# Java Swing !

—

# Mise en Œuvre

**Ronan Keryell**

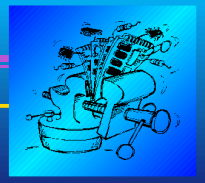
—

**Centre de Recherche en Informatique de l'École des Mines de Paris**

**4 novembre 1998**

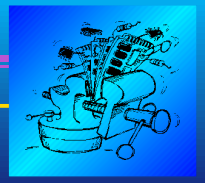
- Besoin de faire de belles interfaces
- AWT assez basique
- Portabilité (multiplateformes & Internet)
- Éviter de devoir réécrire l'interface graphique
- Pouvoir changer le/hériter du « *Look & Feel* »





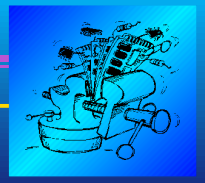
- Composants « *heavyweight* »
  association composant graphique $\leftrightsquigarrow$ pair en code natif
- Changer l'aspect $\Longrightarrow$ changer le code natif
- Bugs proportionnels au nombre de codes natifs





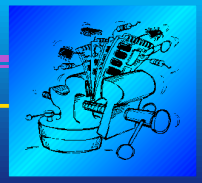
- Philosophie de Swing
- Description des objets graphiques
- Mise en pratique





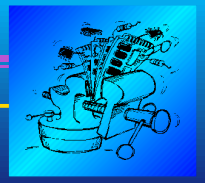
- http://java.sun.com
- http://java.sun.com/docs/index.html
- http://java.sun.com/docs/books/tutorial/ui/index.html
- http://java.sun.com/products/jfc/tsc/swingdoc-static/intro.html
- S'enregistrer comme développeur puis
  - http://developer.java.sun.com/
  - (http://www.MageLang.com/)
    http://developer.java.sun.com/developer/onlineTraining/index.html
- ... Lisez les sources !





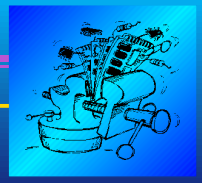
- Composants « *lightweight* » :
  - Pas de code graphique natif
  - Implémenté en Java + AWT
- Aspect « local » de la même application quelle que soit la machine :
  - Metal (Unix)
  - Motif (Unix)
  - MacIntosh
  - Windows
- Facilement extensible
- 100 % pur Java & compatible JavaBeans, utilisable avec outil de génie logiciel
- Fait partie des JFC (Java Foundation Classes) « *designed to help developers to build full-featured enterprise-ready applications* »





- Modèle MVC (modèle, vue, contrôleur)
- Sources disponibles
  - Exemples pour extension
  - Lorsque la documentation ne suffit pas
  - En cours d'évolution : tout n'est pas implémenté...
- Swing étend mais ne remplace pas AWT
  - Mélange AWT & Swing possible
- Né d'une collaboration entre Sun (AWT) et Netscape (Internet Foundation Class)
- Inclus à partir du JDK 1.1.2
- Swing 1.1 vient avec JDK 1.2

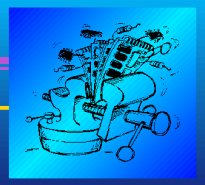


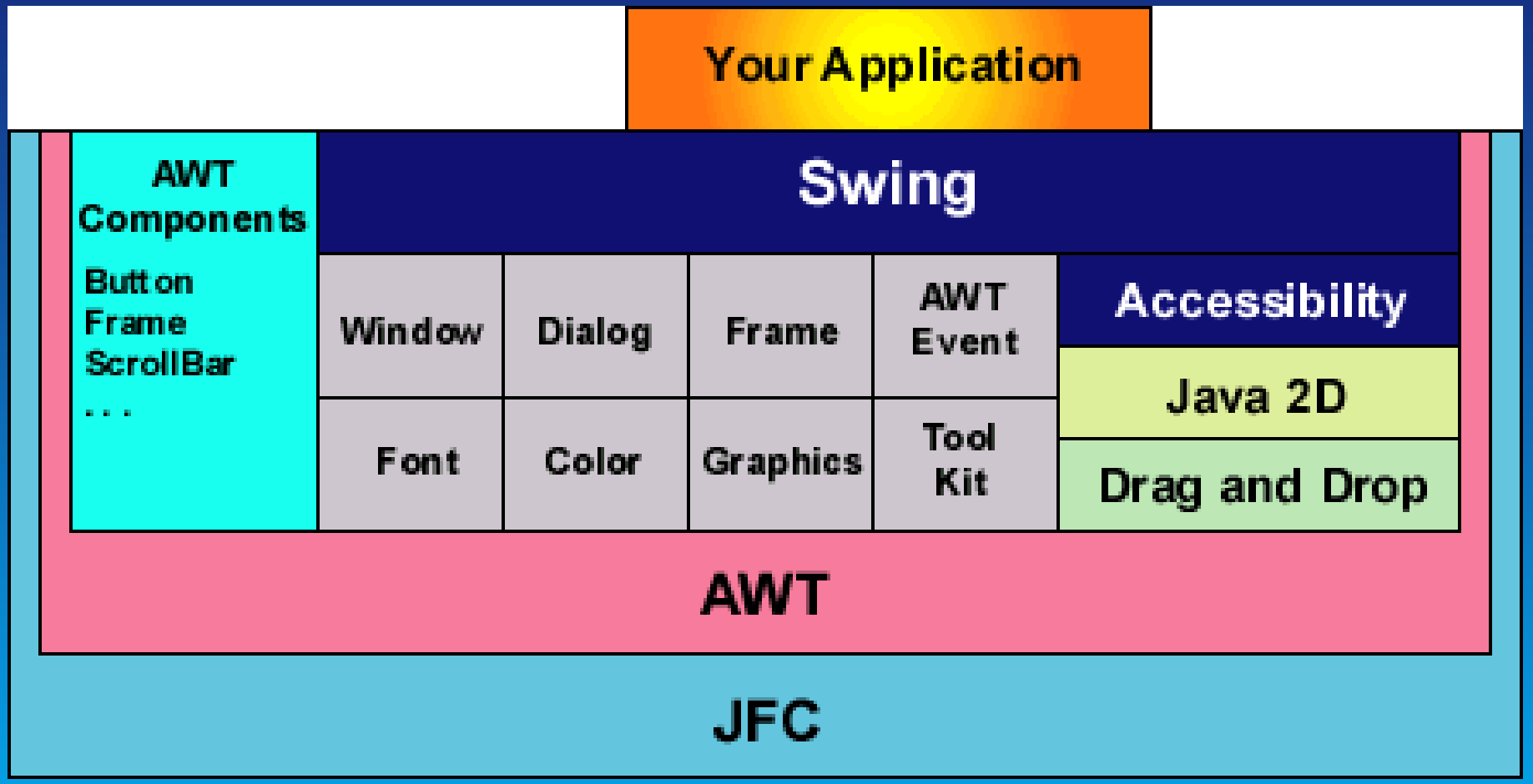


*Java Foundation Classes*

- AWT
- Swing
- Java2D : images, dessins complexes (formes, rendu) (IBM/Taligent)
- Drag & Drop : tranferts de données entre applications Java ou natives
- Accessibility API : aide aux handicapés : loupes, lecteurs de textes,...

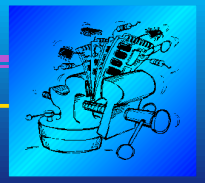







Java2D et Drag & Drop ont besoin de code natif ⇝ pas dans le code de Swing





Swing éclaté en 15 packages :

**javax.swing** le plus haut niveau du package Swing

- Composants
- Adaptateurs
- Modèles par défaut
- Interfaces

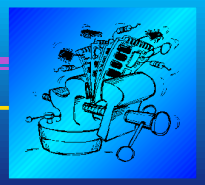
**javax.swing.border** classes et interfaces pour dessiner des bordures autour des composants

**javax.swing.colorchooser** classes et interfaces utilisées par le composant JColorChooser

**javax.swing.event** types d'événements et gestionnaires spécifiques en plus de ceux de `java.awt.event`

**javax.swing.filechooser** classes et interfaces utilisées par le composant `JFileChooser`





**javax.swing.plaf** définit une interface et plusieurs classes abstraites pour fournir le *pluggable look-and-feel*

**javax.swing.plaf.basic** objets de l'interface utilisateur par défaut. Permet un sous-classage pour changer de *look*

**javax.swing.plaf.metal** objets de l'interface utilisateur avec un style *metal*

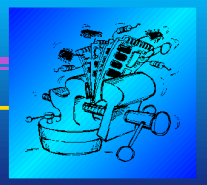
**javax.swing.plaf.multi** multiplexage de plusieurs styles d'interfaces pour mélanger différents styles

**javax.swing.table** classes et interfaces utilisées par le composant `JTable`

**javax.swing.text** classes et interfaces gérant les composants textuels éditables ou non. Introduit la notion de document

**javax.swing.text.html** fournit la classe `HTMLEditorKit` et des classes de support pour la création d'éditeurs HTML



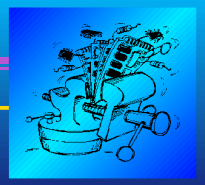


**javax.swing.tree** classes et interfaces utilisées par le composant `JTree`

**javax.swing.undo** fournit les services *undo-redo*

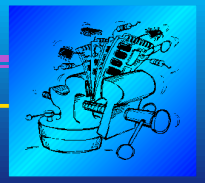
**javax.accessibility** définit une relation entre les composants de l'interface graphique utilisateur et les technologies d'assistance fournissant l'accès à ces composants





¡ Essayer la démonstration SwingSet !



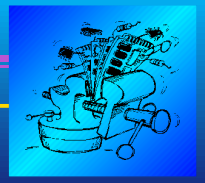


Swing définit 2 types de composants

- Conteneurs globaux (`JFrame`, `JApplet`, `JWindow`, `JDialog`)
- Des composants *lightweight* (`Jeverything-else` , tels que `JButton`, `JPanel`, `JMenu`,...)

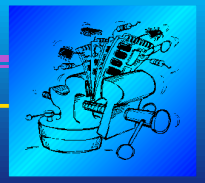
`http://deauville.ensmp.fr/tutorial/ui/swing/generalCompRules.html`

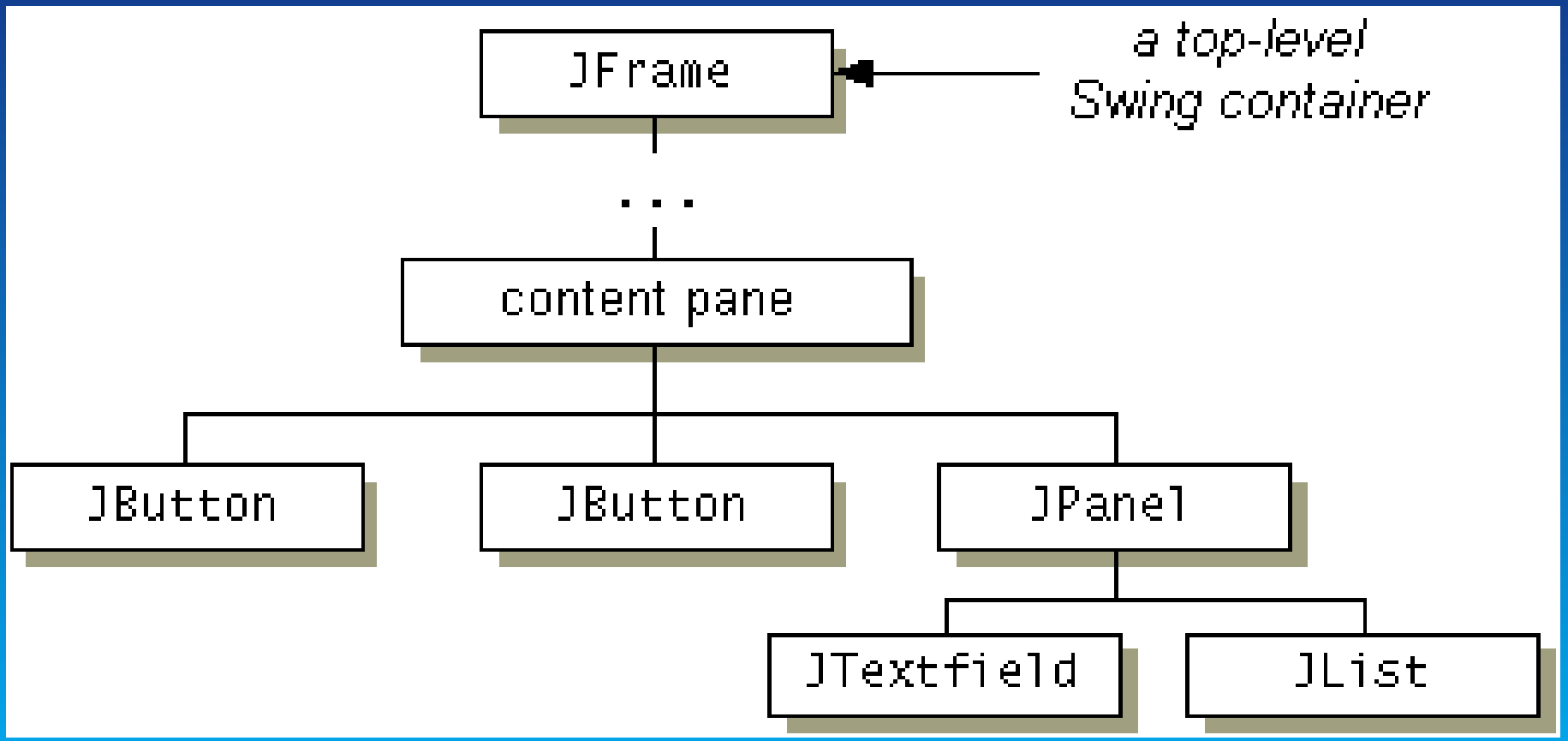




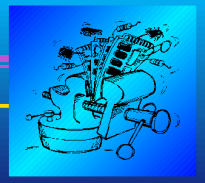
- Infrastructure permettant aux composants poids-plume d'exister
- Contient une zone où les composants peuvent se dessiner
- Barre de menu éventuelle
- Gestion d'événements et dessins plus avancés
- Support pour l'assistance
- Chaque composant doit généralement avoir un conteneur global dans sa hiérarchie de conteneur :
  - Une *applet* contenant du Swing devrait sous-classer `JApplet`
  - Une fenêtre principale devrait sous-classer `JFrame`

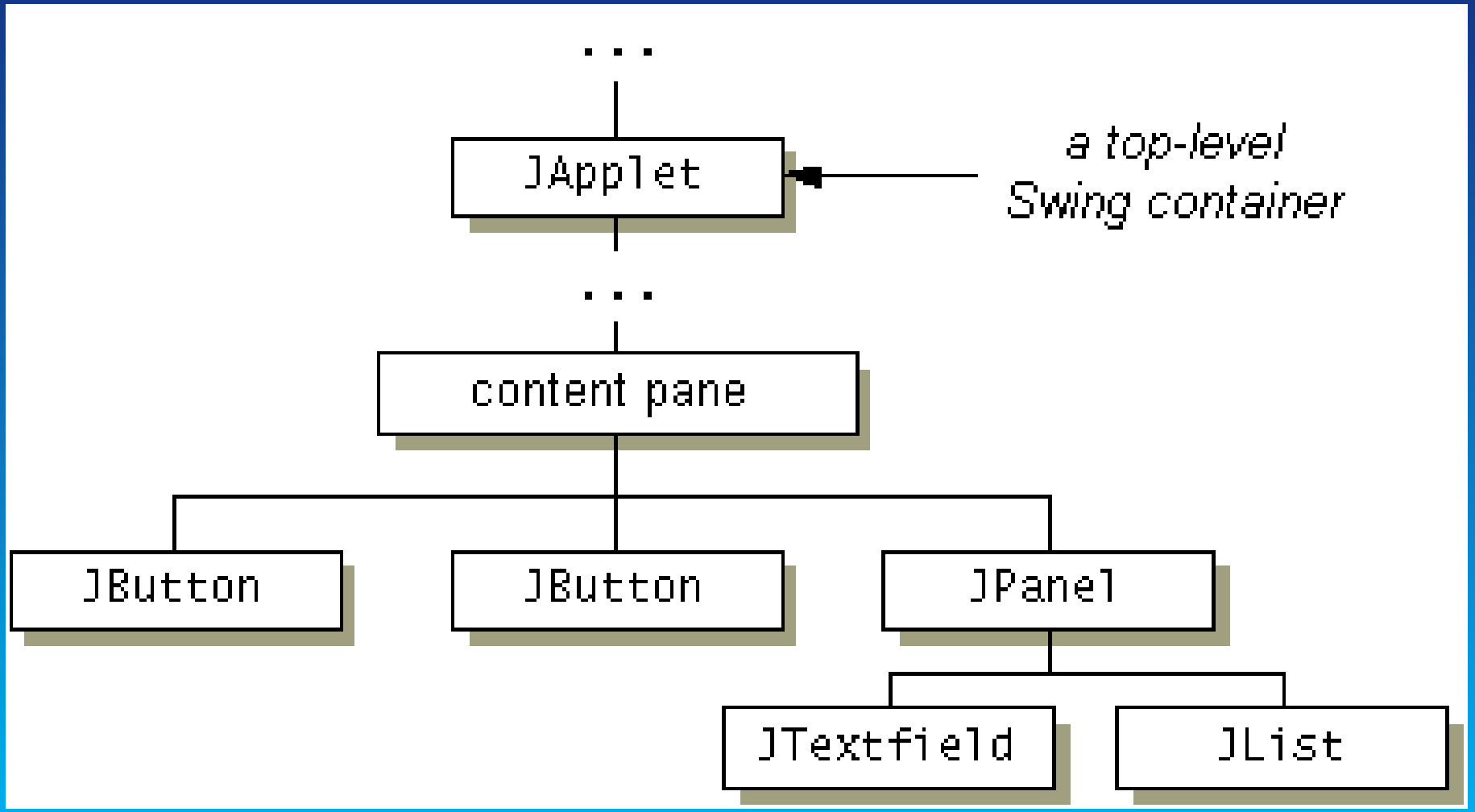




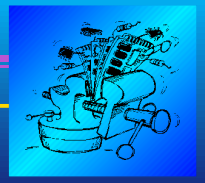

JFrame
a top-level
Swing container
...
content pane
JButton
JButton
JPanel
JTextfield
JList





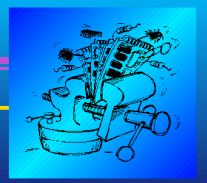

...
JApplet
a top-level
Swing container
...
content pane
JButton
JButton
JPanel
JTextfield
JList





```
//Set up the JPanel, which contains the text field and list.
JPanel panel = new JPanel();
panel.setLayout(new SomeLayoutManager());
panel.add(textField);
panel.add(list);

//topLevel is an instance of JApplet or JFrame
Container contentPane = topLevel.getContentPane();
contentPane.setLayout(new AnotherLayoutManager());
contentPane.add(button1);
contentPane.add(button2);
contentPane.add(panel);
```





- La plupart des composants Swing descendent de `JComponent` qui hérite de Container
- Caractéristiques d'un `JComponent`

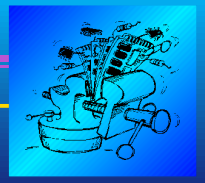
**Bordure** Méthode `setBorder()` pour spécifier une bordure autour du composant. `BorderFactory` permet de créer des bordures

**Double-buffering** 2 mémoires d'affichages existent : celle que l'on voit et celle où les composants Swing sont en train d'être dessinés. Évite le clignotement du tracé. `setDoubleBuffered(false)` permet de l'enlever

**Tool tips** `setToolTipText` spécifie l'affichage d'un message d'aide lorsque le curseur est sur un objet

**Navigation au clavier** `registerKeyboardAction` autorise l'usage du clavier en plus de la souris pour manœuvrer





**Propriétes** `putProperty` associe une propriété au composant

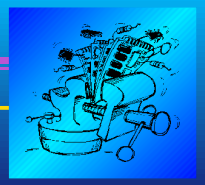
**Style** chaque composant a un *pluggable look and feel* permettant au `UIManager.setLookAndFeel` global de fonctionner

**Placement** contrôlé par `setPreferredSize`, `setMinimumSize`, `setMaximumSize`, `setAlignmentX`, and `setAlignmentY`

**Accessibilité** pour les technologies d'assistance

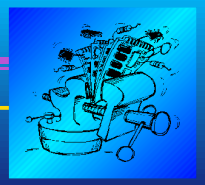
**Localisation** permet des comportements différents en fonction du pays, de la langue, etc.





```
import javax.swing.*;
import java.awt.*;

public class HelloSwingApplet extends JApplet {
    public void init() {
        JLabel label = new JLabel(
                            "You are successfully running a Swing applet!");
        label.setHorizontalAlignment(JLabel.CENTER);
        label.setBorder(BorderFactory.createLineBorder(Color.black));
        getContentPane().add(label);
    }
}
```



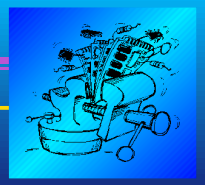


- Compiler avec `javac HelloSwingApplet.java`
- Créer un fichier `HelloSwingApplet.html`

```
<APPLET CODE=HelloSwingApplet.class WIDTH=300 HEIGHT=200>
Cela n'a pas march&eacute;...
</APPLET>
```

- Appeler l'applet avec `appletviewer HelloSwingApplet.html`

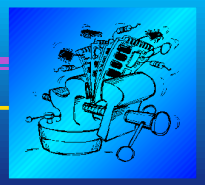




- Conteneur générique de composants Swing
- Double-tamponnage
- `public JPanel(LayoutManager layout)` crée un panneau avec un gestionnaire de placement (`FlowLayout` par défaut)

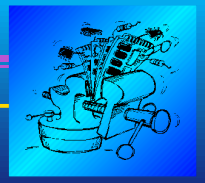
`http://deauville.ensmp.fr/tutorial/ui/swing/panel.html`






```
public ButtonDemo() {
    super();
    ...
    create the three buttons
    ...
    //Add Components to this container, using the default FlowLayout
    add(b1);
    add(b2);
    add(b3);
}
```





D'autres conteneurs plus spécifiques existent

**Box** `http://deauville.ensmp.fr/tutorial/ui/swing/box.html`

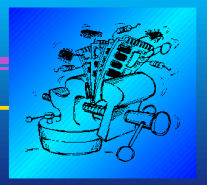
- Utilise un `BoxLayout`
- Poids super-léger car étend Container au lieu de JPanel
- Faux composant Swing : pas possible de choisir la taille minimale/maximale,...

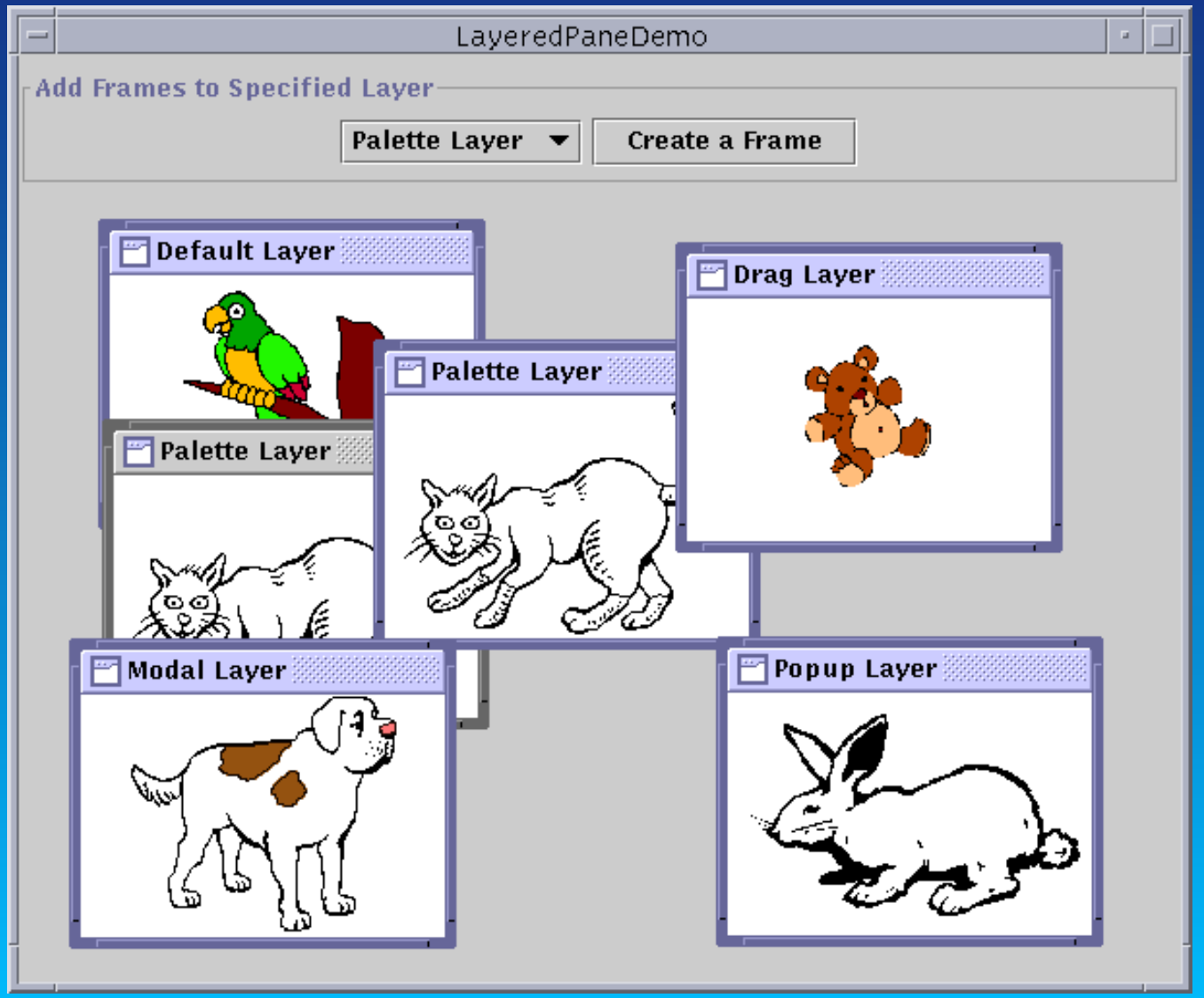
**JLayeredPane**

`http://deauville.ensmp.fr/tutorial/ui/swing/layeredpane.html`

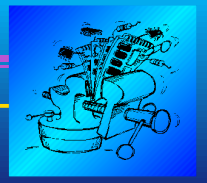
- Rajoute la troisième dimension : place des objets l'un sur l'autre
- Tout objet contenant panneau racine (`JFrame`, `JDialog`,...) possède un `JLayeredPane`
- `JDesktopPane` est une version spécialisée pour gérer des fenêtres en interne






LayeredPaneDemo
Add Frames to Specified Layer
Palette Layer
Create a Frame
Default Layer
Palette Layer
Palette Layer
Drag Layer
Modal Layer
Popup Layer

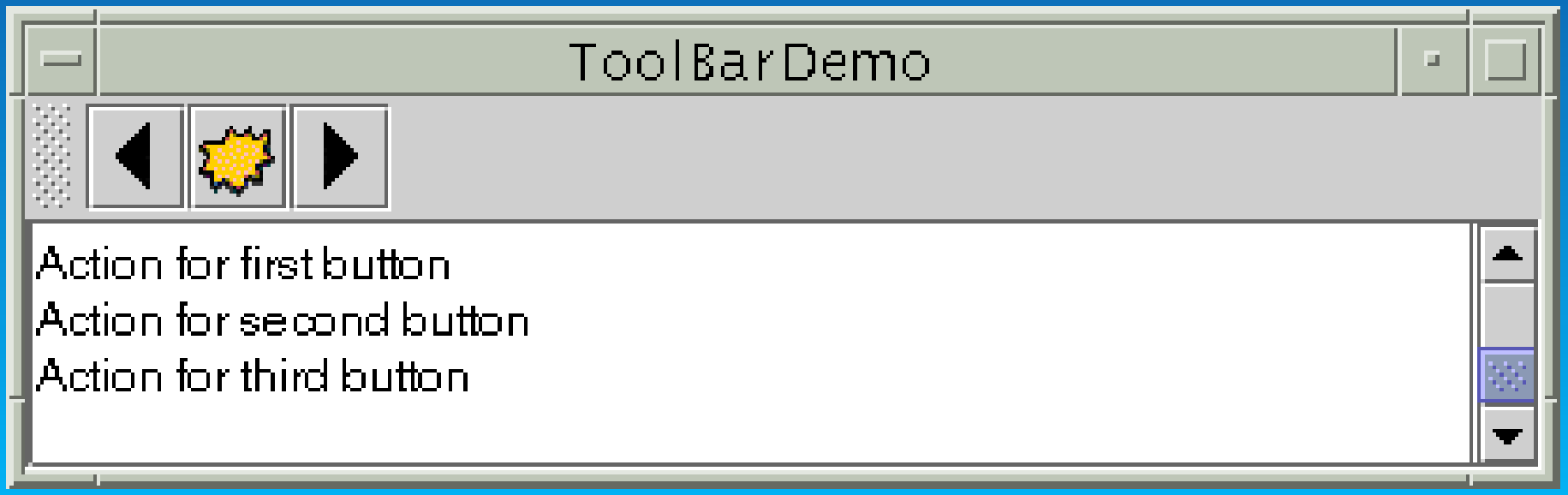




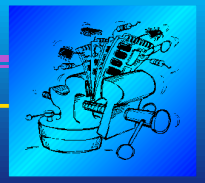
## JScrollPane

`http://deauville.ensmp.fr/tutorial/ui/swing/scrollpane.html`

- Vue avec ascenseurs
- Par défaut prend la taille préférée du client
- Certains clients (listes, tables,...) renvoient une taille préférée plus petite s'ils sont contenus dans un `JScrollPane`





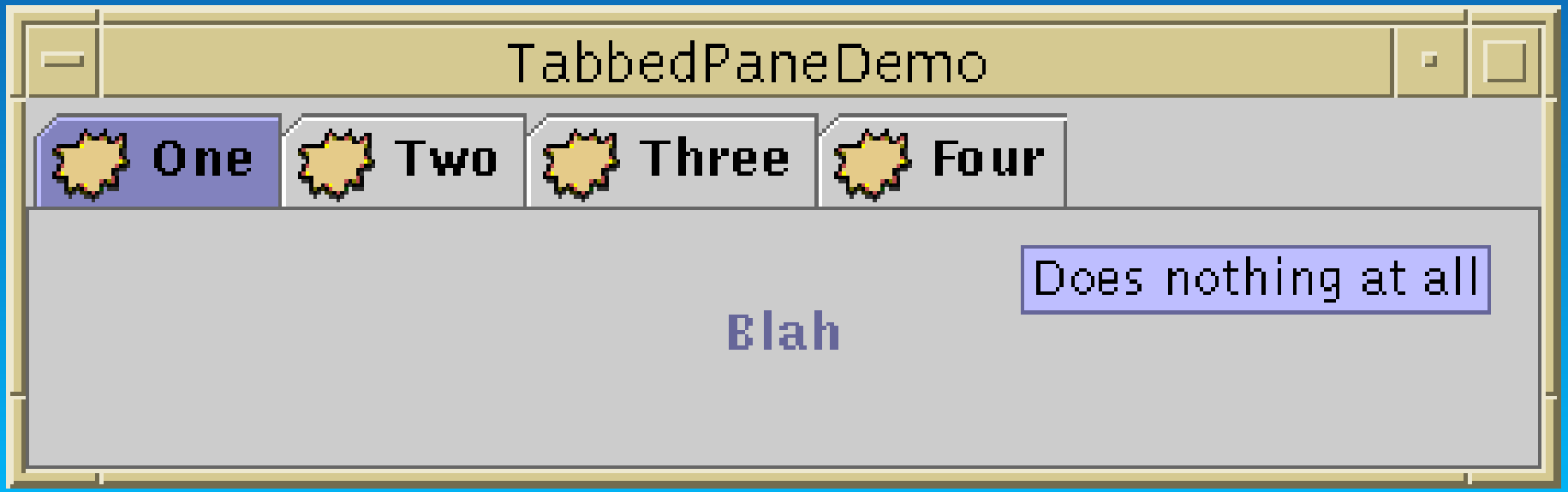
### JSplitPane

`http://deauville.ensmp.fr/tutorial/ui/swing/splitpane.html`

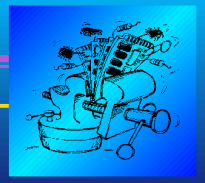
- Contient 2 composants légers
- Partagés par un diviseur mobile

### JTabbedPane

`http://deauville.ensmp.fr/tutorial/ui/swing/tabbedpane.html`





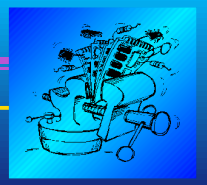
- Classe abstraite `Icon` : petite image de taille fixe pour décorations
- `ImageIcon` génère une icône à partir de donnée, d'une URL (GIF, JPEG,...)

```
Icon tinyPicture = new ImageIcon("images/TinyPicture.gif");
```

Si du GIF89a animé rajouter un observateur à l'image pour mettre à jour l'utilisateur de l'icône

```
tinyPicture.setImageObserver(button);
```

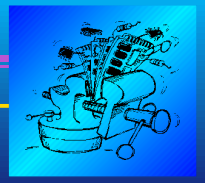




- Affiche du texte ou une image
- Non sélectionnable

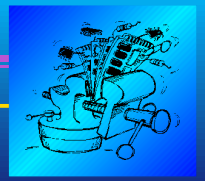

http://deauville.ensmp.fr/tutorial/ui/swing/label.html





```
public class LabelPanel extends JPanel {
  public LabelPanel() {
    // Create and add a JLabel
    JLabel plainLabel = new JLabel("Plain Small Label");
    add(plainLabel);
    // Create a 2nd JLabel
    JLabel fancyLabel = new JLabel("Fancy Big Label");
    // Instantiate a Font object to use for the label
    Font fancyFont = new Font("Serif", Font.BOLD | Font.ITALIC, 32);
    // Associate the font with the label
    fancyLabel.setFont(fancyFont);
    // Create an Icon
    Icon tigerIcon = new ImageIcon("SmallTiger.gif");
    // Place the Icon in the label
    fancyLabel.setIcon(tigerIcon);
    // Align the text to the right of the Icon
    fancyLabel.setHorizontalAlignment(JLabel.RIGHT);
    // Add to panel
    add(fancyLabel);
  }
}
```



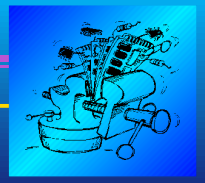



- Conteneur global

- Fournit une fenêtre aux applets et applications
- Décorations (par rapport à `JWindow`)
  - Bord
  - Titre
  - Boutons pour fermer et icônifier
  - Barre de menu éventuelle
- Contient un panneau racine (*root pane*)

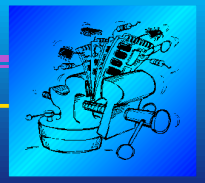
`http://deauville.ensmp.fr/tutorial/ui/swing/frame.html`





```
public static void main(String s[]) {
    JFrame frame = new JFrame("A Basic Frame");

    WindowListener l = new WindowAdapter() {
        public void windowClosing(WindowEvent e) {
        System.exit(0);
        }
    };
    frame.addWindowListener(l);

    JLabel aLabel = new JLabel("Something to look at",
                               new ImageIcon("images/beach.gif"),
                               JLabel.CENTER);
    aLabel.setVerticalTextPosition(JLabel.TOP);
    aLabel.setHorizontalTextPosition(JLabel.CENTER);
    frame.getContentPane().add(aLabel, BorderLayout.CENTER);

    frame.pack();
    frame.setVisible(true);
}
```

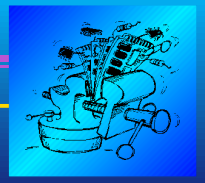




- Fenêtre plus limitée
- Dépend d'une fenêtre principale (destruction, fermeture, (dés)icônification)
- Peut bloquer l'entrée des autres fenêtre (*modal*)
- Opération de fermeture par défaut
- `JOptionPane.showMessageDialog` fournit des fenêtres de dialogue par défaut (erreurs, avertissements, questions, information)

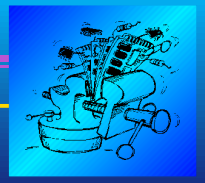
`http://deauville.ensmp.fr/tutorial/ui/swing/dialog.html`



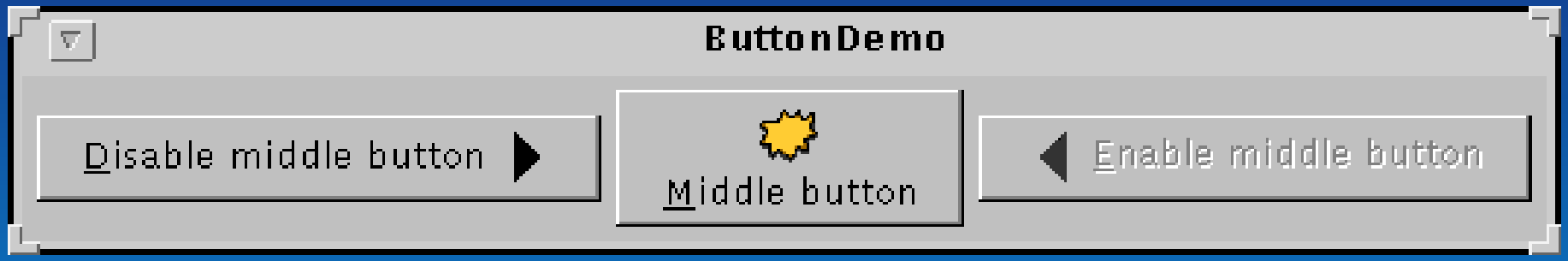


```
JOptionPane.showMessageDialog(frame,
                              "Eggs aren't supposed to be green.");
```



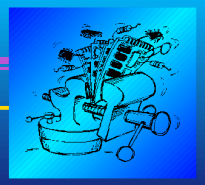



- Bouton de base
- Implémente `AbstractButton`



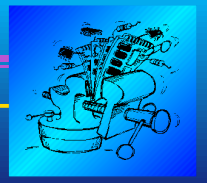

```
//In initialization code:
    ImageIcon leftButtonIcon = new ImageIcon("images/right.gif");
    ImageIcon middleButtonIcon = new ImageIcon("images/middle.gif");
    ImageIcon rightButtonIcon = new ImageIcon("images/left.gif");

    b1 = new JButton("Disable middle button", leftButtonIcon);
    b1.setVerticalTextPosition(AbstractButton.CENTER);
    b1.setHorizontalTextPosition(AbstractButton.LEFT);
    b1.setMnemonic('d');
    b1.setActionCommand("disable");
```





```
    b2 = new JButton("Middle button", middleButtonIcon);
    b2.setVerticalTextPosition(AbstractButton.BOTTOM);
    b2.setHorizontalTextPosition(AbstractButton.CENTER);
    b2.setMnemonic('m');

    b3 = new JButton("Enable middle button", rightButtonIcon);
    //Use the default text position of CENTER, RIGHT.
    b3.setMnemonic('e');
    b3.setActionCommand("enable");
    b3.setEnabled(false);

    //Listen for actions on buttons 1 and 3.
    b1.addActionListener(this);
    b3.addActionListener(this);
    . . .
}

public void actionPerformed(java.awt.event.ActionEvent e) {
    if (e.getActionCommand().equals("disable")) {
        b2.setEnabled(false);
```

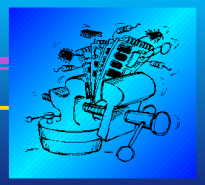




```
            b1.setEnabled(false);
            b3.setEnabled(true);
        } else {
            b2.setEnabled(true);
            b1.setEnabled(true);
            b3.setEnabled(false);
        }
    }
```

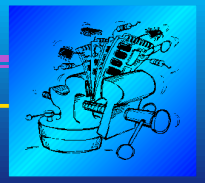
http://deauville.ensmp.fr/tutorial/ui/swing/button.html





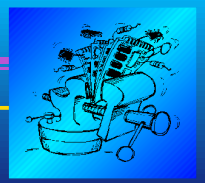
- Fonctionnalités d'un bouton générique
- `setMnemonic(char)` raccourci clavier
- `setSelected(boolean)` (dé)sélectionne le bouton
- `doClick()` simule un clic de souris
- `setActionCommand(String)` définit l'action du bouton
- `addActionListener(ActionListener)` définit le gestionnaire de l'action
- `addItemListener(ItemListener)` gestionnaire d'événement
- `setDisabledIcon(Icon)` lorsque non fonctionnel
- `setPressedIcon(Icon)` lorsque pressé
- `setSelectedIcon(Icon)` quand sélectionné
- `setDisabledSelectedIcon(Icon)`



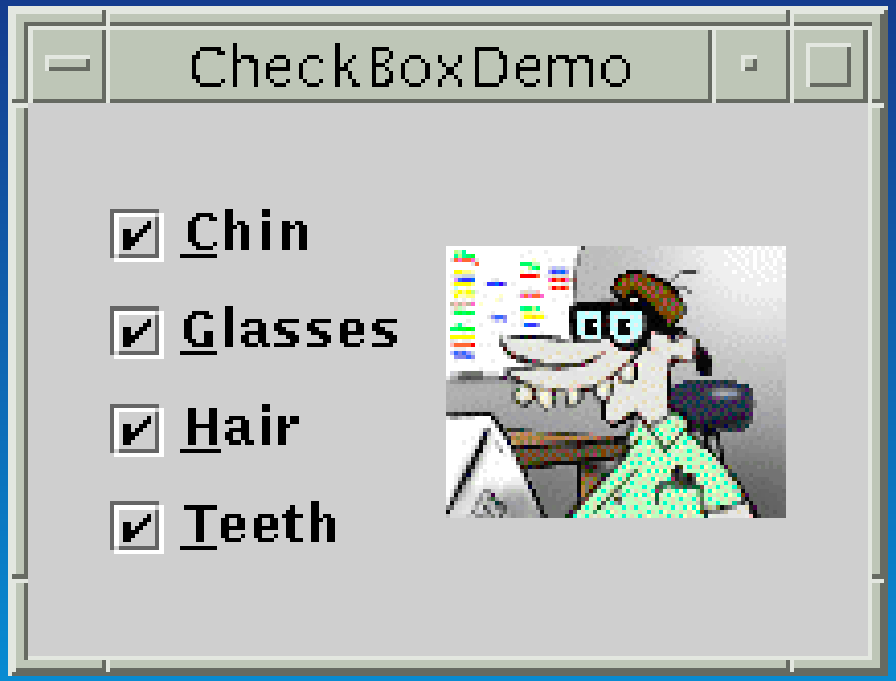


- `setRolloverIcon(Icon)` lorsque la souris passe dessus
- `setRolloverSelectedIcon(Icon)`





# Peut apparaître dans des menus avec `JCheckBoxMenuItem`




```
//In initialization code:
    chinButton = new JCheckBox("Chin");
    chinButton.setMnemonic('c');
    chinButton.setSelected(true);

    glassesButton = new JCheckBox("Glasses");
    glassesButton.setMnemonic('g');
    glassesButton.setSelected(true);

    hairButton = new JCheckBox("Hair");
    hairButton.setMnemonic('h');
    hairButton.setSelected(true);

    teethButton = new JCheckBox("Teeth");
    teethButton.setMnemonic('t');
    teethButton.setSelected(true);

    // Register a listener for the check boxes.
    CheckBoxListener myListener = new CheckBoxListener();
    chinButton.addItemListener(myListener);
    glassesButton.addItemListener(myListener);
    hairButton.addItemListener(myListener);
    teethButton.addItemListener(myListener);
...
class CheckBoxListener implements ItemListener {
    public void itemStateChanged(ItemEvent e) {
        ...
    Object source = e.getItemSelectable();

        if (source == chinButton) {
            //...make a note of it...
        } else if (source == glassesButton) {
            //...make a note of it...
        } else if (source == hairButton) {
            //...make a note of it...
        } else if (source == teethButton) {
```

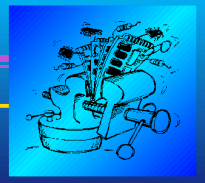


```
        //...make a note of it...
    }

    if (e.getStateChange() == ItemEvent.DESELECTED)
        //...make a note of it...
```

```
            picture.setIcon(/* new icon */);
            ...
        }
    }
```

http://deauville.ensmp.fr/tutorial/ui/swing/checkbox.html



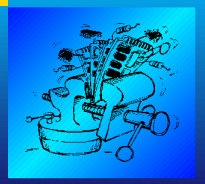


- Groupe de boutons dont 1 seul peut être sélectionné à la fois
- Peut apparaître dans des menus avec `JRadioButtonMenuItem`


```
//In initialization code:
    // Create the radio buttons.
    JRadioButton birdButton = new JRadioButton(birdString);
    birdButton.setMnemonic('b');
    birdButton.setActionCommand(birdString);
    birdButton.setSelected(true);
```

```
JRadioButton catButton = new JRadioButton(catString);
catButton.setMnemonic('c');
catButton.setActionCommand(catString);

JRadioButton dogButton = new JRadioButton(dogString);
dogButton.setMnemonic('d');
dogButton.setActionCommand(dogString);

JRadioButton rabbitButton = new JRadioButton(rabbitString);
rabbitButton.setMnemonic('r');
rabbitButton.setActionCommand(rabbitString);

JRadioButton teddyButton = new JRadioButton(teddyString);
teddyButton.setMnemonic('t');
teddyButton.setActionCommand(teddyString);

// Group the radio buttons.
ButtonGroup group = new ButtonGroup();
group.add(birdButton);
group.add(catButton);
group.add(dogButton);
group.add(rabbitButton);
group.add(teddyButton);

// Register a listener for the radio buttons.
RadioListener myListener = new RadioListener();
```

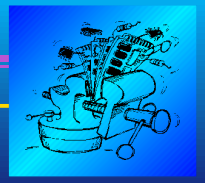




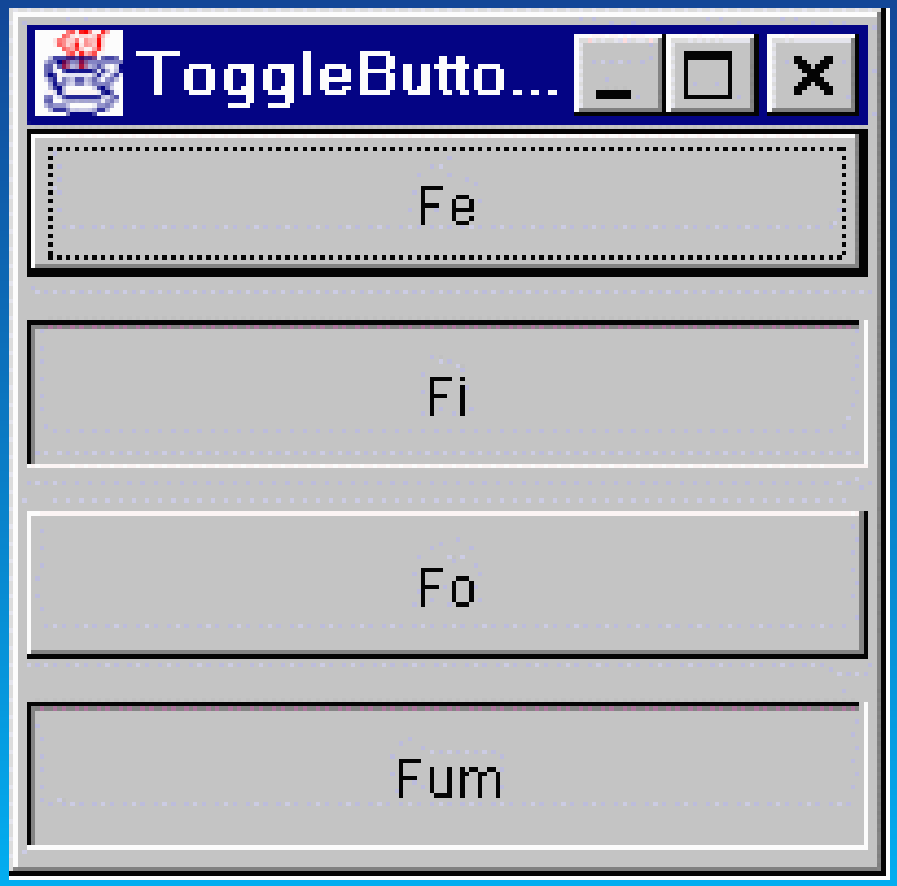
```
    birdButton.addActionListener(myListener);
    catButton.addActionListener(myListener);
    dogButton.addActionListener(myListener);
    rabbitButton.addActionListener(myListener);
    teddyButton.addActionListener(myListener);
...
class RadioListener implements ActionListener ... {
    public void actionPerformed(ActionEvent e) {
        picture.setIcon(new ImageIcon("images/"
                                      + e.getActionCommand()
                        + ".gif"));
    }
}
```

http://deauville.ensmp.fr/tutorial/ui/swing/radiobutton.html.html



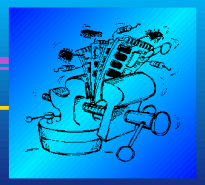


Intermédiaire entre `JCheckBox` et `JRadioButton`

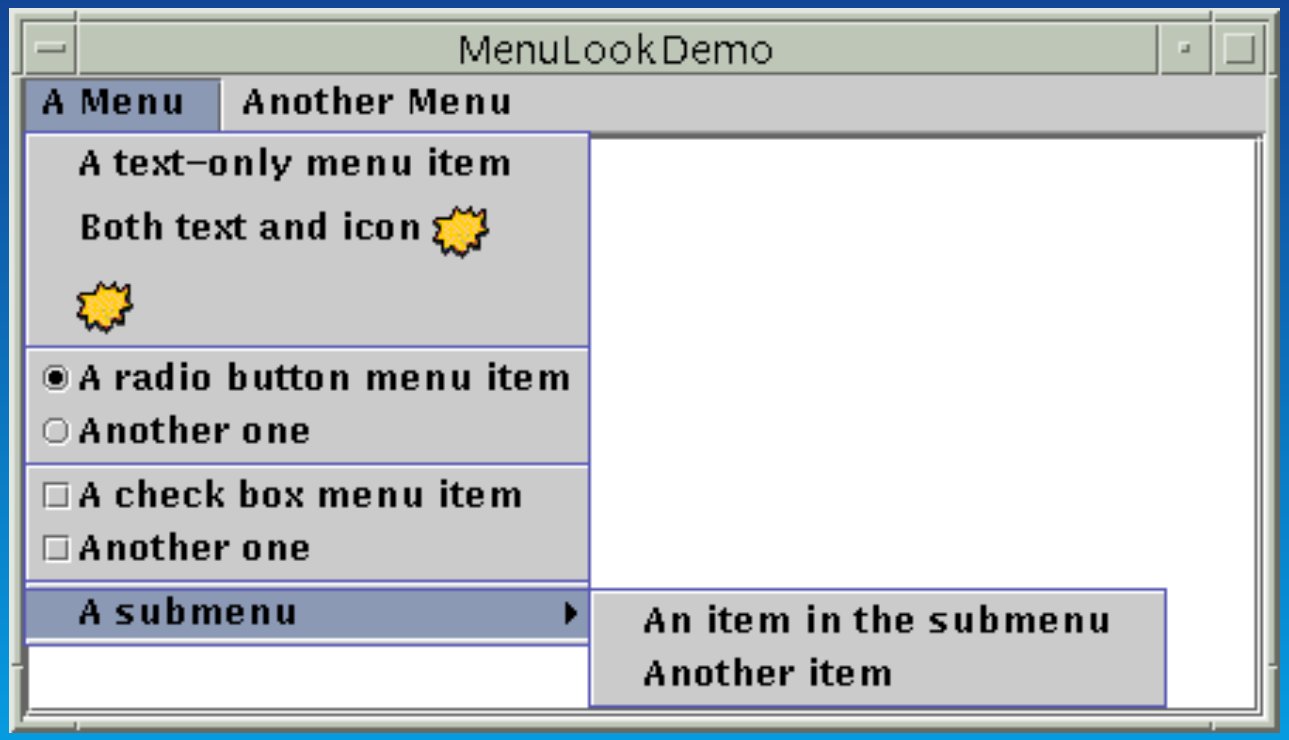



```
public class ToggleButtonPanel extends JPanel {
   public ToggleButtonPanel() {
     // Set the layout to a GridLayout
     setLayout(new GridLayout(4,1, 10, 10));
     add (new JToggleButton ("Fe"));
     add (new JToggleButton ("Fi"));
     add (new JToggleButton ("Fo"));
     add (new JToggleButton ("Fum"));
   }
 }
```





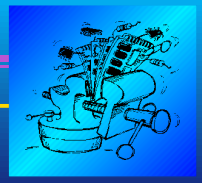
Très génériques & peuvent être des `JPopupMenu`




Accélérateurs avec `setAccelerator(KeyStroke)`

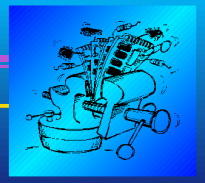
`http://deauville.ensmp.fr/tutorial/ui/swing/menu.html`





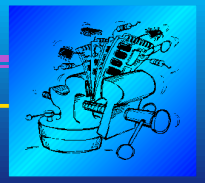
```
//in the constructor for a JFrame subclass:
JMenuBar menuBar;
JMenu menu, submenu;
JMenuItem menuItem;
JCheckBoxMenuItem cbMenuItem;
JRadioButtonMenuItem rbMenuItem;
...
//Create the menu bar.
menuBar = new JMenuBar();
setJMenuBar(menuBar);

//Build the first menu.
menu = new JMenu("A Menu");
menuBar.add(menu);

//a group of JMenuItems
menuItem = new JMenuItem("A text-only menu item");
menu.add(menuItem);
menuItem = new JMenuItem("Both text and icon",
new ImageIcon("images/middle.gif"));
menu.add(menuItem);
menuItem = new JMenuItem(new ImageIcon("images/middle.gif"));
menu.add(menuItem);

//a group of radio button menu items
menu.addSeparator();
ButtonGroup group = new ButtonGroup();
rbMenuItem = new JRadioButtonMenuItem("A radio button menu item");
rbMenuItem.setSelected(true);
group.add(rbMenuItem);
menu.add(rbMenuItem);
rbMenuItem = new JRadioButtonMenuItem("Another one");
group.add(rbMenuItem);
menu.add(rbMenuItem);

//a group of check box menu items
menu.addSeparator();
cbMenuItem = new JCheckBoxMenuItem("A check box menu item");
menu.add(cbMenuItem);
cbMenuItem = new JCheckBoxMenuItem("Another one");
menu.add(cbMenuItem);

//a submenu
menu.addSeparator();
submenu = new JMenu("A submenu");
menuItem = new JMenuItem("An item in the submenu");
submenu.add(menuItem);
menuItem = new JMenuItem("Another item");
submenu.add(menuItem);
menu.add(submenu);

//Build second menu in the menu bar.
menu = new JMenu("Another Menu");
menuBar.add(menu);
```



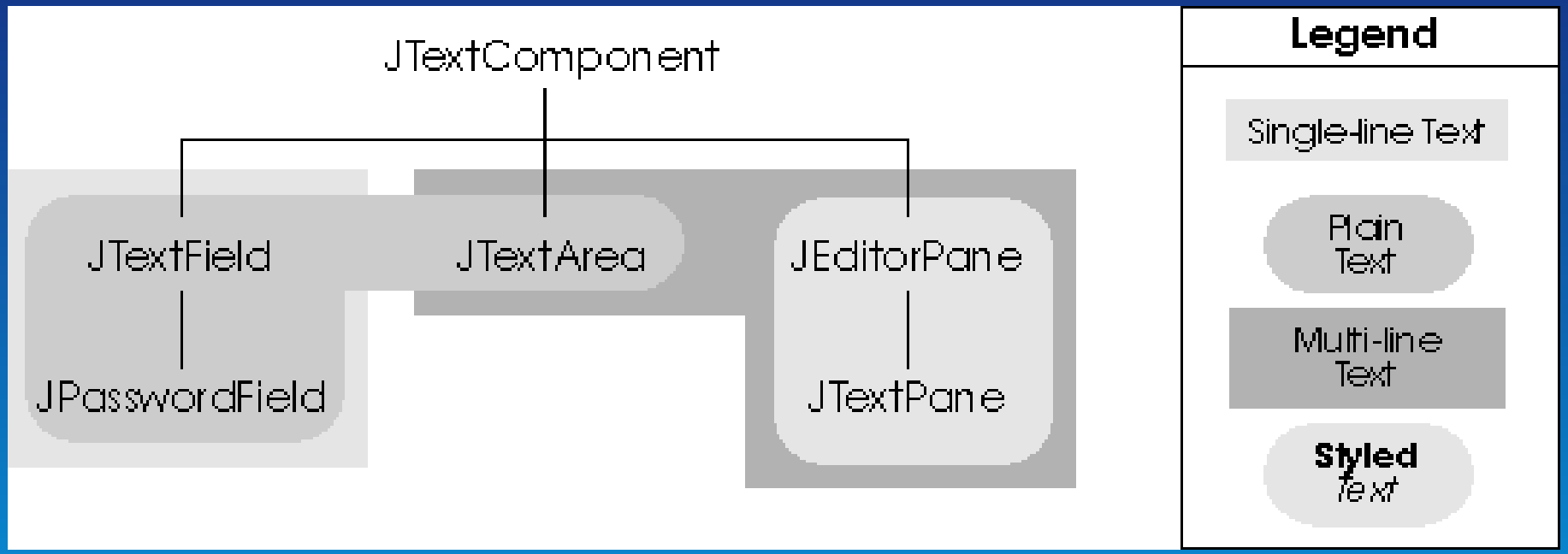


- Dépasser le cadre des composants AWT à fonctionnalités simples
- Modèle de conception d'applications
- 3 objets communicants : modèle, vue & contrôleur
  - Modèle ≡ représentation logique sous-jacente
  - Vue ≡ représentation visuelle
  - Contrôleur ≡ comment gérer les entrées
- Si un modèle change il avertit les vues qui dépendent de lui
- Plusieurs vues possibles d'un même modèle
- On peut changer le type de vue sans casser le modèle



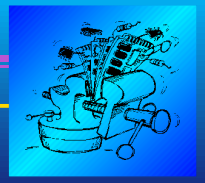


- Composants textuels éditables




- Notion de document séparée (MVC)
- Edition via des actions
- Fonctions clavier redéfinissables
- *Undo/Redo*





- Plusieurs éditeurs pour lire/écrire

  **DefaultEditorKit** texte simple

  **StyledEditorKit** texte enrichi (*Tim's Rich Text Format*)

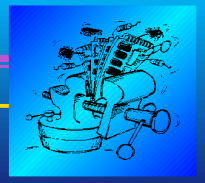
  **HTMLEditorKit** HTML

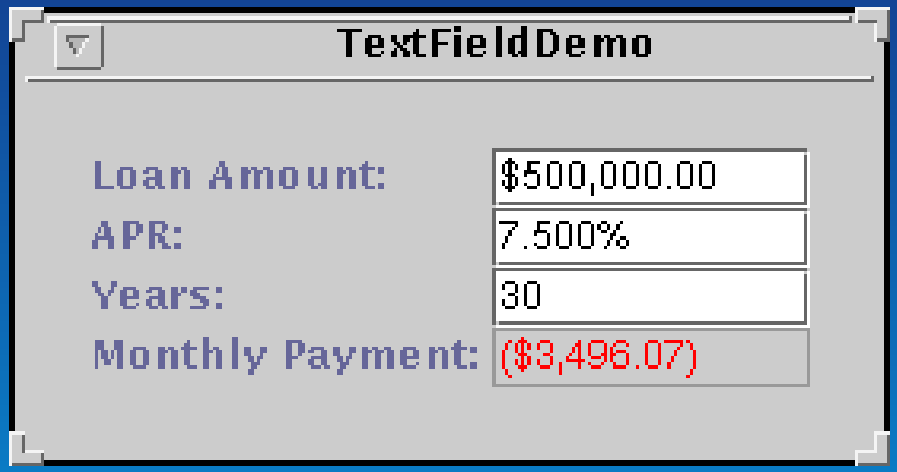
  **RTFEditorKit** RTF

- Un des objets les plus compliqués de Swing...
- Trop complet pour être détaillé ici...

`http://deauville.ensmp.fr/tutorial/ui/swing/text.html`





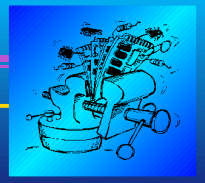



Validation de la date entrée :

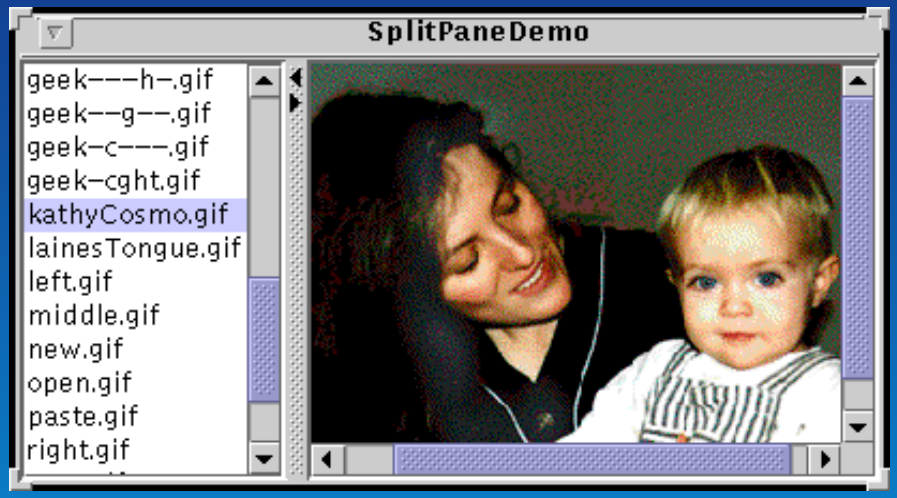
```
protected class WholeNumberDocument extends PlainDocument {
    public void insertString(int offs, String str, AttributeSet a)
                    throws BadLocationException {
        char[] source = str.toCharArray();
        char[] result = new char[source.length];
        int j = 0;

        for (int i = 0; i  result.length; i++) {
        if (Character.isDigit(source[i]))
            result[j++] = source[i];
        else {
            toolkit.beep();
            System.err.println("insertString: " + source[i]);
        }
        }
        super.insertString(offs, new String(result, 0, j), a);
    }
}
```

http://deauville.ensmp.fr/tutorial/ui/swing/textfield.html





# Groupe d'articles sélectionnables

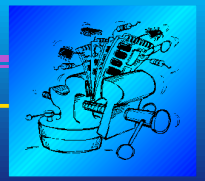



```
...where member variables are declared
   this Vector is initialized from a properties file...
static Vector imageList;
...where the GUI is created...
// Create the list of images and put it in a scroll pane
JList listOfImages = new JList(imageList);
```

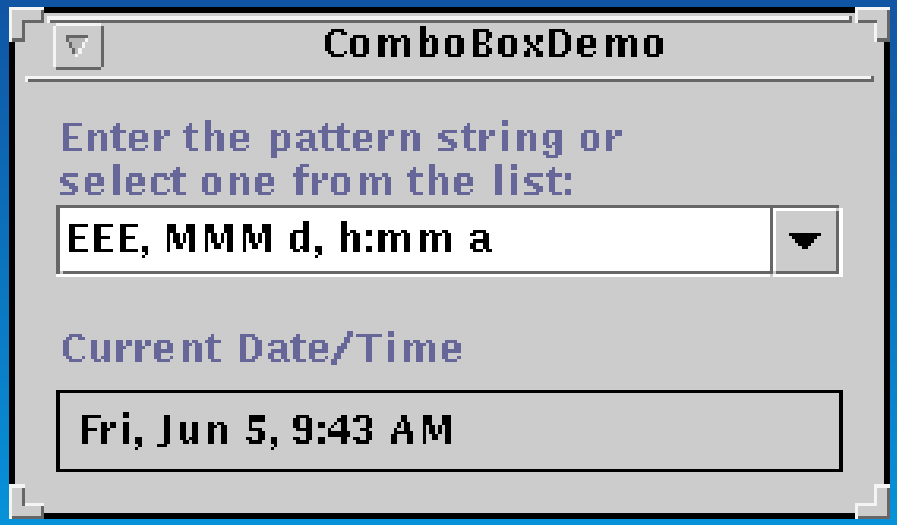
```
listOfImages.setSelectionMode(ListSelectionModel.SINGLE_SELECTION);
listOfImages.setSelectedIndex(0);
listOfImages.addListSelectionListener(this);
JScrollPane listScrollPane = new JScrollPane(listOfImages);

public void valueChanged(ListSelectionEvent e) {
    if (e.getValueIsAdjusting())
        return;

    JList theList = (JList)e.getSource();
    if (theList.isSelectionEmpty()) {
        picture.setIcon(null);
    } else {
        int index = theList.getSelectedIndex();
        ImageIcon newImage = new ImageIcon("images/" +
                                    (String)imageList.elementAt(index));
        picture.setIcon(newImage);
        picture.setPreferredSize(new Dimension(newImage.getIconWidth(),
                                               newImage.getIconHeight()
        picture.revalidate();
    }
}
```

http://deauville.ensmp.fr/tutorial/ui/swing/list.html





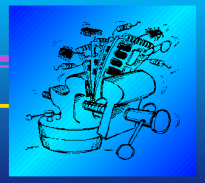
- Bouton + menu descendant + champ textuel éventuellement éditable
- Possibilité d'utiliser un modèle de données `ComboBoxModel`

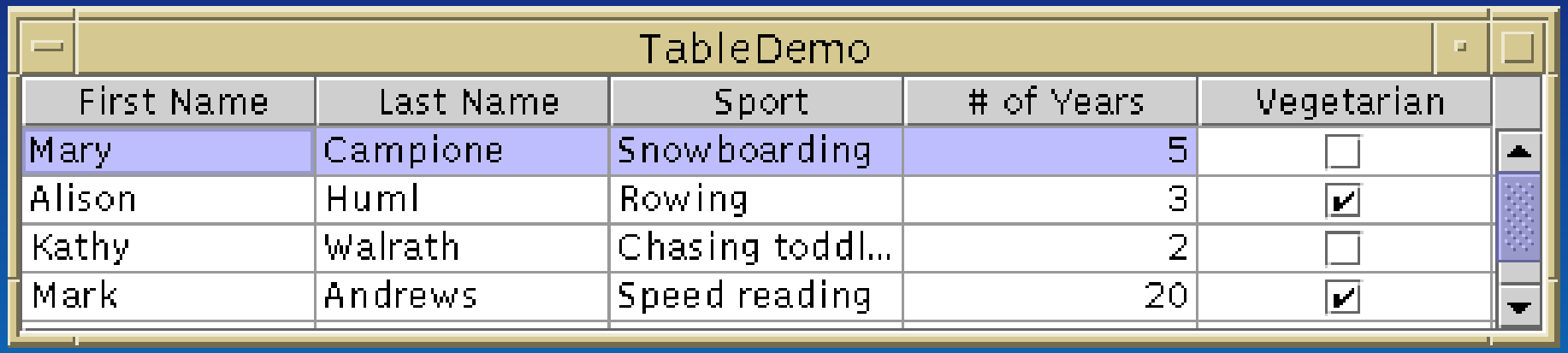



```
currentPattern = patternExamples[0];
. . .
JComboBox patternList = new JComboBox(patternExamples);
patternList.setEditable(true);
patternList.setSelectedIndex(0);
patternList.setAlignmentX(Component.LEFT_ALIGNMENT);
PatternListener patternListener = new PatternListener();
patternList.addActionListener(patternListener);

... // In PatternListener:
public void actionPerformed(ActionEvent e) {
    JComboBox cb = (JComboBox)e.getSource();
    String newSelection = (String)cb.getSelectedItem();
    currentPattern = newSelection;
    reformat();
}
```

http://deauville.ensmp.fr/tutorial/ui/swing/combobox.html

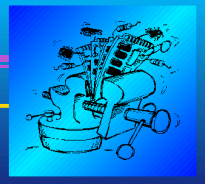







- Affiche un tableau générique
- Chaque cellule peut avoir son propre afficheur et éditeur
- L'utilisateur peut changer la taille et la position des colonnes avec la souris
- Modèle de données via `AbstractTableModel`
- Un des objets les plus compliqués de Swing...

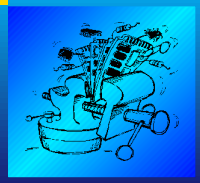




```
public class TableDemo extends JFrame {
    public TableDemo() {
        super("TableDemo");

        MyTableModel myModel = new MyTableModel();
        JTable table = new JTable(myModel);
        table.setPreferredScrollableViewportSize(new Dimension(500, 70));

        //Create the scroll pane and add the table to it.
        JScrollPane scrollPane = new JScrollPane(table);

        //Add the scroll pane to this window.
        setContentPane(scrollPane);

        addWindowListener(new WindowAdapter() {
            public void windowClosing(WindowEvent e) {
                System.exit(0);
            }
        });
    }

    class MyTableModel extends AbstractTableModel {
        final String[] columnNames = {"First Name",
                                      "Last Name",
                                      "Sport",
                                      "# of Years",
                                      "Vegetarian"};
        final Object[][] data = {
            {"Mary", "Campione",
             "Snowboarding", new Integer(5), new Boolean(false)},
            {"Alison", "Huml",
             "Rowing", new Integer(3), new Boolean(true)},
            {"Kathy", "Walrath",
             "Chasing toddlers", new Integer(2), new Boolean(false)},
            {"Mark", "Andrews",
             "Speed reading", new Integer(20), new Boolean(true)},
            {"Angela", "Lih",
             "Teaching high school", new Integer(4), new Boolean(fals
        };

        public int getColumnCount() {
            return columnNames.length;
        }

        public int getRowCount() {
            return data.length;
        }

        public String getColumnName(int col) {
            return columnNames[col];
        }

        public Object getValueAt(int row, int col) {
            return data[row][col];
        }

        /*
         * JTable uses this method to determine the default renderer/
         * editor for each cell.  If we didn't implement this method,
         * then the last column would contain text ("true"/"false"),
         * rather than a check box.
         */
        public Class getColumnClass(int c) {
            return getValueAt(0, c).getClass();
```

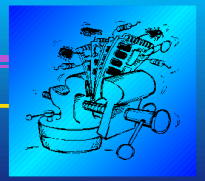






```
    }

    /*
     * Don't need to implement this method unless your table's
     * editable.
     */
    public boolean isCellEditable(int row, int col) {
        //Note that the data/cell address is constant,
        //no matter where the cell appears onscreen.
        if (col < 2) {
            return false;
        } else {
            return true;
        }
    }

    /*
     * Don't need to implement this method unless your table's
     * data can change.
     */
    public void setValueAt(Object value, int row, int col) {
        if (data[0][col] instanceof Integer) {
            //If we don't do something like this, the column
            //switches to contain Strings.
            try {
                data[row][col] = new Integer((String)value);
            } catch (NumberFormatException e) {
                if (SwingUtilities.isEventDispatchThread()) {
                    JOptionPane.showMessageDialog(TableDemo.this,
                        "The \"" + getColumnName(col)
                        + "\" column accepts only integer values.")
                } else {
                    System.err.println("User attempted to enter non
                                       + " value (" + value
                                       + ") into an integer-only column
                }
            }
        } else {
            data[row][col] = value;
        }
    }

    public static void main(String[] args) {
        TableDemo frame = new TableDemo();
        frame.pack();
        frame.setVisible(true);
    }
```



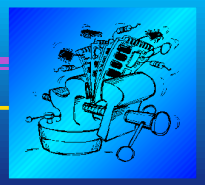


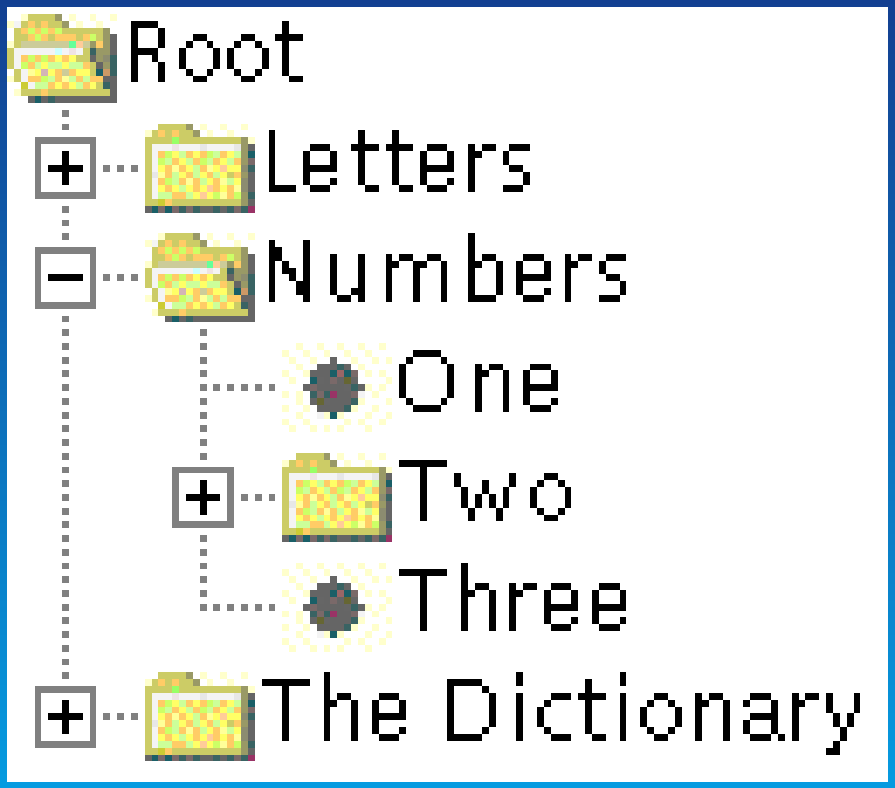
Si les données changent autrement que par l'édition, il faut prévenir en appelant des méthodes `fire`*`XYZT`* de l'`AbstractTableModel` :

`fireTableRowsInserted`, `fireTableStructureChanged`,...

`http://deauville.ensmp.fr/tutorial/ui/swing/table.html`



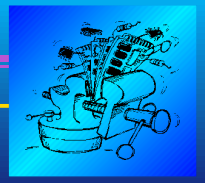


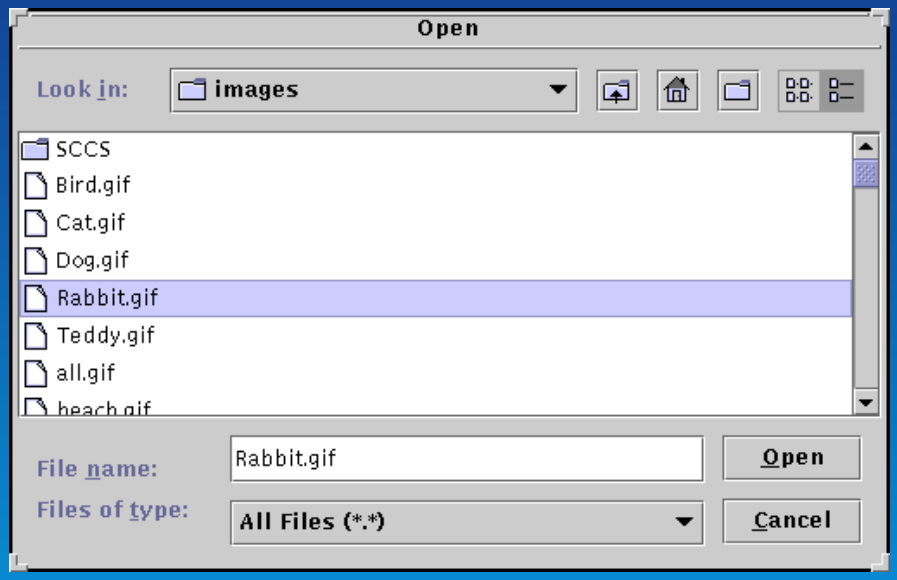



- Affiche des données hiérarchiques
- Modèle `TreeModel`

`http://deauville.ensmp.fr/tutorial/ui/swing/tree.html`





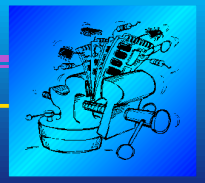



```
private JFileChooser filechooser = new JFileChooser();
...// Open...
int returnVal = filechooser.showOpenDialog(FileChooserDemo.this);
...// Save...
int returnVal = filechooser.showSaveDialog(FileChooserDemo.this);
if (returnVal == JFileChooser.APPROVE_OPTION) {
    File file = filechooser.getSelectedFile();
    log.append("Saving: " + file.getName() + "." + newline);
    } else {
    log.append("Save command cancelled by user." + newline);
    }
}
```

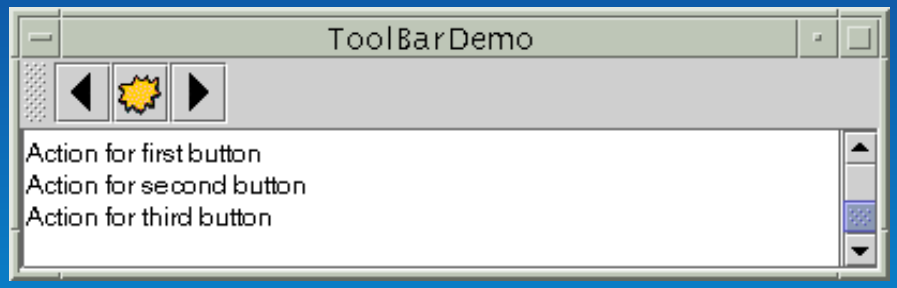
http://deauville.ensmp.fr/tutorial/ui/swing/filechooser.html

`JColorChooser` dans le même style





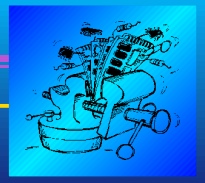
Peut être déplacée, voire sortie

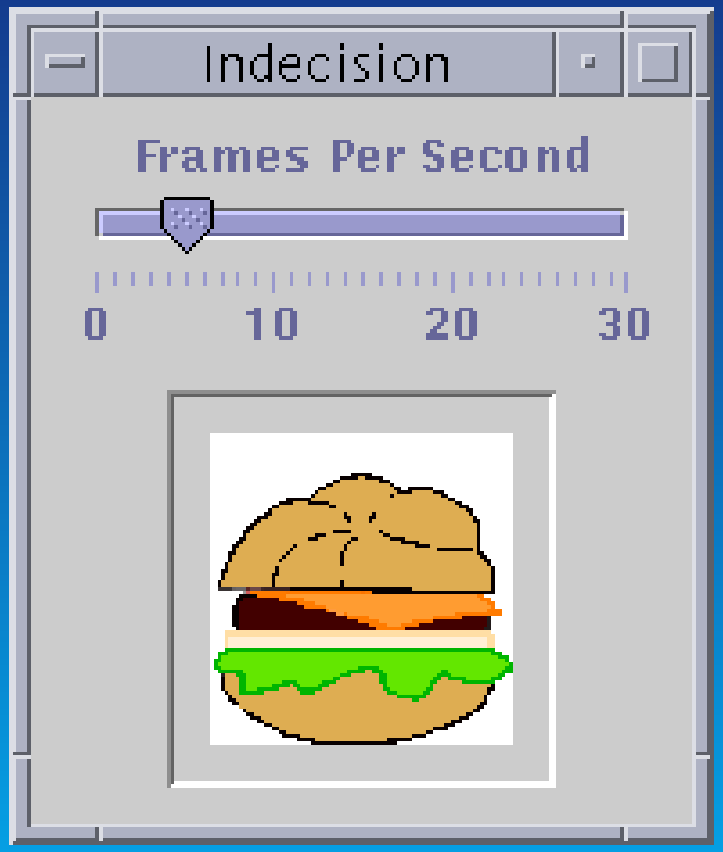



```
ToolBar toolBar = new JToolBar();
button = new JButton(new ImageIcon("images/left.gif"));
    ...
toolBar.add(button);
    ...
JPanel contentPane = new JPanel();
contentPane.setLayout(new BorderLayout());
...
contentPane.add(toolBar, BorderLayout.NORTH);
contentPane.add(scrollPane, BorderLayout.CENTER);
}
```

http://deauville.ensmp.fr/tutorial/ui/swing/toolbar.html



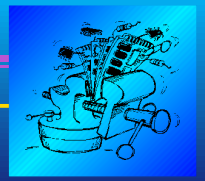





```
JSlider framesPerSecond = new JSlider(JSlider.HORIZONTAL, 0, 30, FPS_IN
framesPerSecond.addChangeListener(new SliderListener());
framesPerSecond.setMajorTickSpacing(10);
framesPerSecond.setMinorTickSpacing(1);
framesPerSecond.setPaintTicks(true);
framesPerSecond.setPaintLabels(true);
framesPerSecond.setBorder(BorderFactory.createEmptyBorder(0,0,10,0));
. . .
//add the slider to the content pane
contentPane.add(framesPerSecond);
```

http://deauville.ensmp.fr/tutorial/ui/swing/slider.html

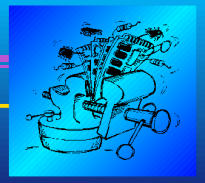


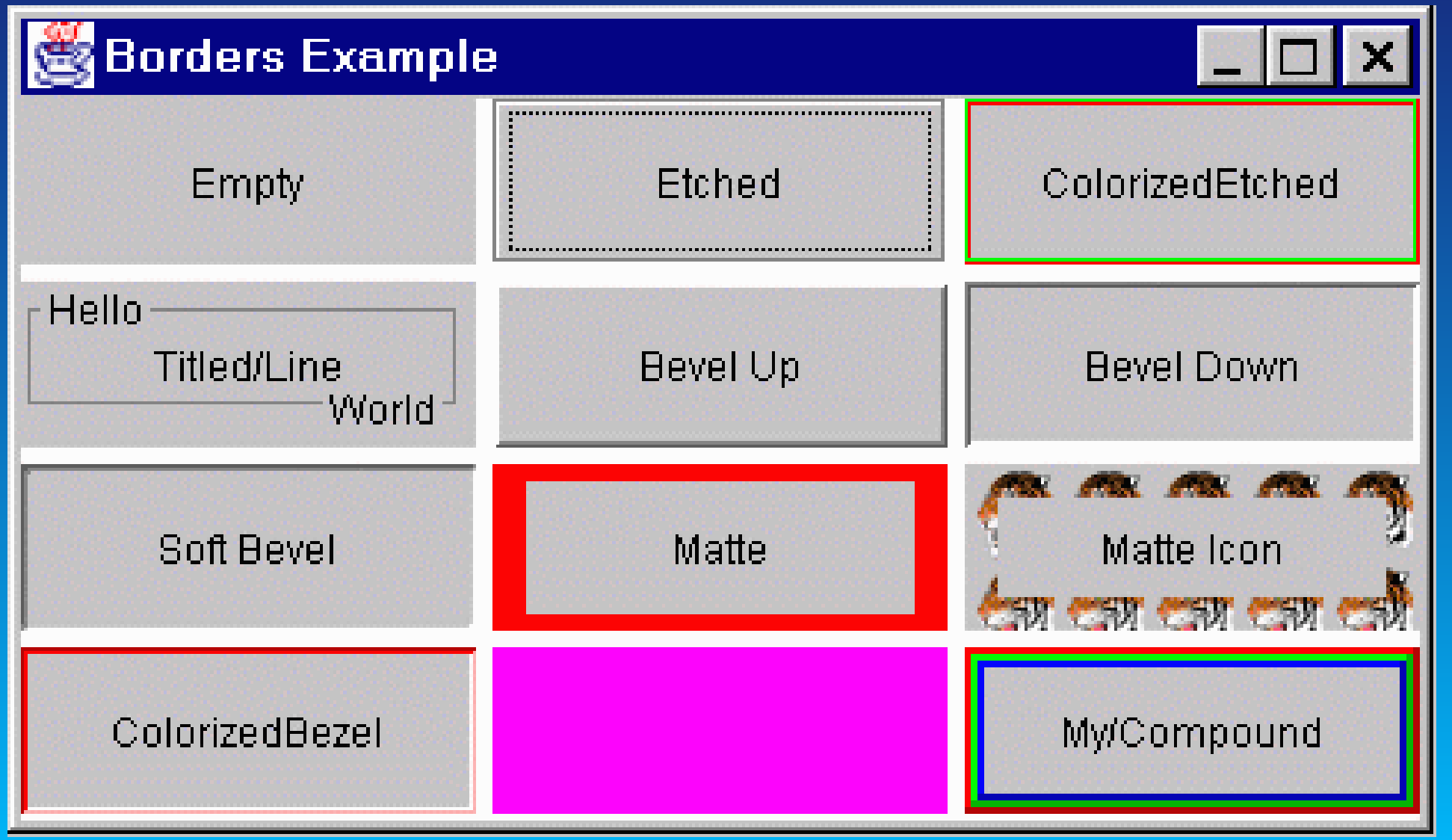



```
b1.setToolTipText("Click this button to disable the middle button.");
b2.setToolTipText("This middle button does nothing when you click it.");
b3.setToolTipText("Click this button to enable the middle button.");
```

http://deauville.ensmp.fr/tutorial/ui/swing/tooltip.html

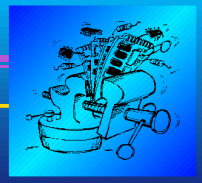


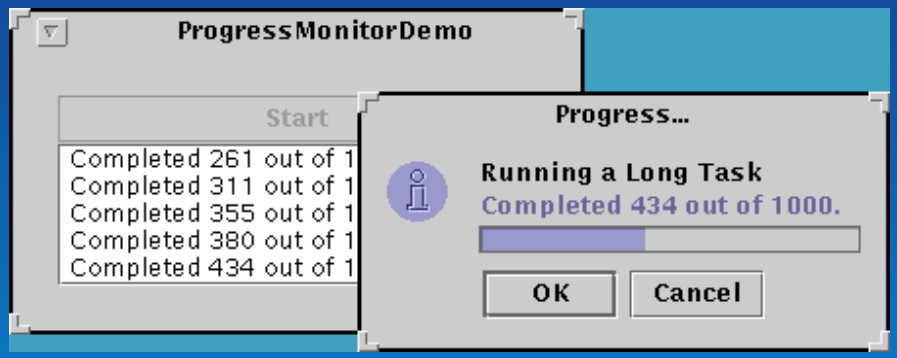





Possibilité de définir aussi ses propres bordures avec `Border`







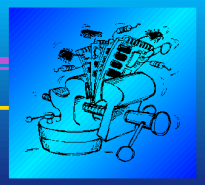

**JProgressBar** affiche l'avancement d'une tâche

**ProgressMonitor** surveille l'avancement d'une tâche

**ProgressMonitorInputStream** surveille l'avancement de la lecture d'un *stream*

`http://deauville.ensmp.fr/tutorial/ui/swing/progressbar.html`

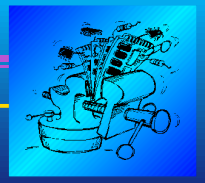




- Swing est basé sur AWT
- Possible de mélanger du poids léger avec du poids lourd
- Un composant léger peut être transparent, un composant lourd est toujours opaque
- Un composant lourd est toujours au dessus d'un composant léger
- À éviter néanmoins

`http://java.sun.com/products/jfc/tsc/swingdoc-archive/mixing.html`

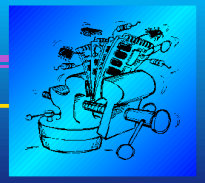




- Bien d’autres choses (Java3D...)
- Jolies interfaces graphiques
- Portable
- Tout (ou presque) est dans Java et ses classes

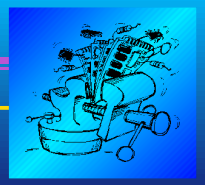
⇝ Travaux pratiques





- Jouer avec la démonstration SwingSet
- Créer un tableur





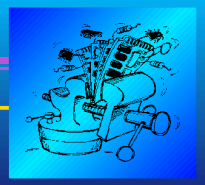
- Charger la dernière version si nécessaire

  `http://developer.java.sun.com/developer/earlyAccess/jfc.html`

- Essayer toutes les possibilités

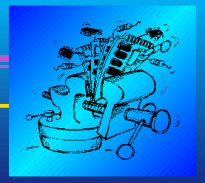
```
cd /usr/local/java/java1.2/demo/jfc/SwingSet/
java SwingSet
```





- Créer une fenêtre avec un tableau éditable
- « Câbler en dur » les opérations à effectuer
- Utiliser le modèle de données pour faire les calculs





# List of Slides