

4. Motif : les widgets conteneurs

Widgets conteneur

Motif: les widgets conteneur

- o Conteneurs
- o BulletinBoard
- o Form
- o RowColumn
- o PanedWindow

1

Jean Berstel - Institut Gaspard Monge, Université Marne-la-Vallée

Widgets conteneur

Classes de widgets conteneur

- o Constraint et Shell sont des classes Xt Intrinsics

```
classDiagram
    Composite --|> Shell
    Composite --|> Constraint
    Constraint --|> Manager
    Manager --|> BulletinBoard
    Manager --|> RowColumn
    Manager --|> PanedWindow
    Manager --|> DrawingArea
    Manager --|> Frame
    Manager --|> Scale
    Manager --|> ScrolledWindow
    DrawingArea --|> Form
    Form --|> Form
    Form --|> SelectionBox
    Form --|> Command
    Form --|> FileSelectionBox
    SelectionBox --|> MessageBox
    ScrolledWindow --|> MainWindow
```

2

Jean Berstel - Institut Gaspard Monge, Université Marne-la-Vallée

4. Motif : les widgets conteneurs

Widgets conteneur

Conteneurs

- o Usage
 - | agencer ses widgets filles selon ses règles de placement;
 - | réorganiser le placement si la "géométrie" change.
- o Principe
 - | une widget ne modifie jamais sa position et sa taille de son propre chef;
 - | la gestion de la géométrie est de la *responsabilité* de sa *mère*;
 - | chaque classe a ses propres *règles* de gestion de ses filles.
- o Exemples :
 - | `BulletinBoard` ne modifie pas la géométrie de ses filles.
 - | `RowColumn`, les dispose en lignes et/ou en colonnes
 - | `ScrolledWindow` a toujours au plus 3 filles, dont deux ascenseurs à emplacements fixes.
- o Widget à contraintes ("constraint widgets")
 - | ont des ressources de placement spécifiques: *ressources de contraintes*.
 - | Ces ressources des ressources des *filles*; *gérées* par les filles
 - | *maïs exploitées* par leur *mère* (une widget à contraintes).
 - | C'est un mécanisme Intrinsic.

Jean Berstel - Institut Gaspard Monge, Université Marne-la-Vallée

3

Widgets conteneur

Manager

- o La superclasse Motif des conteneurs.
- o Détermine le "look and feel" des conteneurs.
- o Procure des ressources
 - | pour la gestion d'ombres
 - | pour les couleurs d'activation
 - | pour le parcours avec touches

<code>XmNbottomShadowColor</code>	Pixel	Couleurs pour les ombres du bord
<code>XmNtopShadowColor</code>		
<code>XmNshadowThickness</code>	Dimension	Épaisseur de l'ombre
<code>XmNforeground</code>	Pixel	Couleur d'encre de la widget
<code>XmNhighlightColor</code>	Pixel	Couleur pour tracer le rectangle d'activation
<code>XmNinitialFocus</code>	Widget	La widget qui est foyer du clavier
<code>XmNnavigationType</code>	enum	Détermine le mode de parcours avec les touches
<code>XmNtraversalOn</code>	Boolean	Si <code>True</code> (défaut) permet parcours par touches
<code>XmNuserData</code>	XtPointer	permet d'accrocher des données

Jean Berstel - Institut Gaspard Monge, Université Marne-la-Vallée

4

4. Motif : les widgets conteneurs

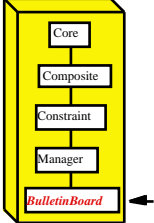
Widgets conteneur

BulletinBoard

```
#include <Xm/BulletinB.h>
```

```
XtVaCreateManagedWidget(nom, xmBulletinBoardWidgetClass, mère, ..., NULL)
XmCreateCascadeButton(mère, nom, args, nargs)
```

Conteneur le plus simple, sans contraintes de position ni de dimension



- La valeur par défaut de la position d'une fille est (0,0). Par conséquent, deux filles sans position explicite se superposent !
- Souvent utilisé comme "formulaire", notamment dans les dialogues. A ce titre, dispose de quelques ressources particulières.

Jean Berstel - Institut Gaspard Monge, Université Marne-la-Vallée

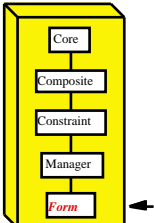
5

Widgets conteneur

Form

```
#include <Xm/Form.h>
```

```
XtVaCreateManagedWidget(nom, xmFormWidgetClass, mère, ..., NULL)
XmCreateForm(mère, nom, args, nargs)
```



- Une classe de **widgets à contraintes**. Chaque fille peut choisir un type d'**ancrage** : elles peuvent être attachées
 - | à leur mère,
 - | à une sœur,
 - | à une "position" relativement à une grille.
- **Chacun des 4 côtés d'une fille peut être ancré**. Les ancrages sont conservés même si la mère est retaillée. Les côtés sont nommés par
 - XmTopAttachment
 - XmRightAttachment
 - XmBottomAttachment
 - XmLeftAttachment
- L'ancrage relativement à la mère ou à une sœur signifie l'**alignement** de ce côté sur le même côté ou le côté opposé de la widget concernée.
- La relation d'ancrage n'est pas symétrique.

Jean Berstel - Institut Gaspard Monge, Université Marne-la-Vallée

6

4. Motif : les widgets conteneurs

Widgets conteneur

Ressources d'ancrage

- o Chaque côté d'une fille peut être ancrée.
- o L'ancrage est relatif
 - | à la **mère**
XmATTACH_FORM ou XmATTACH_OPPOSITE_FORM
 - | à une **sœur**
XmATTACH_WIDGET ou XmATTACH_OPPOSITE_WIDGET
 - | à une **position**
XmATTACH_POSITION
 - | à une **valeur initiale**
XmATTACH_SELF
- o L'absence de spécification vaut
XmATTACH_NONE
- o L'ancrage se fait modulo un **décalage** ("offset") qui peut être spécifié pour chaque côté de chaque fille, ou globalement pour toutes les filles par
XmhorizontalSpacing et XmverticalSpacing

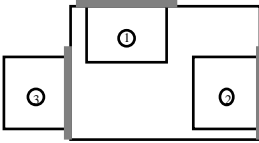
7

Jean Berstel - Institut Gaspard Monge, Université Marne-la-Vallée

Widgets conteneur

Ancrage à la mère

ATTACH_FORM, ATTACH_OPPOSITE_FORM :
alignement du côté sur le même côté ou le côté opposé de la mère.



- ① XmNtopAttachment XmATTACH_FORM
- ② XmNrightAttachment XmATTACH_FORM
- ③ XmNleftAttachment XmATTACH_OPPOSITE_FORM

8

Jean Berstel - Institut Gaspard Monge, Université Marne-la-Vallée

4. Motif : les widgets conteneurs

Widgets conteneur

Exemple

- Les ressources sont spécifiées pour la widget *filie*.

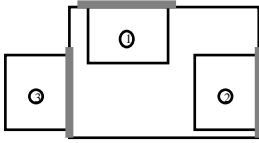
```
forme = XtVaCreateManagedWidget("forme",  
    xmFormWidgetClass, racine, NULL);
```

- En dur

```
un = XtVaCreateManagedWidget("1", xmPushButtonClass, forme,  
    XmNtopAttachment, XmATTACH_FORM, NULL);  
deux = XtVaCreateManagedWidget("2", xmPushButtonClass, forme,  
    XmNrightAttachment, XmATTACH_FORM, NULL);  
trois = XtVaCreateManagedWidget("3", xmPushButtonClass, forme,  
    XmNrightAttachment, XmOPPOSITE_ATTACH_FORM, NULL);
```

- Dans le fichier ressources

```
*1.topAttachment : XmATTACH_FORM  
*2.rightAttachment : XmATTACH_FORM  
*3.rightAttachment : XmOPPOSITE_ATTACH_FORM
```



9

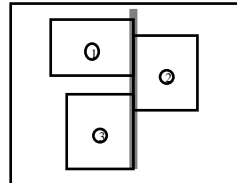
Jean Berstel - Institut Gaspard Monge, Université Marne-la-Vallée

Widgets conteneur

Ancrage à une sœur

- ATTACH_WIDGET, ATTACH_OPPOSITE_WIDGET
 - attachement au côté *opposé* pour ATTACH_WIDGET
 - attachement au *même* côté pour ATTACH_OPPOSITE_WIDGET
- De plus, il faut spécifier la sœur sur laquelle on s'aligne par:
 - XmNtopWidget XmNrightWidget
 - XmNleftWidget XmNbottomWidget

```
deux = XtVaCreateManagedWidget("2",  
    xmPushButtonClass, forme,  
    XmNleftAttachment, XmATTACH_WIDGET,  
    XmNleftWidget, un,  
    NULL);  
trois = XtVaCreateManagedWidget("3",  
    xmPushButtonClass, forme,  
    XmNrightAttachment, XmATTACH_OPPOSITE_WIDGET,  
    XmNrightWidget, un,  
    NULL);
```



10

Jean Berstel - Institut Gaspard Monge, Université Marne-la-Vallée

4. Motif : les widgets conteneurs

Widgets conteneur

Un exemple

```
nord_ouest = XtVaCreateManagedWidget("nord-ouest", xmPushButtonGadgetClass, forme,
    XmNtopAttachment,      XmATTACH_FORM,
    XmNleftAttachment,    XmATTACH_FORM, NULL);

nord_est = XtVaCreateManagedWidget("nord-est",
    xmPushButtonGadgetClass, forme,
    XmNtopAttachment,      XmATTACH_FORM,
    XmNrightAttachment,    XmATTACH_FORM,
    XmNleftAttachment,    XmATTACH_WIDGET,
    XmNleftWidget,        nord_ouest,
    NULL);

sud_ouest = XtVaCreateManagedWidget("sud-ouest",
    xmPushButtonGadgetClass, forme,
    XmNbottomAttachment,   XmATTACH_FORM,
    XmNleftAttachment,     XmATTACH_FORM,
    XmNtopAttachment,     XmATTACH_WIDGET,
    XmNtopWidget,         nord_ouest,
    NULL);

sud_est = XtVaCreateManagedWidget("sud-est", xmPushButtonGadgetClass, forme,
    XmNbottomAttachment,   XmATTACH_FORM,
    XmNrightAttachment,    XmATTACH_FORM,
    XmNleftAttachment,    XmATTACH_WIDGET,
    XmNleftWidget,        sud_ouest,
    NULL);
XtVaSetValues(nord_est, XmNbottomAttachment, XmATTACH_WIDGET,
    XmNbottomWidget,     sud_est,
    NULL);
```

Jean Berstel - Institut Gaspard Monge, Université Marne-la-Vallée

11

Widgets conteneur

ancrage en position

ATTACH_POSITION

- | la forme est quadrillée par une grille carrée;
- | la widget est attachée à (un côté d') une case de la grille.
- | la grille est déformée avec la mère : la position de chaque fille reste proportionnelle.

- o XmNfractionBase donne le nombre de cases par ligne ou par colonne de la grille (valeur par défaut 100).
- o Le type d'attachement pour le côté concerné, et sa coordonnée correspondante sont donnés. Par exemple :

```
XmNtopAttachment,    XmATTACH_POSITION
XmNtopPosition,      3
```

Jean Berstel - Institut Gaspard Monge, Université Marne-la-Vallée

12

4. Motif : les widgets conteneurs

Widgets conteneur

ancrage absolu

- o ATTACH_SELF
 - | la position est donnée par ses coordonnées x, y ;
 - | si la mère est retaillée, la position évolue proportionnellement.

13

Jean Berstel - Institut Gaspard Monge, Université Marne-la-Vallée

Widgets conteneur

Pièges et problèmes

- o Le graphe des attachements doit être *sans circuit*.
- o Les attachements relativement à une sœur (XmATTACH_WIDGET) peuvent être donnés dans le fichier de ressources, en spécifiant le nom (chaîne de caractères) de la widget.
- o Une widget doit avoir été créée avant qu'on peut y attacher une autre.
- o L'attachement concerne la widget *racine* dans des widgets composées:
 - | CreateScrolledList ou CreateScrolledText retourne la widget liste ou texte,
 - | c'est leur mère (ScrolledWindow) qu'il faut attacher !

```
forme = XtVaCreateManagedWidget("forme",
    xmFormWidgetClass, racine,
    XmNfractionBase, 5, NULL);
txt = XmCreateScrolledText(forme, "bla", NULL, 0);
XtVaSetValues(XtParent(txt),
    XmNleftAttachment, XmATTACH_FORM,
    XmNtopAttachment, XmATTACH_FORM,
    NULL);
```

14

Jean Berstel - Institut Gaspard Monge, Université Marne-la-Vallée

4. Motif : les widgets conteneurs

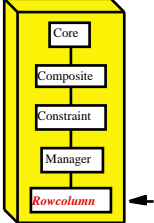
Widgets conteneur

RowColumn

```
#include <Xm/RowColumn.h>
```

```
XtVaCreateManagedWidget(nom, XmRowColumnWidgetClass, mère, ..., NULL)  
XmCreateRowColumn(mère, nom, args, nargs)
```

Une widget qui agence ses filles en lignes et/ou colonnes.



```
graph TD
  Core[Core] --> Composite[Composite]
  Composite --> Constraint[Constraint]
  Constraint --> Manager[Manager]
  Manager --> Rowcolumn[Rowcolumn]
```

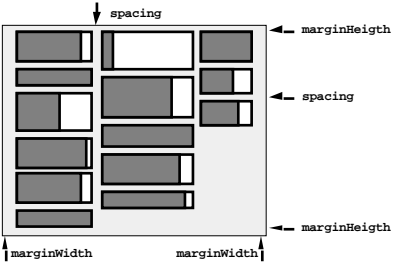
- o L'agencement peut être libre ou forcé.
- o Aussi utilisé de façon interne par Motif pour les menus et les barres de menus.
 - l les fonctions de création spécifiques fixent certaines ressources.

Jean Berstel - Institut Gaspard Monge, Université Marne-la-Vallée

15

Widgets conteneur

Deux agencements



↓ spacing

← marginHeight

← spacing

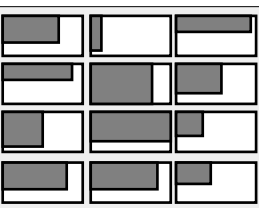
← marginHeight

↑ marginWidth

marginWidth ↓

taille naturelle

PACK_TIGHT



PACK_COLUMN

Jean Berstel - Institut Gaspard Monge, Université Marne-la-Vallée

16

4. Motif : les widgets conteneurs

Widgets conteneur

Ressources des RowColumn

- o Orientation : `XmNorientation`
 - | par défaut verticale (en colonnes);
 - | pour changer en horizontale (en lignes): `XmHORIZONTAL`
 - | tout paramètre sur les colonnes (nombre, ajustement,...) porte alors sur les lignes.
- o Assemblage ("packing") : `XmNpacking`
 - | `XmPACK_TIGHT` (défaut): les filles sont réparties "au mieux", éventuellement en plusieurs colonnes, selon la place disponible;
 - | `XmPACK_COLUMN` : le nombre de colonnes choisies est respecté; s'il n'y a pas assez de place, une partie des filles n'est pas vue.
 - | `XmPACK_NONE` : les filles sont placées aux coordonnées indiquées, (0 , 0) par défaut. Utilité obscure.
- o Nombre de colonnes : `XmNnumColumns`
 - | suppose que `XmNpacking` vaut `XmPACK_COLUMN` ;
 - | les colonnes ont même largeur sauf peut-être la dernière.
- o Pour ajuster aussi la dernière colonne : `XmNadjustLast`
 - | si `False`, la dernière colonne a la même largeur que les autres;
 - | si `True` (défaut), cette colonne est élargie jusqu'au côté.

Jean Berstel - Institut Gaspard Monge, Université Marne-la-Vallée

17

Widgets conteneur

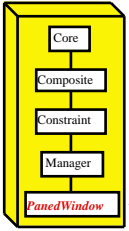
PanedWindow

```
#include <Xm/PanedW.h>
```

```
XtVaCreateManagedWidget(nom, XmPanedWindowWidgetClass, mère, ..., NULL)
```

```
XmCreatePanedWindow(mère, nom, args, nargs)
```

Agence ses filles en une colonne



```
graph TD; Core[Core] --- Composite[Composite]; Composite --- Constraint[Constraint]; Constraint --- Manager[Manager]; Manager --- PanedWindow[PanedWindow];
```

- o Entre deux filles consécutives est ajouté un séparateur muni d'une *poignée* ("sash").
 - | le séparateur et la poignée sont ajoutés automatiquement;
 - | en agissant sur la poignée, l'utilisateur peut agrandir ou rétrécir la hauteur d'une fille.
- o On peut jouer sur la taille extrême d'une fille par des ressources de contraintes (i.e. de la fille)
 - `XmNpaneMinimum`, `XmNpaneMaximum` : hauteur minimale, maximale de la fille;
 - `XmNresizeHeight` : si `False`, empêche changement de hauteur;
- o La poignée peut avoir une taille autre que par défaut (10 sur 10):
 - `XmNsashHeight`, `XmNsashWidth` contrôlent cette taille;
 - `XmNseparatorOn` : si `False`, fait disparaître les lignes séparatrices.

Jean Berstel - Institut Gaspard Monge, Université Marne-la-Vallée

18