

## 6. Motif : messages et dialogues

Dialogues et messages

### Motif: messages et dialogues

- o Messages et dialogues
- o Les messages
- o Les dialogues
- o Modalité

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

1

Dialogues et messages

### Messages et dialogues

- o Les messages et dialogues sont le **moyen de communication privilégié** entre un programme et son utilisateur. Ils servent
  - | à **informer** l'utilisateur (messages),
  - | à **communiquer** (commande, sélection).
- o Un message ou dialogue
  - | fait **apparaître** une fenêtre pour l'utilisateur;
  - | la fenêtre **disparaît** lorsque l'utilisateur répond ou annule la communication.
- o L'application peut être suspendue en attendant la réponse :
  - | dialogue **modal** : la réponse est requise avant que l'application puisse continuer;
  - | dialogue **amodal** sinon.

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

2

## 6. Motif : messages et dialogues

Dialogues et messages

### Structure

- o Un message ou dialogue est composé :
  - l d'une widget Shell, de la classes DialogShell. Créée automatiquement par les fonctions utilitaires.
  - l d'un conteneur, fille unique de la shell. D'une classe dérivée de BulletinBoard
- o Un **message** ou dialogue *d'information* a un conteneur MessageBox.
- o Un **dialogue** de *communication* un conteneur SelectionBox.
- o Des fonctions utilitaires créent des dialogues spécifiques

```
graph LR; BulletinBoard --> MessageBox; BulletinBoard --> SelectionBox; BulletinBoard --> Command; BulletinBoard --> FileSelectionBox; SelectionBox --> MessageDialog; SelectionBox --> ErrorDialog; SelectionBox --> InformationDialog; SelectionBox --> QuestionDialog; SelectionBox --> WarningDialog; SelectionBox --> WorkingDialog; SelectionBox --> SelectionDialog; SelectionBox --> PromptDialog;
```

3

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

Dialogues et messages

### Messages

```
graph LR; MessageBox --> MessageDialog; MessageBox --> ErrorDialog; MessageBox --> InformationDialog; MessageBox --> QuestionDialog; MessageBox --> WarningDialog; MessageBox --> WorkingDialog;
```

- o Un message ou dialogue est composé de deux régions
  - l **région de contrôle** (textes, boutons radio, listes...) permettant des choix;
  - l **région de commande** (boutons accord, annulation..).
- o Les messages et dialogues sont des widgets *composites*.
- o La région de commande contient 3 boutons, étiquetés par OK, Cancel, Help. Un quatrième (Apply) dans les dialogues
- o La région de contrôle contient message, icône, ou plus dans les dialogues.
- o Il faut bien distinguer le *conteneur* (MessageBox) et la *shell* de dialogue (TransientShell).

4

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

## 6. Motif : messages et dialogues

Dialogues et messages

### Création de messages

- o Les dialogues des messages ne diffèrent que par leur logo (icône).
- o Les fonction se terminant en `Dialog` créent la widget Shell automatiquement, et retournent l'identifiant du conteneur (comme pour `scrolled text`).
- o Exemple, création d'un dialogue d'information

```
dial = XmCreateInformationDialog(bouton, "info", NULL, 0);
XtVaSetValues(dial,
  XtVaTypedArg, XmNmessageString, "Bienvenue.", 10, NULL);
```

`<Xm/MessageB.h>`

```
XmCreateErrorDialog()
XmCreateInformationDialog()
XmCreateQuestionDialog()
XmCreateWarningDialog()
XmCreateWorkingDialog()

XmCreateMessageBox()
XmCreateMessageDialog()
```

5

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

Dialogues et messages

### Gestion des messages

- o **Création** par une fonction utilitaire retourne la widget conteneur.
- o **Affichage**, par  

```
XtManageChild(dial);
```
- o **Disparition automatique** par activation du bouton OK ou Cancel, mais pas si l'on clique sur Help ou Apply.
  - Les widgets de la classe `BulletinBoard` (et donc ses sous-classes) possèdent une ressource de nom `XmNautoUnmanage`, par défaut à `True`, qui produit l'effet cherché.
- o **Accès aux filles** par `XmMessageBoxGetChild`

<code>XmDIALOG_OK_BUTTON</code>	<code>XmDIALOG_DEFAULT_BUTTON</code>
<code>XmDIALOG_CANCEL_BUTTON</code>	<code>XmDIALOG_HELP_BUTTON</code>
<code>XmDIALOG_MESSAGE_LABEL</code>	<code>XmDIALOG_SEPARATOR</code>
<code>XmDIALOG_SYMBOL_LABEL</code>	
- o **Ecarter** les filles inutilisées
  - soit en les enlevant  

```
XtUnmanageChild(XmMessageBoxGetChild(d, XmDIALOG_HELP_BUTTON));
```
  - soit en les insensibilisant  

```
XtSetSensitive(XmMessageBoxGetChild(d, XmDIALOG_HELP_BUTTON), False);
```

6

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

## 6. Motif : messages et dialogues

Dialogues et messages

### Exemple

```
x = XtVaCreateManagedWidget("x",xmPushButtonWidgetClass, tete,NULL);
XtAddCallback(x, XmNactivateCallback, activercreer, "un message");

void activercreer(Widget w, String s) {
    static Widget dial;

    if (! dial) { /* on le cree */
        dial = XmCreateMessageDialog(w, "Dialogue" , NULL, 0);
        XtVaSetValues(dial, XtVaTypedArg,
            XmNmessageString, XmRString, s, l+strlen(s), NULL);
        XtUnmanageChild(XmMessageBoxGetChild(dial,XmDIALOG_CANCEL_BUTTON));
        XtUnmanageChild(XmMessageBoxGetChild(dial,XmDIALOG_HELP_BUTTON));
    }
    XtManageChild(d);
}
```

- o Création du message *à la première activation* du bouton x.
- o Affichage systématique *à chaque activation*.
- o Disparaît si on clique sur le bouton OK.

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

Dialogues et messages

### Variante : création séparée

- o Autre style de programmation : création séparée de l'activation :

```
x = XtVaCreateManagedWidget("Aide",xmPushButtonWidgetClass, tete,NULL);
/* Creation */
dial = XmCreateMessageDialog(x, "aide",NULL,0);
XtVaSetValues(dial, XtVaTypedArg, XmNmessageString, XmRString,
    "Aide",5, NULL);
XtUnmanageChild(XmMessageBoxGetChild(dial,XmDIALOG_CANCEL_BUTTON));
XtUnmanageChild(XmMessageBoxGetChild(dial,XmDIALOG_HELP_BUTTON));
/* Adjunction de "activer" */
XtAddCallback(x, XmNactivateCallback, activer, dial);

/* Activation */
void activer (Widget w, Widget dial) {
    XtManageChild(dial);
}
```

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

## 6. Motif : messages et dialogues

Dialogues et messages

### Dialogues : création

```
<Xm/SelectionB.h>
XmCreateSelectionBox()
XmCreateSelectionDialog()
XmCreatePrompt()
XmCreatePromptDialog()

<Xm/FileSB.h>
XmCreateFileSelectionBox()
XmCreateFileSelectionDialog()

<Xm/Command.h>
XmCreateCommand()
```

```
graph TD
    SelectionBox[SelectionBox] --> SelectionDialog[SelectionDialog]
    SelectionBox --> PromptDialog[PromptDialog]
    SelectionBox --> Command[Command]
    SelectionBox --> FileSelectionBox[FileSelectionBox]
```

- o Un dialogue est comme un message, composé d'une
  - l région de *contrôle* (textes, boutons radio, listes...) mais plus sophistiquée;
  - l région de *commande* (accord, annulation...). Dans les sélections, il y a, en plus des 3 boutons, étiquetés OK, Cancel, Help, un quatrième bouton, étiqueté Apply. Ce dernier est créé mais non "managé" par défaut.

9

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

Dialogues et messages

### Gestion de dialogues

- o Elle est la même que pour le messages.
- o L'accès aux filles se fait par  
XmSelectionBoxGetChild,  
XmFileSelectionBoxGetChild
- o Chaque *filles* a un nom symbolique (valable si elle existe dans la widget)

**FileSelectionBox**

```
XmDIALOG_APPLY_BUTTON,
XmDIALOG_CANCEL_BUTTON,
XmDIALOG_DEFAULT_BUTTON,
XmDIALOG_DIR_LIST,
XmDIALOG_DIR_LIST_LABEL,
XmDIALOG_FILTER_LABEL,
XmDIALOG_FILTER_TEXT,
XmDIALOG_HELP_BUTTON,
XmDIALOG_LIST,
XmDIALOG_LIST_LABEL,
XmDIALOG_OK_BUTTON,
XmDIALOG_SELECTION_LABEL,
XmDIALOG_SEPARATOR,
XmDIALOG_TEXT,
XmDIALOG_WORK_AREA
```

10

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

## 6. Motif : messages et dialogues

Dialogues et messages

### Ressources utiles

- o Ressources générales
  - XmNokLabelString
  - XmNcancelLabelString
  - XmNhelpLabelString
  - XmNokCallback
  - XmNcancelCallback
  - XmNhelpCallback
- o MessageDialog
  - XmNmessageString
  - XmNsymbolPixmap
- o SelectionDialog
  - XmNapplyLabelString
  - XmNselectionLabelString
  - XmNlistLabelString
  - XmNapplyCallback
- o FileSelectionDialog
  - XmNfilterLabelString
  - XmNfileListLabelString
  - XmNdirListLabelString
- o Command
  - XmNpromptString

11

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

Dialogues et messages

### Modalité

- o Un dialogue est
  - | *modal*, si l'exécution du programme est suspendue tant que la réponse de l'utilisateur n'est pas donnée;
  - | *amodal*, sinon.
- o Les deux types de dialogues ont leur utilité :
  - | *modal* : confirmation de destruction, modification irréversible, choix d'un paramètre,...
  - | *amodal* : la plupart des dialogues de sélection.
- o Les dialogues Motif permettent de spécifier le mode choisi :
  - DIALOG\_SYSTEM\_MODAL : réponse requise avant toute interaction dans toute application;
  - DIALOG\_FULL\_APPLICATION\_MODAL : réponse requise avant interaction avec une widget de l'application;
  - DIALOG\_PRIMARY\_APPLICATION\_MODAL
  - DIALOG\_APPLICATION\_MODAL : réponse requise avant interaction avec la widget qui a lancé le dialogue;
  - DIALOG\_MODELESS : (défaut) pas de suspension.
- o Choix par XmNdialogStyle.

12

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

## 6. Motif : messages et dialogues

Dialogues et messages

### Gestion des dialogues

- o Il y a deux logiques de programmation des dialogues:
  - | logique par "délégation" ou "continuation":
  - | logique "séquentielle"
- o **Par délégation:** Une fonction (p. ex. un réflexe associé à un bouton)
  - | **poste le dialogue**
  - | puis retourne à la boucle d'évènements (MainLoop) pour que l'application puisse gérer les évènements qui suivent
  - | La réponse aux questions du dialogue est enregistrée dans des fonctions réflexes du dialogue.

**Logique par délégation**

```
{ ... dans fonction réflexe du bouton
d = CréerMessage();
AddCallback(d, GererOK);
AddCallback(d, GererCANCEL);
PosterMessage(d);
return;
}
void GererOK() {
// faire en cas de oui
}
void GererCANCEL(){
// faire en cas de non
}
```

13

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

Dialogues et messages

- o **Logique séquentielle** La fonction réflexe
  - | poste le dialogue
  - | **attend la réponse**
  - | entreprend l'action en fonction de la réponse
  - | La logique séquentielle demande l'installation d'une *boucle locale de gestion d'évènements*, pour attendre la réponse.

**Logique séquentielle**

```
{ ... dans fonction réflexe du bouton
d = CréerMessage();
PosterMessage(d);
reponse = AttendreReponse();
si (reponse == OK)
faire_oui();
si (reponse == CANCEL)
faire_cancel();
...
}
```

14

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

## 6. Motif : messages et dialogues

Dialogues et messages

### Exemple (par délégation)

```
/* dans la fonction reflexe */
Widget dial;
dial = creerDialogue(w,question);
XtAddCallback(dial, XmNokCallback, gerer_ok, NULL);
XtManageChild(dial);
return;
}

void gerer_ok(Widget w, XtPointer cl, XtPointer ca)
{
    faire_oui();
    XtDestroyWidget(w);
}

Widget creerDialogue(Widget w, String s)
{
    Widget d;
    d = XmCreateQuestionDialog(w, "Dialogue", NULL, 0);
    XtVaSetValues(d, XtVaTypedArg, XmNmessageString, XmRString,
        s, 1+strlen(s), NULL);
    XtVaSetValues(d, XmNdialogStyle, XmDIALOG_FULL_APPLICATION_MODAL, NULL);
    XtUnmanageChild(XmMessageBoxGetChild(d, XmDIALOG_CANCEL_BUTTON));
    XtUnmanageChild(XmMessageBoxGetChild(d, XmDIALOG_HELP_BUTTON));
    return d;
}
```

15

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

Dialogues et messages

### Exemple (séquentiel)

```
Boolean Question(Widget w, char* question) {
    Widget dial;
    Boolean reponse = FALSE;

    dial = creerDialogue(w,question);
    XtManageChild(dial);
    XtAddCallback(dial, XmNokCallback, dire_oui, (XtPointer) &reponse);

    while (XtIsManaged(dial))
        XtAppProcessEvent(app, XtIMAll);

    XtDestroyWidget(dial);
    return reponse;
}

o Reflexe
void dire_oui( Widget w, XtPointer
    cd, XtPointer ca)
{
    int* adresse = (int *) cd;
    *adresse = TRUE;
}

o Variante explicite pour boucle locale de
gestion d'évènements:
while (XtIsManaged(d)) {
    XEvent e;
    XtAppNextEvent(app, &e);
    XtDispatchEvent(&e);
}
```

16

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