
FocusEvent
FocusAdapter
InputEvent
KeyListener
KeyEvent
KeyAdapter
MouseListener
MouseEvent
MouseAdapter
MouseMotionListener
PaintEvent
MouseMotionAdapter
WindowListener
WindowEvent
WindowAdapter
ItemListener
ItemEvent
TextListener
TextEvent

Un exemple

- Réaliser un listener qui change la couleur du bouton qui possède le focus
  java.awt.event.FocusListener

```
public void focusGained(FocusEvent e){
}
public void focusLost(FocusEvent e){
}
```

- Modifiez votre classe Appli afin que tous les boutons soient abonnés à une instance de votre FocusListener

```
addFocusListener(<unFocusListener>);
```

Exercice

I) L'exemple inévitable (HelloWorld)

1) développer une fenêtre HelloWord qui affiche « Hello !! » dans un label

2) Ajouter un bouton à la fenêtre. Le label affichera « Hello (n) » où n est le nombre de clics sur le bouton

II) Interface graphique pour la gestion des étudiants

Développer une interface graphique pour ajouter, supprimer et afficher un étudiant à la classe GesEtudHash du TD précédent.