

# **MUIBuilderFrançais**

Eric Totel

Copyright © CopyrightÂ©1994 Eric Totel

COLLABORATORS

	TITLE : MUIBuilderFrançais		
ACTION	NAME	DATE	SIGNATURE
WRITTEN BY	Eric Totel	July 26, 2024	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

# Contents

<b>1</b>	<b>MUIBuilderFrançais</b>	<b>1</b>
1.1	MUIBuilder . . . . .	1
1.2	COPYRIGHT . . . . .	1
1.3	INTRODUCTION . . . . .	2
1.4	MUI . . . . .	2
1.5	avantages . . . . .	3
1.6	Utilisation . . . . .	3
1.7	Principes généraux . . . . .	4
1.8	Les Objets . . . . .	4
1.9	Configuration . . . . .	5
1.10	La sauvegarde du contexte . . . . .	6
1.11	poids . . . . .	6
1.12	application . . . . .	7
1.13	Fenêtre . . . . .	7
1.14	liste temporaire . . . . .	8
1.15	groupe . . . . .	8
1.16	bouton . . . . .	9
1.17	liste . . . . .	9
1.18	Liste de répertoires . . . . .	10
1.19	chaîne . . . . .	10
1.20	label . . . . .	11
1.21	cycle . . . . .	11
1.22	radio . . . . .	11
1.23	image . . . . .	12
1.24	espace . . . . .	12
1.25	checkmark . . . . .	12
1.26	slider . . . . .	12
1.27	jauge . . . . .	13
1.28	Règle . . . . .	13
1.29	Texte . . . . .	13

---

1.30	Gadget Proportionnel . . . . .	14
1.31	Caractères Spéciaux . . . . .	14
1.32	code . . . . .	15
1.33	options . . . . .	16
1.34	code-application . . . . .	16
1.35	code-object . . . . .	16
1.36	remove-label . . . . .	17
1.37	add-label . . . . .	17
1.38	Localisation . . . . .	17
1.39	Catalogue . . . . .	17
1.40	Langage C . . . . .	18
1.41	gencodec_old . . . . .	19
1.42	gencodec . . . . .	19
1.43	Langage E . . . . .	20
1.44	guide . . . . .	20
1.45	L'enregistrement . . . . .	21
1.46	Futur . . . . .	22
1.47	Remerciements . . . . .	22

## Chapter 1

# MUIBuilderFrançais

### 1.1 MUIBuilder

MUI-BUILDER  
V1.1

Ecrit par Eric Totel  
en 1994

Introduction  
Avantages de MUIBuilder  
Utilisation  
Génération du code  
Génération AmigaGuide  
Améliorations futures  
Copyright  
Enregistrement  
Remerciements

### 1.2 COPYRIGHT

(C) Copyright 1993 Eric Totel. Tous droits réservés.

Vous pouvez me contacter à :

E-Mail : [totel@laas.fr](mailto:totel@laas.fr)

or Eric Totel  
5 rue Riquet  
31000 Toulouse  
France

MUI-Builder est Giftware.  
( cadeau-ware pour les non-anglicistes !!! )

Ce programme peut être librement distribué, tant que personne ne tire aucun bénéfice de cette distribution. Tout autre type de

vente ne peut en aucun cas être effectuée sans l'autorisation ECRITE de l'auteur.  
MUI-Builder peut être inclus dans des collections de logiciels du domaine public, tant que les conditions ci-dessus restent vérifiées.

L'auteur se dégage ici de toute responsabilité quant à d'éventuelles dégradations de données occasionnées par l'utilisation de MUI-Builder. Ce logiciel est délivré 'tel quel', sans aucune garantie d'aucune sorte.

## 1.3 INTRODUCTION

Merci d'avoir récupéré MUI-Builder !

Vous avez maintenant entre vos mains un outil qui, je l'espère devrait vous être très utile.

MUI-Builder est un outil qui va vous permettre d'écrire des Applications @{ " MUI " link MUI }, sans avoir à taper des lignes et des lignes de code, et sans avoir à connaître la syntaxe ( somme toute relativement simple ) de @{ " MUI " link MUI } .

Grâce à MUI-Builder, vous allez enfin pouvoir créer vos interfaces graphiques sans la moindre arrière pensée, et sans plus d'effort que de réfléchir à ce que vous voulez réaliser.

Le but premier de ce programme était uniquement de satisfaire à mes besoins ( en créant un outil qui devrait être intégré dans @{ " MUI " link MUI } ↔ )  
et d'apprendre à utiliser cet outil fantastique que nous a réalisé Stefan Stuntz ( que je remercie d'ailleurs ici ).

J'espère que ce soft vous sera aussi utile qu'il me l'a déjà été jusqu'à présent.

## 1.4 MUI

Cette application utilise

MUI - MagicUserInterface

(c) Copyright 1993 by Stefan Stuntz

MUI est un système pour générer et maintenir des interfaces graphiques. Avec l'aide d'un programme de préférences, l'utilisateur d'une application a la possibilité d'en modifier l'apparence extérieure pour qu'elle s'adapte à ses goûts.

MUI est distribué en tant que shareware. Pour obtenir une archive contenant de

---

nombreux exemples et plus d'informations en ce qui concerne l'enregistrement procurez-vous un fichier appelé "muiXXusr.lha" (où XX représente le numéro de version le plus récent) sur votre BBS local ou sur des disks du domaine public.

Si vous voulez vous enregistrer directement, vous pouvez envoyer

DM 20.- ou US\$ 15.-

à

Stefan Stuntz  
Eduard-Spranger-Straße 7  
80935 München  
GERMANY

## 1.5 avantages

Beaucoup se demanderont quel est l'intérêt d'un tel programme compte tenu de la simplicité d'utilisation de MUI.

A tous ces gens j'expose ici les avantages que moi et tous ceux qui ont eu l'amabilité de tester MUI-Builder ont pu remarquer :

1. MUI-Builder est, pour les débutants, LE moyen de maîtriser RAPIDEMENT MUI, en regardant le code généré par le Builder. Son utilisation en fait presque un didacticiel destiné à vous guider dans la découverte de MUI.
2. MUI-Builder se comporte, pour ceux qui maîtrisent déjà MUI, comme une sorte d'interpréteur du langage associé à MUI. Vous pouvez directement tester l'allure de votre interface graphique pour connaître à l'avance le résultat. Tous ceux qui ont utilisés régulièrement MUI se sont aperçus en effet de certaines limitations ( en particulier en ce qui concerne le resizing et la taille fixe de certains objets qui affectent parfois le look de l'interface ).
3. MUI-Builder offre une grande flexibilité dans la génération du code ( chaque objet peut être généré indépendamment des autres, avec ou sans les déclarations et initialisations des variables associées aux objets MUI ).
4. Il est enfin possible, depuis un interface builder de construire l'aide en ligne correspondant à l'interface graphique. Ce qui est tout à fait nouveau ( à ma connaissance) sur l'Amiga.
5. C'est le premier programme de ce type à permettre une localisation aisée de votre programme !

## 1.6 Utilisation

---

Les principes d'utilisation de MUI-Builder peuvent être classés de la manière suivante:

- Principes généraux
- Les Objets
- Configuration
- La sauvegarde du contexte courant

Pour obtenir de l'aide à n'importe quel moment dans le programme, pressez tout simplement la touche 'HELP'.

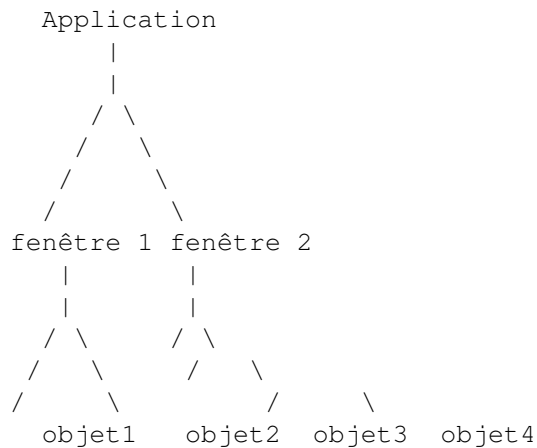
## 1.7 Principes généraux

MUI-Builder conserve le même type d'idée de conception de l'interface que MUI lui-même ( auquel il est d'ailleurs très lié ).

Je ne peux donc que vous conseiller de vous référer à la documentation de MUI, pour mieux appréhender le principe.

MUI-Builder permet de créer l'interface complète d'une application, et, par la même occasion, de créer toutes les fenêtres attachées à l'application.

En fait, chaque application est un arbre de dépendance des objets d'interface qui la compose. Ainsi, une application comportant deux fenêtres, qui contiennent chacune un objet quelconque, pourra être schématiquement représentée par l'arbre suivant:



L'idée de MUI-Builder est de vous permettre de créer cet arbre en utilisant une interface graphique ( que j'espère suffisamment simple et conviviale ).

## 1.8 Les Objets

MUI-Builder permet de créer tous les objets que vous voudrez bien attacher à votre application. Ces objets peuvent être classés comme suit :

Application	
Fenêtre	
Groupe	Bouton
Liste	Dirlist
Chaine	Label
Cycle	Radio
Image	Espace
CheckMark	Slider
Jauge	Scale
Texte	Prop

Il est à noter qu'un objet, outre le fait d'être créé, peut également être modifié. Pour cela, il suffit de double-cliquer sur son nom dans une liste où il apparaît.

Pour certains objets, vous pourrez modifier l'attribut 'Weight' ( poids en Français )

## 1.9 Configuration

Le panneau de configuration offre un certain nombre de choix quand à la manière dont le programme va fonctionner par la suite. Tous ces choix sont sauvegardés dans une variable d'environnement ( ENV:MUIBuilder.env ), cette variable étant créée pendant l'installation du logiciel.

Les choix que vous serez amené à faire sont les suivants :

- Fenêtres fantômes : si vous sélectionnez cette option, vous n'aurez jamais plus d'une fenêtre de MUIBuilder présente à l'écran en même temps ( elles s'effaceront automatiquement lorsqu'elles ne sont pas utilisées ). Cette option peut être utile si vous n'aimez pas avoir beaucoup de fenêtres à l'écran ou si vous avez une machine relativement lente ( c'est-à-dire à base de 68000 ).
- Icônes : si vous sélectionnez cette option, les sauvegardes seront accompagnées d'un icône par défaut : cette icône se trouve normalement dans 'ENV:Sys/' et se nomme 'def\_MUIBuilder.info'. Elle est normalement copiée à sa place à l'installation du logiciel.
- Requêtes : indique si vous voulez ou non des requesters à chaque fois que vous faites une action importante ( effacement d'un objet par exemple ).
- Code : indique quel générateur de code vous voulez utiliser.
- Editeur : Donnez le nom de votre éditeur favori ! Assurez-vous que cet éditeur se lance de manière synchrone ( c'est-à-dire

qu'il ne vous rende pas la main juste après lancement ! )

Les éditeurs suivants devraient fonctionner sans problème :

- Vim (v2.0) : mettez l'option '-x'
  - CygnusEd : utilisez l'option '-keepio'
  - GoldEd : utilisez l'option 'STICKY'
  - TurboText : en mettant l'option WAIT
  - Ed : éditeur par défaut
- GetString : ATTENTION ceci est IMPORTANT !!!  
Si vous rentrez une chaîne ici, elle désignera le nom de la procédure 'GetString' utilisée pour la localisation. Cette chaîne sera copiée dans le gadget chaîne correspondant de la fenêtre de génération de code la PREMIERE fois que vous y rentrerez ( pas les autres, si bien que vous pourrez en fait passer outre cette option de configuration ).  
Si cette chaîne est laissée VIDE, alors le nom par défaut sera construit à partir du nom de votre application ('GetApplicationNameString').

Une fois les options positionnées, vous pouvez les sauver en sélectionnant le bouton correspondant !

## 1.10 La sauvegarde du contexte

Les boutons 'Sauver' et 'Charger' vous permettront respectivement de sauvegarder et récupérer l'interface de l'application en cours de création.

De cette manière vous serez en mesure de poursuivre un travail interrompu, ou bien de restaurer une ancienne application dont vous voulez modifier l'interface.

Il est à remarquer que la fenêtre principale de MUIBuilder est une appwindow : vous pouvez donc amener une icône sur la liste de la fenêtre principale pour lire un fichier !

## 1.11 poids

Le poids est l'un des attributs les plus intéressants des objets : il va vous permettre de définir de manière fine les tailles relatives des objets les uns par rapport aux autres.

Tous les objets ont un poids par défaut de 100 .

Si vous désirez par exemple qu'un groupe ait une taille double de son groupe voisin, vous lui affecterez un poids double.  
De cette manière, on peut éviter de se retrouver avec des boutons de taille phénoménale !!!

De plus un poids de zéro indique que l'objet doit garder ( dans tous les cas de figure ) sa taille minimale.

## 1.12 application

L'application est en fait le noeud racine de l'arbre de dépendance constituant l'interface graphique complète.

En appuyant sur le bouton 'Appli' dans la fenêtre principale de MUI-Builder, vous pouvez déterminer :

- la 'base' de votre application, c'est-à-dire le nom du serveur AREXX de votre application.
- le nom de l'auteur
- le titre de votre application
- la version
- un texte relatif au copyright
- une description de votre application

L'application ne peut avoir qu'un seul type de fils, à savoir les Fenetres .

## 1.13 Fenêtre

Vous savez tous déjà ce qu'est une fenêtre ! Notre probleme va être d'apprendre à la construire grâce à MUI-Builder ...

Deux pages se partagent l'espace réservé : l'un nommé 'Création', l'autres 'attribut' :

La page de création de la fenêtre se divise en deux parties distinctes :

1. Une barre de texte ou apparaitra les liens de parentés entre les objets que vous allez sélectionner.
2. Enfin l'ensemble des fils, petits fils, petits petits ... fils de la fenêtre qui sont disposés en trois groupes :
  - 2.1 Dans la liste de gauche :

L'ensemble des groupes ayant un lien de parenté avec la fenêtre ( Y compris le groupe principal, aussi appelé groupe\_racine qui est le seul fils direct de la fenêtre ).
  - 2.2 Dans la liste du milieu:

L'ensemble des fils du groupe sélectionné dans la

fenêtre de gauche.

### 2.3 Une liste d'objets 'temporaire' ( voir tmp list ).

Les groupes constituent les éléments de base dans lesquels vont être intégrés tous les autres objets . Ainsi, à chaque fois que vous ajouterez un objet, vous devrez le déclarer comme étant le fils du groupe que vous aurez précédemment sélectionné.

Dans la page 'Attributs' se trouve :

- Le label de la fenêtre, ainsi que son titre
- Les attributs de création de la fenêtre :
  - AppWindow : si vous voulez que votre fenêtre soit une fenêtre d'application
  - Pas de bord : si vous ne voulez pas de bordure !
  - Gadget profondeur : à sélectionner si vous voulez ce gadget
  - Gadget taille : à sélectionner si vous voulez ce gadget
  - Fenêtre en fond : pour que votre fenêtre se trouve toujours derrière toutes les autres !
  - Gadget Fermeture : à sélectionner si vous voulez ce gadget
  - Barre de mouvement: pour être autorisé à bouger la fenêtre

## 1.14 liste temporaire

Dans cette liste vous pouvez amener des objets, afin de les passer d'un groupe à l'autre, ou d'une fenêtre à l'autre.

## 1.15 groupe

Lorsque vous créez une nouvelle fenêtre, un groupe lui est déjà attaché : le groupe principal, encore nommé 'groupe racine'.

Vous allez ensuite construire votre fenêtre en attachant des objets à ce groupe racine.

Parmi les fils pourront, bien entendu, exister d'autres groupes qui, eux-mêmes, contiendront d'autres fils ... etc ...

Enfin, vous devrez, pour chaque groupe, définir l'ensemble de ses attributs :

- Horizontal : pour que le groupe ait une disposition horizontale
- Registre : Le groupe ne pourra 'voir' qu'un seul de ses fils à la fois.  
Vous devez alors indiquer dans la liste des entrées du registre les titres qui

doivent apparaître sur les fiches du registre

- Même hauteur: Tous les fils du groupe auront la même hauteur
- Même largeur: Tous les fils du groupe auront la même largeur
- Même taille : Tous les fils du groupe auront la même taille
- Virtuel : Le groupe sera un groupe virtuel
- Titre : Le groupe sera surmonté d'un titre que vous devez donner dans la barre 'title'
- Poids : Permet de définir le @{ "poids" LINK Poids} du groupe
- Colonnes : affiche le groupe sous forme de colonnes ( le nombre de colonnes devant être entré dans la chaine 'number' )  
Cet attribut permet d'aligner des éléments dans une fenêtre.
- Lignes : affiche le groupe sous forme de lignes ( le nombre de lignes devant être rentré dans la chaine 'number' )
- Espacement : Horizontal : permet de définir l'espacement horizontal entre les objets  
  
Vertical : permet de définir l'espacement vertical entre les objets  
  
L'espacement est à rentrer dans la chaine correspondante.

## 1.16 bouton

Pour définir complètement un bouton, il suffit de donner :

- Le label que portera le bouton dans le programme
- Le texte qui apparaîtra dans le bouton
- Le raccourci clavier qui sera supporté par le bouton

Vous pouvez également donner le @{ "poids" LINK Poids } du bouton, grâce au slider 'Poids'.

## 1.17 liste

---

Pour définir complètement une liste, il suffit de donner :

- Son label dans le programme C généré.
- Son `@{ "poids" LINK Poids }`.
- Si le 'double click' est autorisé dans cette liste.
- Si la multi-sélection est autorisée dans cette liste.
- Le type de la liste, qui peut être :
  - une liste standard
  - une liste de floattext, permettant d'afficher un texte banal.
  - une liste de 'Volumes', qui contient tous les assigns et volumes de votre configuration.

Il est à remarquer qu'il existe un autre type de liste, appelé `@{ "Directory" LINK DirList }`.

Ce type de liste peut être créé indépendamment en cliquant sur son bouton dans la fenêtre de choix des objets.

## 1.18 Liste de répertoires

La liste de répertoire affiche à l'écran, les fichiers et répertoires présents dans le directory spécifié.

Les options possibles pour cet objet sont les suivantes :

- le poids ( cf `@{ " Poids " link Poids }` )
- 'Répertoires uniquement' : à sélectionner si vous ne voulez QUE les répertoires
- 'Fichiers uniquement' : à sélectionner si vous ne voulez QUE les fichiers
- 'Multi Sélection' : autorise une multi-sélection dans la liste
- 'Rejeter les Icônes' : n'affiche pas les '.info'
- 'Inverser le tri' : inversion de l'ordre de tri
- 'Type de tri' : choix du type du tri ( nom, date, taille )

La chaîne 'Répertoire' doit contenir le répertoire de lecture pour la DirList.

## 1.19 chaîne

Le gadget Chaîne est défini par les données suivantes :

- Le titre qui apparaîtra directement à gauche du gadget `string ...` ( et que vous pouvez enlever )
- le label de la variable associée au gadget chaîne
- le contenu initial du gadget chaîne lors de l'affichage
- une chaîne de caractères contenant les caractères acceptés par votre gadget chaîne.
- une chaîne de caractères contenant les caractères refusés par votre gadget `string`.
- la longueur maximale de la chaîne qui devra être acceptée

On peut également redéfinir le `@{ " poids " LINK Poids }` de la chaîne.

## 1.20 label

Pour définir complètement un label, il faut donner :

- son label dans le programme
- le contenu du texte constituant le label
- on peut également redéfinir son `@{ "poids" LINK Poids }`.

## 1.21 cycle

Pour définir complètement un cycle, il faut donner :

- la liste des entrées du cycle
- le label du gadget cycle dans le programme
- on peut redéfinir le `@{ "poids" LINK Poids }` du cycle

## 1.22 radio

Pour définir complètement des boutons radios, il faut :

- la liste des boutons radios
  - le label du gadget radio dans le programme
  - on peut également redéfinir le `@{ "poids" LINK Poids }`
-

## 1.23 image

Avant toute chose vous devez cliquer sur une des images pour sélectionner celle qui vous intéresse.

Les choix à effectuer sont les suivants :

- 'Libre Verticalement' : l'image se redimensionnera dans le sens vertical
- 'Libre Horizontalement' : l'image se redimensionnera dans le sens horizontal
- 'Mode Input' : Votre image pourra être sélectionnée par l'utilisateur.
- 'Fixer la Hauteur' : Détermine la hauteur de l'image une fois pour toutes ( indiquée dans le gadget string correspondant ). Rend inutile 'Free Vertical' lorsqu'il est activé.
- 'Fixer la Largeur' : Détermine la largeur de l'image une fois pour toutes ( indiquée dans le gadget string correspondant ). Rend inutile 'Free Horizontal'

Comme pour la plupart des objets, il faut donner le label de la variable qui sera générée dans le programme.

## 1.24 espace

Cet objet sert uniquement à insérer un espace entre deux autres objets, afin de permettre le redimensionnement de la fenêtre.

## 1.25 checkmark

Vous pouvez, pour les CheckMarks ( comme pour les chaînes et les slider ) indiquer si un titre doit être écrit ou non juste devant le CheckMark.

Si ce titre doit apparaître, vous devez l'indiquer dans la chaîne 'Title'.

... et bien sûr il faut une fois de plus donner le label de l'objet.

## 1.26 slider

Pour le slider, il faut spécifier :

- si le niveau courant doit être affiché ( 'Slider Silencieux' )
- si le slider doit être inversé ou non
- si un titre doit apparaître devant le slider

Il faut alors rentrer :

- la valeur maximale qui peut être atteinte
- la valeur minimale
- la valeur initiale

vous pouvez également remplir :

- le titre qui peut éventuellement apparaître devant le slider
- le label de l'objet

## 1.27 jauge

Pour définir une jauge, il faut donner :

- le sens de la jauge ( horizontale ou verticale )
- si l'on veut fixer la hauteur de la jauge ( utilisé en particulier pour les jauges horizontales )  
Cette hauteur doit être spécifiée dans le gadget String correspondant.
- si l'on veut fixer la largeur de la jauge ( utilisé en particulier pour les jauges verticales )  
Cette largeur doit être spécifiée dans le gadget String correspondant.
- si son contenu va être 'divisé'
- le maximum que peut atteindre la jauge
- Le label de la jauge dans le programme généré

## 1.28 Règle

Le gadget Scale doit être utilisé en conjonction avec le gadget Jauge. Son but est de dessiner une graduation sous ce dernier.

Le seul élément à donner à la création de ce gadget est le sens de positionnement ( horizontal et vertical ).

## 1.29 Texte

Les gadgets textes peuvent être définis grâce aux attributs suivants :

- 'Text\_SetMax' : la taille maximale du gadget texte est la taille initiale du gadget.
  - 'Text\_SetMin' : la taille minimale du gadget texte est la taille initiale du gadget.
  - 'Fonds' : testez et vous verrez !
  - 'Encadré' : idem
-

Le texte doit être rentré dans le gadget chaîne correspondant et peut comporter tous les caractères spéciaux que l'on peut en particulier trouver dans des sources C.

Vous devrez également rentrer le nom de label du gadget.

### 1.30 Gadget Proportionnel

Les options à choisir sont les suivantes :

- 'Horizontal' : indique si le gadget doit être horizontal
- 'Fixer largeur' : Largeur fixe ( à indiquer dans le gadget string correspondant )
- 'Fixer Hauteur' : Hauteur fixe ( à indiquer dans le gadget string correspondant )

Ensuite, vous devez indiquer :

- le nombre d'entrées pour le gadget proportionnel
- le numéro de la 1ère entrée
- le nombre d'entrées visibles

Et il faut bien sûr donner au programme le label de l'objet.

### 1.31 Caractères Spéciaux

dans les textes que vous rentrez ( TOUS les textes qui apparaissent à l'écran et non pas seulement les gadgets texte ) , vous pouvez insérer des caractères spéciaux :

- \n nouvelle ligne
- \r retour chariot
- \t tabulation
- \e escape
- \ le caractere anti-slash
- \" un guillemet
- \xNN le caractere de code ascii NN ( en hexadecimal )
- \nnn le caractere de code ascii nnn ( en octal )
- \c c si c est n'importe quel autre caractere

Voici quelques exemples d'utilisation :

```
\033b pour un texte en gras
\033n pour revenir à un texte normal
\0338 pour mettre le texte en blanc
\033c pour centrer le texte
\033l pour justifier le texte à gauche
\033r pour justifier le texte à droite
```

...

Il est à remarquer qu'il faut rentrer '\"' et non '"' lorsque vous

voulez un guillemet. Sinon vous aurez des problèmes dans le code généré à la compilation.

## 1.32 code

Lorsque vous appuyez sur le bouton 'Code' de la fenêtre principale de MUI-Builder, vous lancez la génération du code de l'interface graphique.

MUI-Builder va alors vérifier que vous n'avez pas défini plusieurs fois la même variable, puis va vous prévenir si c'est le cas.

Sinon, vous allez vous retrouver devant la liste, d'une part de tous les objets que vous avez créés, d'autre part de tous les noms de variables qui seront générés par le programme.

Ces deux listes vont vous permettre de contrôler de manière très précises les variables qui vont être générées par le programme. Si vous n'en voyez pas l'intérêt : ne vous en faites pas, MUI-Builder s'occupe de tout pour vous, sans que vous n'ayez rien à y faire.

Vous devez, avant de générer le code, définir les options  
Les boutons disponibles dans cette fenêtre sont :

- Code App
- Code Objet
- Effacer Label
- Ajout Label

Le programme génère actuellement un fichier-programme 'générique' qui correspond plus à une description d'un programme qu'à un source réel, d'ailleurs ...

Lorsque ce source est généré ( fichier temporaire dans T: ), MUIBuilder appelle l'un des modules de création de source ( actuellement un module de génération de code C, et un de code E ) qui va utiliser le fichier temporaire pour créer le source dans le langage cible.

Par conséquent, MUIBuilder devrait pouvoir, dans l'avenir, générer du code source dans tous les langages où MUI est utilisable.

Le module de génération de l'assembleur est en cours de réalisation

Par contre : si VOUS désirez un langage bien spécifique, et que vous vous sentez capable de créer le générateur de code pour ce langage bien particulier, CONTACTEZ-MOI !!!! : je vous expliquerai dans le détail ce qui constitue le code générique et comment vous y prendre pour la création automatique de votre code.

Grâce au panneau de configuration ( bouton 'Config' ) vous pouvez déterminer le type de langage dans lequel sera généré le code. Pour l'instant, sont disponibles :

```
@{ " Langage C " link Langage_C }  
@{ " Langage E " link Langage_E }
```

### 1.33 options

Quatres options vous sont proposées :

- 'Déclarations' : si cette option est activée, vous obtiendrez la déclarations des variables et leur initialisation dans le code généré.
- 'Environnement' : détermine si vous voulez ou non tout ce qui concerne les includes, la boucle de gestion des évènements, la déclaration de la procédure ... etc ...
- 'Code' : indique que vous voulez la génération du code MUI.
- 'Locale' : montre que vous voulez que MUIBuilder vous génère un code localisé.  
( cf @{ " Localisation " link Localisation } )

Il est à remarquer que chacune de ces options peut être validée complètement indépendamment des autres ... permettant ainsi de générer exactement la partie de code que l'on veut.

Prenons un exemple bien concret :

Supposons que vous ayez créé une fenêtre et que vous désiriez rajouter un bouton dans votre code : vous sélectionnerez alors uniquement 'Code' et générerez le code pour le bouton uniquement. Puis apres avoir inséré ce texte dans votre programme, vous aurez besoin de rajouter la déclaration uniquement : sélectionnez alors uniquement 'Declaration' ... et vous insérez directement le code généré à l'endroit adéquat dans le texte de votre programme.

### 1.34 code-application

Ce bouton vous permet de générer le code pour toute l'application.

Bien sur, le code généré dépend des options que vous aurez choisies.

### 1.35 code-object

Ce bouton vous permet de générer le code de l'objet sélectionné dans la liste 'Objects labels'.

Bien sur, le code généré dépend des options que vous aurez choisies.

---

## 1.36 remove-label

Vous pouvez grâce à ce bouton, indiquer au programme de ne pas générer le nom de la variable que vous avez sélectionnée dans la liste 'Generated Labels'.

MUI-Builder sait automatiquement si, oui ou non, il doit générer un nom de variable. Ainsi, il est par exemple inutile ( dans la majorité des cas ) de garder une variable liée à un groupe MUI, à moins d'avoir besoin de rajouter dynamiquement des objets dans ce groupe durant l'exécution du programme.

Ce bouton, ainsi que le bouton `Add Label` permettent de changer les définitions standards de MUI-Builder sur certains objets.

## 1.37 add-label

Vous pouvez, grâce à ce bouton, indiquer au programme de générer une variable pour un objet donné.

Voir également `Remove Label`

## 1.38 Localisation

Lorsque vous validez l'option 'Locale' dans la fenêtre de génération de code, vous choisissez la génération d'un code localisé.

En fait toutes les chaînes de caractères et les raccourcis claviers seront remplacés par des appels à une fonction de type 'GetString' dont le rôle est de chercher une chaîne localisée dans le catalogue de votre programme.

De nombreux programmes peuvent d'ailleurs générer cette fonction pour vous ( je citerai ici CatComp, ainsi que Flexcat ).

Le nom de la fonction GetString peut être déterminé en modifiant la chaîne qui apparaît dans le gadget situé dans la fenêtre de code ou dans la fenêtre de configuration ( cf @{ "Configuration"link ↔ Configuration } )

A ceux qui ne connaissent pas très bien comment s'y prendre, je conseille de jeter un coup d'oeil à l'un des exemples intégrés dans l'archive de MUIBuilder.

Le fichier de description du catalogue peut être généré en cliquant sur le bouton @{ "Catalogue" link Catalogue } de la fenêtre de génération de code.

## 1.39 Catalogue

---

Le nom du fichier '.cd' doit être entré dans la fenêtre de code avant d'effectuer sa génération.

Je vais ici faire un bref résumé sur ce qu'il est nécessaire de savoir à propos du fichier de description du catalogue ( fichier 'xxxxx.cd' ).

Dans ce fichier, vous trouverez toutes les chaînes de votre programme dans la langue par défaut.

Grâce à ce fichier, vous allez pouvoir créer automatiquement ( grâce à [ par exemple ] CatComp ou Flexcat ) :

- le fichier de traduction ( fichier 'xxxxx.ct' ) dans lequel se trouve les chaînes correspondant à une langue différente de la langue par défaut

- les fichiers programmes correspondant à votre langage favori

( veuillez vous référer à la documentation de ces programmes pour plus de détails )

Le fichier de description de catalogue utilise la convention suivante pour les raccourcis claviers :

\_e Exemple

signifie : Exemple

## 1.40 Langage C

Un certain nombre de choses doivent être remarquées à propos du code généré :

- Certains objets nécessitent des variables auxiliaires, autres que celles qui les définissent directement. Ainsi, un texte doit référer à une chaîne de caractères pour définir son contenu. De la même manière, une liste de type FloatText doit également pointer sur une chaîne de caractère.

Ces variables auxiliaires sont générées et initialisées par le programme et portent, pour les chaînes de caractères le nom 'STR\_"label\_de\_l'objet"'.

- lors de la génération d'un code localisé, vous allez avoir en entête de votre programme un include du fichier header contenant la définition de toutes les chaînes : le nom de ce fichier est construit en utilisant le nom du catalogue.  
exemple : si votre catalogue s'appelle 'MUIBuilderStrings.cd',  
en début de fichier vous trouverez '#include MUIBuilderStrings.h'

D'autre part, deux modules de génération pour le langage C sont fournis avec MUIBuilder :

---

- @{ "GenCodeC\_Old" link GenCodeC\_Old } : reprend les avantages et ↵  
inconvenients  
du générateur de la version 1.0.
- @{ "GenCodeC" link GenCodeC } : un nouveau générateur qui va vous  
permettre de réaliser du code séparé pour l'interface graphique.
- AUCUNE notification n'est effectuée dans le programme généré. Vous  
devez donc, si vous voulez tester rapidement le programme, au  
moins ouvrir une fenêtre avant la boucle événementielle.  
( ce qui s'effectue avec set( WI\_window, MUIA\_Window\_Open, TRUE ) )  
Si vous faites uniquement cette modification, vous devez sortir  
du programme en utilisant la commodité 'Exchange'.

## 1.41 gencodec\_old

Ce module inclut en début de fichier le contenu du fichier  
'C-header\_Old', qui doit demeurer dans le répertoire Modules, mais que  
vous pouvez modifier à votre guise, de manière à ce que MUIBuilder  
génère EXACTEMENT ce que VOUS voulez !!

Sinon, il utilise toujours les mêmes principes que le module C  
inclut dans l'archive de MUIBuilderV1.0.

## 1.42 gencodec

Ce module de génération est de loin le plus intéressant des deux !

Grâce à lui, vous allez créer un fichier d'interface indépendant  
de votre programme.

Il inclut en début de fichier le contenu du fichier 'C-header',  
qui doit demeurer dans le répertoire Modules, mais que  
vous pouvez modifier à votre guise, de manière à ce que MUIBuilder  
génère EXACTEMENT ce que VOUS voulez !!

Le générateur va créer deux fichiers :

- un fichier .c contenant deux procédures :
  - CreateObject qui vous renvoie un pointeur sur une  
structure.
  - DisposeObject qui libère la mémoire de l'objet créé
- un fichier .h que vous devrez inclure dans votre programme,  
qui contient la définition d'une structure dont les champs  
sont les noms des objets MUI que vous avez créés dans votre  
interface.

Ces fichiers s'utiliseront de la manière suivante :

```
#include <libraries/mui.h>
```

---

```

#include <clib/muimaster_protos.h>
#include <clib/alib_protos.h>
#include <clib/dos_protos.h>
#include <clib/exec_protos.h>

#include <pragmas/muimaster_pragmas.h>
#include <pragmas/exec_pragmas.h>

#include "essai.h"      /* ---- Généré par MUIBuilder ---- */

struct Library * MUIMasterBase;

main()
{
    struct ObjApp * app;
    BOOL  running = TRUE;
    ULONG signal;
    extern struct Objapp * CreateSmall_Example( void );

    Init();      /* Vos initialisations */
                /* ouverture de librairies ( locale, MUI ... ) */

    app = CreateApp();      /* ---- Généré par MUIBuilder ---- */

                /* Incluez ici toutes vos notifications */

    set( app->WI_main, MUIA_Window_Open, TRUE );      /* Ouverture d'une fenêtre */

    while (running)      /* Boucle principale du programme */
    {
        switch (DoMethod(app->app,MUIM_Application_Input,&signal))
        {
            case MUIV_Application_ReturnID_Quit:
                running = FALSE;
                break;
        }
        if (signal) Wait(signal);
    }
    DisposeApp(app);      /* ---- Généré par MUIBuilder ---- */
}

```

## 1.43 Langage E

Veuillez vous reporter à la documentation séparée.

Merci.

## 1.44 guide

Grâce à MUI-Builder, plus personne ne pourra plus avoir de raisons valables pour réaliser ce que l'on trouve sur toutes les autres machines ( y compris les PC, ce qui n'est pas peu dire !!! ), à

savoir une aide en ligne, une documentation hypertexte de votre programme.

En effet, vous pouvez attacher à tout objet MUI créé depuis MUI-Builder un petit texte d'aide.

Ensuite, MUI-Builder générera automatiquement une documentation hypertexte au format AmigaGuide ( que vous pourrez même admirer depuis MUI-Builder grâce au bouton 'Voir Doc' ).

Au même titre que la génération de code, vous pouvez ne générer qu'une partie de la documentation pour l'insérer directement dans une documentation déjà écrite.

Ainsi, si vous effectuez des modifications dans la documentation générée par MUI-Builder, elle ne seront pas perdues : vous ne générerez que la partie du texte concernant les éléments de l'interface que vous avez modifiés, et vous les insérerez dans le document.

Vous pouvez éditer le texte de la documentation

- le texte principal avec 'Noeud Appli'
- le texte pour une fenêtre avec 'Noeud fenêtre'
- le texte d'un objet avec 'Noeud Objet'

Vous pouvez générer la documentation :

- pour toute l'application avec 'Générer tout'
- pour une fenêtre avec 'Générer fenêtre'
- pour un objet avec 'Générer Objet'

En double-cliquant sur un nom d'objet ou de fenêtre, vous pouvez éditer le titre du texte d'aide correspondant, et voir le texte attaché à l'objet. ( même chose avec les deux boutons 'Editer' ).

## 1.45 L'enregistrement

Rien de plus simple que de s'enregistrer !

MUI-Builder est GiftWare.

Ce qui signifie que vous n'êtes absolument pas tenu de verser quelque chose à l'auteur pour continuer à utiliser ce logiciel.

Si vous aimez ce logiciel, qu'il vous est ou vous a été particulièrement utile, alors vous pouvez m'envoyer un modeste don ( de l'ordre de 50FF ou 15\$US ) à l'adresse suivante :

Eric Totel  
5 rue Riquet  
31000 Toulouse  
France

A tous ceux qui ne se sentent pas assez philanthrope pour envoyer ce genre de don ( les étudiants fauchés par exemple ! ), je suggère de m'envoyer une simple lettre, une carte postale, un mail ( adresse : totel@laas.fr ), pour m'encourager, me donner vos impressions sur le logiciel, vos

---

idées ... etc ...

N'HESITEZ surtout PAS : je veux absolument savoir si MUI-Builder est utilisé ou apprécié. En particulier, j'aimerais recevoir beaucoup plus d'avis de la part de francophone ... Il semblerait que la dynamique Amiga, si j'en crois tout le courrier reçu, se développe bien plus à l'étranger qu'en France !!!! :-)

D'autre part, si vous avez traduit la documentation dans votre propre langue ( ou une autre qui n'est pas fournie dans l'archive actuelle de MUIBuilder ) ... alors ( s'il vous plait ) envoyez moi ce que vous avez fait de toute urgence !!!!

## 1.46 Futur

Parmi les améliorations futures de MUI-Builder, il est prévu d'implémenter :

- La notification ( génération du code correspondant aux évènements ) sera directement implémentable à partir de MUIBuilder.
- La possibilité d'échanger des données avec des éditeurs de texte par l'intermédiaire du port AREXX, afin d'insérer directement du code généré par MUI-Builder dans votre programme.
- Toutes les modifications que vous auriez envie de voir apporter à MUI-Builder ( n'hésitez surtout PAS à me faire parvenir TOUTES vos idées et remarques, de manière à ce que le programme évolue suivant vos besoins ).

## 1.47 Remerciements

Je voudrais remercier ici tous ceux qui m'ont aidé et ont contribué à la réalisation de MUI-Builder :

- Pierre Carrette pour son aide immodérée et ses idées plus intéressantes les unes que les autres !!! :-)
  - Lionel Vintenat pour l'idée du code générique et la réalisation du générateur de code pour le langage E.
  - Gael Marziou, Pascal Pensa, Christian Brandel, Daniel Murrel pour les tests attentifs qu'ils ont réalisés, et pour toutes les ( nombreuses ) idées qu'ils m'ont suggéré.
  - Pascal Rabier et Mike Manzano pour m'avoir aidé dans la traduction de la documentation anglaise.
  - Christian Brandel pour ses brillantes traductions, d'une part de la documentation, d'autre part du catalogue Allemand !
-

- Tobias Ferber pour son superbe icône MagicWB. ( non, non ... pas superbe : FANTASTIQUE !!!! :-) )
- Michal Kara pour sa traduction du catalogue en Checkoslovaque, et la réalisation du module externe pour le langage d'assemblage 680X0.