

# Programme

- Introduction à Swing
- Composants et Conteneurs
- Gestion de la Mise en Page
- Gestion des Événements et Ecouteurs
- Modèles
- Dialogues
- Menus
- Outils Graphiques
- Autour des `JTables`
- *Sérialisation*

# Problématique de la Mise en Page

*Comment spécifier l'emplacement d'un composant dans la fenêtre, sa taille ainsi que la manière dont il est géré lors du déplacement/redimensionnement de la fenêtre ?*

- Positionnement Absolu
  - Placement des composants en utilisant un système de coordonnées (x,y)
  - Problème : pas pratique
    - Si ajout de composants, changement des coordonnées des autres composants
    - Difficile de faire du contenu redimensionnable avec cette méthode à moins de recalculer les coordonnées de tous les composants à chaque redimensionnement
- Gestionnaire de Placement
  - Place les composants dans la fenêtre en fonction des paramètres donnés et des composants eux-mêmes
  - Plus souple et plus pratique que le positionnement absolu

# Gestionnaires de Disposition

## Aspects Généraux

- Gestion des composants ajoutés
- Détermine taille et position
- Chaque conteneur a un gestionnaire de disposition (très souvent)
- Problématiques abordées :
  - Types des gestionnaires de disposition
  - Création des gestionnaires de disposition
  - Description des gestionnaires de disposition
  - Critères guidant le choix du gestionnaire de disposition
- **A lire**  
<http://java.sun.com/docs/books/tutorial/uiswing/layout/using.html>

# Typologie des Gestionnaires de Disposition

- BorderLayout
- FlowLayout
- GridLayout
- BoxLayout
- Extra :
  - CardLayout
  - GridBagLayout

# Création d'un Gestionnaire de Disposition

- Gestionnaires de disposition par défaut
  - JFrame, JDialog, JApplet considèrent le BorderLayout
  - JPanel a le FlowLayout
    - Sauf lorsqu'il est utilisé comme *Content Pane* (BorderLayout)
- Mise en place du gestionnaire de disposition pour un conteneur
  - `JFrame frame = new JFrame();`  
`frame.setLayout(new FlowLayout());`
  - `JPanel contentPane = new JPanel();`  
`contentPane.setLayout(new BorderLayout());`

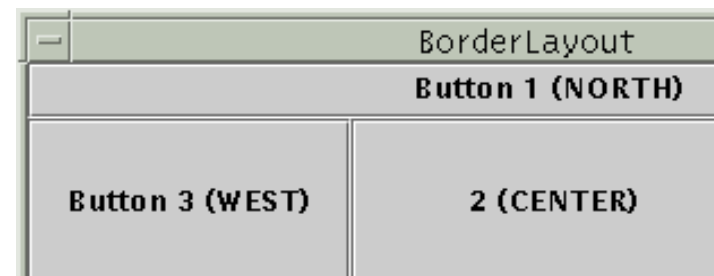
# Taille Préférée des Composants

- Les objets composants de Swing ont chacun une taille 'préférée' – i.e. une taille qui convient parfaitement à la mise en place de leurs contenus (texte, icônes, etc.)
- Certains types de gestionnaires de disposition (e.g. `FlowLayout`) choisissent d'adopter la taille préférée des composants qu'ils gèrent alors que d'autres (e.g. `BorderLayout`, `GridLayout`) ne la considèrent pas et utilisent un autre procédé

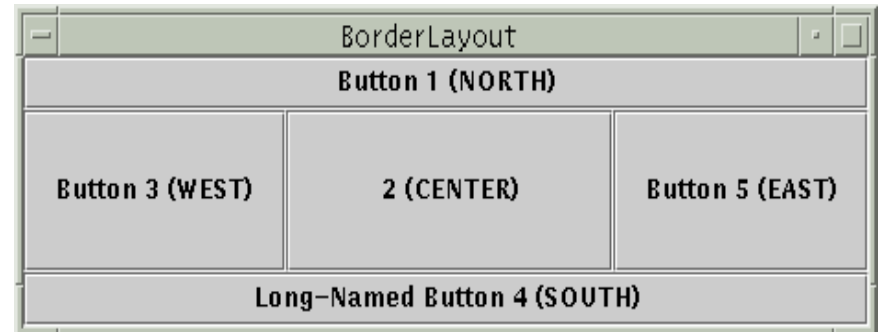
*Boutons avec taille 'préférée'*



*Pas de taille 'préférée'*



# BorderLayout



- Cinq Zones
  - NORTH, SOUTH, EAST, WEST et CENTER
  - Possibilité de ne pas caractériser certaines zones
  - Pas de zone par défaut pour les composants
  - Zone centrale occupe l'espace le plus grand possible
- Spécification de l'emplacement comme paramètre de la méthode d'ajout
  - `pane.setLayout(new BorderLayout());`
  - `pane.add(new JButton("Button 1 (NORTH)"), BorderLayout.NORTH);`
- Mise en place d'espacements entre composants (par défaut, 0)
  - `BorderLayout.setHgap(int gap);`
  - `BorderLayout.setVgap(int gap);`
  - `BorderLayout(int horizontalGap, int verticalGap)`

# FlowLayout

```
public FlowLayout ()
```

- Gère le conteneur dans le sens gauche-droite, haut-bas
- Attribue aux composants leur taille “préférée”
- Composants positionnés selon leur ordre d’ajout
- Si dépassement potentiel d’une ligne, le composant est disposé en début de ligne suivante

```
Container panel = new JPanel(new FlowLayout());  
panel.add(new JButton("Button 1"));
```

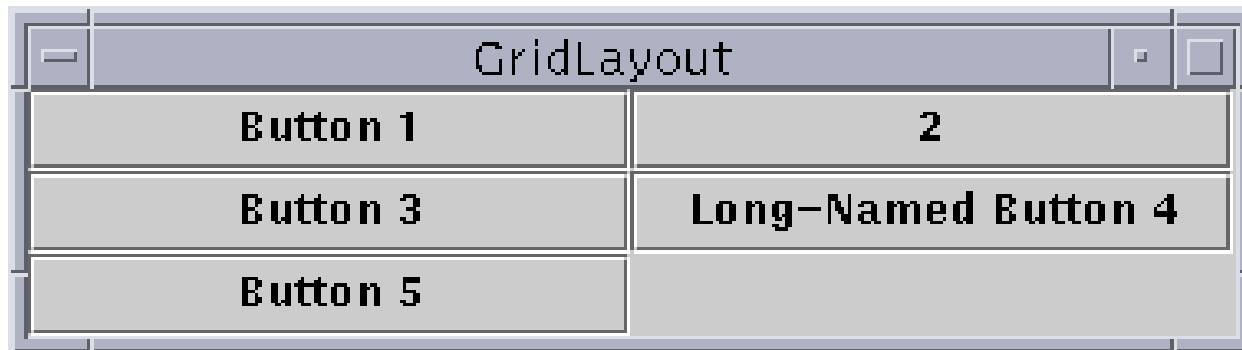




# GridLayout

```
public GridLayout(int rows, int columns)
```

- Gère le conteneur comme une grille de lignes et de colonnes de même taille
- Attribue aux composants la même taille horizontale / verticale, ignorant la taille “préférée”

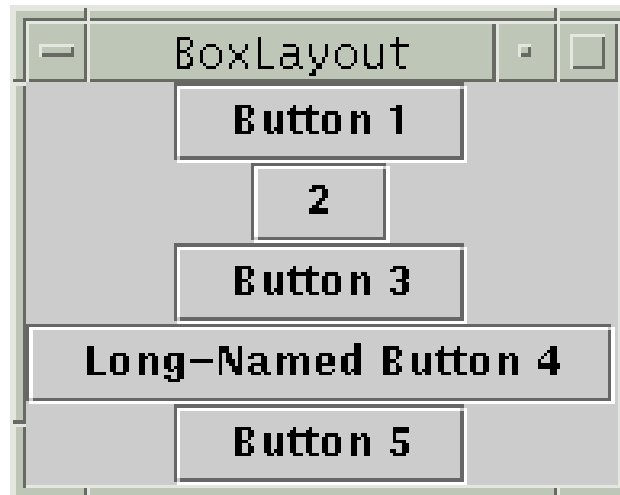


# BoxLayout

- Aligne composants sur une seule ligne ou colonne
- Composants utilisent leur taille “préférée” et sont alignés selon leur alignement préféré
- Construction d'un conteneur avec BoxLayout

```
BoxLayout (panel, BoxLayout.X_AXIS)
```

```
BoxLayout (panel, BoxLayout.Y_AXIS)
```



# Autres

- CardLayout  
couches de "cartes" empilées  
les unes sur les autres;  
une seule visible au temps t



- GridBagLayout  
compliqué, à ne pas utiliser



- sur mesure / pas de disposition spécifique
  - Mise en oeuvre de positions absolues en utilisant `setX/Y` et `setWidth/Height`

# Méthodes des Gestionnaires de Disposition

- Méthodes n'aboutissant pas sur une nouvelle disposition
  - `add()`, `remove()`, `removeAll()`
  - `getAlignmentX()`, `getAlignmentY()`
  - `getPreferredSize()`, `getMinimumSize()`, `getMaximumSize()`
- Méthodes à l'origine d'une nouvelle disposition
  - `JFrame.pack()` ;
    - Force la fenêtre à s'ajuster à la taille préférée et aux dispositions de ses composants
  - `JFrame.show()` & `JFrame.setVisible(true)` ;
    - Montrent le composant
  - `JComponent.revalidate()` ;
    - Méthode appelée automatiquement sur le composant lors d'une modification. Examine les composants dépendants et appelle la méthode `validate()` pour ces derniers. `Validate()` force un conteneur à reconsidérer la disposition de ses composants.

# Choix du Gestionnaire de Disposition (1)

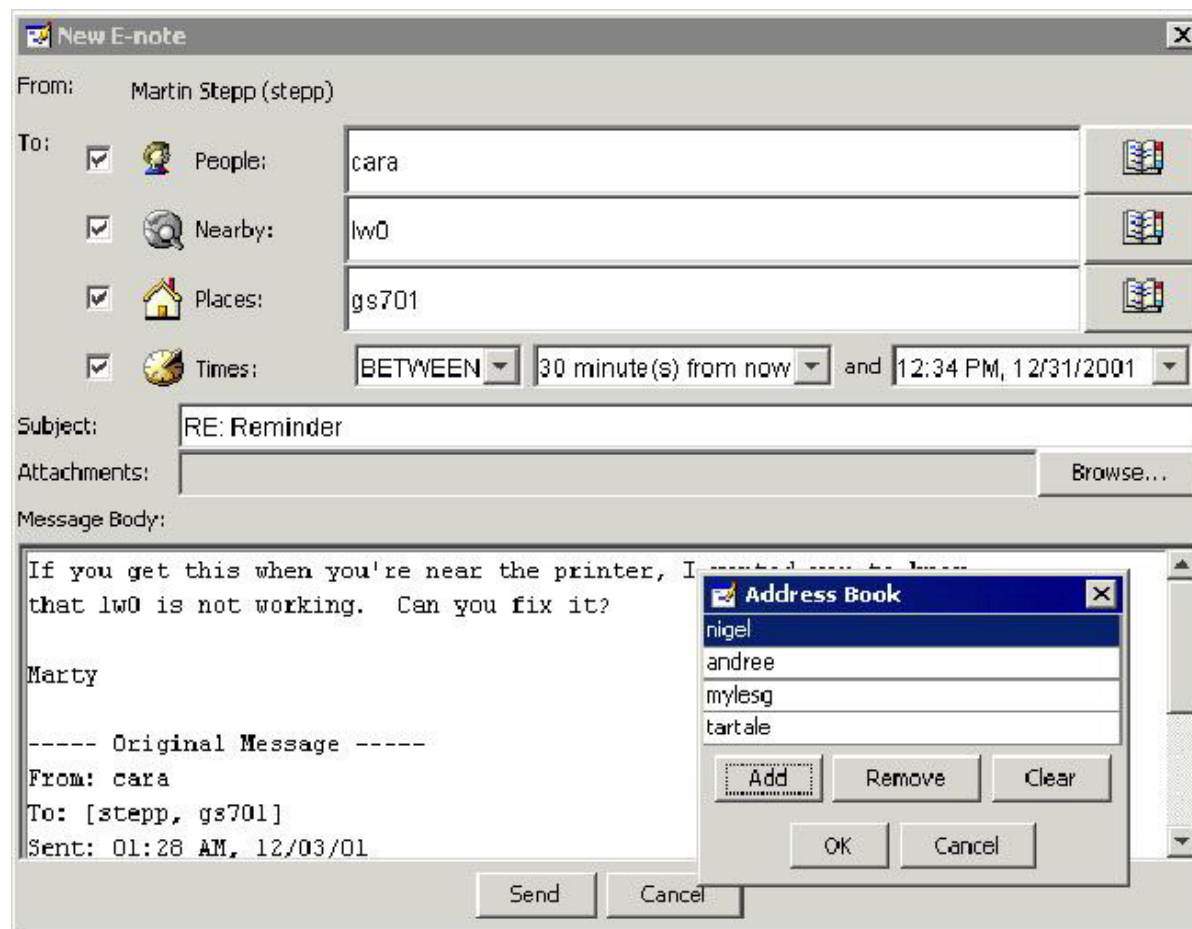
- Affichage d'un composant dans l'espace maximum pouvant lui être alloué
  - BorderLayout
    - Composant dans la zone CENTER
  - BoxLayout
    - Composant spécifie des tailles préférées/maximum très larges
- Affichage de composants sur une ligne compacte
  - FlowLayout
  - BoxLayout
- Affichage de composants de même taille dans des lignes et colonnes
  - GridLayout

## Choix du Gestionnaire de Disposition (2)

- Affichage de composants sur une ligne ou colonne, avec différents espacements entre eux et tailles sur mesure
  - BoxLayout
- Affichage d'une disposition complexe avec plusieurs composants...

# Problème avec Gestionnaires de Disposition

Comment créer une fenêtre complexe en utilisant les gestionnaires de disposition ?



# Solution : Composition de Questionnaires

- Création de *panels* à l'intérieur de *panels*
- Chacun a une disposition propre et en combinant les dispositions, obtention d'une disposition plus complexe
- Exemple :
  - Combien de *panels* ?
  - Quelle disposition pour chacun ?
  - A réaliser

