Utilisation des Resources dans X/MOTIF
Leurs applications dans Xspect

Gauthier LEBRUN

10 juin 1997

Résumé

Une particulartité des programmes X/MOTIF est qu'ils sont entièrement paramétrables par l'utilisateur grâce au mécanisme des resources. Ainsi, l'utilisateur peut créer un fichier de resources dans lequel il indiquera la couleur de ses barres de menus par exemple.
Pour le programmeur, c'est un moyen très simple pour adapter un programme dans une langue différente, en définissant par exemple les libellés des menus dans un fichier de resources. Il n'aura ensuite qu'à modifier ce fichier pour changer la langue du programme, sans devoir le recompiler.
Ce document nous décrit succintement le mécanisme de la gestion des resources sous X/MOTIF, et comment il a été utilisé dans Xspect.

1 Généralités

Les widgets et les applications peuvent déclarer toutes leurs variables comme resources. Dans la pratique, seules celles ayant besoin de valeurs entrées par l'utilisateur seront définies en resources (comme les configurations de fenêtres par exemple).
La bibliothèque Xt permet de définir les resources en utilisant une structure `XtResource` déclarée comme suit:

```
typedef struct {
    String resource_name;       /* specifiee par le symbole XmN   */
    String resource_class;      /* specifiee par le symbole XmC   */
    String resource_type;       /* type de la variable actuelle   */
    Cardinal resource_size;     /* taille determinee par sizeof() */
    Cardinal resource_offset;   /* determinee par XtOffsetOf()    */
    String default_type;        /* valeur immediate ou fonction   */
    XtPointer default_address;  /* adr. de la valeur par defaut   */
} XtResource, *XtResourceList;
```

Pour utiliser des resources dans des widgets existants, il suffit d'écrire un fichier de resources.
Normalement, ce fichier se trouve dans le répertoire */usr/lib/X11/app-defaults* et a pour nom le même que l'argument `classname` de `XtVaAppInitialize()`. Par convention, ce nom est le même que celui de l'application, avec la première lettre en majuscule. Si l'application commence par X, les deux premières lettres seront en majuscules. Par exemple pour *xclock*, ce serait *XClock*.

La gestion des resources s'effectuent en plusieurs étapes :

1. Au démarrage de l'application, par un appel `XtAppInitialize`, la bibliothèque Xt lit les fichiers de resources, que ce soit .Xdefaults ou un fichier utilisateur tel que `xspect.ad` (en utilisant le programme *xrdb*). Puis, Xt les fusionne dans une seule base interne au programme qui sera utilisée à chaque création de widget.
2. Si aucun fichier de resources n'est utilisé (c.-à-d. qu'on ne s'est pas servi de *xrdb* avant de lancer le programme), alors la base de resources est remplie par les resources codées en "dur" : c'est le champ *fall-back_resources* (pointeur sur un tableau XtResourceList).
3. Quand on crée un widget, l'appel à `XtVaCreateManagedWidget()` lit la base de resources et affecte automatiquement les resources des widgets aux valeurs contenues dans la base.

Note:

- Si la valeur d'une resource est codée en "dur" par `XtVaManagedWidget` ou `XtVaSetValues`, cette valeur écrase celle lue dans la base.
- Pour récupérer les valeurs des resources de l'application, on doit faire un appel à `XtGetApplicationResources`.
- Si la ligne de commande comporte des resources, elles écrasent celles spécifiées par la base. Par exemple :

  ```
  xspect +askBeforeExit
  ```

  lancera Xspect avec l'option *askBeforeExit* même si dans le fichier de resource cette option est désactivée.

2 Syntaxe

- Chaque entrée de la base est un couple spécification / valeur.
  `spécification = nom de l'application + '.' + Nom de la resource`

valeur = ':' + chaîne de caractéres
Exemple : `xterm.scrollBar:on`

- Une * peut être utilisée comme "joker" à la place du nom de l'application (ex.:`*scrollBar`). On affecte alors la valeur à toutes les resources ayant la propriété (ici `scrollBar`) dans leur base.
- Une ligne commentaire comence par !.

3 Utilisation des resources dans Xspect

3.1 Le fichier de resource xspect.ad

L'utilisateur peut (re-)définir l'environnement de **Xspect** en modifiant le fichier de resources `xspect.ad`.
Le fichier original `xspect.ad` se trouve dans le répertoire */u/formes/src/xspect/Xspect/src/app-default*. Il comporte les resources de **Xspect** que l'on peut personaliser telles que les couleurs, mais aussi les racourcis claviers.

IMPORTANT: il est indispensable de copier ce fichier dans son compte avant de le modifier.

Pour charger le fichier de resources, on utilise le programme *xrdb* :

```
xrdb -merge xspect.ad
```

3.2 Le fichier include resource.h

Le fichier *include* **resource.h** comporte les resources par défaut de Xspect. On peut donc les modifier dans un fichier de resources ; par exemple :

```
xspect.margintop:10
xspect.askBeforeExit:false
```

Voici la liste des resources définies dans resource.h :

```
askBeforeExit
audioDisplay (Alpha seulement)
autoPlaceWindow
buttonBarSize
createViewWarning
firstViewCreated
fileFormat
frameActiveBkgColor
frameActiveShadowType
frameBkgColor
frameShadowType
idActiveBkgColor
idBkgColor
idLeftPosition
marginBottom
marginLeft
marginRight
marginTop
```

maxSecScreen	sfDir
maxSecVerify	signalFullPage
mode	splitPercent
nColors	watchbufferLimits
openFilesIn	zoomMode
rawWarning	
selectionQuestion	

3.3 Initialisation des resources

L'initialisation des resources dans le programme se fait par l'appel de la fonction `AppInitialize` (qui se trouve dans le module **resources.c**). Cette fonction comporte l'appel suivant :

```
widget=XtVaAppInitialize(appContext,''XXspect'',
       optionDesc, XtNumber(optionDesc),
       pargc, pargv, resourcesString,NULL);
```

La variables `resourcesString` passée en argument représente donc la liste des resources par défaut de l'application. Cette variable a été initialisée dans **resources.h**.

3.4 Ajout d'une option sous Xspect

Cette partie s'adresse essentiellement au programmeur qui désire ajouter une option au programme.
Les options sont les resources propres à Xspect et trouvent leur valeur par défaut dans le fichier `resource.h`.
Une option doit donc pouvoir être spécifiée dès le lancement du programme mais peut aussi être modifiée en cours d'exécution par l'intermédiaire de la boîte de context d'options.

L'ajout d'une option dans Xspect se fait en plusieurs étapes. Nous prendrons exemple sur l'option privateColormap qui permet de passer en palette de couleurs privées :

1. Modification de la structure Options (fichier typedefs.h). Cette structure regroupe toutes les options propres à Xspect. Ceci permet de les centraliser et de pouvoir les manipuler rapidement si l'utilisateur les modifie par l'intermédiaire du context d'Options :

   ```
   typedef struct _Options
   {
        ....
        Boolean      privateColormap;
   } Options;
   ```

2. Cette option aura une valeur par défaut qu'il faut définir comme constante dans le fichier defines.h :

   ```
   #define PRIVATE_COLORMAP    False
   ```

3. Il faut ensuite modifier le fichier resources.h. Nous avons besoin de deux constantes XtNprivateColormap et XtCprivateColormap qui seront nécessaires dans le tableau resources. Ces constantes permettent de donner des noms aux resources :

   ```
   #define XtNprivateColormap "privateColormap"
   #define XtCprivateColormap "PrivateColormap"
   ```

4. Dans le même fichier, ilfaut modifier le tableau resources en ajoutant les caractéristiques de la nouvelle resource : noms, type, taille, et valeur par défaut.

   ```
   static XtResource resources[] = {
     ...
     { XtNprivateColormap, XtCprivateColormap,
       XtRBoolean, sizeof(Boolean),
       XtOffset(OptionsPtr, privateColormap), XtRImmediate,
       (XtPointer) PRIVATE_COLORMAP
     },
   };
   ```

5. Toujours dans resources.h, il faut modifier le tableau des options optionDesc qui permet de spécifier ce qui devra être tapé pour lancer une option au démarrage de Xspect.

```
static XrmOptionDescRec optionDesc[] = {
  ....,
  {"-privateColormap", XtNprivateColormap, XrmoptionNoArg, "false"},
  {"+privateColormap", XtNprivateColormap, XrmoptionNoArg, "true"},
};
```

6. En lançant Xspect avec l'options -h, on obtient la liste de toutes les options avec leur descriptifs. Il faut donc rajouter cette nouvelle option dans cette liste, ce qui se fait dans le fichier resource.c dans la fonction usage.

```
fprintf(stderr, " -/+privateColormap  Launch with the private
colormap.\n");
```

7. Pour terminer, il faut rajouter cette option dans la boîte de context Option, en rajoutant un champ context.