MUIBuilder

version 2.2
Documentation française

Eric Totel

1 Introduction

Tout d'abord merci d'utiliser MUIBuilder!

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

MUIBuilder est un outil qui va vous permettre d'écrire des applications **MUI**, sans avoir à taper des lignes et des lignes de code, et sans avoir à connaitre la syntaxe (somme toute relativement simple) de **MUI**. Grâce à MUIBuilder, vous allez enfin pouvoir créér vos interfaces graphiques sans la moindre arrière pensée, et sans plus d'effort que de réfléchir à ce que vous voulez réaliser.

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

1.1 Avantages de MUIBuilder

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

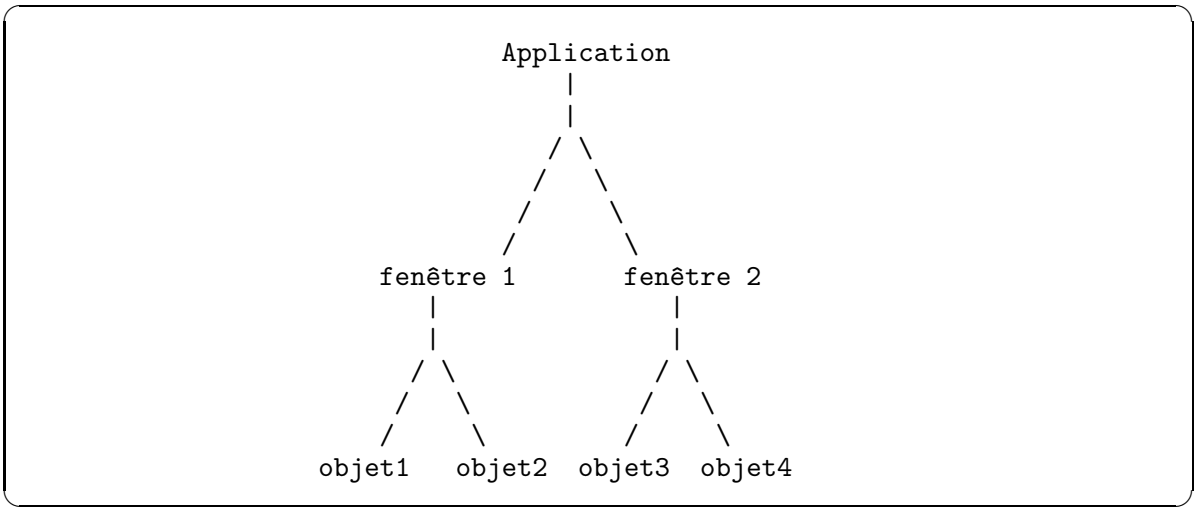
A tous ces gens j'expose ici les avantages que les utilisateurs ont pu mettre en évidence.

1. MUIBuilder est, pour les débutants, LE moyen de maitriser 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. MUIBuilder se comporte, pour ceux qui maitrisent déjà MUI, comme une sorte d'interpréteur du langage associé à MUI. Vous pouvez directement tester l'allure de votre graphique pour connaitre à l'avance le résultat. Tous ceux qui ont utilisé régulièrement MUI se sont aperçus en effet de certaines limitations (en particulier en ce qui concerne le dimensionnement et la taille fixe de certains objets qui affectent parfois le look de l'interface).
3. MUIBuilder 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!
6. Dans cette nouvelle version apparaissent de nouvelles fonctionnalités complètement nouvelles dans le monde de l'Amiga. En effet, il est désormais possible de gérer les notifications d'évènements depuis MUIBuilder, ET DE LES TESTER depuis le générateur d'interface.

2 Principes

MUIBuilder 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 referer à la documentation de MUI, pour mieux appréhender le principe. MUIBuilder 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 composent. Ainsi, une application pourra être schématiquement représentée par un arbre similaire à celui figurant sur cette page.



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

3 Distribution

Cette distribution est constituée d'une archive au format lharc. Cette archive ne doit en aucun cas être altérée, ou modifiée. Ce programme ne doit EN AUCUN CAS être distribué sous une autre forme, sans l'autorisation écrite de l'auteur.

D'autre part, ce programme ne doit jamais être distribué pour un prix supérieur à celui d'une disquette + frais de ports (c'est-à-dire environ 12FF, 3 $-US, 4.DM). Dans tous les cas de figures, AUCUN bénéfice ne peut et ne doit être retiré de cette vente.

MUIBuilder est un logiciel Giftware[1].

Ce qui signifie que vous n'êtes absolument pas tenu de verser quelque chose à l'auteur pour continuer à utiliser ce logiciel. Sachez également que si vous avez payez une somme substancielle pour obtenir ce logiciel (somme ne devant pas normalement dépasser les montants indiqués ci-dessus), je n'en bénéficie d'aucune manière.

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
26 route de Montsuzain
10150 Voue
France
E-Mail  : Eric.Totel@Ramses.fdn.org
Fidonet : 2:320/104.56
```

A tous ceux qui ne se sentent pas assez philantrope pour envoyer ce genre de don (les étudiants fauchés par exemple! :-)), je suggère de m'envoyer une simple lettre, une carte postale, pour m'encourager, me donner vos impressions sur le logiciel, vos idées ... etc ... Certaines personnes ont eu l'excellente idée de m'envoyer les programmes qu'elles ont elles-mêmes réalisés : je trouve cette initiative excellente et je dois avouer que je l'encourage !!! J'apprécie énormément ce retour de la

[1] Cadeaugiciel pour les francophiles :-)

part des utilisateurs ! J'apprécierais encore plus si vous pouviez indiquer dans votre documentation que je vous avez utilisé MUIBuilder durant le développement de votre programme.

S'il vous plait, n'envoyez pas de fichiers à mon adresse e-mail !! (catalogues, fichiers .MUIB, générateurs de code ...) Envoyez d'abord un message et je vous ferai savoir où il vous sera possible de l'envoyer.

4 Garantie

Aucune garantie n'est donnée quand au bon fonctionnement de ce programme. Ce programme est distribué 'tel quel' sans aucune garantie d'aucune sorte, qu'elle soit explicite ou implicite, y compris (mais sans qu'elles puissent être considérées comme une limite) les garanties implicites généralement attachées à un produit. Tous les risques inhérents à la qualité de ce programme ou à ses performances sont à votre charge. Si celui-ci devait se trouver être défaillant, vous devrez assumer vous-même tous les services de réparations et corrections nécessaires.

En aucun cas l'auteur, ou tout autre tiers impliqué dans la distribution de ce produit dans les limites des contraintes imposées pour la distribution, ne pourront être tenues responsables des dommages généraux, spécifiques, accidentels ou conséquents de l'utilisation de ce logiciel (incluant la perte de données, la détérioration de données, des défaillances du logiciel, et tout autre type de défaillance), même si l'auteur ou un tiers tel qu'il est défini précédemment a été averti de la possibilité de tels dommages.

En conséquence tout utilisateur se sert de MUIBuilder à ses risques et périls ...

5 Tutorial

De manière à ce que les débutants puissent rapidement comprendre comment le logiciel fonctionne, je me permets d'inclure ici la marche à suivre pour la réalisation d'un petit exemple.

Nous allons réaliser pas-à-pas une interface relativement simple dont le but sera d'afficher un simple message dans un objet texte lorsque l'on appuiera sur des boutons.

- Tout d'abord créons une fenêtre (bouton `Nouv. fenêtre` dans la fenêtre principale).
- Sélectionnez l'unique groupe présent dans la liste des objets de l'interface, et choisissez `Ajouter objet` parmi les boutons disponibles.
- Choisissez un objet Texte, définissez ses paramètres (inutile de changer quoi que ce soit en fait pour l'instant) et appuyez sur `Ok`.
- Votre objet se retrouve inséré dans votre interface
- Maintenant, reproduisez cette opération en ajoutant un groupe dont vous mettrez l'attribut `horizontal` à Vrai. (groupe horizontal)
- Si tout se passe bien vous vous retrouvez avec un groupe en dessous du gadget texte : sélectionnez le, et ajoutez lui successivement trois boutons, en définissant leurs titres successivement par '1er bouton', '2eme bouton', '3eme bouton' et en leur affectant des raccourcis claviers '1', '2', et '3'.
- Votre interface est désormais terminée. Maintenant nous allons définir les notifications sur les objets.
- Sélectionnez le premier bouton, et appuyez sur `Notifier`
- Dans la fenêtre qui apparait, choisissez l'évènement `Relâcher bouton`, puis l'objet Texte que vous avez créé, et enfin l'action `Mettre une valeur constante`. Ajouter cette notification en double-cliquant sur l'action, ou en appuyant sur `Ajouter Notification`. MUIBuilder va vous demander la chaine de caractères qui doit être écrite dans le gadget texte. Entrez simplement '1er bouton sélectionné'. Une fois tout ceci réalisé, sortez de la fenêtre des notifications.
- Répétez l'opération précédente sur les deux autres boutons, en définissant la chaine constante respectivement par '2eme bouton sélectionné' et '3eme bouton sélectionné'.
- Une fois de retour dans la fenêtre de création, toutes ces opérations étant réalisées, appuyez sur le bouton `Test`, et appuyez sur les boutons de votre interface ... Amusant, non ?? :)

Bien sûr, cette interface était particulièrement petite et simple à réaliser, mais en fait, l'ensemble des possibilités que vous offre MUIBuilder devrait vous permettre de réaliser l'ensemble de vos GUI tout aussi rapidement!

6 Utilisation de MUIBuilder

Lors de l'utilisation de MUIBuilder, de nombreuses possibilités s'offrent à vous, tant du point de vue de la configuration que des possibilités propres au logiciel. Ces possibilités vont être énumérées dans les paragraphes qui suivent.

6.1 Configuration de MUIBuilder

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 :

1. 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).
2. Icônes : si vous sélectionnez cette option, les sauvegardes seront accompagnées d'une 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.
3. Requêtes : indique si vous voulez ou non des boites de requêtes à chaque fois que vous faites une action importante (effacement d'un objet par exemple).
4. Test : indique que vous voulez voir le résultat de votre travail pendant que vous procédez à la construction de votre interface.
5. Code : indique quel générateur de code vous voulez utiliser.
6. 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

7. GetString : ATTENTION ceci est IMPORTANT!!! Si vous rentrez une chaine ici, elle désignera le nom de la procédure 'GetString' utilisée pour la localisation. Cette chaine sera copiée dans le gadget chaine 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 chaine est laissée VIDE, alors le nom par défaut sera construit à partir du nom de votre application ('GetApplicationNameString').
8. Profondeur : permet de définir la profondeur minimale de l'affichage de l'arbre de la fenêtre de création principale lors de son ouverture.
9. caractère : ce caractère est inséré lors de l'affichage des arbres, de manière à aider à la visualisation.
10. Label : Si cette option est positionnée, un label désignant le nom de l'objet (et indiquant son raccourci clavier) est inséré devant l'image de chaque objet, dans la fenêtre de sélection des objets disponibles.
11. Colonnes : Indique de quelle manière les images des objets vont être organisées dans la fenêtre. Vous pouvez adapter ce choix en fonction de la taille de votre écran, et de la taille des images que vous aurez choisies.
12. Location images : Le répertoire dans lequel se trouvent les images représentant les objets.

Une fois les options positionnées, vous pouvez les sauver en sélectionnant le bouton correspondant! Il faut remarquer que lors de la sauvegarde, le nom du répertoire dans lequel vous avez effectué vos dernières lectures ou sauvegardes de répertoire sera lui-même sauvegardé.

6.2 La sauvegarde d'une application

Les boutons 'Sauver' et 'Charger' vous permettront respectivement de sauvegarder et récupérer l'interface de l'application en cours de création. Le requester Asl s'ouvrira automatiquement sur votre répertoire de projet courant.

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!

D'autre part, beaucoup remarqueront que le fichier de sauvegarde peut être directement édité depuis un simple éditeur de texte : je le déconseille, puisqu'il y a de fortes chances que vous rendiez invalide le format du fichier de sauvegarde.

6.3 Génération du code

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

MUIBuilder 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 précise les variables qui vont être générées par le programme et que vous allez pouvoir référencer dans votre application. Si vous n'en voyez pas l'intérêt : ne vous en faites pas, MUIBuilder s'occupe de tout pour vous, sans que vous n'ayez rien à y faire.

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 (fichiers temporaires dans T:), MUIBuilder appelle l'un des modules de création de source qui va utiliser le fichier temporaire pour créer le source dans le langage cible. Ces modules externes utilisent alors les fonctions mises à leur disposition dans la muibuilder.library pour générer le source.

Par conséquent, MUIBuilder devrait pouvoir, dans l'avenir, générer du code source dans tous les langages supportés par MUI (ce qui n'est pas encore tout à fait le cas).

Si VOUS désirez un langage bien spécifique, et que vous vous sentez capable de créér 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. La librairie muibuilder.library permet justement de simplifier la création de tels modules externes.

Vous devez, avant de générer le code, définir les options de génération de code. (voir Section 6.3.1 [Options], page 13)

Grâce au panneau de configuration vous pouvez déterminer le type de langage dans lequel sera généré le code. Pour l'instant, sont disponibles, le langage C, le langage E, l'assembleur, et l'Obéron.

6.3.1 Options de génération du code

Cinq options vous sont proposées :

- `Déclarations`: si cette option est activée, vous obtiendrez la déclaration 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.(voir Section 6.4 [Localisation], page 15)
- `Notifications`: permet de déterminer si oui ou non vous voulez que le code concernant les notifications soit généré.

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 source pour le bouton uniquement. Puis apres avoir inséré ce texte dans votre programe, vous aurez besoin de rajouter la déclaration uniquement : sélectionnez alors uniquement 'Declaration' ... et vous insérez directement par un copy-paste le code généré à l'endroit adéquat dans le texte de votre programme.

6.3.2 Code de l'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.

6.3.3 Code d'un objet

Ce bouton vous permet de générer le code de l'objet sélectionné dans la liste 'labels Objets'. Cette opération est beaucoup plus importante qu'elle n'y parait. En effet, il peut être très utile de créer une fonction générant un objet, pour pouvoir, par exemple, l'inclure dans plusieurs fenêtres. Il peut être également intéressant d'avoir une fonction de création pour chaque fenêtre existante, de manière à allouer et désallouer les fenêtres uniquement lorsque cela est nécessaire.

A nouveau le code généré dépend des options que vous aurez choisies.

6.3.4 Effacer un 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'.

MUIBuilder 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 'Ajouter un label' (voir Section 6.3.5 [Ajouter un label], page 15) permettent de changer les définitions standards de MUIBuilder sur certains objets.

6.3.5 Ajouter un label

Vous pouvez, grâce à ce bouton, indiquer au programme de générer une variable accessible à votre application pour un objet donné. (voir Section 6.3.4 [Effacer un label], page 15)

6.4 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 chaines de caractères et les raccourcis claviers seront remplacés par des appels à une fonction de type 'GetString' dont le rôle est de chercher une chaine localisée dans le catalogue

de votre programme. De nombreux programmes peuvent d'ailleurs générer cette fonction à votre place (je citerai ici CatComp, ainsi que Flexcat).

Le nom de la fonction GetString peut être déterminé en modifiant la chaine qui apparait dans le gadget situé dans la fenêtre de code ou dans la fenêtre de 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' de la fenêtre de génération de code.

6.4.1 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 chaines 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 trouvent les chaines 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)

A tous ceux qui ne connaissent pas encore cet aspect de la programmation sur Amiga, je ne peux que conseiller l'utilisation de Flexcat (écrit par Jochen Wiedmann) qui a l'avantage de s'adapter à tous les langages, et propose une documentation qui devrait vous permettre de comprendre de quoi il retourne.

6.5 Generation de l'aide en ligne

Grâce à MUIBuilder, 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 MUIBuilder un petit texte d'aide. Ensuite, MUIBuilder génèrera automatiquement une documentation hypertexte au format AmigaGuide (que vous pourrez même admirer depuis MUIBuilder 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 MUIBuilder, 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').

Lors de la génération du code, MUIBuilder va générer un attribut MUIA_Help_Node pour tout objet pour lequel vous aurez écrit un petit texte d'aide. Il faut bien remarquer que le fichier généré est bien un fichier destiné à l'aide en ligne : je ne pense pas que l'on puisse le considérer comme une documentation digne de ce nom ...

6.6 Remarques diverses

Un certain nombre de détails n'ont pas encore été abordés dans cette documentation. Vous allez donc trouver dans ce chapitre, dans un ordre tout à fait arbitraire un ensemble de renseignements qui vous aideront à mieux utiliser MUIBuilder.

6.6.1 Caractères spéciaux

Dans les textes que vous entrez (TOUS les textes qui apparaissent à l'écran et non pas seulement les gadgets texte) , vous pouvez insérer des caractères spéciaux. Les conventions utilisées sont les suivantes :

`\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. De plus, '\e' et '\033' sont équivalents

(ils désignent tous deux le caractère d'échappement). L'une des erreurs les plus communément commises est d'utiliser '\33' au lieu de '\033' ...

6.6.2 Aligner des objets

Pour aligner un ensemble d'objets, il suffit de les inclure, soit dans un groupe à plusieurs colonnes (alignement vertical), soit dans un groupe à plusieurs lignes (alignement horizontal).

Certains objets (Slider, CheckMark, String) peuvent être accompagné d'un titre au niveau de MUIBuilder. Ces titres n'existent pas au niveau de MUI, en tant qu'attributs attachés à un objet.. Pour proposer cette option, MUIBuilder construit un groupe horizontal contenant un label, et l'objet dont il est question. Ainsi, si vous utilisez ce titre, vous n'arriverez pas à aligner vos objets. Utilisez donc ces objets sans leur titre dans ce cas, en les incluant accompagnés d'objets Label dans un groupe multi-colonne.

6.6.3 Interface redimensionnable

Une question qui revient souvent est la suivante: 'après avoir ajouté un certains nombre d'objets dans mon interface, il n'était plus possible de la redimensionner'. A cette question : une seule réponse : l'objet `Espace`! Insérez des objets `Espace` dans votre interface, et elle deviendra dimensionnable! Prenons un exemple : vous avez mis dans votre interface un checkmark et son titre : la checkbox n'est pas redimensionnable horizontalement, de meme que le Label qui lui sert de titre. Dans ce cas, insérez votre objet dans un groupe horizontal en l'encadrant de deux espaces : ce groupe sera alors redimensionnable, au contraire de l'objet checkmark seul !

7 Les objets

MUIBuilder permet de créer tous les objets que vous voudrez bien attacher à votre application. Ils peuvent être créés ou modifiés depuis la fenêtre principale de création de fenêtres.

remarque : il suffit de double-cliquer sur un objet pour être en mesure de le rééditer.

7.1 Objet 'Aire'

Tous les objets qui, dans MUI, héritent de l'objet Area, se voient, à partir de la version 2.0 de MUIBuilder, attacher un RegisterGroup contenant les attributs de cette classe qui peuvent être positionnés à l'initialisation de l'objet. Dans la version française 'Area' est devenu 'Aire' ... Cette traduction n'est à mon avis pas forcément heureuse, mais elle a au moins le mérite d'être claire !

- `Cacher` : l'objet n'apparaitra pas dans l'interface, sauf s'il est seul
- `Désactiver` : l'objet sera désactivé à l'ouverture de la fenêtre. Il apparaitra grisé dans l'interface
- `Mode Input` : l'objet peut être sélectionnable comme un bouton
- `Fantôme` : le pourtour de l'objet sera invisible.
- `Frame` : permet de choisir le pourtour d'un objet
- `Fond` : indique le type de fond que vous voulez dans un objet
- `Caractère de contrôle` : positionnez ici le caractère qui vous permettra d'activer l'objet par l'intermédiaire du clavier
- `Titre` : ce titre viendra se positionner sur le haut du pourtour de l'objet si il existe.

Pour certains objets, certains de ces attributs ne seront pas accessibles : ce n'est pas une erreur dans le programme, j'ai simplement essayé d'empêcher que l'on puisse réaliser des interfaces complètement délirante (par exemple un objet bouton avec un pourtour de chaine). Cette philosophie n'engage que moi, et certaines des personnes qui ont eu le mérite de tester cette version dans sa phase bêta ne semblent pas tout à fait d'accord sur le principe, alors que d'autres poussent à encore plus de restrictions ... J'ai essayé de faire la part des choses : toute remarque est bien sûr la bienvenue !

7.2 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 MUIBuilder, 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

A l'application peut être affectées des notifications ainsi que des menus accessibles par leurs boutons respectifs.

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

7.3 Les fenêtres

Vous savez tous déjà ce qu'est une fenêtre! Notre problème va être d'apprendre à la construire grâce à MUIBuilder ...

Deux pages se partagent l'espace réservé : l'une nommée 'Création', l'autre 'Attributs' :

1. La page de Création comporte 2 listes:
   - Dans la première vous réalisez votre interface graphique elle-même. La représentation s'effectue de manière hiérarchique sous la forme d'un arbre. Pour plier ou déplier une partie de l'arbre, il suffit de double-clicker sur la petite flèche se trouvant à la gauche du nom de l'objet. Grâce aux boutons entre les deux listes, vous pouvez éditer/copier/bouger les objets de l'interface. De plus les menus proposent deux entrées `Plier` et `Déplier` qui permettent respectivement de replier complètement la hiérarchie de l'arbre, et de de la déplier sur toute sa profondeur. A gauche du nom de chaque objet se trouve deux lettres

'H' et 'G'. Leur signification est issue directement de l'anglais : 'H' pour 'Help' et 'G' pour 'Generate'. Si le 'H' est présent, le noeud correspondant à l'objet sera généré dans la documentation amigaguide. Si le 'G' est présent, le label correspondant à l'objet sera généré dans les objets que vous pourrez référencer dans votre code.

- La liste de droite permet, elle, de stocker temporairement des objets. (voir Section 7.3.1 [Liste temporaire], page 23).

Les Groupes (voir Section 7.4 [Groupe], page 23) constituent les éléments de base dans lesquels vont être intégrés tous les autres objets (voir Chapter 7 [Objets], page 21). Ainsi, à chaque fois que vous ajouterez un objet, vous devrez le déclarer comme étant le fils du groupe que vous aurez précédemment sélectionné.

2. 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
     - `Ouvrir à la création`: (IMPORTANT) cet attribut vous permet de définir si wous voulez que votre fenêtre s'ouvre lorsqu'elle est créée. Cet attribut est pris en compte lorsque vous testez votre application depuis MUIBuilder. Toutefois, pendant l'édition d'une fenêtre donnée celle-ci s'ouvrira toujours pendant le test.

7.3.1 Liste temporaire

Cette liste est persistante, c'est-à-dire qu'elle n'est pas effacée lorsque vous changez d'application, ni d'ailleurs lorsque vous effacez l'application courante. Vous pouvez donc y amener des objets, afin de les passer d'un groupe à l'autre, ou d'une fenêtre à l'autre.

7.4 Groupe

Lorsque vous créez une nouvelle fenêtre, un groupe lui est déja attaché : le groupe principal, encore nommé 'groupe racine'. Vous allez ensuite construire votre fenêtre en attachant des objets

(voir Chapter 7 [Objets], page 21) à ce groupe racine. En termes de structure d'arbre, un groupe constitue un noeud de l'arbre. Parmi les fils d'un groupe pourront, bien entendu, exister d'autres groupes qui, eux-mêmes, contiendront d'autres fils ... etc ...

Vous devrez, pour chaque groupe, définir les attributs suivants:

- `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 apparaitre 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
- `Colonnes`: affiche le groupe sous forme de colonnes (le nombre de colonnes devant être entré dans la chaine 'nombre'). 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` L'espacement est à rentrer dans la chaine correspondante.
  - Horizontal : permet de définir l'espacement horizontal entre les objets
  - Vertical : permet de définir l'espacement vertical entre les objets

L'objet groupe hérite de l'objet Aire (voir Section 7.1 [Objet Aire], page 21).

7.5 Liste

Pour définir complètement une liste, il suffit de donner :

- Son label dans le programme C généré.
- Si le 'double click' est autorisé dans cette liste.
- Si la multi-sélection est autorisée dans cette liste.
- Si la liste est sélectionnable ou pas (mode input)
- Si la taille de la liste doit être ajustée verticalement à sa création
- Si la taille de la liste doit être ajustée horizontalement

- Le positionnement de la liste lors de sa création (attention: il s'agit bien de positionnement, pas de sélection !) :
  - Pas de positionnement spécifique
  - Positionnement au niveau de la première entrée
  - Positionnement au niveau de la dernière entrée
- Le type de la liste, qui peut être :
  - une liste standard
  - une liste de floattext, permettant d'afficher un texte banal. Dans le cas où vous choisissez un floattext, vous pouvez entrer la chaine qu'il contiendra dans le gadget chaine 'Texte flottant'.
  - une liste de 'Volumes', qui contient tous les assigns et volumes de votre configuration.
- Les hooks[1] de fonctions peuvent être définis à partir de MUIBuilder, mais ne sont pas nécessaires au bon fonctionnement d'une liste ordinaire.
  - Construction : appelée lors de l'insertion d'un objet dans la liste
  - Destruction : appelée lors de la destruction d'un objet de la liste
  - Comparaison : appelée lors du tri de la liste, pour comparer les objets qu'elle contient
  - Visualisation : appelée lors de l'affichage d'un objet de la liste
  - Multitest : appelée lorsque vous faites une multisélection dans une liste
- Le format de la liste (permet d'utiliser des listes multi-colonnes). Se reporter à la documentation de MUI our plus de détails.
- Le titre de la liste.

Il est à remarquer qu'il existe un autre type de liste, appelé DirList (voir Section 7.6 [Liste de répertoires], page 25) . Ce type de liste peut être créé indépendamment en cliquant sur son bouton dans la fenêtre de choix des objets.

L'objet liste hérite de l'objet Area (voir Section 7.1 [Aire], page 21)

7.6 Liste de fichiers et répertoires

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

---

[1] 'Hook' peut être traduit en Français par 'Crochet' : il s'agit en fait en programmation d'une structure désignant une fonction et utilisée pour la référencer

Les options possibles pour cet objet sont les suivantes :

- `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)
- `Tri du répertoire` : trois possibilités
  - Les répertoires sont positionnés en tête de liste
  - Les répertoires sont positionnés en queue de liste
  - Les répertoires sont mélangés avec les noms de fichier (suivant le critère `Type de tri`)
- La chaine 'Répertoire' doit contenir le répertoire de lecture pour la DirList.
- Accepter : la pattern à accepter
- Rejeter : la pattern à rejeter

L'objet 'Liste de répertoires' hérite de l'objet Aire (voir Section 7.1 [Aire], page 21).

7.7 Chaine

Le gadget Chaine est défini par les données suivantes :

- Le titre qui apparaitra directement à gauche du gadget string.
- le label de la variable associée au gadget chaine.
- le contenu initial du gadget chaine lors de l'affichage.
- une chaine de caractères contenant les caractères acceptés par votre gadget chaine.
- une chaine de caractères contenant les caractères refusés par votre gadget string.
- la longueur maximale de la chaine qui devra être acceptée.

L'objet Chaine hérite de Area. (voir Section 7.1 [Aire], page 21).

7.8 Bouton

Pour définir complètement un bouton, il suffit de donner :

- Le label que portera le bouton dans le programme
- Le texte qui apparaitra dans le bouton

et bien sûr tous les paramètres liés aux objets 'Area' de MUI. (voir Section 7.1 [Aire], page 21).

7.9 Label

Pour définir complètement un label, il faut donner :

- son label dans le programme
- le contenu du texte constituant le label

L'objet Label hérite de Area. (voir Section 7.1 [Aire], page 21).

7.10 Gadget Cycle

Pour définir complètement un cycle, il faut donner :

- la liste des entrées du cycle
- le label du gadget cycle dans le programme

Si vous affectez un caractère de contrôle à un gadget cycle, il est conseillé de l'associer à un label possédant le même caractère de contrôle, de manière à ce que ce dernier soit mis en évidence dans l'interface graphique.

L'objet Cycle hérite de Area. (voir Section 7.1 [Aire], page 21).

7.11 Boutons Radios

Pour définir complètement des boutons radios, il faut :

- la liste des boutons radios : l'ajout, le retrait, le renommage des éléments de cette liste s'effectue de manière relativement intuitive grâce aux boutons se trouvant sur sa droite.
- le label du gadget radio dans le programme

L'objet Radio hérite de Area. (voir Section 7.1 [Aire], page 21).

7.12 CheckMark

Vous pouvez, pour les CheckMarks (comme pour les chaines et les sliders) indiquer si un titre doit figurer ou non juste devant le CheckMark.

Si ce titre doit apparaitre, vous devez l'indiquer dans la chaine 'Title'.

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

L'objet CheckMark hérite de Area. (voir Section 7.1 [Aire], page 21).

7.13 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.

Deux points à noter dans cette version 2.2 :

- un gadget popasl permet de saisir un nom de fichier : si ce nom existe, il doit etre celui d'une image correspondant à un type supporté par les datatypes installés sur votre système.
- cette image, parfois, ne sera pas visualisée pendant les tests : désolé, apparemment je n'y suis pour rien. Le code généré sera par contre toujours correct.

L'objet Image hérite de Area. (voir Section 7.1 [Aire], page 21).

7.14 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 apparaitre devant le slider

Il faut alors entrer :

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

vous pouvez également remplir :

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

L'objet Slider hérite de Area. (voir Section 7.1 [Aire], page 21).

7.15 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' : on indique alors la valeur de division
- le maximum que peut atteindre la jauge
- le texte d'information de la jauge : ce petit texte va se placer à l'intérieur de la jauge, de manière à produire une information. On peut y faire figurer le niveau de la jauge en incluant `%ld` dans la chaine de caractères.
- Le label de la jauge dans le programme généré

L'objet Jauge hérite de Area. (voir Section 7.1 [Aire], page 21).

7.16 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.
- `Label` : Le nom du label de l'objet

Le texte doit être entré dans le gadget chaine correspondant et peut comporter tous les caratères spéciaux (voir Section 6.6.1 [Caractères spéciaux], page 17) que l'on peut en particulier trouver dans des sources C.

L'objet Texte hérite de Area. (voir Section 7.1 [Aire], page 21).

7.17 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). Remarquez qu'il y a un bug dans MUI actuellement : cet objet ne fonctionne que dans son état horizontal.

L'objet Règle hérite de Area. (voir Section 7.1 [Aire], page 21).

7.18 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.

L'objet Proportionnel hérite de Area. (voir Section 7.1 [Aire], page 21).

7.19 Espace

Cet objet sert uniquement à insérer un espace entre deux autres objets, afin de permettre le redimensionnement de la fenêtre. Il peut être de trois types :

- Horizontal
- Vertical
- Horizontal et vertical à la fois

Aux deux premiers types on peut associer la taille de l'espacement.

7.20 PopAsl

L'objet PopAsl vous permet d'inclure des requesters standards dans votre interface. Utilisez les, et abusez d'eux ! Les attributs disponibles sont:

- L'image popup que vous désirez
- Le type du requester, à savoir:
  - Requester de fichiers
  - Requester de fontes
  - Requester de mode d'écran
- Les hooks disponibles:
  - Début : la fonction appelée à l'ouverture du requester
  - Fin : la fonction appelée à la fermeture du fichier
- Le label de l'objet PopAsl

L'objet PopAsl dépend de Area (voir Section 7.1 [Aire], page 21).

7.21 Objet PopUp

L'objet popasl vous permet d'associer à une liste et une image popup un objet qui apparaitra lorsque vous avtiverez cette image. Cet objet peut-être de tout type (y compris un autre objet PopUp!!). L'objet en question apparaitra dans la hiérarchie comme fils du PopUp, et pourra être édité comme n'importe quel autre objet. En particulier s'il s'agit d'un groupe, vous pourrez lui ajouter des fils, comme s'il était un simple objet de l'interface.

Pour définir l'objet PopUp dans son intégralité, vous avez à votre disposition:

- Une liste avec les images possibles
- Quelques attributs:
  - `Suivre` : L'objet popup suit la fenêtre
  - `Légé` : L'objet lorsqu'il affiché, l'est dans une fenêtre sans bord
  - `volatile` : L'objet disparait lorsque vous sélectionnez l'image popup
- Les hooks:
  - Ouverture : fonction appelée à l'ouverture du fichier
  - Fermeture : fonction appelée à la fermeture du fichier
- Le label de l'objet

L'objet PopUp hérite de Area (voir Section 7.1 [Aire], page 21).

7.22 Rectangle

Cet objet est relativement complexe, dans la mesure où il permet de réaliser plusieurs choses radicalement différentes. En fait il autorise, par l'intermédiaire du gadget cycle présent dans l'interface, la création:

- D'une zone rectangulaire dans une fenêtre
- D'une barre horizontale avec ou sans titre permettant de séparer des éléments de l'interface
- D'une barre verticale permettant également une séparation de gadgets de l'interface

Vous pouvez également fixer la hauteur ou la largeur de ces éléments si nécessaire.

L'objet Rectangle hérite de Area (voir Section 7.1 [Aire], page 21).

7.23 Case colorée

Cet objet relativement simple a pour but de visualiser une couleur. Vous devez juste définir:

- si vous voulez ou non figer la hauteur de l'objet
- si vous voulez ou non figer sa largeur

- la couleur qu'il contiendra

L'objet Case colorée hétire de Area (voir Section 7.1 [Aire], page 21).

7.24 Menus

Les menus peuvent être attachés à une fenêtre ou à l'application. Un menu se compose d'un ou plusieurs titres, comprenant eux-mêmes des sous-menus et des entrées. La hiérarchie qui en découle est représentée de manière similaire à celle utilisée dans la fenêtre principale de création des objets.

Les seules limitations remarquables tient à la profondeur de l'arbre des menus. En effet, un sous-menu ne peut avoir lui-même de sous-menu. La profondeur de l'arbre est donc réduite à 2 au maximum.

D'autre part:

- Les titres ou sous-menus peuvent être rendus actifs ou inactifs grâce au checkmark correspondant
- Les éléments ont, de leur côté un peu plus d'attributs qui les définissent:
  - Il est possible d'autoriser son marquage
  - On peut le définir 'marqué'ou 'non marqué'
  - on peut y associer un raccourci clavier
  - on peut définir son label

8 Notifications

La gestion des évènements constitue, à mon avis, la grande nouveauté de MUIBuilder2.0. Vous allez enfin voir sur Amiga quelque chose de nouveau dans le domaine des constructeurs d'interfaces graphiques. Le but est de pouvoir définir des liens entre, d'une part, des évènements et, d'autre part, des actions déterminées. Les évènements proviennent des objets MUI : vous allez donc définir pour un évènement donné l'action qui devra être engendrée, et sur quel objet de l'interface cette action devra avoir lieu.

Une fois un lien évènement-action construit, vous pouvez le tester depuis MUIBuilder, en cliquant simplement sur le bouton `Test` de la fenêtre de création principale. Toutes les actions qui ne mettent pas en jeu l'application et/ou des variables définies dans votre programme peuvent être testées sans problème (tout du moins je l'espère) depuis l'interface builder.

La fenêtre de notification comporte les éléments suivants:

- La liste des évènements qui peuvent être générés par l'objet dont le nom apparait en haut de la fenêtre.
- La liste des objets sur lesquels une action peut être effectuée. Pour vous aider à trouver un objet précis, les objets ont été organisés suivant un ordre hiérarchique : fenêtre par fenêtre. La fenêtre dans laquelle se trouve l'objet dont vous éditez les notifications apparait en première position de cette liste. Ensuite vous trouvez l'objet application (sur lequel nous allons revenir), puis toutes les autres fenêtres de votre application.
- Une fois que vous avez sélectionné un objet cible, la troisième liste contient la liste de toutes les actions qui peuvent être effectuées sur cet objet. Certaines de ces actions nécessitent des arguments qui vous seront demandés lorsque vous les validerez

Pour ajouter une notification à l'objet que vous éditez, il suffit d'effectuer un double-click sur un élément de la liste des actions, ou bien d'appuyer sur le bouton 'Ajouter notification'.

L'objet application qui vient s'ajouter dans la liste des objets a pour rôle de rassembler toutes les actions qui constituent une référence vers votre programme. Ainsi, vous trouverez comme action valide l'appel de fonction, l'envoi d'un évènement au programme ...

Lorsque vous validez une action, si celle-ci demande un paramètre le programme va ouvrir une fenêtre de requête. Les paramètres peuvent être de plusieurs types:

Le choix d'un objet auquel on veut ajouter des notifications peut s'effectuer de deux manières différentes :

– Pour une fenêtre : en appuyant sur le bouton `Notifier` de la page `Attributs`
– Pour un objet : en sélectionnant l'objet dans la fenêtre de création, et en appuyant sur le bouton `Notifier` de cette page.

8.1 Fonctions

Lorsque le paramètre d'une action est un `Hook` d'une fonction, le programme va vous demander le nom de cette fonction. Il va alors ouvrir une fenêtre où apparaitront toutes les fonctions disponibles dans la base des fonctions de MUIBuilder. Vous pouvez, soit en choisir une qui existe déjà, soit en créer une nouvelle. Lors de la génération du source, MUIBuilder prendra en compte ces définitions de fonctions (à voir suivant le langage cible).

8.2 Variables

Certaines actions peuvent avoir une variable comme paramètre. De manière à récupérer ce nom de variable, MUIBuilder vous ouvre une fenêtre, dans laquelle se trouve (tout comme lors de la requête d'un nom de fonction) la liste de toutes les possibilités déjà enregistrées par MUIBuilder. Vous pouvez donc renvoyer une variable déjà déclarée ou en déclarer une nouvelle.

8.3 Constantes

Une constante peut être de plusieurs types:

- un booléen : 2 valeurs possible True/False
- Une chaine de caractères
- Un entier
- Un caractère

Suivant le type de la constante, l'un des gadgets de la fenêtre de requête des constantes va être activé. C'est grâce à ce gadget que vous entrerez la valeur qui vous est demandée.

9 Evolution Future

De nombreuses choses restent à faire dans ce programme. Certaines auraient même dû être incluses dans cette version. Mais malheureusement, je n'ai pas eu le temps de tout implémenter, et dans la mesure où je ne suis pas sûr de pouvoir y consacrer du temps dans les prochains mois, il était nécessaire de sortir une version maintenant.

Dans les prochaines version, j'envisage d'ajouter:

- Quelques attributs manquant encore sur certains objets (tel que l'objet PopAsl)
- Quelques objets manquant toujours à l'appel (la palette par exemple)
- Des fonctions ARexx pouvant donner accès à des fonctions internes de MUIBuilder
- Probablement (si personne d'autre ne se décide à le faire) un support pour les objets externes MUI
- Parfaire un peu le système de notifications, permettant ainsi de créer des notifications très complexes, tout en conservant la même interface qu'actuellement
- Amélioration des générateurs de code : ajout d'une interface graphique, et ajout de TRES nombreuses options de génération
- Toujours à propos des générateurs : possibilité d'avoir des préférences locales et globales
- Interfacer MUIBuilder avec des éditeurs de textes semble envisageable et est toujours à l'ordre du jour
- Dès que le Drag&Drop sera disponible dans MUI, il sera utilisé dans la construction de l'interface graphique
- Les boutons correspondant au choix des objets deviendront probablement des images
- La sauvegarde évoluera probablement, de manière à permettre des search&replace directement sur cette dernière
- La création du catalogue sera plus 'intelligente' : elle traitera l'ajout et la suppression de chaines, et n'effacera plus les chaines que vous aurez ajoutées 'à la main'

Comme vous pouvez le voir, j'ai encore de quoi m'occuper quelques mois avant de pouvoir commencer un autre projet !!! :) Bien sûr toute suggestion intéressante viendra s'ajouter à cette liste dès que vous aurez eu l'amabilité de me la faire parvenir!!!

Remerciements

MUIBuilder est dorénavant **dédié à la mémoire de Pierre Carrette**, ce formidable programmeur sans qui ce soft n'aurait jamais été ce qu'il est, et qui nous a malheureusement quittés, à notre grand désespoir.

Je voudrais remercier ici tous ceux qui m'ont aidé et ont contribué à la réalisation de MUIBuilder (pas d'ordre particulier):

- **Lionel Vintenat** pour l'idée du code générique et la réalisation du générateur de code pour le langage E.
- **Albert Weinert** pour sa réalisation du module externe dédié à l'OberonII.
- **Stefan Schulz** pour le module externe dédié au Modula2.
- **Stefan Sommerfeld** pour le module d'assembleur.
- **Dirk-Michael Brosig** pour le module externe dédié à GCC.
- **Bernd Noll** pour son aide efficace lors du debuggage de MUIBuilder, et ses gigantesques rapports de bugs et suggestions
- **Andreas Jung** pour ses plus que fréquents rapports de bugs et ses inestimables suggestions
- **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.
- **Christian Brandel** pour ses brillantes traductions, d'une part de la documentation, d'autre part du catalogue Allemand !
- **Pascal Rabier** et **Mike Manzaro** pour m'avoir aidé dans la traduction de la documentation anglaise.
- **Tobias Ferber** pour son superbe icône MagicWB et les images des objets utilisées dans MUIBuilder.
- **Jérôme Chesnot** pour ses images d'objets utilisées dans MUIBuilder.
- **Marco Musso** pour sa traduction du catalogue en Italien.
- **Marcin Orlowski** (Mr Soft/W.F.M.H.) pour sa traduction du catalogue en Polonais.
- **Stanislav Kneifl** pour la traduction tcheque.
- **Soeren Berg Glasius** pour la traduction Danoise.
- Enfin je tiens à remercier toutes les personnes qui m'ont envoyé des rapports de bugs, des suggestions, et qui ont contribué d'une manière ou d'une autre à l'évolution de ce logiciel. **Merci à tous !**

Index

R

S

V

Résumé du contenu du document

Table des matières