

ClassAction

Gasmi Salim

COLLABORATORS

	<i>TITLE :</i> ClassAction		
<i>ACTION</i>	<i>NAME</i>	<i>DATE</i>	<i>SIGNATURE</i>
WRITTEN BY	Gasmi Salim	May 28, 2025	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

1	ClassAction	1
1.1	Guide de ClassAction 2.8	1
1.2	Qu'est-ce que ClassAction ?	1
1.3	Un peu plus sur ClassAction	2
1.4	Exigences système	2
1.5	Configuration de ClassAction	3
1.6	ClassAction Prefs	7
1.7	Utiliser la fonction apprendre	8
1.8	Qu'est-ce qu'une classe	9
1.9	Créer une nouvelle classe	9
1.10	Définir une nouvelle action	12
1.11	Mode CLI	13
1.12	Mode WB	14
1.13	Mode NO CLI	14
1.14	Mode ARexx	14
1.15	Arguments	14
1.16	Commandes	15
1.17	Améliorations futures	17
1.18	Remerciements	17
1.19	L'auteur	18
1.20	Utiliser ClassAction	18
1.21	Informations Techniques	19
1.22	Installation	20
1.23	Enregistrement	20
1.24	Licence	21
1.25	Historique	21
1.26	Les commandes ARexx	26
1.27	Après l'installation	28
1.28	Message pour Philippe THOMAS	29
1.29	Bugs référencés	29

Chapter 1

ClassAction

1.1 Guide de ClassAction 2.8

ClassAction
Version 2.8

Qu'est-ce que ClassAction ?	Lisez tout d'abord ceci
Exigences système	Ce dont vous avez besoin
Installation	Comment installer ClassAction
Après l'installation	Installation Partie II
Utiliser ClassAction	Comment utiliser cet outil
Configuration	Changer les ToolTypes
ClassAction Prefs	Configuration des classes et des actions
Les commandes ARexx	Le port ARexx et les commandes
Améliorations futures	Ce qui devra être rajouté
Historique & Fonctionnalités	Depuis le début...
Enregistrement	Pourquoi et comment s'enregistrer
Licence	L'auteur
Remerciements	Bugs connus

1.2 Qu'est-ce que ClassAction ?

ClassAction est un petit utilitaire qui va simplifier la vie de tous les utilisateurs de disque dur.

Lorsque vous possédez un disque dur, vous avez forcément une tonne de fichiers : exécutables, modules, images, sources, animations, sons...

ClassAction détermine pour vous le type de fichier sélectionné et propose une liste d'actions à effectuer sur ce fichier.

Par exemple, si vous sélectionnez une image GIF, elle sera reconnue par

ClassAction comme étant de la classe GIF, et une liste d'action sera affichée avec des actions telles que 'Afficher' ou 'Editer'.

ClassAction est fortement configurable, vous pouvez ajouter vos propres classes et actions.

Les actions utilisent des programmes externes, vous pouvez donc utiliser votre afficheur de Gif préféré (ou n'importe quoi d'autre).

ClassAction a une AppIcône, un port ARexx, est localisé et est une commodité.

ClassAction utilise la xfdmaster.library pour décompresser automatiquement les fichiers compressés.

Avec cette possibilité, même les classes compressées peuvent être reconnues.

ClassAction utilise très peu de mémoire et de temps CPU.

Si vous prenez un peu de temps à configurer ClassAction, vous pourrez faire n'importe quoi avec !!! C'est une façon simple de gérer ses fichiers.

Avant de décider de le virer, essayez le !!!!!!!

Si vous voulez en savoir plus sur ClassAction, cliquez [ici](#)

1.3 Un peu plus sur ClassAction

ClassAction et ClassActionPrefs sont (C) 1994-95 par Gasmi Salim

Ce logiciel est distribué en tant que ShareWare. Vous êtes libre de l'utiliser !!! et de le diffuser tant que vous ne modifiez aucun des fichiers de l'archive.

MAIS si vous l'utilisez régulièrement, vous devez vous enregistrer !!!

Les distributeurs de DP sont autorisés à inclure le package de ClassAction dans leur collection tant qu'ils me préviendront qu'ils l'ont inclu.

Ceci est la version 2.8 de ClassAction.

Si vous la trouvez sympa, vous DEVEZ vous Enregistrer

J'espère que vous trouverez cet outil utile.

(au moins, je le trouve utile :))

Pour des informations techniques sur ClassAction :

Informations Techniques

1.4 Exigences système

Pour utiliser ClassAction, vous avez besoin des trucs suivants :

- o Amiga OS 3.0 ou supérieur
(Peut-être qu'une version OS 2.0 sera développée un jour)
- o un disque dur
(ClassAction sans intérêt sans disque dur)

That's all folks !!!

1.5 Configuration de ClassAction

L'interface et les fonctionnalités de ClassAction sont configurables via les ToolTypes.

Pour changer une valeur d'un ToolType, sélectionner simplement l'icône de ClassAction et utiliser l'option 'Information' dans le menu 'Icones' du Workbench.

Voici une liste de tous les ToolTypes :

Note : La valeur par défaut d'un ToolType est utilisée lorsque le ToolType n'est pas trouvé.

=====

DONOTWAIT

N'enlevez pas ce ToolType, il est nécessaire si ClassAction est lancé depuis la WBStartup.

=====

CX_PRIORITY

Priorité de ClassAction.

Valeur par défaut : 0

=====

CX_HOTKEY

Ceci est la Hotkey qui permet a ClassAction de cacher/montrer son interface.

Si vous l'utiliser quand ClassAction est en AppIcône ou caché alors la fenetre apparaitra, par contre si vous l'utilisez

quand ClassAction est en mode fenetre alors il sera caché.

Valeur par défaut : LALT C (ALT gauche + Shift + c)

REQBUG

Certains programmes comme MagicMenu font planter les commandes REQF,REQD,REQV quand elles sont utilisees a partir de l'AppIcône

Si c'est votre cas et que vous tenez absolument a conserver le programme fautif, mettez ce tooltype a YES (REQBUG=YES)

Ceci forcera ClassAction a utiliser REQT a la place de REQF,REQD et REQV.

APPSTART

Vous pouvez mettre ce ToolType a : YES,NO ou HIDE .

NO : ClassAction démarre en mode fenetre.
YES : ClassAction démarre en mode AppIcône.
HIDE : ClassAction démarre caché.

Valeur par défaut : NO

STARTDIR

Ce ToolType détermine le répertoire à afficher la première fois que ClassAction est lancé.

Valeur par défaut : répertoire actuel

WBFONT

Mettez ce ToolType à YES (i.e. WBFONT=YES) si vous voulez que ClassAction utilise les fontes actuelles du Workbench.
Si ce ToolType est à YES, la largeur réelle de la fonte doit être inférieure à 15, sinon ClassAction utilise la fonte topaz 8.

Si vous mettez ce ToolType à NO, ClassAction utilisera topaz 8.

Valeur par défaut : YES

DECRUNCH

Si vous mettez ce ToolType à YES (i.e. DECRUNCH=YES), ClassAction essaiera de décompresser les fichiers compressés, grâce à la

xfdmaster library.

Cela est nécessaire si vous souhaitez reconnaître les fichiers compressés.

Bien sûr, dans ce cas, le fichier sera pré-chargé en mémoire et un buffer sera alloué pour les données décompressées. Ceci consomme donc de la mémoire (environ 2.5 x la taille du fichier).

Donc si vous n'avez pas suffisamment de mémoire, vous ne devriez pas utiliser cette possibilité.

Valeur par défaut : NO

HEIGHT

Ce ToolType est la hauteur de la fenêtre de ClassAction.

La hauteur minimale dépend de la taille de vos fontes WB. Si vous indiquez une taille trop petite, ClassAction utilisera la hauteur la plus petite possible.

Valeur par défaut : 0 (utilise la plus petite fenêtre possible)

WINX et WINY

Ces ToolTypes sont les coordonnées par default de la fenetre.

Valeurs par default : 10

PUBSCREEN

Mettez ce Tooltype a YES (PUBSCREEN=YES) si vous voulez que la fenetre de ClassAction soit une fille de l'écran public en cours.

Valeur par défaut : NO

ICONX et ICONY

Ces ToolTypes sont les coordonnées de l'AppIcône.

Notez que s'il existe déjà un icône aux coordonnées spécifiées, le Workbench les ignorera et placera l'AppIcône le plus près possible de cette position.

Valeurs par défaut : pas de position fixe

=====

ICONNAME

Ce ToolType est le texte apparaissant sous l'AppIcône.
Par exemple, si vous spécifiez ICONNAME=Drop, le texte affiché sous l'icône sera : 'Drop'.
Le texte peut être vide (i.e. ICONNAME=), ainsi, aucun texte n'apparaît sous l'AppIcône.

Valeur par défaut : ClassAction

=====

ICONFILE

Ce ToolType est le fichier utilisé pour l'AppIcône.
De cette façon, vous pouvez utiliser n'importe quel icône (image) pour ClassAction.

Note : Ne mettez pas d'extension .info au nom du fichier icône.

Exemple : ICONFILE=Sys:icons/head
 utilisera l'icône head.info du répertoire Sys:Icons/
 comme AppIcon.

Si ClassAction ne parvient pas à charger votre icône, il utilisera celui par défaut (ClassAction.info).

Valeur par défaut : "" (Utilise l'icône par défaut)

=====

CLISIZE

C'est la taille de la fenêtre de sortie CLI, lorsque SystemTags() est utilisé.

La syntaxe est : DEVICE:TopX/TopY/Width/Height/Title

De cette façon, vous pouvez utiliser une autre unité à la place de CON: , ou définir de nouvelles position et dimensions pour la fenêtre de sortie CLI.

ATTENTION :

Si ce ToolType comporte une valeur erronée, ClassAction ne n'ouvrira pas de CLI.
NE modifiez pas cette valeur à moins que vous ne sachiez ce que vous faites.

** NE JAMAIS ** ajouter de commandes telle que AUTO, CLOSE, WAIT à cet argument car ClassAction le fait automatiquement.

**** NE JAMAIS **** mettre d'espaces dans le titre.

Dans la version 2.0, un ToolType similaire : OUTPUT était utilisé.
Il n'est plus nécessaire actuellement.

Valeur par défaut :
CON:0/0/640/100/ClassAction_Output_Window

=====

DRIVE1 à DRIVE11

Mettez ces ToolTypes à un chemin valide et ils seront affichés
dans les boutons du Requester de fichier.
C'est utile pour aller rapidement vers un répertoire.

La syntaxe est :

DRIVEEx=<Texte du Bouton>,<Chemin>

i.e. DRIVE9=Jpeg,dh0:gfx/pictures/jpeg

De cette façon, le texte du bouton 9 sera 'Jpeg', mais le chemin
sera dh0:gfx/pictures/jpeg.

Vous pouvez également fournir uniquement le <Texte du Bouton>
sans le <Chemin>.

i.e. DRIVE3=dh0:libs

Dans ce cas, le texte du bouton sera le même que le chemin.
Mais dans ce cas, n'utilisez pas de chemin trop long car ils ne
seront pas affichés dans les petits boutons.

Valeur par défaut : "" (rien)

1.6 ClassAction Prefs

La définition de nouvelles classes et actions est le coeur du programme.
Pour faire ceci, il faut utiliser ClassActionPrefs.

L'utilisation de ClassActionPrefs devrait être simple...
Alors allons-y !!

La fenêtre est divisée en deux parties : les Classes et les Actions.

D'abord, choisissez une classe, les actions associées seront affichées
dans la partie des actions.

Pour ajouter ou supprimer une classe, cliquez simplement sur le bouton
correspondant.

C'est la même chose pour les actions...

1. Qu'est-ce qu'une classe
2. Créer une nouvelle classe
3. Définir une nouvelle action
4. La fonction Apprendre

1.7 Utiliser la fonction apprendre

La fonction Apprendre est destinée a vous aider a definir de nouvelles classes.

Quand vous definissez une nouvelle classe, il vous faut parfois aussi definir des offsets pour la classe, et cela peut être long et ennuyeux d'editer les fichiers pour essayer de deviner quels pourraient bien être les offsets .

C'est pour cette raison que le merveilleuse fonction Apprendre a été écrite.

Pour l'utiliser, suivez cette procedure :

- 1 : definissez votre classe, remplissez le nom.
- 2 : cliquer sur le bouton 'Apprendre'.
- 3 : Selectionner autant de fichiers que vous voulez avec le requester qui apparait Tant qu'ils appartiennent tous a la classe que vous voulez definir.
Plus vous selectionnerez de fichiers, meilleur sera le resultat.

ClassAction va essayer de trouver les offsets !
(Pour selectionner plusieurs fichiers, utilisez la touche shift)
- 4: Apres analyse, vous avez les offsets trouvés.
- 5: Modifiez les a la main si neccessaire.
- 6: cliquez sur 'Accepter' pour les utiliser pour votre classe ou cliquez sur 'Annuler' pour annuler.

ATTENTION :

Pour que ceci fonctionne , il faut impérativement être sur que :

* TOUS LES FICHIERS CHOISIS SOIENT DE LA MEME CLASSE.

* TOUS LES FICHIERS CHOISIS SOIENT NON CRUNCHES.

C'est tout !! facile non ?

1.8 Qu'est-ce qu'une classe

Une classe est une famille de fichiers. Par exemple, l'ensemble des sources C peut être considéré comme une classe, appelons la Classe C.

Avec ClassActionPrefs, vous pouvez définir autant de classes que vous le souhaitez, tant que vous décrivez la façon de la reconnaître.

Pour décrire comment reconnaître une classe, il y a deux méthodes : le nom du fichier (motif ou match name) et l'analyse du contenu du fichier.

L'attribut MatchName est utilisé pour reconnaître un fichier en fonction de son nom.

Les attributs Offsets sont utilisés pour reconnaître un fichier en fonction de son contenu.

Il y a deux classes internes que vous ne pouvez pas supprimer, elles sont affichées en blanc dans la liste des classes.

La première est appelée "Classe inconnue", mais vous pouvez la renommer. Cette classe contient tous les fichiers que ClassAction ne peut pas reconnaître.

La seconde est appelée "Action génériques" et vous ne pouvez pas la renommer. Cette classe contient des actions qui seront affichées pour TOUTES les autres classes...

Intérêt : si vous voulez avoir une action 'Copier' pour toutes les classes, vous pouvez la définir pour toutes les classes existantes, mais loooooong et ennuyeux.

Une façon plus appropriée de faire ceci est de rajouter cette action dans la liste des "Action génériques". 'Copier' sera alors affiché pour toutes les classes.

Les actions génériques sont affichées en blanc dans la liste des actions de ClassAction et ne sont visibles que lorsque la fenêtre de ClassAction est ouverte.

Elles ne sont pas accessibles lorsque ClassAction est AppIconifié, ou via la commande ARexx : Load.

1.9 Créer une nouvelle classe

Cliquez simplement sur le bouton 'Ajout' dans la partie Classes pour ajouter une nouvelle classe.

Une classe a trois propriétés :

- un nom
- un motif
- des offsets

- o Le nom est simplement le nom de la classe, c'est à vous de le choisir.
ATTENTION : un nom de classe doit être unique.

- o Le motif est une expression régulière AmigaDos, telle que :

`#?.c , mod.#? , #?.c|#?.h , #?b[a|c] , #?toto?`

(lisez le manuel AmigaDOS pour tous les caractères génériques)

ATTENTION : N'utilisez pas le caractère `*` mais plutôt `#?` .

Le motif ne différencie pas minuscules et majuscules, donc `toto.C` peut correspondre à `#?.c`

Si vous vous définissez une classe grâce aux motifs, il faut être sûr que cette définition est toujours correcte.

Exemple : si vous définissez la classe GIF avec le motif `#?.gif` tous les fichiers terminant par `.gif` seront reconnus comme des images GIF.

Mais êtes vous certain que tous les fichiers `.gif` sont des images GIFs, ou que toutes vos images GIF dispose de l'extension `.gif` ?

En fait, il ne faut utiliser les motifs que pour ces deux situations :

- il y a une bijection entre le motif et les noms des fichiers de la classe
(ex: `#?.info` est un bon motif pour la classe des Icones)
- vous n'avez pas le choix
(ex: comment reconnaître un source C, à part avec `#?.c`)

- o Dans les autres cas, il vaut mieux utiliser les Offsets.

Un Offset est un endroit dans le fichier où l'on doit trouver quelque chose de particulier pour l'identifier.

Par exemple, les images GIF commencent toujours par la chaîne `'GIF'` à l'Offset 0.

Il y a trois syntaxes pour définir les Offsets :

=====

Syntax `n\textdegree{}1` : `Offset,ChaineHexa`

Offset est un nombre DECIMAL contenant l'Offset.
ChaineHexa est une chaine HEXADECIMALE qui doit se trouver à cet Offset.

Exemple 1 : `0,4f4a` signifie que le fichier doit commencer avec les octets `$4f` et `$4a`

Exemple 2 : `9,448b3c` signifie que, à l'octet `n\textdegree{}9` doit se trouver : `$44 $8b $3c`

=====

Syntax `n\textdegree{}2` : `Offset,'Chaine'`

Offset est un nombre DECIMAL contenant l'Offset.
Chaine est une chaine ASCII qui doit se trouver à cet offset.

Exemple 1 : `0,'GIF'` signifie que le fichier doit commencer par la chaine `'GIF'`

Exemple 2 : `9,'FuBar'` signifie que, à l'octet `n\textdegree{}9` doit se trouver la chaine `'FuBar'`

=====

Syntax `n\textdegree{}3` : `Offset,"Chaine"`

Offset est un nombre DECIMAL contenant l'Offset.
Chaine est une chaine ASCII qui doit se trouver à cet offset.

Notez la différence avec la syntaxe précédente, ici nous utilisons `"` pour définir la chaine, et dans la syntaxe `n\textdegree{}2`, nous utilisons `'`.

C'est le même concept que la syntaxe `n\textdegree{}2`, mais dans ce cas ↵
,
il n'y a pas de différenciation entre minuscules et majuscules lors de la comparaison.

Par exemple : un fichier `AmigaGuide` commence toujours par la chaine `@database` en majuscules ou minuscules.

Si vous utilisez la deuxième méthode pour reconnaître les fichiers `AmigaGuide`, avec `0,'@database'`, `ClassAction` ne reconnaitra pas un fichier commençant par `@DATABASE` comme étant un `AmigaGuide`.

Par contre, cela fonctionne si l'Offset est défini avec la syntaxe `n\textdegree{}3` : `0,"@database"`

=====

Vous pouvez préciser jusqu'à 5 Offsets pour définir une classe.
Un fichier est reconnu pour la classe si tous les Offsets
correspondent.

Ex : si la classe X est définie par :

Offset n\textdegree{}1 : 0,4a8b6c
Offset n\textdegree{}2 : 58,14

Tous les fichiers commençant par \$4a8b6c ET ayant \$14 à l'octet 58
appartiendront à X.

Pour définir plusieurs Offsets, cliquez simplement sur le bouton
cyclique 'Offset #' pour activer le prochain Offset.

Remarque : La classe ASCII

Il existe une command interne pour les Offsets, nommée : ASCII[]

Si vous insérez cette commande dans l'Offset n\textdegree{}1
(i.e Offset#1=ASCII[]), elle va reconnaître les fichiers ASCII.
ClassAction n'essayera cette commande que lorsque tout le reste
aura échoué.
Grâce à ceci, les documents Amigaguide (qui sont des fichiers ASCII)
ne seront pas interprétés comme ASCII si vous avez défini une classe
Amigaguide.

Normalement, vous ne devriez pas utiliser cette commande car j'ai
fourni un fichier de préférences standard qui contient déjà la
classe ASCII, définie grâce à cette commande.

1.10 Définir une nouvelle action

Une fois la classe définie, il faut y rajouter des actions.
Chaque classe peut contenir autant d'actions que vous le souhaitez.

Cliquez sur le bouton 'Ajout' dans la partie des actions pour ajouter
une action.

Une action a 5 propriétés :

- un nom
- un mode d'exécution
- une taille de pile (seulement si le mode d'exécution est CLI)
- un délai (seulement si le mode d'exécution est CLI)
- une commande à exécuter

- o Nom est le nom de l'action.

- o Mode d'exécution peut être : Cli , WB , No Cli or ARexx .
- o Commande est une ligne de commande AmigaDOS valide, qui peut contenir des paramètres.
VOUS DEVRIEZ toujours utiliser un chemin absolu pour les exécutables.

Exemple : utilisez C:Copy au lieu de Copy dans la ligne de commande.

Vous pouvez insérer dans la ligne de commandes, des Arguments et des Commandes internes à ClassAction.

Boutons

- o Les boutons 'H' Haut et 'B' Bas permettent de trier les actions.
- o Le bouton 'Charg' vous demande de choisir un exécutable pour la ligne de commandes.
- o Le bouton 'Comm' affiche une fenêtre qui liste tous les arguments ou commandes internes possibles pour la ligne de commandes.
- o Avec le bouton 'Copie', vous pouvez copier toutes les actions d'une autre classe vers la classe actuelle.
Cliquez sur le bouton Copie et choisissez la classe source, d'où les actions doivent être copiées.
C'est utile lorsqu'il y a des actions similaires dans des classe différentes.

1.11 Mode CLI

**** Mode CLI ****

Si 'Cli' est choisi, l'action sera lancée à partir d'un CLI et la taille de la pile sera déterminée par la valeur donnée dans : Pile
(par défaut : 4096)

Ce mode d'exécution n'ouvrira de CLI que si cela est nécessaire (lorsque l'exécutable affiche quelque chose).

Vous pouvez définir la propriété Délai pour ce mode :

Si Délai est négatif (i.e. Delai = -1), le CLI restera ouvert jusqu'à ce que vous le fermiez grâce au gadget de fermeture dans le coin supérieur gauche de la fenêtre.

Si Délai est nul (i.e Delai = 0), le CLI se fermera de lui-même dès que la tâche se terminera.

Si Délai est positif (i.e. Delai = n avec n>0), le CLI attendra n secondes avant de se refermer, mais vous pouvez forcer la fermeture grâce au gadget correspondant.

La dimension du CLI utilisé est défini par le ToolType CLISIZE.
(l'ancien ToolType OUTPUT est maintenant obsolète).

1.12 Mode WB

*** Mode WB ***

Si 'WB' est choisi, aucun CLI se sera ouvert, et ClassAction simulera une exécution de l'action depuis le Workbench.
L'outil sera exécuté avec comme arguments, ceux contenus dans son icône.

ATTENTION : ce mode est uniquement valable pour les exécutables qui disposent d'une icône.

1.13 Mode NO CLI

*** Mode NO CLI ***

Si 'No CLI' est choisi, aucun CLI ne sera ouvert, même si le programme affiche quelque chose, mais la tâche est tout de même exécutée depuis un CLI.

1.14 Mode ARexx

*** Mode ARexx ***

Si 'ARexx' est choisi, la commande rx est exécutée avec la ligne d'exécution spécifiée. Cette ligne DOIT être un script ARexx.
Bien sûr, REXXMaster doit être actif et and Rx doit se trouver dans le répertoire Sys:rexxc/ pour fonctionner.

1.15 Arguments

Actuellement 8 commandes Arguments sont disponibles :

les 4 premières commandes sont en minuscules : [f] [s] [b] [x]
et incluent le résultat entre guillemets .

[f] : Chemin complet du fichier sélectionné avec guillemets
[s] : Chemin complet du fichier sélectionné sans suffixe avec guillemets
[b] : Nom du fichier sélectionné avec guillemets
[x] : Nom du fichier sélectionné sans suffixe avec guillemets

Les 4 commandes suivantes font la meme chose mais sans les guillemets

```
[F] : Chemin complet du fichier selectionne
[S] : Chemin complet du fichier selectionne sans suffixe
[B] : Nom du fichier selectionne
[X] : Nom du fichier selectionne sans suffixe
```

Exemple : supposons que vous ayez choisi le fichier ram:env/sys.prefs

```
[f] = "ram:env/sys.prefs"
[s] = "ram:env/sys"
[b] = "sys.prefs"
[x] = "sys"

[F] = ram:env/sys.prefs
[S] = ram:env/sys
[B] = sys.prefs
[X] = sys
```

Exemple : supposons que vous avez choisi le fichier ram:main.c

```
* La ligne Exec
      c:copy [f] [F].bak
sera remplacee par :
      c:copy "ram:main.c" ram:main.c.bak

* La ligne Exec
      c:copy [f] [S].bak
sera remplacee par :
      c:copy "ram:main.c" ram:main.bak
```

1.16 Commandes

Actuellement, cinq commandes sont possibles :

```
REQD[texte] : Demande un repertoire.
REQF[texte] : Demande un fichier.
REQV[texte] : Demande un volume.
REQT[texte] : Demande un texte.
SURE[texte] : Demande à l'utilisateur de confirmer.
```

=====

Commandes REQ

Les commandes REQ affichent une requête Reqtools ayant pour titre :

[texte]. Elle sont utiles lorsque vous avez besoin de commandes interactives.

Exemple :

```
bin:lha x [f] to REQD[Choisissez un répertoire où extraire]
```

Ceci affichera une requête de répertoire vous permettant de choisir le répertoire cible. Le fichier sélectionné [f] sera alors extrait dans ce répertoire.

REQF[] est identique, mais il demande un fichier.

Exemple :

```
c:dir [f] > REQF[Choisissez un fichier]
```

REQV[] demande à l'utilisateur de choisir un volume.

REQT[] demande à l'utilisateur de rentrer un texte, c'est utile pour entrer des arguments, par exemple :

```
c:cpu REQT[entrer les arguments de cpu]
```

ATTENTION :

Les commandes REQD, REQF et REQV peuvent être incompatibles avec le mode AppIcône, quand certains programmes comme MagicMenu sont chargés en mémoire.

Si à l'utilisation de ces commandes vous remarquez de fréquents gurus avec le mode AppIcône, essayez de trouver le programme fautif ou alors mettez le tooltype REQBUG à YES.

Ce qui transformera toutes les commandes REQ dangereuses en une commande REQT pour l'AppIcône.

=====

Commande SURE

La commande SURE[texte] affiche une requête avec le texte [texte] et avec deux boutons : oui / non.
Si l'utilisateur choisi non, la ligne de commande est annulée.
Si l'utilisateur choisi oui, ClassAction exécutera la ligne de commande sur la partie DROITE de la commande Sure.

Exemple :

```
SURE[Détruisons nous ce fichier ?]C:delete [f]
```

Ceci affichera une requête demandant à l'utilisateur de répondre oui ou non à la question : "Détruisons nous ce fichier ?".
Si l'utilisateur répond non, rien ne se passe ; si l'utilisateur répond oui, alors C:delete [f] est exécuté.

=====

Bien sûr, vous pouvez combiner autant d'arguments / commandes dans une ligne de commande, que vous le voulez.

Exemple :

```
SURE[Je le renomme ?]c:rename [f] REQF[Donne moi un nouveau nom]
```

1.17 Améliorations futures

Voici ce que j'aimerais rajouter à ce programme dans les futures versions :

- o Concept de Famille de Classes :
Par exemple, la famille Images contient GIF, IFF, TARGA, JPEG... et la possibilité de choisir un filtre selon la famille dans la requête de répertoires. Grâce à cela, il sera possible de filtrer les images, sons...
- o Un nombre illimité de boutons de chemin.
- o Utiliser les fontes WB dans ClassActionPrefs.
- o Et bien sûr, tout ce que vous me demanderez :)

1.18 Remerciements

Je voudrais remercier les personnes suivantes :

- o Mireille (pour sa patience...)
 - o Philippe Thomas (pour des suggestions, aides, beta-tests, le guide en Français et UTT utilisé lors de l'install)

Hey Phil, si tu lis ça, clique [ici](#)
 - o Richier Pierre (pour les icones MagicWB de ClassAction)
 - o Jean-Michel et Georges (pour des beta-tests sur un A4000/40)
 - o Obvious Implementations Corp (pour Dice C Pro)
 - o Nico Francois pour la ReqTools library
 - o Georg Hörmann pour la *FANTASTIQUE* xfdmaster library
 - o Tous les personnes qui m'ont contactes pour me reporter des bugs ou pour me faire des suggestions.
 - o Tous les utilisateurs enregistrés
-

1.19 L'auteur

Vous pouvez me contacter à l'adresse suivante :

Gasmi Salim
4b rue des petits champs

67300 Shiltigheim
France

Irc: Dr_Unix (#amigafr, #amiga)

Web: <http://www.gasmi.net>

E-Mail: salim@gasmi.net or salim@sdv.fr

1.20 Utiliser ClassAction

L'utilisation de ClassAction est VRAIMENT simple.

Sélectionner simplement un fichier dans le requester correspondant.
Vous pouvez aller sur le répertoire précédent grâce au bouton 'Parent' ou la ligne <Parent> dans le requester.
Le bouton droit de la souris affiche la liste des volumes montés et des assignations.

Après avoir sélectionné un fichier, apparaîtra dans la liste de droite, la classe du fichier et les actions correspondantes.
Sélectionnez simplement l'action souhaitée...

Si vous double-cliquez sur un fichier, la première action définie sera automatiquement lancée.

Vous pouvez Zoomer/Dézoomer la fenêtre grâce au petit gadget de zoom dans le coin supérieur droit de la fenêtre.

Pour quitter, utiliser le bouton 'Fin'.

Pour réduire la fenêtre en une AppIcône, il suffit de la fermer.
Double cliquez sur l'AppIcône pour réouvrir la fenêtre.

Une fois ClassAction transformé en AppIcône, il suffit de jeter des icônes dessus pour que le type du fichier soit reconnu.
Si la classe dispose d'une seule action possible, alors celle-ci est automatiquement exécutée.
Sinon, une fenêtre propose la liste des actions, il ne reste plus qu'à en choisir une.

C'est tout, facile non ???

1.21 Informations Techniques

ClassAction est écrit à 100 % avec DICE C 3.0

Informations diverses :

Le fichier de configuration est un fichier ASCII appelé :
ENVARC:ClassAction.prefs

Les actions génériques sont sauveées dans ENVARC:ClassAction_Gen.prefs

ClassAction crée un exécutable nommé ClassAction_RunTask, stocké dans T:

Cet exécutable est utilisé pour lancer des tâches depuis le WB.

La vitesse du requester de fichier est due à l'algorithme de tri utilisé (tri récursif basé sur les arbres appelé 'Tri du Chat')

Informations sur les bibliothèques :

Bibliothèques ROM utilisées :

exec.library	V37+
dos.library	V37+
intuition.library	V37+
graphics.library	V37+
gadtools.library	V39+
workbench.library	V37+
utility.library	V39+

Bibliothèques DISK nécessaires :

rexsyslib.library	V39+
commodities.library	V37+
asl.library	V39+
icon.library	V37+
reqtools.library	V38+

Bibliothèques DISK utilisées (si disponibles) :

locale.library	V38+
datatypes.library	V39+
xfdmaster.library	V30+

Comment ClassAction détermine une classe :

- 1- test si le nom du fichier correspond au motif défini pour la classe

- 2- test si les offsets définis dans la classe correspondent au fichier
- 3- décompression du fichier grâce à la xfdmaster.library
- 4- test si les offsets définis dans la classe correspondent au fichier décompressé
- 5- test si le fichier est ASCII (si la commande d'Offset ASCII est utilisée)

Si tout cela échoue, le fichier est déclaré comme 'Type inconnu'.

1.22 Installation

Pour installer ce logiciel :

Normalement, vous devriez utiliser le scripts Installer (script d'installation standard) fourni dans l'archive et simplement suivre les instructions.

Mais, si vous ne l'avez pas, voila comment installer ClassAction :

- Copiez ClassAction, ClassActionPrefs et les icones où vous voulez.
- Copiez les fichiers .prefs fournis dans ENVARC:
- Copiez le fichier ClassAction.guide où vous voulez.

C'est tout...

1.23 Enregistrement

Si vous lisez ces lignes, c'est que vous voudriez être un utilisateur enregistré de ClassAction.

Laissez moi vous expliquer pourquoi vous devriez vous enregistrer :

Tout d'abord, pour soutenir la meilleure machine jamais créée, parce que le futur de l'Amiga dépend des logiciels futurs. Lorsque vous soutenez un programmeur dans son travail, vous soutenez votre ordinateur et son futur !!!!

Aussi parce que je passe tout mon temps libre à faire des sharewares. Si vous les utilisez, pourquoi ne pas envoyer l'enregistrement ? Cela me fera continuer à faire des programmes.

Mais la version MailWare est utilisable à 100 %, rien n'a été enlevé !!! Seulement quelques requesters "ennuyeux" en plus. Donc si vous l'appréciez vraiment, ENREGISTREZ VOUS !!

De plus l'enregistrment est GRATUIT !!! vous devez juste executer

le programme CA_Register et suivre la procedure suivante :

- Entrez votre nom, votre adresse Email (si vous en avez une).
 - Ceci creera un fichier dans S: appelle CA-Code_request.txt
envoyez le moi par mail a classaction@sdv.fr
- ou par courrier standard a mon adresse
(dans ce cas incluez 2\$ pour couvrir les frais d'envoi
et n'oubliez pas de me communiquer votre adresse)
- Je vous enverrai (par Email ou courrier) votre Licence Code
Relancez CA_register, entrez le licence code et votre keyfile
sera cree ...

Merçi d'avance pour votre support !

Sincèrement,

Salim

1.24 Licence

Copyright

ClassAction et ClassActionPrefs sont Copyright © 1994-1995
par Gasmi Salim.

ClassAction est un programme shareware. Le package ne doit pas être
altéré de quelque façon que ce soit et ne peut pas être utilisé à des
fins commerciales sans la permission écrite préalable de l'auteur.
Ce message de copyright doit être préservé.

Garantie

Aucune responsabilité ne sera acceptée pour des dommages qui pourraient
résulter de l'utilisation de ce programme. Toute utilisation est à vos
risques et périls. Le logiciel est fourni "tel quel" sans aucune
garantie implicite. La documentation est censée être correcte, mais
l'auteur se réserve le droit de mettre à jour le programme et/ou la
documentation sans en notifier l'utilisateur.

1.25 Historique

Historique de ClassAction V2.8 (c) Salim Gasmi

25/09/95 : V2.8

- ClassAction dispose maintenant d'une gestion TOTALE de sa commoditee et propose une HotKey pour Cacher/Monter l'interface.
- Le Tooltype CX_HOTKEY a ete rajoute pour gerer la Hotkey
- Le Tooltype APPSTART peut etre maintenant mis a HIDE (APPSTART=HIDE) si on veut que ClassAction démarre caché.
- Le Tooltype PUBSCREEN a ete rajoute pour permettre l'utilisation des ecran publics.
- Le selecteur de fichiers interne de ClassAction ne liberait pas toute la memoire allouée, c'est maintenant corrigé .

11/09/95 : V2.75

- Les commandes REQT,REQV et REQF n'etaient pas incompatibles avec le mode AppIcône mais avec certains programmes comme MagicMenu principalement.

Enormement d'utilisateurs se sont plaints du changement en REQT meme si ils n'utilisaient pas un programme creeant le probleme.

J'ai maintenant rajoute un tooltype (REQBUG) pour laisser le choix a l'utilisateur d'utiliser les REQs ou pas a partir de l'AppIcône.

03/09/95 : V2.7

- Commandes [b],[x],[B],[X],[F],[S] rajoutees
- la commande Arexx ACTION a ete rajoutée
- Bug corrige quand on jetait plusieurs icones sur l'appicone
- Quelques bugs dans les routines de 'render' on etes supprimees
- Les commandes REQD,REQF et REQV sont maintenant changees en REQT quand elles sont utilisees a partir de l'appicone ou d'Arexx. Ces commandes ne sont ompatibles qu'avec le mode fenetre.

28/08/95 : V2.6

- La fenetre de ClassAction est maintenant redimensionnable.
 - ClassAction et ClassActionPrefs utilisent la ReqTools.library.
 - Ajout de la commande REQT pour choisir un texte.
 - Ajout de la commande REQV pour choisir un Volume.
 - Ajout des tooltypes WINX et WINY.
 - Le requester de la commande 'Apprendre' a maintenant un bouton ALL.
-

- Le nom du fichier sélectionné apparaît dans tous les requesters REQ.
- ClassAction utilise maintenant un 'Key File' pour les versions enregistrées qui se place dans S: .

17/07/95 : V2.5

- ClassAction et ClassActionPrefs sont maintenant localisés et un catalogue français est fourni dans l'archive.
- Ajout de la fonction Apprendre dans ClassActionPrefs.
- Pour choisir la première action d'un fichier, il faut maintenant double-cliquer dessus au lieu de le resélectionner.
- Les Actions Génériques sont maintenant synchro et relisent le répertoire courant.
- Le ToolType AUTOSELECT n'est plus utilisé.
- Les requêtes REQ s'ouvrent maintenant dans le répertoire courant.
- Quelques améliorations mineures.

12/06/95 : V2.1

- Ajout de la classe interne 'Actions Génériques'.
 - Ajout de la commande exec SURE[].
 - Ajout de la commande d'offset ASCII[] pour reconnaître les fichiers ASCII.
 - Ajout des commandes ARexx : AppIconify, Show, Status, GetClass.
 - Changement du système de lancement des tâches : maintenant j'utilise Systemtags(). Il n'y a plus besoin de fichiers temporaires.
 - Vous pouvez maintenant définir un délai pour le mode d'exécution CLI.
 - Ajout de 'string' et "string" pour la définition des offsets.
 - Le ToolType OUTPUT est devenu obsolète et n'est plus utilisé, on utilise le nouveau ToolType CLISIZE à la place.
 - Le requester des actions est maintenant correctement dimensionné, apparaît sous la souris et utilise l'écran public le plus en avant.
 - Echange des boutons 'Utiliser' et 'Sauver' et ajout du bouton 'Cancel' dans ClassActionPrefs, pour se conformer au look standard des Prefs. Transformation du bouton 'About' en '?'.
Transformation du bouton 'About' en '?'.
 - Les routines de gestion des Cycles Gadgets de ClassActionPrefs n'étaient pas conformes à 100 % et des patchs comme Cycle2Menu
-

faisaient bugger ClassActionPrefs ; c'est enfin corrigé.

- Amélioration du code du requester et la routine d'information : ils sont environ 400 % plus rapides.
- Erreur de la couleur de surbrillance dans la Listview corrigée.
- ClassAction ne bloque plus l'écran Workbench lorsqu'il est AppIcconnifié
- Un méchant bug a été trouvé et supprimé dans ClassActionPrefs.
- Le bouton de droite ne montre les Assigns que lorsque la souris se trouve dans le requester et recliquer sur le bouton de droite revient au répertoire courant.
- ClassAction mémorise la position de la fenêtre.

23/05/95 : V2.00 (Mise à jour majeure)

- ClassAction dispose maintenant d'une AppIccone.
- ClassAction est maintenant une commodité.
- ClassAction dispose d'un port ARexx.
- ClassAction utilise les fontes par défaut du WB.
- Ajout du mode exec 'Arexx'.
- Des couleurs différentes pour les répertoires/fichiers.
- Ajout de gadgets Haut/Bas pour les actions dans ClassActionPrefs.
- Ajout du gadget 'Utiliser' dans ClassActionPrefs.
- Les classes sont maintenant triées dans la liste de ClassActionPrefs.
- Ajout des ToolTypes : APPSTART, ICONNAME, ICONX, ICONY, CX_PRIORITY, WBFONT, OUTPUT, ICONFILE.
- Lorsque la fenêtre est zoomée, elle a maintenant la bonne hauteur.
- Nouveau format de sauvegarde (CASF20).
- Suppression du bouton Suffix/Prefix, remplacé par le gadget MatchName qui accepte les wildcards (motifs).
- Déplacement du fichier de config dans ENVARC:
- A script installer est maintenant fourni avec l'archive.
- Optimisations du code.

05/05/95 : V1.43

- Nettoyage du code du requester, il est environ 5% plus rapide.
- ClassAction recherche son nom dans la structure WBstartup et peut donc être renommé.

02/05/95 : V1.42

- Ajout du mode exec 'No Cli' dans ClassActionPrefs.
- Des "" sont toujours ajoutés autour des noms de fichiers, même si cela n'est pas nécessaire.
C'est plus simple pour les scripts ARexx.

06/04/95 : V1.4

- Ajout du ToolType HEIGHT.
- Optimisation du code optimization.

22/02/95 : V 1.31

- Si vous cliquez deux fois sur le même fichier, la première action sera lancée (c'est plus facile que de sélectionner l'action à la main).
- Cette version est à présent ShareWare et vous devez vous enregistrer afin de recevoir la version enregistrée.

15/01/95 : V 1.3

- Ajout des commandes interactives REQD[] et REQF[].
- Ajout du gadget 'Copy' dans ClassActionPrefs.
- Quelques améliorations mineures.
- Les Beta-testeurs indiquent que cette version est vraiment stable.

25/11/94 : V 1.22

- Première Version Publique.
 - Ajout du bouton 'Info'.
 - ClassAction verrouillait les répertoires qu'il lisait, sans les déverrouiller ensuite. Corrigé.
 - Optimisation du code.
 - Quelques autres bugs mineurs corrigés.
-

08/11/94 : V 1.21

- Le programme se plantait avec des unités de disque vides... Corrigé.
- Bug corrigé avec Volume/Name.
- La ligne <...> n'apparaît plus lorsqu'on est sur la racine d'un volume.

07/11/94 : V 1.2

- Nouvelle interface, j'ai inclu mon propre et rapide requester de fichiers.
- Ajout des ToolTypes STARTDIR et DRIVE1 à DRIVE11.

01/11/94 : V 1.1

- Utilise maintenant la xfdmaster.library pour reconnaître et décompresser les fichiers.
- Utilisation des ToolTypes pour la configuration (DECRUNCH,AUTOSELECT).
- 'Unknown Class' est maintenant une classe interne avec un nombre illimité d'actions.
- Nouveau format de sauvegarde (CASF11).

16/10/94 : V 1.0

- Nouveau format de sauvegarde (CASF10).
- La fenêtre a maintenant un gadget de Zoom.
- Ajout de nombreuses définitions de classes.

10/10/94 : Beta Version

- Correction du bug de fichier temporaire.
- Correction d'une erreur dans l'incrémentation de l'Offset.
- Utilisation de l'asl.library pour le file requester.

01/10/94 : Alpha Version

1.26 Les commandes ARexx

ClassAction dispose d'un port ARexx appelé : ClassAction.01

Voici une liste de toutes les commandes ARexx :

=====

Quit

Ne fait que quitter ClassAction...

=====

Use

Force ClassAction à recharger le fichier de config.

=====

Ver

Retourne la version de ClassAction.

=====

Status

Retourne l'état actuel de ClassAction.

Retourne 0 : ClassAction est AppIconnifié.

Retourne 1 : ClassAction est en mode fenêtre.

=====

AppIconify

Force ClassAction à s'AppIconifier.

Si ClassAction est déjà en AppIcône, cette commande ne fait rien.

=====

Show

Force ClassAction à afficher la fenêtre principale.

Si ClassAction est déjà en fenêtre, cette commande ne fait rien.

=====

Load <nom de fichier>

ClassAction essaie de charger le fichier <nom de fichier>, et affiche le requester des actions, pour que l'utilisateur puisse

choisir l'une d'entre elles.

Si le fichier n'existe pas, cette commande ne fait rien.

=====

GetClass <nom de fichier>

ClassAction essaie de charger le fichier <nom de fichier> et retourne la classe de ce fichier.

Si le fichier n'existe pas, cette commande ne fait rien.

=====

Action <nom de fichier> <Action Generique>

ClassAction va essayer de lancer la premiere action qui repondra a <action generique> sur le fichier <nom de fichier>

un exemple :

Action ram:toto.lha extr

lancera la premiere action dont le nom contient extr sur le fichier ram:toto.lha

la formule generique est pratique , car vous n'avez pas a donner le nom complet de l'action a utiliser , un bout de son nom suffit.

1.27 Après l'installation

Après l'installation, vous devriez avoir une ClassAction en état de marche, avec un grand nombre de classes prédéfinies, mais avec très peu d'actions (la plupart des classe n'ont même pas d'action définie).

C'est à vous de configurer les actions en fonction de votre système et de vos préférences.

Pour cela, chargez simplement ClassActionPrefs et configurez le à votre goût...

Si vous ne savez pas comment configurer les actions et les classes, lisez ce guide :)).

Après avoir configuré tout cela, essayez encore de configurer le comportement de ClassAction avec ses ToolTypes.

Cela peut prendre du temps avant de parvenir à une 'chouette' config. Mais une fois que s'est fait, c'est vraiment SUPER !!!

Bien, maintenant lisez ce guide et Bonne Chance.

1.28 Message pour Philippe THOMAS

Salut Philippe !!!

Je voulais simplement te remercier pour toute l'aide que tu m'as apporté à la création de ce programme.

Quasiment toutes les améliorations de la version 2.0, c'est toi qui me les a proposées, parfois même avec insistance, style le resize de la fenêtre que je n'ai toujours pas fait.. :(

Encore merci pour tous les appels téléphoniques ; parfois plus d'une heure à discuter des Hooks, SystemTagList, de bugs etc... et ce, même en periode d'exams.

Franchement si ClassAction commence à être cool, c'est beaucoup grâce à toi, je crois que j'aurais eu la flemme de le paufinner autant si tu n'étais pas là.

Bref, ce programme est aussi un peu le tien.

Okay Phil, à la prochaine.

Salim.

PS : Non, non ton processeur il est bien, il est pas buggé... :)).

1.29 Bugs référencés

Enormement d'utilisateurs m'ont demande d'inclure une fenetre redimensionnable dans ClassAction V2.6+ .

J'ai fait de mon mieux pour les contenter mais c'est une tache ardue avec la bibliotheque GadTools.

Tres rarement, quand on redimensionne la fenetre, le rafraichissement graphique stoppe et il faut fermer la fenetre a la main pour deplanter.

Je ne sais vraiment pas pourquoi d'autant plus que je n'ai jamais vu le phenomene moi-meme et que ce sont les Beta-Testeurs qui ont remarques le probleme (2 fois en un mois de tests).

Bien sur, je travaille a la recherche et a l'elimination de ce bug mais je n'ai pas voulu priver les utilisateurs d'une fenetre redimensionnable pour un bug qui apparait 2 fois par mois.

De plus certains programmes comme MagicMenu font planter les commandes REQD,REQF et REQV quand elles sont utilisees a partir du mode AppIcône...

J'ai cree un tooltype REQBUG pour forcer ClassAction a ne plus les utiliser a partir de l'AppIcône mais de prendre REQT a la place pour tous ceux qui souhaitent continuer d'utiliser MagicMenu.