**Cours des Universités**

**Conception et programmation graphique en Java**

**L'API Swing**

**Les gestionnaires de placement**

**Table des matières**

Les composants Swing et leur placement 3

Fonctionnement d'un gestionnaire de placement 4

Le gestionnaire BorderLayout 5

Le gestionnaire BorderLayout sans précision de région 6

Le gestionnaire BorderLayout : région EAST 7

Le gestionnaire BorderLayout : région NORTH 8

Le gestionnaire BorderLayout : région CENTER 9

Le gestionnaire FlowLayout 10

Exemple 1 de gestionnaire FlowLayout 11

Exemple 2 de gestionnaire FlowLayout 12

Le gestionnaire BoxLayout 13

Un exemple de gestionnaire BoxLayout 14

Le gestionnaire CardLayout 15

Un exemple de gestionnaire CardLayout 16

Le gestionnaire GridLayout 17

Un exemple de gestionnaire GridLayout 18

Le gestionnaire GridBagLayout 19

Un exemple de gestionnaire GridBagLayout 20

Des panneaux les uns dans les autres (1/2) 21

Des panneaux les uns dans les autres (2/2) 22

Résumé (1/3) 23

Résumé (2/3) 24

Résumé (3/3) 25

# Les composants Swing et leur placement

Un widget est un composant Swing dérivant de la classe JComponent

ce sont des éléments que l'utilisateur voit et avec lesquels il interagit

champs de texte, bouton radio, …

Tous les composants sont capables de contenir d'autres composants mis à part les composants atomiques

on place des composants interactifs sur des composants d'arrière-plan (cadre, panneau)

un JPanel sur un autre JPanel

Les gestionnaires d'agencement contrôlent la taille et l'emplacement des widgets via la méthode getPreferredSize() des composants (la taille préférée est la taille qui s'adapte à du texte)

il est possible de se passer des gestionnaires et de coder en dur le placement et la taille des composants

Ces gestionnaires sont associés dans un composant d'arrière-plan et contrôlent les composants inclus et uniquement ceux inclus dans le composant d'arrière-plan

pas de contrôle sur des composants hiérarchiquement plus bas

si un panneau A contient un panneau B alors le gestionnaire de A contrôle B mais pas les composants inclus dans B

# Fonctionnement d'un gestionnaire de placement

Les gestionnaires d'agencement appliquent des politiques de placement

aligner des composants sur une grille, à l'horizontale, uniformiser leur taille, …

Certains layouts respectent les souhaits des composants, d'autres respectent qu'une partie des souhaits (taille verticale respectée mais taille horizontale égale à celle du panneau), ...

Le scénario pour utiliser un layout est

création d'un panneau et ajout d'un layout

ajout de composants dans le panneau

le layout du panneau demande aux composants la taille qu'ils souhaitent avoir

le layout du panneau applique ses politiques et décide s'il doit respecter les souhaits des composants

Il y a 7 gestionnaires d'agencement (dans le paquetage awt)

BorderLayout, FlowLayout, BoxLayout, CardLayout, GridLayout, GridBagLayout, GroupLayout

# Le gestionnaire BorderLayout

Un BorderLayout divise un composant d'arrière-plan en cinq régions

centre, nord, sud, est, ouest (si aucune région n'est précisé alors tout le centre est utilisé)

On ne peut ajouter qu'un seul composant par région

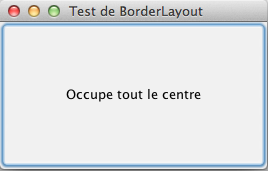
Les composants gérés par ce layout n'obtiennent pas la taille désirée

en est et ouest : largeur désirée ; en sud et nord : hauteur désirée

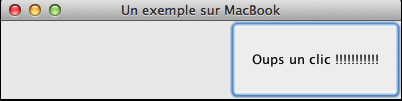
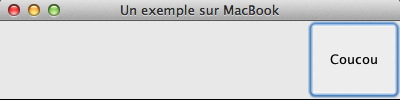
au centre : ce qui reste

Ce layout est le gestionnaire par défaut pour un JFrame

# Le gestionnaire BorderLayout sans précision de région

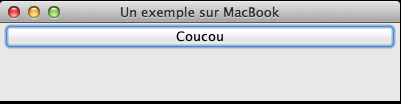
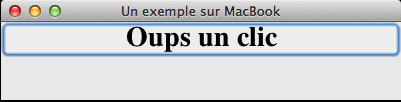


# Le gestionnaire BorderLayout : région EAST

Le bouton a la hauteur du panneau !!!

Le bouton a la largeur demandée (taille de « Coucou »)

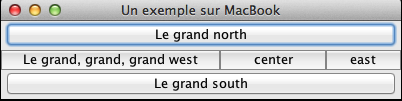
# Le gestionnaire BorderLayout : région NORTH

Le bouton a la hauteur demandée

Le bouton a la largeur du panneau !!!

Le bouton a la hauteur demandée (taille de « Coucou »)

# Le gestionnaire BorderLayout : région CENTER

Le centre a ce qui reste

sauf si on utilise la méthode pack()

Cf infra

# Le gestionnaire FlowLayout

Les composants sont ajoutés de gauche à droite et passe à la ligne si nécessaire

il est possible d'indiquer l'alignement des composants

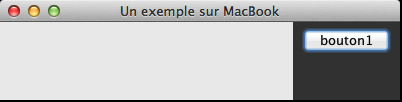
gauche, droit, centre

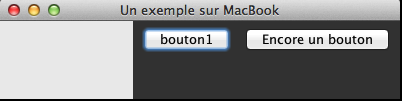
new FlowLayout(FlowLayout.CENTER)

La taille souhaitée des composants est respectée

Ce layout est le gestionnaire par défaut des JPanel

# Exemple 1 de gestionnaire FlowLayout



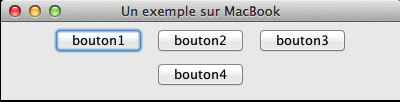


Ce panneau est ajouté en EAST dans le panneau de

la JFrame

Par défaut le JPanel à un FlowLayout

# Exemple 2 de gestionnaire FlowLayout



Le FlowLayout écrase le BorderLayout

# Le gestionnaire BoxLayout

Les composants sont ajoutés verticalement ou horizontalement

La création d'un Box horizontal ou vertical se réalise de 2 façons

par une méthode statique

Box ligne = Box.createHorizontalBox() ;

Box colonne = Box.createVerticalBox() ;

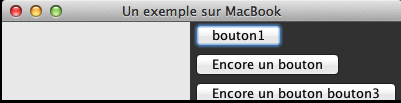
par spécification d'un axe

new BoxLayout(BoxLayout.Y\_AXIS)

new BoxLayout(BoxLayout.X\_AXIS)

La taille souhaitée des composants est respectée

# Un exemple de gestionnaire BoxLayout



# Le gestionnaire CardLayout

Les composants sont ajoutés suivant une pile de telle sorte que seul le composant en haut de pile soit visible

Lors de l'ajout d'un composant, on doit fournir une chaîne identifiant le composant

CardLayout pile = new CardLayout() ;

pile.add(new Buton(), "Bouton num 1") ;

Le composant visible est le premier ajouté au conteneur

Il est possible de faire apparaître un autre composant

pile.next(conteneur) ;

pile.previous(conteneur) ;

pile.first(conteneur) ;

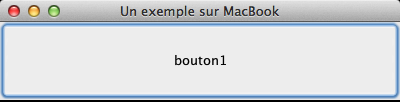
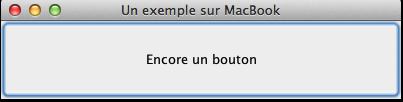
pile.last(conteneur) ;

pile.show(conteneur, "Bouton num 1") ;

La taille souhaitée des composants n'est pas respectée

les composants occupent tout l'espace

# Un exemple de gestionnaire CardLayout



# Le gestionnaire GridLayout

Les composants sont ajoutés suivant une grille de telle sorte que chaque composant occupe une cellule

La grille est gérée comme une matrice

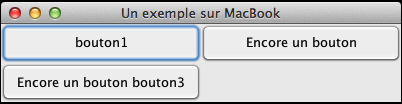
les composants sont ajoutés dans l'ordre de création ligne par ligne

La grille est définie lors de la création

new GridLayout(5, 4)

La taille souhaitée des composants n'est pas respectée

# Un exemple de gestionnaire GridLayout



# Le gestionnaire GridBagLayout

Les composants sont ajoutés suivant une grille de telle sorte que chaque composant occupe une ou plusieurs cellules

La grille est gérée comme une matrice virtuelle

Les composants sont ajoutés en fournissant leur position par l'intermédiaire d'un objet c de type GridBagConstraints

add(new Buton(), c) ;

Les paramètres de position sont des champs d'un GridBagConstraints

gridx et gridy : abscisse et ordonnée du coin supérieur gauche du composant

gridwidth et gridheight : largeur et hauteur du composant

weightx et weighty : poids horizontal et vertical du composant

fill : comment le composant occupe l'espace disponible

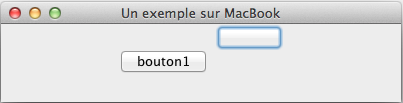
GridBagConstraints.HORIZONTAL : largeur ajustée à l'espace disponible

GridBagConstraints.VERTICAL : hauteur à l'espace disponible

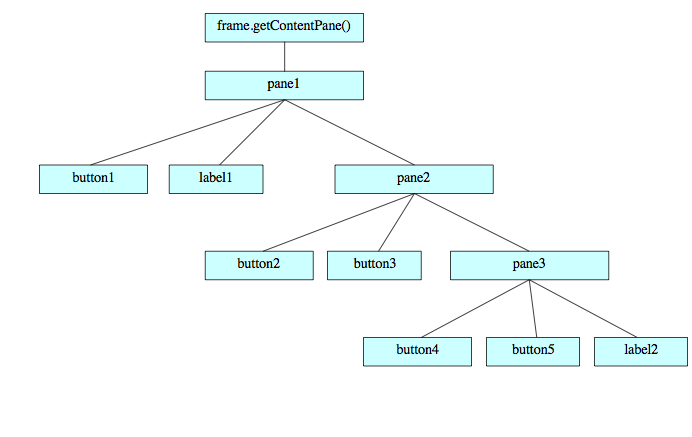
GridBagConstraints.BOTH : largeur et hauteur à l'espace disponible

GridBagConstraintsNONE : aucun ajustement

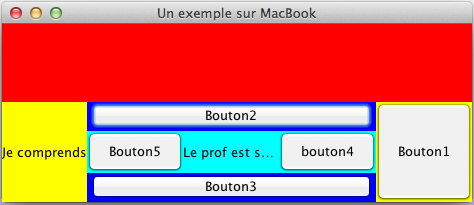
# Un exemple de gestionnaire GridBagLayout



# Des panneaux les uns dans les autres (1/2)



# Des panneaux les uns dans les autres (2/2)



# Résumé (1/3)

Les gestionnaires d'agencement contrôlent la taille et le placement des composants

Il est possible de changer le gestionnaire d'agencement attaché au panneau du cadre en créant un nouveau panneau via setLayout()

Il est possible de personnaliser la taille des composants avec la méthode setSize() mais les gestionnaires d'agencement peuvent ignorer cette taille

La taille des composants est gérée par leur taille préférée mais les gestionnaires d'agencement peuvent l'ignorer

Les gestionnaires d'agencement utilisent la taille préférée des composants via la méthode getPreferedSize()

les gestionnaires d'agencement contrôlent la taille des composants

Il est possible de placer les composants où l'utilisateur le souhaite en désactivant le gestionnaire d'agencement via la méthode setLayout(null)

C'est le gestionnaire d'agencement du composant d'arrière-plan qui contrôle les composants ajoutés à ce dernier

Un JPanel peut contenir un autre JPanel qui contient un autre JPanel ...

# Résumé (2/3)

BorderLayout permet d'ajouter un composant dans cinq régions

add(BorderLayout.EAST, panneau)

les composants du nord et du sud ont leur hauteur préférée mais pas leur largeur

les composants d'est et d'ouest ont leur largeur préférée mais pas leur hauteur

le composant du centre obtient ce qu'il reste (sauf si on utilise pack())

FlowLayout permet d'ajouter des composants de gauche à droite et de haut en bas ; il donne la taille préférée des composants

BoxLayout permet d'ajouter des composants verticalement ; il donne la taille préférée des composants

FlowLayout est le gestionnaire par défaut des panneaux

Les méthodes show() et hide() sont deprecated ; il faut utiliser setVisible()

La méthode revalidate() de JComponent permet de reconstruire la taille et la position des composants au sein d'un layout en cas de modification (suppression de composants, ajouts de composants, modification de la taille de composants, ...)

validate() est liée à AWT

La méthode pack() de la classe JFrame utilise la taille préférée des composants afin de calculer la taille du JFrame pour que tous les composants soient visibles

# Résumé (3/3)

La méthode getParent() retourne le container d'un composant