D[i] | Département Informatique

UNIVERSITÉ TOULOUSE III PAUL SABATIER
Université de Toulouse
Faculté Sciences et Ingénierie

Licence Informatique

Librairie Java Swing & Programmation événementielle

Florent CABRIC

Florent.cabric@irit.fr

Principaux composants conteneurs

Les composants / containers

- Composants qui ont pour but principal de contenir d'autres composants

- Les principaux conteneurs :
  - JPanel
  - JScrollPane
  - JSplitPane
  - JTabbedPane
  - JDesktopPane

Conteneur de composants - **JPanel**

- Contenant générique
- N'est pas visuellement détectable
- Permet de structurer une interface en rassemblant un ensemble de composants liés à une activité de l'utilisateur
- Principaux constructeurs

JPanel()

JPanel(LayoutManager layout)

JPanel

- Modifier le layout utilisé

```
void setLayout(LayoutManager layout)
```

- Ajouter un composant au panneau

```
void add(Component c)
```

- Supprimer un composant du panneau

```
void remove(Component c)
```

Gestion des barres de défilement - **JScrollPane**

- Conteneur qui prend 1 composant

- Principaux constructeurs

  JScrollPane(Component view)

  JScrollPane(Component view, int vsb, int hsb)

- vsb et hsb : gestion de l'affichage des barres de défilement
  - Les différentes utilisations possibles sont enregistrées sous forme de *static final int* dans **JScrollPane**

  - VERTICAL_SCROLLBAR_AS_NEEDED
  - VERTICAL_SCROLLBAR_NEVER
  - VERTICAL_SCROLLBAR_ALWAYS
  - HORIZONTAL_SCROLLBAR_AS_NEEDED
  - HORIZONTAL_SCROLLBAR_NEVER
  - HORIZONTAL_SCROLLBAR_ALWAYS

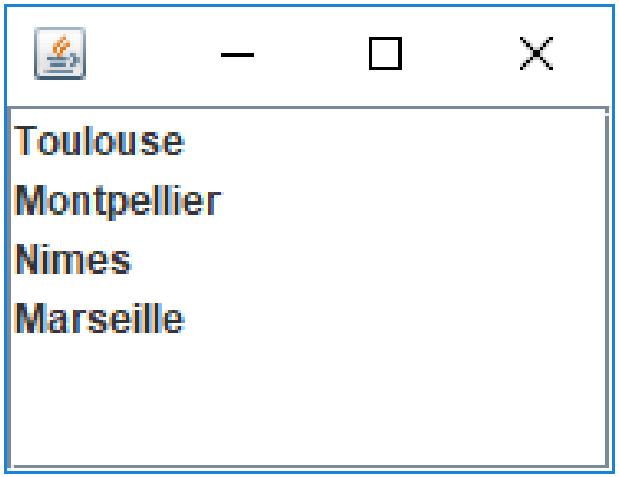
Gestion des barres de défilement - **JScrollPane**

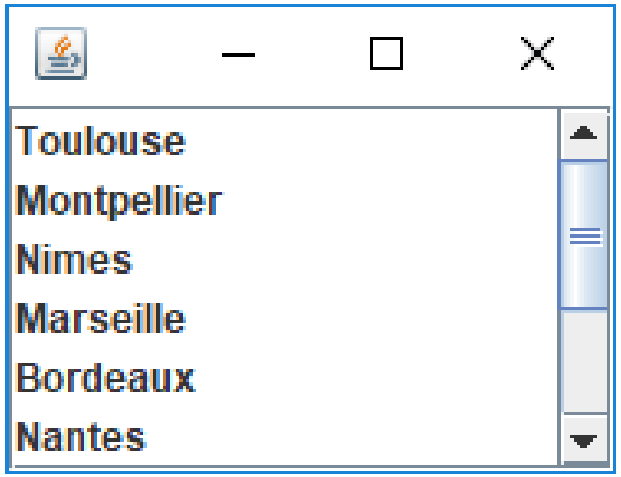
- La partie visible du JScrollPane s'appelle *viewport*

- JScrollPane gère automatiquement la création et la suppression des barres de défilement (scrollbar)
  - Si le composant contenu dans le JScrollPane est plus grand verticalement que le *viewport* → une barre de défilement verticale est créée
  - Si le composant contenu dans le JScrollPane est plus grand horizontalement que le *viewport* → une barre de défilement horizontale est créée
  - Si le composant contenu dans le JScrollPane est plus grand verticalement ET horizontalement que le *viewport* → deux barres de défilement sont créees

JScrollPane - Exemple

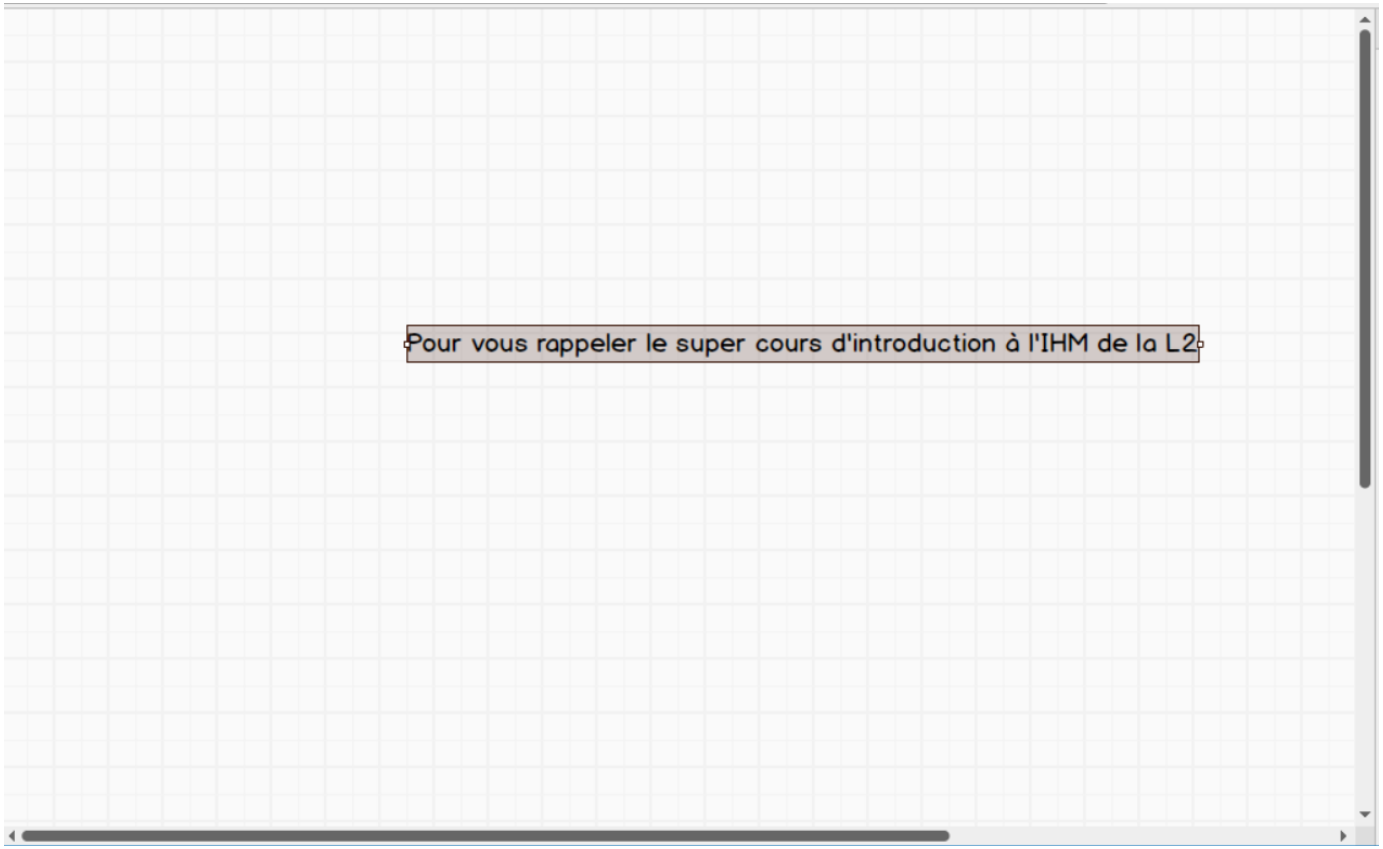
```
String[] nomVille;
...
JList<String> listeVille = new JList<String>(nomVille);
...
frame.add(new JScrollPane(listeVille,
                          JScrollPane.VERTICAL_SCROLLBAR_AS_NEEDED,
                          JScrollPane.HORIZONTAL_SCROLLBAR_AS_NEEDED));
```





JScrollPane – Exemple d'application

Balsamiq



Scinder un espace en deux zones - **JSplitPane**

- Scinder un espace en 2 (et uniquement deux !) zones séparées par une barre qui peut être déplacée dynamiquement

- Par défaut,
  - le premier composant (haut ou gauche selon orientation) prend la taille dont il a besoin
  - Le second prend la place restante

- Principaux constructeurs

  JSplitPane(int orient)

  JSplitPane(int orient, Component compG, Component compD)

  - Les différentes orientations possibles sont enregistrées sous forme de *static final int* dans **JSplitPane**
    - JSplitPane.*VERTICAL_SPLIT*
    - JSplitPane.*HORIZONTAL_SPLIT*

JSplitPane

- Positionner la barre de séparation

  void **setDividerLocation**(int location)

- Donner une taille à la barre de séparation

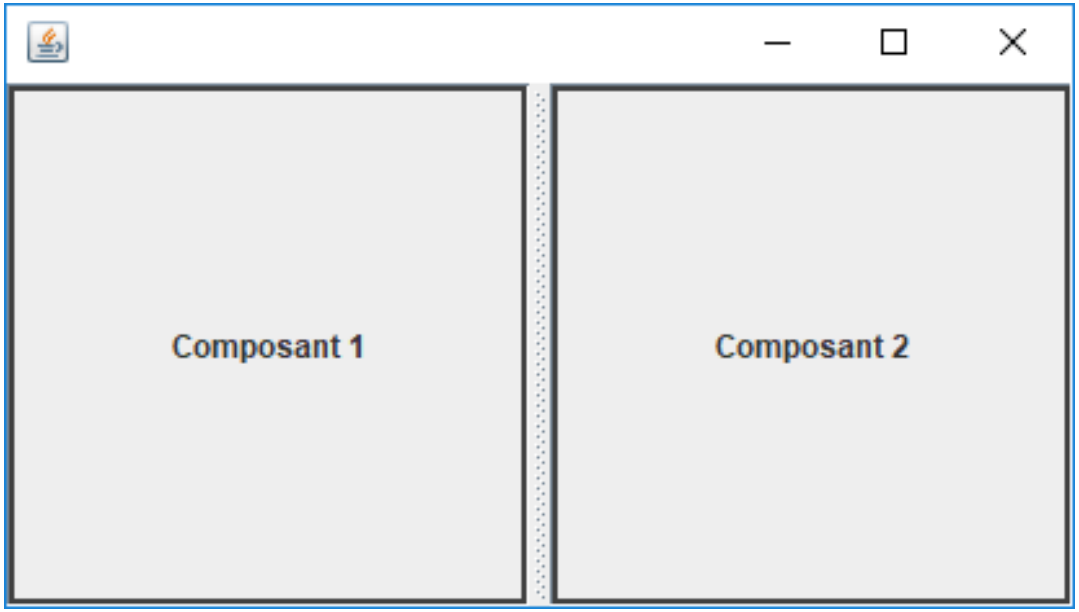
  void **setDividerSize**(int size)

- Modifier les composants à afficher

  void **setBottomComponent**(Component c)
  void **setLeftComponent**(Component c)
  void **setRightComponent**(Component c)
  void **setTopComponent**(Component c)

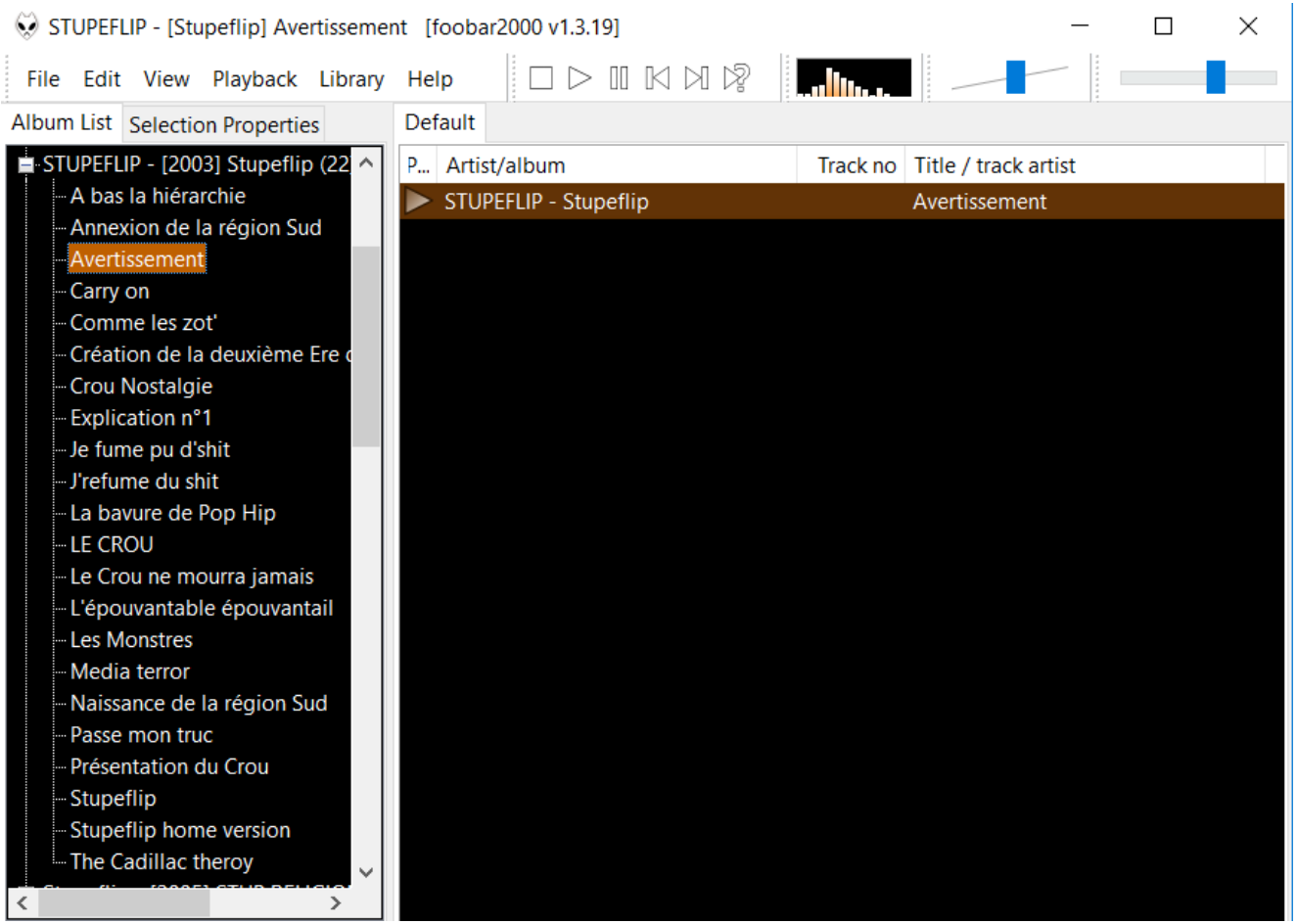
JSplitPane - Exemple

```
JSplitPane splitPane = new JSplitPane(JSplitPane.HORIZONTAL_SPLIT,
                                      genereComposant(),
                                      genereComposant());

frame.setLayout(new BorderLayout());
frame.add(splitPane, BorderLayout.CENTER);
```



JSplitPane – Exemple d'application

Foobar



Système d'onglets - **JTabbedPane**

- Permet d'avoir plusieurs panneaux sur la même surface
  - Chaque panneau est accessible via un onglet

- Principaux constructeurs

JTabbedPane()

JTabbedPane(int tabPlacement)

  - Les différents positionnements possibles pour l'onglet sont enregistrés sous forme de *static final int* dans **JTabbedPane**
    - TOP,
    - BOTTOM,
    - RIGHT,
    - LEFT

JTabbedPane – Gestion des onglets

- Ajout d'un onglet avec un titre pour l'onglet

  void **addTab**(String title, Component c)

- Activer / désactiver un onglet

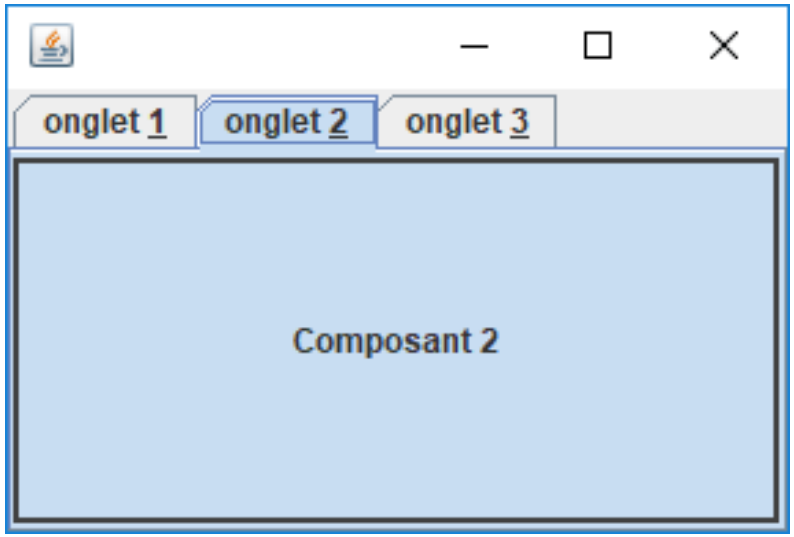
  void **setEnabledAt**(int index, boolean b)

- Modifier le composant d'un onglet

  void **setTabComponentAT**(int index, Component c)

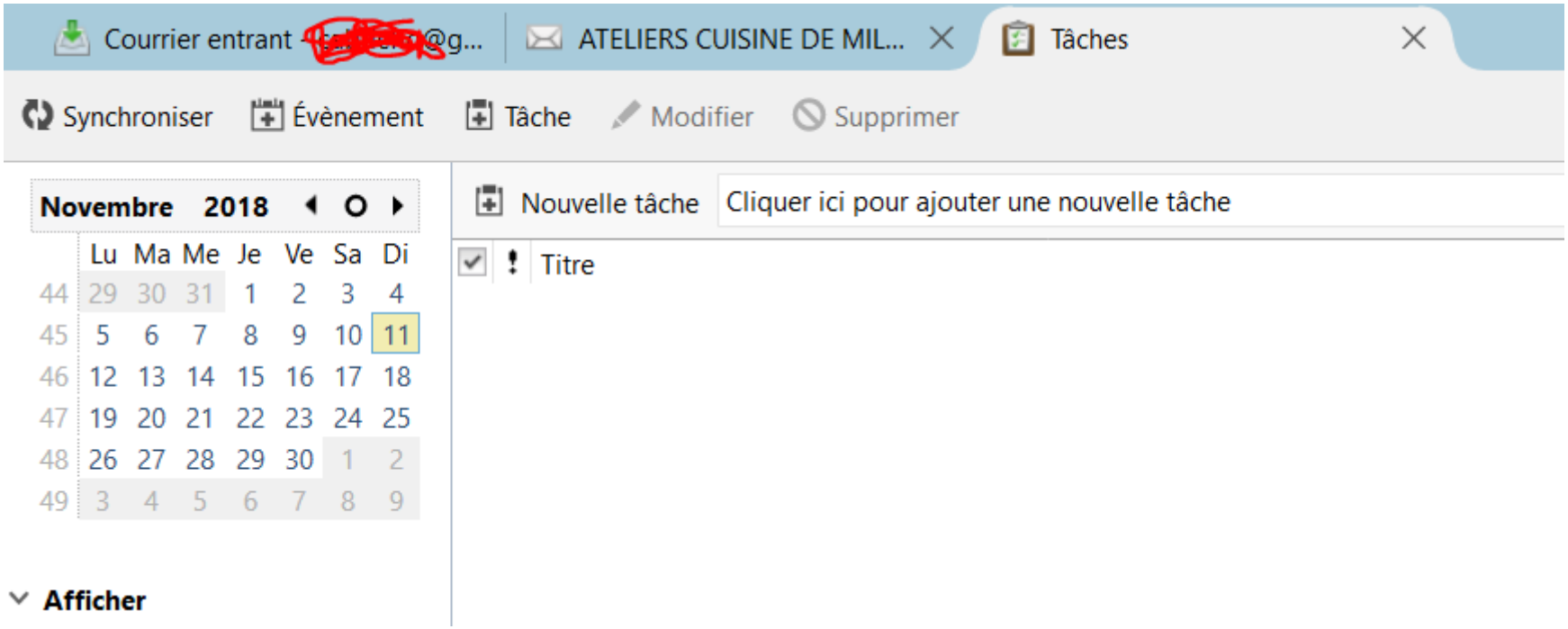
JTabbedPane - Exemple

```
JTabbedPane onglet = new JTabbedPane();
onglet.addTab("onglet 1", genereComposant());
onglet.setMnemonicAt(0, KeyEvent.VK_1);
onglet.addTab("onglet 2", genereComposant());
onglet.setMnemonicAt(1, KeyEvent.VK_2);
onglet.addTab("onglet 3", genereComposant());
onglet.setMnemonicAt(2, KeyEvent.VK_3);
```



Système d'onglets – Exemple d'application

Mozilla Thunderbid



Bureau et fenêtres internes

- Gérer des fenêtres à l'intérieur d'une fenêtre
  - Conçues comme un bureau avec **JDesktopPane**
  - Pour contenir des fenêtres internes **JInternalFrame**

- **JDesktopPane**

```
JDesktopPane()
```

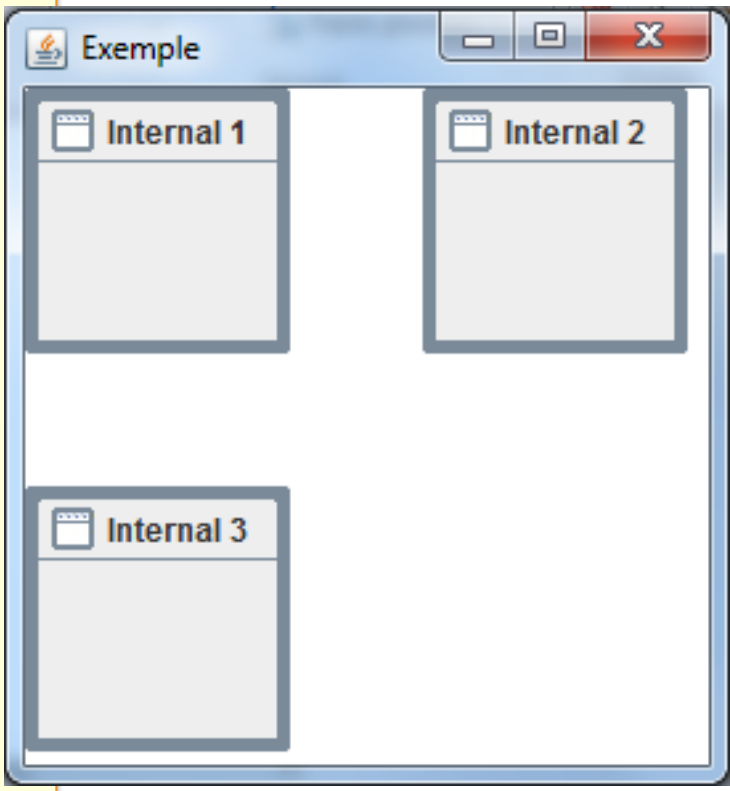
  - Ajouté dans l'interface comme tout autre composant

- **JInternalFrame** : Ajouté au JDesktopPane via la méthode add

```
JInternalFrame(String title,
               boolean resizable,
               boolean closable,
               boolean maximizable,
               boolean iconifiable)
```

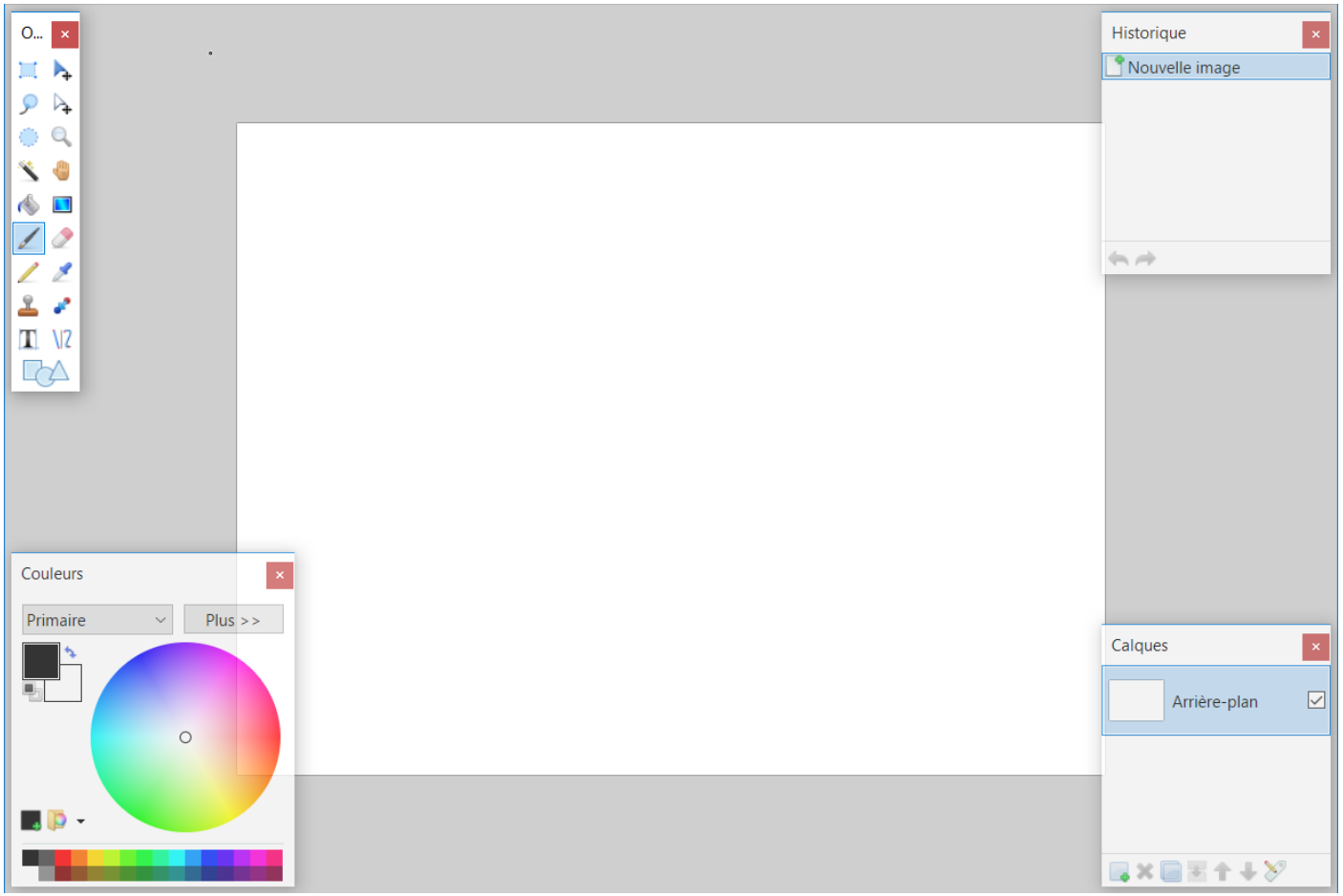
Exemple

```
bureau=new JDesktopPane();
setLayout(new BorderLayout());
add(bureau);

JInternalFrame i1 = new JInternalFrame("Internal 1"
i1.setBounds(0, 0, 100, 100);
i1.setVisible(true);
JInternalFrame i2 = new JInternalFrame("Internal 2"
i2.setBounds(150, 0, 100, 100);
i2.setVisible(true);
JInternalFrame i3 = new JInternalFrame("Internal 3"
i3.setBounds(0, 150, 100, 100);
i3.setVisible(true);

bureau.add(i1);
bureau.add(i2);
bureau.add(i3);
```



JDekstopPane et JInternalFrame : Exemple d'app

Paint.net



Boite à outils - **JToolBar**

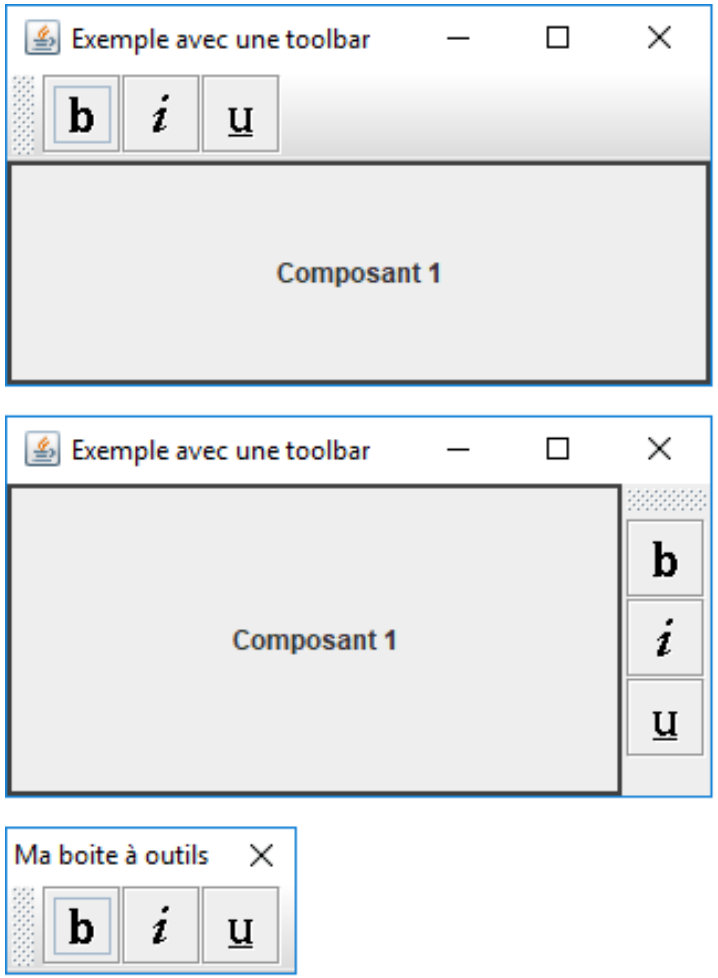
- Boite à outils
  - Ancrable sur un bord de l'interface
  - Détachable

- La Toolbar ne peut contenir **uniquement** des boutons

- **Principal constructeur**

  JToolBar(String name, int orient)

  - Les différentes orientations possibles sont enregistrées sous forme de *static final int* dans **JToolBar**
    - JToolBar.*VERTICAL*
    - JToolBar.*HORIZONTAL*



JToolBar – Gestion des boutons

- Ajouter un bouton à la boite à outils

  void **add**(Action a)

- Ajouter un séparateur entre deux boutons

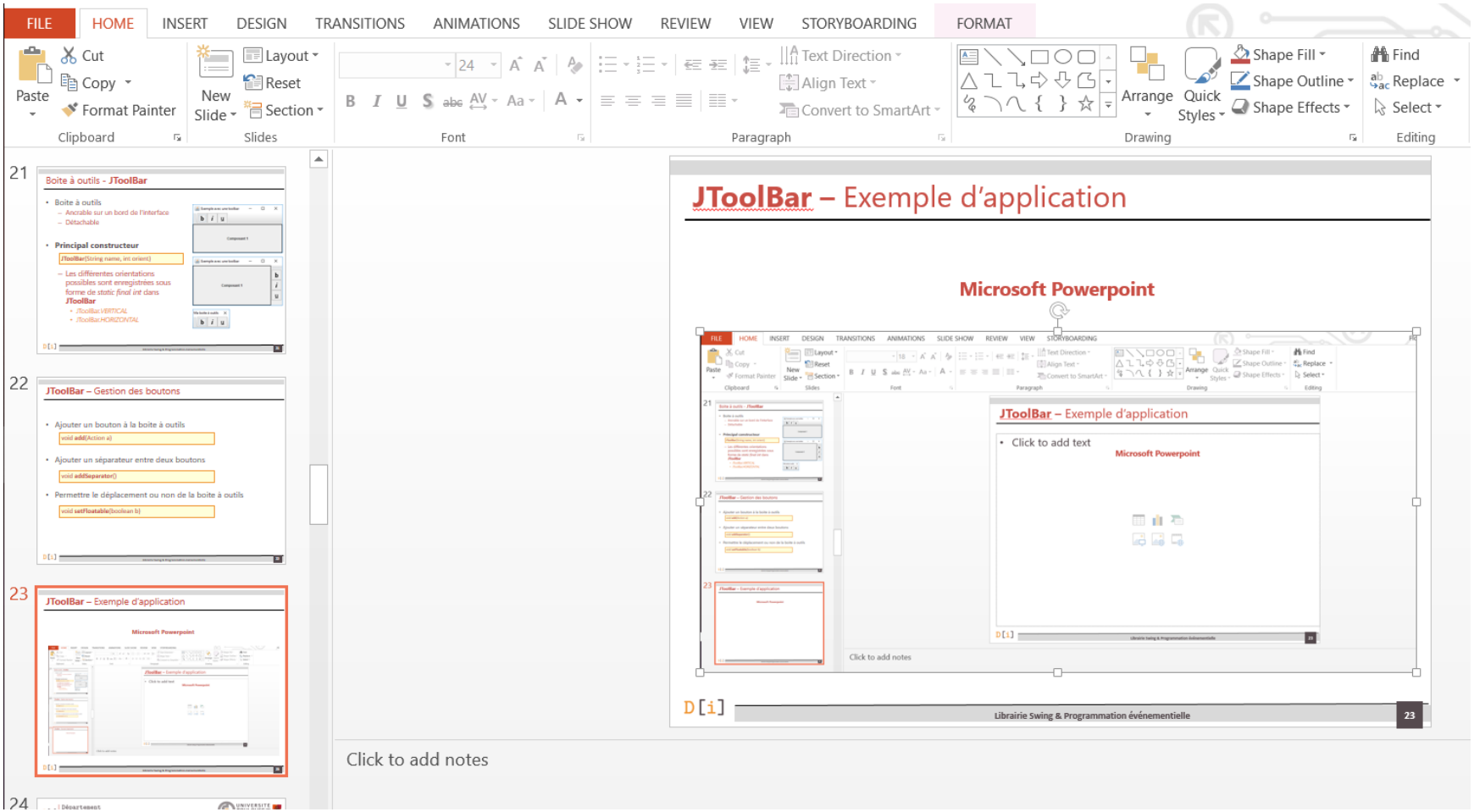
  void **addSeparator**()

- Permettre le déplacement ou non de la boite à outils

  void **setFloatable**(boolean b)

JToolBar – Exemple d'application

Microsoft Powerpoint



Gestion de la disposition des éléments

Container - Disposition

- Comment positionner les composants les uns par rapport aux autres?
- Comment un container agence-t-il les composants qu'il contient?
  - Gestion du redimensionnement dynamique
- Utilisation de différents gestionnaires de géométrie

  → **LayoutManager**

Layout

Une stratégie par type de layout

- Dispositions simples, peu d'éléments graphiques
  - BorderLayout (5 zones)
  - FlowLayout
  - BoxLayout
- Disposition matricielle
  - GridLayout
- Réalisation de formulaires, dispositions recherchées
  - FormLayout (n'appartient pas au JDK)
  - GridBagLayout
  - **GroupLayout (recommandé avec Netbeans)**

Positionner les composants

- Sur des containers
  - Fenêtre
  - Composants / containers
- Au moyen d'un gestionnaire de géométrie : LayoutManager

```
void setLayout(LayoutManager layout)
```

- Ajout des composants sur le conteneur

```
Component add(Component c)
```

Positionnement sans LayoutManager

- Définir qu'il n'y a pas de LayoutManager

```
setLayout(null)
```

- Positionnement et taille de chaque composant au pixel près

```
setLocation(x,y)
setSize(new Dimension(l,h))
```

```
setBounds(x,y,l,h)
```

→ **Problème :** Pas de redimensionnement des composants en cas de redimensionnement de la fenêtre

Normalement vous n'aurez que rarement le besoin de faire cela !

FlowLayout - Principe

- Arrange les composants de gauche à droite

  FlowLayout(int align)

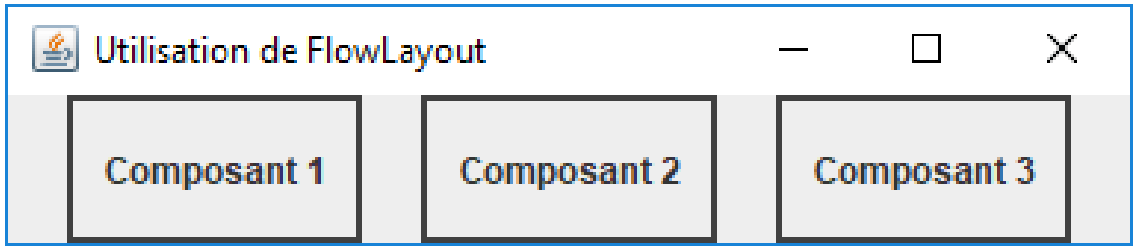
  FlowLayout(int align, int hgap, int vgap)

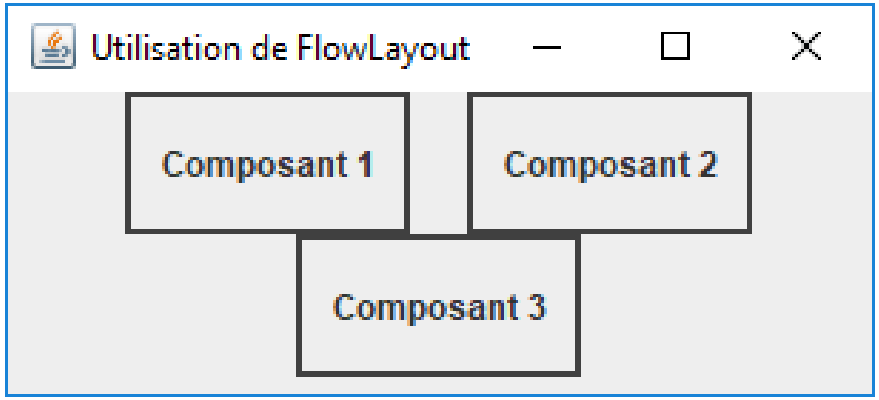
- Alignement des composants sur le conteneur
  - Les différents alignements possibles sont enregistrés sous forme de *static final int* dans **FlowLayout**
    - LEFT
    - CENTER (par défaut)
    - RIGHT

- Hgap et Vgap : Espaces en pixel autour des composants

FlowLayout - Exemple

```
FlowLayout layout = new FlowLayout(FlowLayout.CENTER,20,0);
JPanel panneau = new JPanel(layout);
panneau.add(genereLabel());
panneau.add(genereLabel());
panneau.add(genereLabel());
```





GridLayout - Principe

- Découpe le conteneur en **N** lignes et **M** colonnes
  - Chaque cellule a les mêmes dimensions

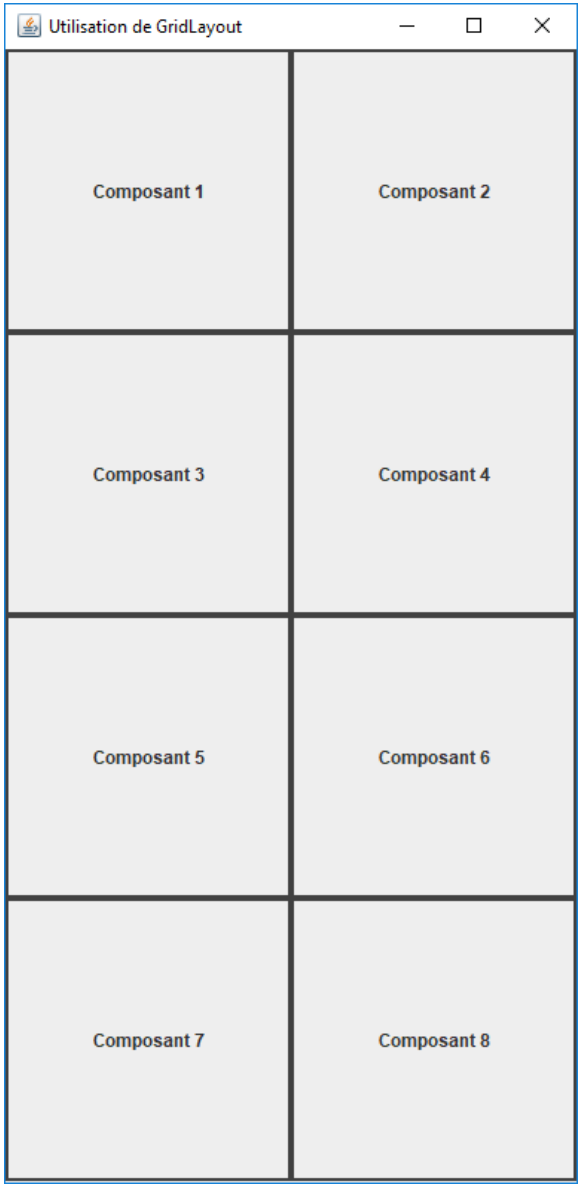
GridLayout(int nbLignes, int nbColonnes)

GridLayout(int nbLignes, int nbColonnes, int hgap, int vgap)

- Un composant ajouté au conteneur prend automatiquement toute l'espace défini pour une cellule
  - Ajouté dans l'ordre de gauche à droite et de haut en bas

GridLayout - Exemple

```
GridLayout layout = new GridLayout(4,2,0,0);
JPanel panneau = new JPanel(layout);
panneau.add(genereLabel());
panneau.add(genereLabel());
panneau.add(genereLabel());
panneau.add(genereLabel());
panneau.add(genereLabel());
panneau.add(genereLabel());
panneau.add(genereLabel());
panneau.add(genereLabel());
```



BoxLayout - Principe

- Arrange les composants sur le conteneur
  - En prenant la taille préférée des composants
  - sur une ligne (X_AXIS)
  - sur une colonne (Y_AXIS)

- Constructeur

  BoxLayout(Container target, int axis)

  - Le conteneur doit être créé avant d'instancier le **BoxLayout**
  - Les différents axes possibles sont enregistrés sous forme de *static final int* dans **BoxLayout**
    - X_AXIS
    - Y_AXIS

Le conteneur **Box**

- Composant transparent pour ajouter plusieurs composants à un endroit donné dans un composant utilisant **BoxLayout**

- Constructeur

  Box(int axis)

- Création au moyen d'une méthode statique

  static Box **createHorizontalBox**()

  static Box **createVerticalBox**()

Le conteneur **Box**

- Il peut contenir
  - Des composants

    Component **add**(Component c)

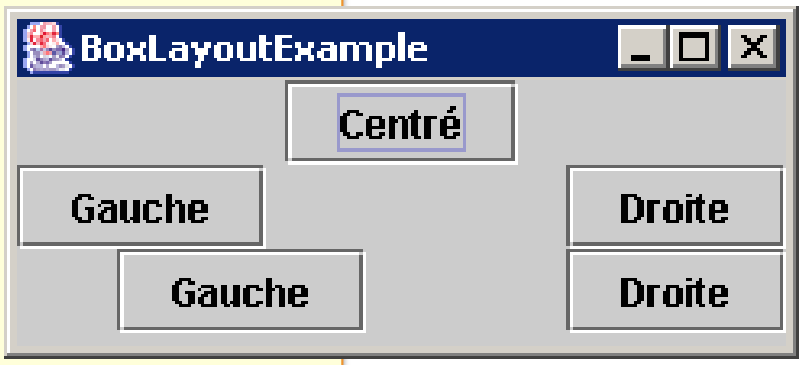
  - Des glues pour « remplir » des espaces entre composants
    - Un composant entre deux glues sera centré
    - Une glue entre deux composants plaquera les composants sur les bords

    static Component **createHorizontalGlue()**

    static Component **createVerticalGlue()**

BoxLayout - Exemple

```
Box line1=new Box(BoxLayout.X_AXIS);
line1.add(Box.createHorizontalGlue());
line1.add(new JButton("Centré"));
line1.add(Box.createHorizontalGlue());
Box line2=new Box(BoxLayout.X_AXIS);
line2.add(new JButton("Gauche"));
line2.add(Box.createHorizontalGlue());
line2.add(new JButton("Droite"));
Box line3=new Box(BoxLayout.X_AXIS);
line3.add(Box.createHorizontalGlue());
line3.add(new JButton("Gauche"));
line3.add(Box.createHorizontalGlue());
line3.add(Box.createHorizontalGlue());
line3.add(new JButton("Droite"));
frame.setLayout(new BoxLayout(frame,BoxLayout.Y_AXIS));
frame.add(line1);
frame.add(line2);
frame.add(line3);
```



BorderLayout - Principe

BorderLayout()

BorderLayout(int hgap, int vgap)

- Peut contenir jusqu'à 5 éléments
  - En haut ou en bas du conteneur
    - → Le composant s'étend sur toute la longueur de la fenêtre
    - → Le composant prend juste la hauteur dont il a besoin
  - A gauche ou à droite du conteneur
    - → Le composant s'étend sur toute la hauteur de la fenêtre
    - → Le composant prend juste la longueur dont il a besoin
  - Au centre du conteneur
    - → Le composant prend toute la place restante

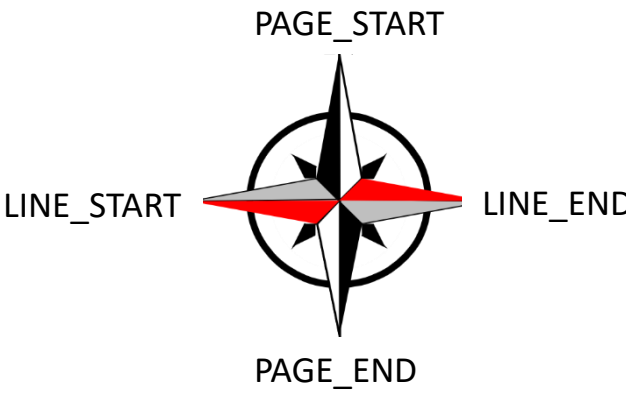
BorderLayout – Ajout des composants

- Ajouté au conteneur avec la contrainte de placement

  void **add**(Component c, Object constraints)

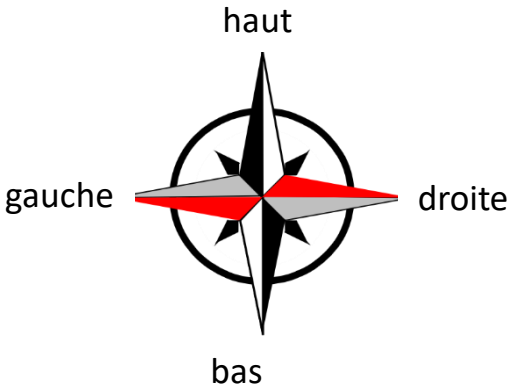
  - Les différents placements possibles sont enregistrés sous forme de *static final String* dans **BorderLayout**
    - CENTER
    - PAGE_START
    - PAGE_END
    - LINE_START
    - LINE_END

En France



Au cas où…

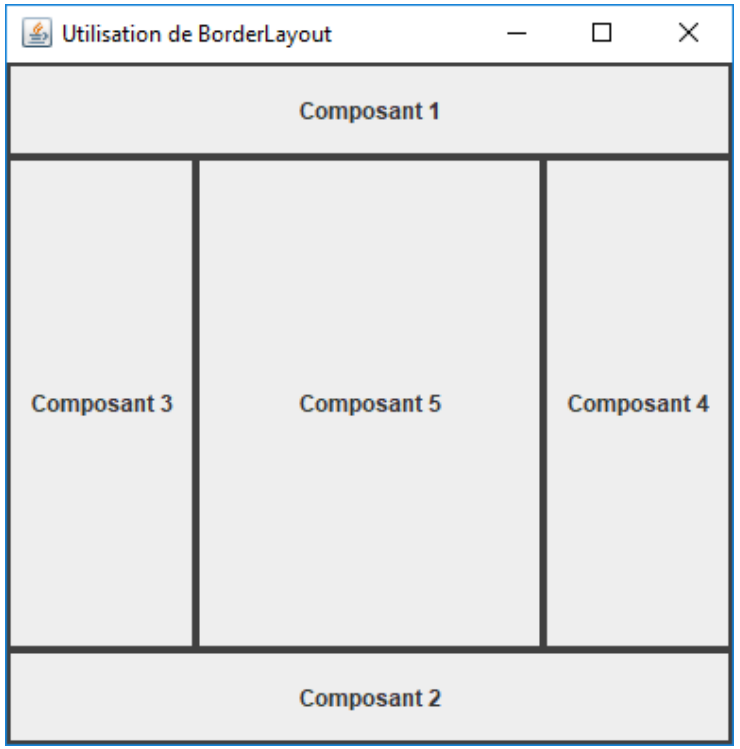
Dans le monde entier



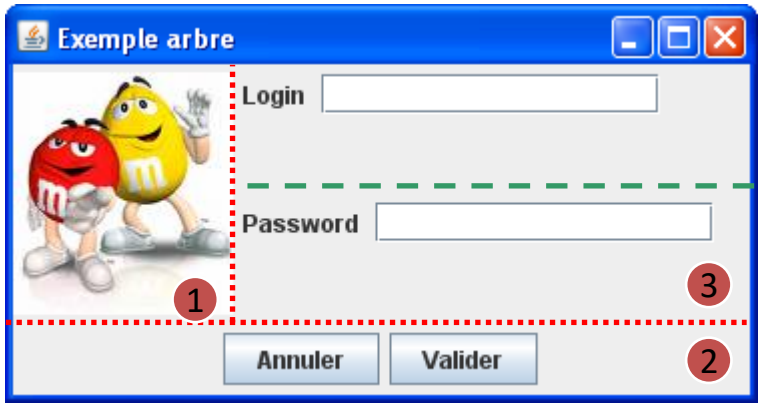
Vraiment au cas où !

BorderLayout - Exemple

```
BorderLayout layout = new BorderLayout();
JPanel panneau = new JPanel(layout);
panneau.add(genereLabel(),
                    BorderLayout. PAGE_START);
panneau.add(genereLabel(),
                    BorderLayout. PAGE_END);
panneau.add(genereLabel(),
                    BorderLayout.LINE_START);
panneau.add(genereLabel(),
                    BorderLayout.LINE_END);
panneau.add(genereLabel(),
                    BorderLayout.CENTER);
```



Exemple d'arbre de composants

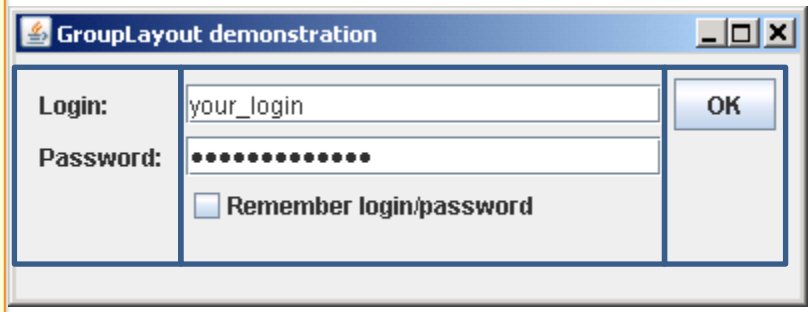


GroupLayout - Principe

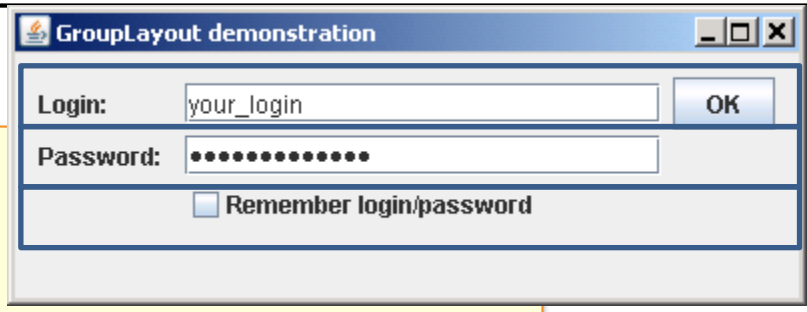
- Hiérarchie de groupes
  - Séquentielle
  - Parallèle
- Positionnement selon les deux axes
  - Horizontal
  - Vertical
- Layout utilisé par les éditeurs logiciels d'interface utilisateur

GroupLayout - Horizontal

```
GroupLayout.SequentialGroup hGroup=
layout.createSequentialGroup();

hGroup.addGroup(layout.createParallelGroup()
.addComponent(label1) /*label : login*/
.addComponent(label2)); /*label : Password*/

hGroup.addGroup(layout.createParallelGroup()
.addComponent(login)
.addComponent(passwd)
.addComponent(remember));

hGroup.addGroup(layout.createParallelGroup()
.addComponent(OK));

layout.setHorizontalGroup(hGroup);
```



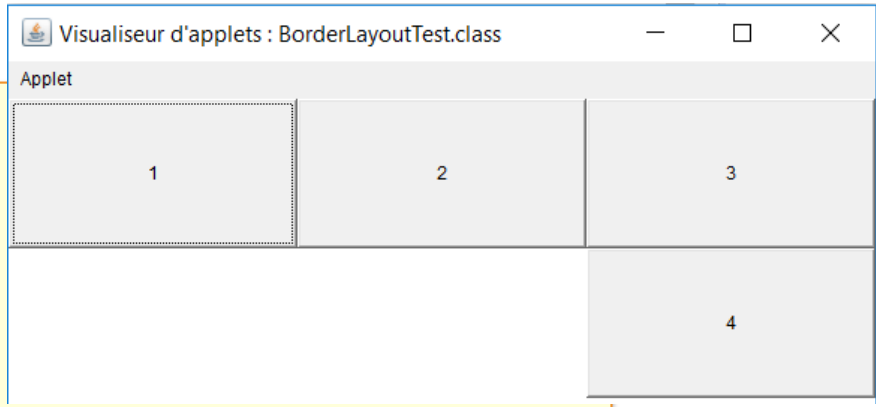
GroupLayout - Vertical




```
GroupLayout.SequentialGroup vGroup=
layout.createSequentialGroup();

vGroup.addGroup(layout.createParallelGroup(Alignment.BASELINE)
.addComponent(label1)
.addComponent(login)
.addComponent(OK));

vGroup.addGroup(layout.createParallelGroup(Alignment.BASELINE)
.addComponent(label2)
.addComponent(passwd));

vGroup.addGroup(layout.createParallelGroup(Alignment.BASELINE)
.addComponent(remember));

layout.setVerticalGroup(vGroup);
```

GroupLayout – Horizontal et vertical

```
Button c1 = new Button("1");
Button c2 = new Button("2");
Button c3 = new Button("3");
Button c4 = new Button("4");
GroupLayout layout = new GroupLayout(this);
setLayout(layout);
layout.setHorizontalGroup(
                        layout.createSequentialGroup()
                        .addComponent(c1)
                        .addComponent(c2)
                        .addGroup(layout.createParallelGroup(GroupLayout.Alignment.LEADING)
                                .addComponent(c3)
                                .addComponent(c4))
                        );
layout.setVerticalGroup(
                        layout.createSequentialGroup()
                        .addGroup(layout.createParallelGroup(GroupLayout.Alignment.BASELINE)
                                .addComponent(c1)
                                .addComponent(c2)
                                .addComponent(c3))
                        .addComponent(c4)
                        );
```

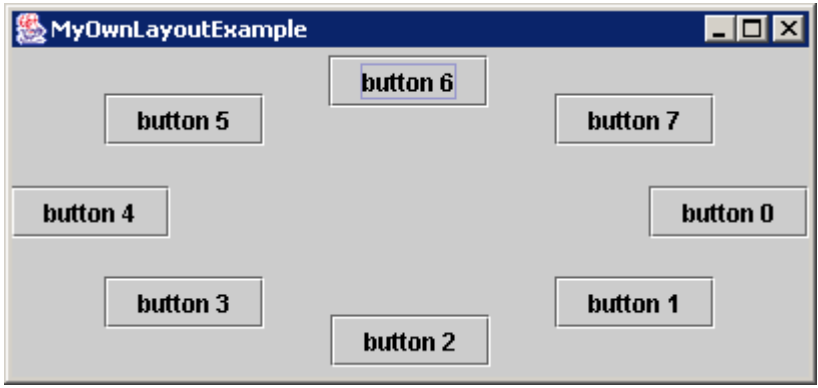


Créer son propre LayoutManager

- Implementer l'interface LayoutManager
  - Définir la taille du container
    - public Dimension preferredLayoutSize(Container parent);
    - public Dimension minimumLayoutSize(Container parent);
  - Ajouter ou supprimer un composant
    - void addLayoutComponent(String name, Component c);
    - void removeLayoutComponent(Component c);
  - placement des fils dans le container (avec setBounds() par ex.)
    - void layoutContainer(Container parent);

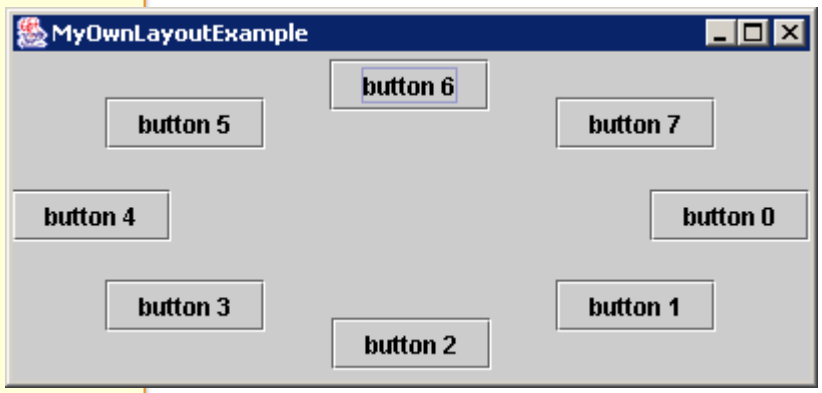
Exemple – Utilisation d'un layout personnalisé

```
content.setLayout(new ExoLayoutManager());
content.add(new JButton("Button 0"));
content.add(new JButton("Button 1"));
content.add(new JButton("Button 2"));
content.add(new JButton("Button 3"));
content.add(new JButton("Button 4"));
content.add(new JButton("Button 5"));
content.add(new JButton("Button 6"));
content.add(new JButton("Button 7"));
```



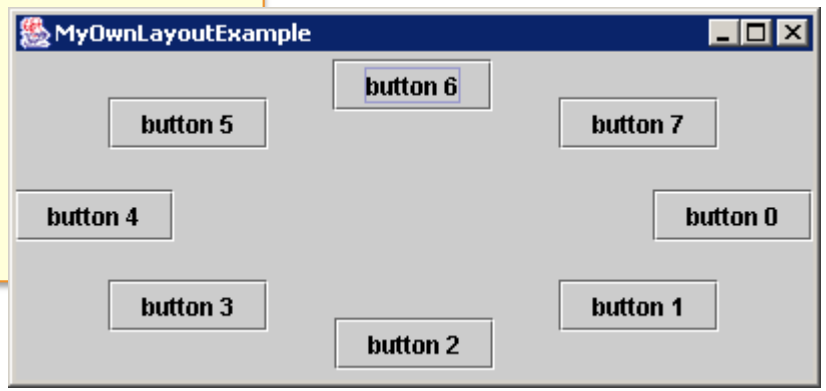
Exemple

```
public void layoutContainer(Container parent)
{
  int width=parent.getWidth()/2;
  int height=parent.getHeight()/2;
  int count=parent.getComponentCount();
  for(int i=0;i<count;i++)
  {
    double angle=2*Math.PI*i/count;
    int x=(int)(width+0.8*width*Math.cos(angle));
    int y=(int)(height+0.8*height*Math.sin(angle));
    Component c=parent.getComponent(i);
    Dimension preferred=c.getPreferredSize();
    c.setBounds(x-preferred.width/2,
                y-preferred.height/2,
                preferred.width,
                preferred.height);
  }
}
```



Exemple – Utilisation d'un layout personnalisé

```
public Dimension preferredLayoutSize(Container parent)
{
  int width=0,height=0;
  int count=parent.getComponentCount();
  for(int i=0;i<count;i++)
  {
    Component c=parent.getComponent(i);
    Dimension preferred=c.getPreferredSize();
    width+=preferred.getWidth();
    height+=preferred.getHeight();
  }
  return new Dimension(width,height);
}
```



Création d'un menu

La barre de menu - **JMenuBar**

- Classe pour la création de la barre de menu : **JMenuBar**

  JMenuBar()

- Espace spécifique pour la barre de menu dans la fenêtre
  - **Ne doit pas être placé sur le contentPane**

  JMenuBar **getJMenuBar**()

  void **setJMenuBar**(JMenuBar)

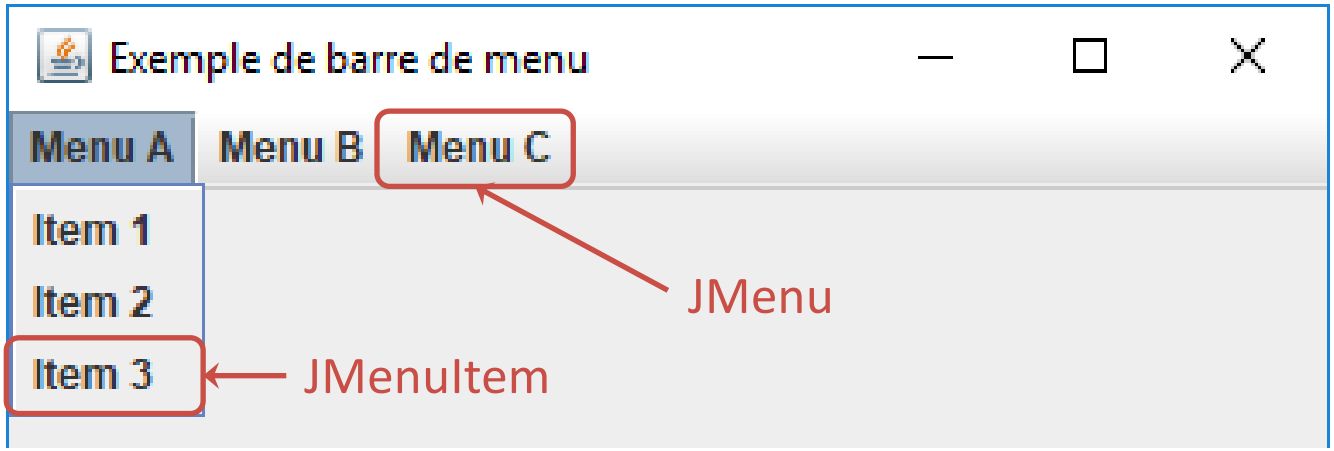
Menu et item d'un menu

- JMenu permet de gérer la liste des items

JMenu(String nom)

- JMenuItem pour créer un item d'un menu

JMenuItem(Icon icon)
JMenuItem(String text)
JMenuItem(String text, Icon icon)



Créer un menu ou un sous menu

- Une barre de menu contient un ensemble de menu
  - Ajout d'un menu à la barre de menu

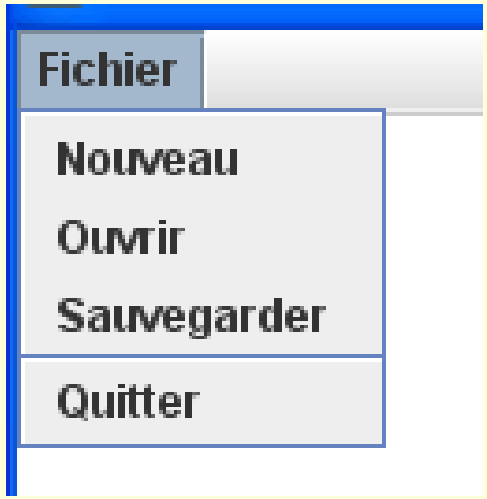
```
JMenu add(JMenu menu)
```

- Un menu peut contenir
  - Des items
    - JMenuItem,
    - JRadioButtonMenuItem,
    - JCheckboxMenuItem
  - Des sous menus (JMenu)
  - Des séparateurs (Jseparator)

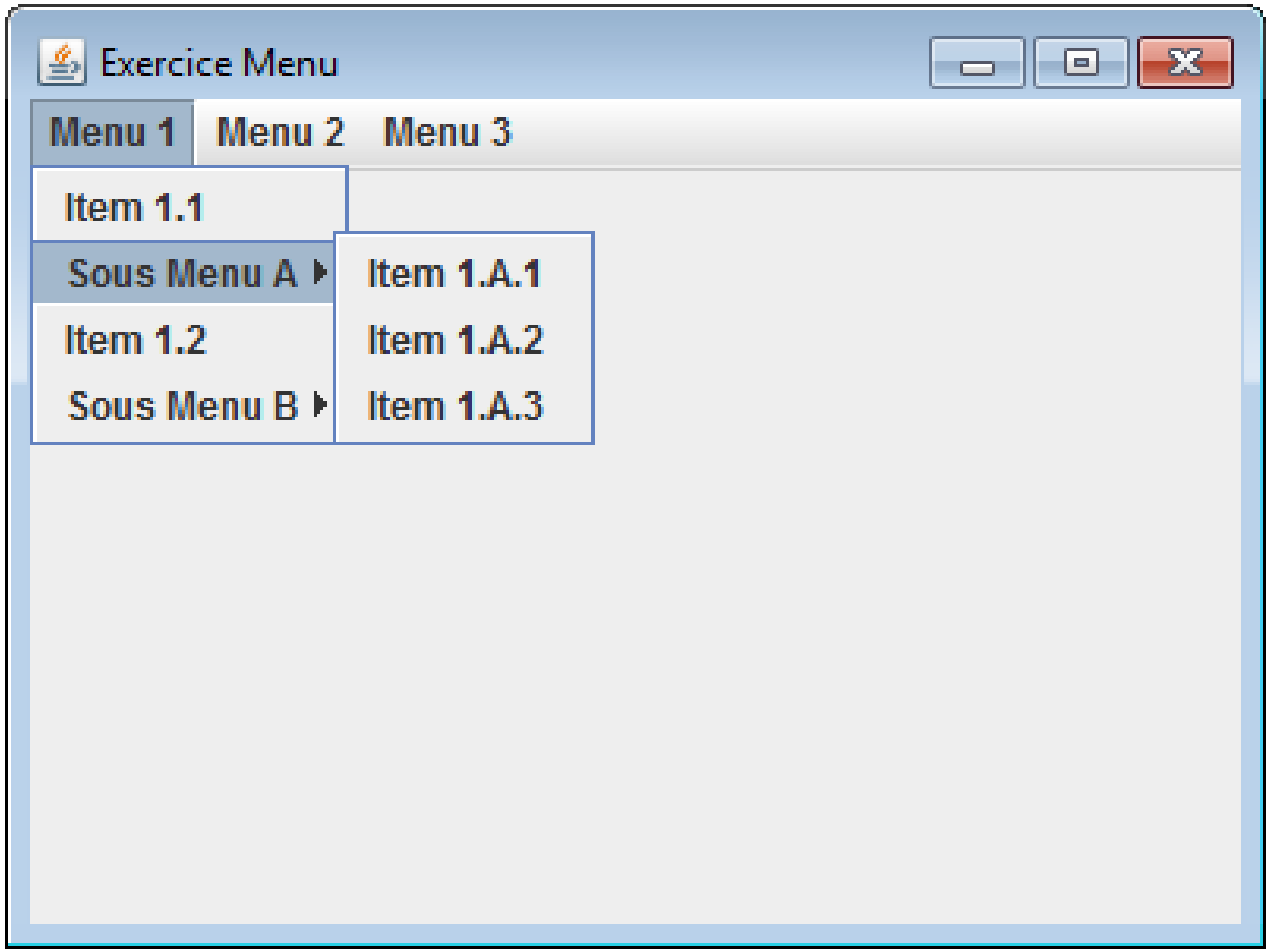
```
JMenuItem add(JMenuItem item)
```

Exemple

```
JMenuBar barreMenu = new JMenuBar();
JMenu menuFichier = new JMenu("Fichier");

JMenuItem itemNouveau = new JMenuItem("Nouveau");
JMenuItem itemOuvrir = new JMenuItem("Ouvrir");
JMenuItem itemSauvegarder = new JMenuItem("Sauvegarder");
JMenuItem itemQuitter = new JMenuItem("Quitter");

menuFichier.add(itemNouveau);
menuFichier.add(itemOuvrir);
menuFichier.add(itemSauvegarder);
menuFichier.add(new JSeparator());
menuFichier.add(itemQuitter);

barreMenu.add(menuFichier);

setJMenuBar(barreMenu);
```



Exercice – Créer la barre de menu



Menu contextuel

- Déclaration du menu contextuel

```
JPopupMenu jPopupMenu = new JPopupMenu();
```

- Ajout d'éléments au menu contextuel

```
JMenuItem jMenuItemCut = new JMenuItem("Copier");
JMenuItem jMenuItemPaste = new JMenuItem("Coller");
jPopupMenu.add(jMenuItemCut);
jPopupMenu.add(jMenuItemPaste);
```

Liens utiles

- Ce cours !
- https://imss-www.upmf-grenoble.fr/prevert/Prog/Java/swing/tableDesMatieres.html
- https://docs.oracle.com/javase/tutorial/uiswing/
- https://docs.oracle.com/javase/7/docs/api/overview-summary.html