FM_ARexx

Dino Papararo

COLLABORATORS			
	TITLE : FM_ARexx		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Dino Papararo	August 10, 2024	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

FM_ARexx

1.1 Benvenuto

Benvenuto nel mondo di FlashMandel-ARexx!

Questa guida ti aiuterà a scrivere script arexx per FlashMandelWOS. Nota che nelle pagine seguenti saranno spiegate solo le caratteristice della interfaccia arexx di FlashMandelWOS. Se non hai familiarità con la sintassi del linuaggio arexx, fai riferimento al manuale arexx distribuito insieme a molti computers Amiga.

Generale informazioni generali per scrivere script per FlashMandelWOS.

Comandi lista di tutti i comandi implementati in FlashMandelWOS.

Esempi alcuni script di esempio.

Problemi conosciuti se arexx rifiuta d funzionare :-(

1.2 Informazioni generali

Informazioni generali per l'interfaccia arexx di FlashMandelWOS:

* Dove salvare gli script.

Il percorso standard è "FLASHMANDEL:arexx". E' più semplice per te salvare tutti gli script in questo cassetto.

* L'hostname

L'hostname è sempre scritto in questo modo: FLASHMANDEL.n. Il primo programma lanciato ha sempre n = 1, il secondo n = 2

e così via...

* Come avviare uno script.

Ci sono diverse possibilità per fare ciò:

1. Ovviamente puoi avviare uno script digitandone il nome nella shell. Per esempio:

> rx FLASHMANDEL:arexx/test.rexx FLASHMANDEL.1

2. Puoi scegliere il menu "ARexx/Launch...". Poi si aprirà un filerequester dal quale potrai scegliere lo script da eseguire.

3. Puoi creare un sotto menu nel menu arexx utilizzando i tooltype dell'icona. Il primo dovrà avere il nome REXXCMD.1, il successivo

REXXCMD.2 e così via...

Non ci sono limiti sulla quantità dei sottomenu se non quello imposto dall'AmigaOS.

IMPORTANTE:

Ogni script avviato da FlashMandelWOS, eredita l'attuale hostname come primo argomento. In questo modo puoi sempre impostare

l'hostname, se hai lanciato FlashMandelWOS più volte.

* Se esiste lo speciale script arexx chiamato "FLASHMANDEL:arexx/startup.rexx", sarà avviato alla partenza di FlashMandelWOS.

Utilissimo per configurare FlashMandelWOS o per ascoltare un suono in background o...

1.3 Problemi conosciuti

Si possono riscontrare i seguenti paricolari problemi con l'interfaccia arexx di FlashMandelWOS:

* FlashMandelWOS utilizza la reaction-arexx-class. Per questo motivo è indispensabile avere almeno l'AmigaOS 3.5+

per utilizzare questa funzionalità.

Non è previsto il supporto per versioni precedenti dell'OS neanche per il futuro, mi dispiace però:

perchè non aggiornoi il tuo sistema? - con AmigaOS 3.5+ avrai tra le altre cose anche una interfaccia reaction migliorata!

* Nessun altro problema conosciuto.

1.4 Esempi

Esempi distribuiti con FlashMandelWOS:

* PlayMovie.rexx:

Fa giusto alcuni zoom nell'immagine a video.

* RunningColor.rexx:

Dimostrazione su come manipolare i colori all'interno della tavolozza.

* SaveAnim.rexx:

Fa dieci zoom nell'immagine a video e salva il tutto in formato IFF-ANIM. Il pecorso può essere scelto tramite il requester.

* startup.rexx:

Esempio di statup-script per FlashMandelWOS. Verrà eseguito alla partenza riproducendo il 'welcome sound'.

* StopSound.rexx:

Interrompe la riproduzine di un suono.

* Test_ARexx.rexx:

Quì troverai il codice di esempio per tutti i comandi arexx implementati in FlashMandelWOS.

Per testare un comando, è necessario cambiare la linea "IF 0" in "IF 1".

1.5 Welcome

Segue una lista di tutti i comandi arexx implementati in FlashMandelWOS:

Comandi menu Progetto:

ABOUT Apre una finestra di informazioni generali.

SYSINFO Apre una finestra sulle risorse del sistema.

HELP Mostra la guida in linea.

LOAD_PICTURE Carica una immagine.

SAVE_PICTURE Salva una immagine.

LOAD_PALETTE Carica una tavolozza.

SAVE_PALETTE Salva una tavolozza.

PRINT Stampa una immagine.

QUIT Termna FlashMandelWOS

Comandi menu Opzioni:

SHOW_TITLE Nasconde la barra del titolo.

SHOW_ELAPSED Mostra il tempo trascorso.

SET_COORDINATES Imposta le coordinate complesse.

GET_COORDINATES Ottiene le coordinate complesse.

SET_ITERATIONS Imposta il numero delle iterazioni.

GET_ITERATIONS Ottiene il numero delle iterazioni.

SET_POWER Imposta l'esponente del polinomio.

GET_POWER Ottiene l'esponente del polinomio.

SET_PRIORITY Imposta la priorità del task.

GET_PRIORITY Ottiene la priorità del task.

SET_COLOR_REMAP Imposta il metodo di mappatura dei colori.

GET_COLOR_REMAP Ottiene il metodo di mappatura dei colori.

SET_FRACTAL_TYPE Imposta il tipo di frattale (Julia/Mandelbrot).

GET_FRACTAL_TYPE Ottiene il tipo di frattale (Julia/Mandelbrot).

SET_CPU_TYPE Imposta il processore con il quale operare (68k/PPC)

GET_CPU_TYPE Ottiene il processore con il quale operare (68k/PPC)

Comandi menu Calcola:

SHOW_PREVIEW Apre una finestra di anteprima.

RENDER Calcola un frattale.

UNDO Annulla l'ultima modifica.

SET_FRAME Imposta il frame di zoom.

GET_FRAME Ottiene il frame di zoom.

Comandi menu Video:

ZOOM Esegue uno zoom nell'immagine.

DO_COLORCYCLING Avvia il ciclo dei colori.

SET_DELAY Imposta il tempo di ritardo per il ciclo dei colori.

GET_DELAY Ottiene il tempo di ritardo per il ciclo dei colori.

SET_PALETTE Imposta la tavolozza per lo schermo.

GET_PALETTE Ottiene la tavolozza dello schermo.

SET_SCREENATTRS Imposta gli attributi dello schermo.

GET_SCREENATTRS Ottiene gli attributi dello schermo.

SET_FONT Imposta i caratteri dello schermo.

GET_FONT Ottiene i caatteri dello schermo.

Comandi generici:

START_TIMER Avvia il cronometro.

STOP_TIMER Interrompe il cronometro.

PLAY_SOUND Riproduce un suono.

STOP_SOUND Interrompe la riproduzione di un suono.

1.6 arexx-command ABOUT

NAME

ABOUT - open about-requester.

SYNOPSIS

"ABOUT"

FUNCTION

Opens the about-requester, just like choosing the 'About...'-menu-item. Must be closed by clicking the 'OK'-button.

INPUTS

-

RETURNS

RC:

error-level, 0 if successfull.

RC2:

error-code, 0 if successfull.

NOTES

-

SEE ALSO

-

EXAMPLE

"ABOUT"

BUGS

-

1.7 arexx-command SYSINFO

NAME

SYSINFO - open system-info-requester.

SYNOPSIS

"SYSINFO"

FUNCTION

Opens the system-info-requester, just like choosing the ’System info...’-menu-item. Must be closed by clicking the ’OK’-button.

INPUTS

-

RETURNS

RC:

error-level, 0 if successfull.

RC2:

error-code, 0 if successfull.

NOTES

-

SEE ALSO

-

EXAMPLE

"SYSINFO"

BUGS

-

1.8 arexx-command HELP

NAME

HELP - open help-guide.

SYNOPSIS

"HELP"

FUNCTION

Opens the help-guide, just like choosing the ’Help...’-menu-item.

INPUTS

-

RETURNS

RC:

error-level, 0 if successfull.

RC2:

error-code, 0 if successfull.

NOTES

-

SEE ALSO

-

EXAMPLE

"HELP"

IF RC THEN SAY "Can’t open help-guide!"

BUGS

-

1.9 arexx-command LOAD_PICTURE

NAME

LOAD_PICTURE - load a FlashMandelWOS-picture.

SYNOPSIS

"LOAD_PICTURE"

"LOAD_PICTURE" "PATH,FAST/S"

FUNCTION

Loads an FlashMandelWOS-Picture into the program, If you call this command without any parameter, it will behave like choosing the menu-item "Load Picture...".

INPUTS

PATH:

path of the desired FlashMandelWOS-picture.

FAST:

If specified, the picture will be loaded as fast as possible, this means no fading.

RETURNS

RC:

error-level, 0 if successfull.

RC2:

error-code, 0 if successfull.

NOTES

-

SEE ALSO

SAVE_PICTURE

EXAMPLE

"LOAD_PICTURE" "FLASHMANDEL:Pictures/Mandelbrot_def.iff" "FAST"

IF RC THEN SAY "Can’t open picture:" ErrorText(RC2)

BUGS

-

1.10 arexx-command SAVE_PICTURE

NAME

SAVE_PICTURE - save a FlashMandelWOS-picture.

SYNOPSIS

"SAVE_PICTURE"

"SAVE_PICTURE" "PATH,FORCE/S"

FUNCTION

Save an FlashMandelWOS-picture, If you call this command without any parameter, it will behave like choosing

the menu-item "Save Picture...".

INPUTS

PATH:

desired path to save the FlashMandelWOS-picture.

FORCE:

If specified, the picture will be saved, even if the specified filename already exists,

RETURNS

RC:

error-level, 0 if successfull.

RC2:

error-code, 0 if successfull.

NOTES

-

SEE ALSO

LOAD_PICTURE

EXAMPLE

"SAVE_PICTURE" "FLASHMANDEL:Pictures/Mandelbrot_test.iff" "FORCE"

IF RC THEN SAY "Can’t save picture:" ErrorText(RC2)

BUGS

-

1.11 arexx-command LOAD_PALETTE

NAME

LOAD_PALETTE - load a FlashMandelWOS-palette.

SYNOPSIS

"LOAD_PALETTE"

"LOAD_PALETTE" "PATH,FAST/S"

FUNCTION

Loads a FlashMandelWOS-palette into the program, If you call this command without any parameter, it will behave like choosing the menu-item "Load Palette...".

INPUTS

PATH:

path of the desired FlashMandelWOS-palette.

FAST:

If specified, the palette will be loaded as fast as possible, this means no fading.

RETURNS

RC:

error-level, 0 if successfull.

RC2:

error-code, 0 if successfull.

NOTES

-

SEE ALSO

SAVE_PALETTE

EXAMPLE

"LOAD_PALETTE" "FLASHMANDEL:Palettes/Palette_def.256" "FAST"

IF RC THEN SAY "Can’t load palette:" ErrorText(RC2)

BUGS

-

1.12 arexx-command SAVE_PALETTE

NAME

SAVE_PALETTE - save a FlashMandelWOS-palette.

SYNOPSIS

"SAVE_PALETTE"

"SAVE_PALETTE" "PATH,FORCE/S"

FUNCTION

Save an FlashMandelWOS-palette, If you call this command without any parameter, it will behave like choosing the menu-item "Save Palette...".

INPUTS

PATH:

desired path to save the FlashMandelWOS-palette.

FORCE:

If specified, the picture will be saved, even if the specified filename already exists,

RETURNS

RC:

error-level, 0 if successfull.

RC2:

error-code, 0 if successfull.

NOTES

-

SEE ALSO

LOAD_PALETTE

EXAMPLE

"SAVE_PALETTE" "FLASHMANDEL:Palettes/Palette_test.256" "FORCE"

IF RC THEN SAY "Can’t save palette:" ErrorText(RC2)

BUGS

-

1.13 arexx-command PRINT

NAME

PRINT - print FlashMandelWOS-picture.

SYNOPSIS

"PRINT" "FORCE/S"

FUNCTION

prints the actual picture on your printer.

INPUTS

FORCE:

If specified, the picture will be printed immediately without any questions.

RETURNS

RC:

error-level, 0 if successfull.

RC2:

error-code, 0 if successfull.

NOTES

-

SEE ALSO

-

EXAMPLE

"PRINT" "FORCE"

IF RC = 5 THEN SAY "User-aborted"

BUGS

-

1.14 arexx-command QUIT

NAME

QUIT - quit FlashMandelWOS.

SYNOPSIS

"QUIT" "FORCE/S,FAST/S"

FUNCTION

When calling this command, FlashMandelWOS will quit.

INPUTS

FORCE:

If specified, FlashMandelWOS will quit without any questions.

FAST:

If specified, the command will be executed as fast as possible, this means no fading.

RETURNS

RC:

error-level, 0 if successfull.

RC2:

error-code, 0 if successfull.

NOTES

-

SEE ALSO

-

EXAMPLE

"QUIT" "FORCE"

BUGS

FAST-parameter has no effect.

1.15 arexx-command SHOW_TITLE

NAME

SHOW_TITLE - turn screen-title on/off.

SYNOPSIS

"SHOW_TITLE" "BOOLEAN/A/N"

FUNCTION

Specify, if the screen-title should be visible or not.

INPUTS

BOOLEAN:

0 = Hide screen-title.

1 = Show screen-title.

RETURNS

RC:

error-level, 0 if successfull.

RC2:

error-code, 0 if successfull.

NOTES

-

SEE ALSO

-

EXAMPLE

"SHOW_TITLE" "0"

BUGS

-

1.16 arexx-command SHOW_ELAPSED

NAME

SHOW_ELAPSED - show elapsed time.

SYNOPSIS

"SHOW_ELAPSED"

FUNCTION

Show needed time for the last calculation in the screen title.

INPUTS

-

RETURNS

RC:

error-level, 0 if successfull.

RC2:

error-code, 0 if successfull.

NOTES

-

SEE ALSO

-

EXAMPLE

"SHOW_ELAPSED"

BUGS

-

1.17 arexx-command SET_COORDINATES

NAME

SET_COORDINATES - set the limits.

SYNOPSIS

"SET_COORDINATES"

"SET_COORDINATES" "STEM"

FUNCTION

Set the limits for calculations. If you call this command without any parameter, it behaves just like choosing the menu-item "Limits...".

INPUTS

STEM:

Name of the stem-variable, which contains all needed data.

Following extensions are defined:

.RMIN: Left complex-coordinate.

.RMAX: Right complex-coordinate.

.IMIN: Bottom complex-coordinate.

.IMAX: Top complex-coordinate.

.JKRE: Julia-real constant.

.JKIM: Julia-imag constant.

RETURNS

RC:

error-level, 0 if successfull.

RC2:

error-code, 0 if successfull.

NOTES

This command only sets the parameter. If it should take effect to your display, you have to call the command RENDER afterwards.

SEE ALSO

GET_COORDINATES

EXAMPLE

```
COORD.RMIN = -2.0
COORD.RMAX = +1.2
COORD.IMIN = -1.2
COORD.IMAX = +1.2
COORD.JKRE = -0.72
COORD.JKIM = -0.26
"SET_COORDINATES" "COORD"
IF RC THEN SAY "Can't set coordinates:" ErrorText(RC2)
```

BUGS

-

1.18 arexx-command GET_COORDINATES

NAME

GET_COORDINATES - get the limits.

SYNOPSIS

```
"GET_COORDINATES" "STEM/A"
```

FUNCTION

Get the limits for calculations.

INPUTS

STEM:

Name of the stem-variable, which will contain all coordinates.

Following extensions are defined:

.RMIN: Left complex-coordinate.

.RMAX: Right complex-coordinate.

.IMIN: Bottom complex-coordinate.

.IMAX: Top complex-coordinate.

.JKRE: Julia-real constant.

.JKIM: Julia-imag constant.

RETURNS

RC:

error-level, 0 if successfull.

RC2:

error-code, 0 if successfull.

NOTES

-

SEE ALSO

SET_COORDINATES

EXAMPLE

'GET_COORDINATES' 'COORDS'

IF RC THEN SAY "Can't get coordinates:" ErrorText(RC2)

ELSE DO

SAY "RMIN: " COORDS.RMIN; /* Left */

SAY "RMAX: " COORDS.RMAX; /* Right */

SAY "IMIN: " COORDS.IMIN; /* Bottom */

SAY "IMAX: " COORDS.IMAX; /* Top */

SAY "JKRE: " COORDS.JKRE; /* Julia Real */

SAY "JKIM: " COORDS.JKIM; /* Julia Imag */

END

BUGS

-

1.19 arexx-command SET_ITERATIONS

NAME

SET_ITERATIONS - set iterations.

SYNOPSIS

"SET_ITERATIONS"

"SET_ITERATIONS" "NUMBER/N"

FUNCTION

Specify the number of iterations. If this command is called without any parameters, it behaves just like choosing the menu-item "Iterations/Custom...".

INPUTS

NUMBER:

a numeric value between 32 and 32768.

RETURNS

RC:

error-level, 0 if successfull.

RC2:

error-code, 0 if successfull.

NOTES

This command only sets the parameter. If it should take effect to your display, you have to call the command RENDER afterwards.

SEE ALSO

GET_ITERATIONS

EXAMPLE

"SET_ITERATIONS" "1024"

IF RC THEN SAY "Can’t set iterations:" ErrorText(RC2)

BUGS

-

1.20 arexx-command GET_ITERATIONS

NAME

GET_ITERATIONS - get number of iterations.

SYNOPSIS

"GET_ITERATIONS" "VAR/A"

FUNCTION

Get the number of iterations.

INPUTS

VAR:

Name of the variable, which should contain the numer of iterations after calling this command.

The result is a number between 32 and 32768.

RETURNS

RC:

error-level, 0 if successfull.

RC2:

error-code, 0 if successfull.

NOTES

-

SEE ALSO

SET_ITERATIONS

EXAMPLE

"GET_ITERATIONS" "ITERATIONS"

IF RC THEN SAY "Can’t set iterations:" ErrorText(RC2)

ELSE SAY "Number of iterations:" ITERATIONS

BUGS

-

1.21 arexx-command SET_POWER

NAME

SET_POWER - set power-parameter.

SYNOPSIS

"SET_POWER" "NUMBER/A/N"

FUNCTION

Specify the power-parameter.

INPUTS

NUMBER:

a numeric value. Following values are possible:

2

4

8

16

32

64

128

256

512

1024

2048

RETURNS

RC:

error-level, 0 if successfull.

RC2:

error-code, 0 if successfull.

NOTES

This command only sets the parameter. If it should take effect to your display, you have to call the command RENDER afterwards.

SEE ALSO

GET_POWER

EXAMPLE

"SET_POWER" "1024"

IF RC THEN SAY "Can’t set power:" ErrorText(RC2)

BUGS

-

1.22 arexx-command GET_POWER

NAME

GET_POWER - get power-parameter.

SYNOPSIS

"GET_POWER" "VAR/A"

FUNCTION

Get the power-parameter.

INPUTS

VAR:

Name of the variable, which should contain the value of the power-parameter after calling this command.

Following results are possible:

2

4

8

16

32

64

128

256

512

1024

2048

RETURNS

RC:

error-level, 0 if successfull.

RC2:

error-code, 0 if successfull.

NOTES

-

SEE ALSO

SET_POWER

EXAMPLE

"GET_POWER" "POWER"

IF RC THEN SAY "Can’t get power-parameter:" ErrorText(RC2)

ELSE SAY "Power-parameter:" POWER

BUGS

-

1.23 arexx-command SET_PRIORITY

NAME

SET_PRIORITY - set the priority of the own task.

SYNOPSIS

"SET_PRIORITY" "PRI/A"

FUNCTION

Specify the tasks priority.

INPUTS

PRI:

a numeric value between -5 and +5.

RETURNS

RC:

error-level, 0 if successfull.

RC2:

error-code, 0 if successfull.

NOTES

-

SEE ALSO

GET_PRIORITY

EXAMPLE

"SET_PRIORITY" "+1"

IF RC THEN SAY "Can’t set priority:" ErrorText(RC2)

BUGS

-

1.24 arexx-command GET_PRIORITY

NAME

GET_PRIORITY - get priority of the own task.

SYNOPSIS

"GET_PRIORITY" "VAR/A"

FUNCTION

Get the priority of this task.

INPUTS

VAR:

Name of the variable, which should contain the value of the priority after calling this command.

The returned values are between -5 and +5.

RETURNS

RC:

error-level, 0 if successfull.

RC2:

error-code, 0 if successfull.

NOTES

-

SEE ALSO

SET_PRIORITY

EXAMPLE

"GET_PRIORITY" "PRIORITY"

IF RC THEN SAY "Can't get the priority:" ErrorText(RC2)

ELSE SAY "Priority:" PRIORITY

BUGS

-

1.25 arexx-command SET_COLOR_REMAP

NAME

SET_COLOR_REMAP - set the color-remap-method.

SYNOPSIS

"SET_COLOR_REMAP" "INDEX/A/N"

FUNCTION

Specify the method, how colors are assigned.

INPUTS

INDEX:

a numeric value between 1 and 8.

The numbers have following meanings:

1 = Linear

2 = Ln (x)

3 = Repeated

4 = Sqrt (x)

5 = x^2-x

6 = Sqrt(x^3-x^2-x)

7 = Sinh (Ln (x^3))

8 = Cosh (Log (x^3))

RETURNS

RC:

error-level, 0 if successfull.

RC2:

error-code, 0 if successfull.

NOTES

This command only sets the parameter. If it should take effect to your display, you have to call the command RENDER afterwards.

SEE ALSO

GET_COLOR_REMAP

EXAMPLE

"SET_COLOR_REMAP" "2"

IF RC THEN SAY "Can't set color-remap-method:" ErrorText(RC2)

BUGS

-

1.26 arexx-command GET_COLOR_REMAP

NAME

GET_COLOR_REMAP - get the color-remap-method.

SYNOPSIS

"GET_COLOR_REMAP" "VAR/A"

FUNCTION

Get the specified the method, how colors are assigned.

INPUTS

VAR:

Name of the variable, which should contain the color-remap-method after calling this command.

Following results are possible:

1 = Linear

2 = Ln (x)

3 = Repeated

4 = Sqrt (x)

5 = x^2-x

6 = Sqrt(x^3-x^2-x)

7 = Sinh (Ln (x^3))

8 = Cosh (Log (x^3))

RETURNS

RC:

error-level, 0 if successfull.

RC2:

error-code, 0 if successfull.

NOTES

-

SEE ALSO

SET_COLOR_REMAP

EXAMPLE

"GET_COLOR_REMAP" "REMAP"

IF RC THEN SAY "Can't get color-remap-method:" ErrorText(RC2)

ELSE SAY "Color-remap-method:" REMAP

BUGS

-

1.27 arexx-command SET_FRACTAL_TYPE

NAME

SET_FRACTAL_TYPE - set the fractal type.

SYNOPSIS

"SET_FRACTAL_TYPE" "INDEX/A/N"

FUNCTION

Specify the type of fractal, which should be rendered.

INPUTS

INDEX:

a numeric value between 1 and 2.

The numbers have following meanings:

1 = Julia rendering formula.

2 = Mandelbrot rendering formula.

RETURNS

RC:

error-level, 0 if successfull.

RC2:

error-code, 0 if successfull.

NOTES

This command only sets the parameter. If it should take effect to your display, you have to call the command RENDER afterwards.

SEE ALSO

GET_FRACTAL_TYPE

EXAMPLE

"SET_FRACTAL_TYPE" "2"

IF RC THEN SAY "Can’t set the fractal-type:" ErrorText(RC2)

BUGS

-

1.28 arexx-command GET_FRACTAL_TYPE

NAME

GET_FRACTAL_TYPE - get the fractal type.

SYNOPSIS

"GET_FRACTAL_TYPE" "VAR/A"

FUNCTION

Get the specified type of the fractal.

INPUTS

VAR:

Name of the variable, which should contain the color-remap-method after calling this command.

Following results are possible:

1 = Julia rendering formula.

2 = Mandelbrot rendering formula.

RETURNS

RC:

error-level, 0 if successfull.

RC2:

error-code, 0 if successfull.

NOTES

-

SEE ALSO

SET_FRACTAL_TYPE

EXAMPLE

"GET_FRACTAL_TYPE" "TYPE"

IF RC THEN SAY "Can’t get the fractal-type:" ErrorText(RC2)

ELSE SAY "Fractal-type:" TYPE

BUGS

-

1.29 arexx-command SET_CPU_TYPE

NAME

SET_CPU_TYPE - set the CPU type.

SYNOPSIS

"SET_CPU_TYPE" "INDEX/A/N"

FUNCTION

Specify the type of CPU, which should be used for rendering.

INPUTS

INDEX:

a numeric value between 1 and 2.

The numbers have following meanings:

1 = 68k-processor.

2 = PPC-processor.

RETURNS

RC:

error-level, 0 if successfull.

RC2:

error-code, 0 if successfull.

NOTES

-

SEE ALSO

GET_CPU_TYPE

EXAMPLE

"SET_CPU_TYPE" "2"

IF RC THEN SAY "Can't set the cpu-type:" ErrorText(RC2)

BUGS

-

1.30 arexx-command GET_CPU_TYPE

NAME

GET_CPU_TYPE - get the CPU type.

SYNOPSIS

"GET_CPU_TYPE" "VAR/A"

FUNCTION

Get the specified type of CPU, which is used for rendering.

INPUTS

VAR:

Name of the variable, which should contain the color-remap-method after calling this command.
Following results are possible:
1 = 68k-processor.
2 = PPC-processor.
RETURNS
RC:
error-level, 0 if successfull.
RC2:
error-code, 0 if successfull.
NOTES
-
SEE ALSO
GET_CPU_TYPE
EXAMPLE
"SET_CPU_TYPE" "2"
IF RC THEN SAY "Can’t set the cpu-type:" ErrorText(RC2)
BUGS
-

1.31 arexx-command SHOW_PREVIEW

NAME
SHOW_PREVIEW - open preview-requester.
SYNOPSIS
"SHOW_PREVIEW"
FUNCTION
Opens the preview-requester, just like choosing the ’Preview’-menu-item. Must be closed by clicking the ’OK’-button.
INPUTS
-
RETURNS
RC:
error-level, 0 if successfull.
RC2:
error-code, 0 if successfull.
NOTES
-
SEE ALSO
-
EXAMPLE
"SHOW_PREVIEW"
BUGS
-

1.32 arexx-command RENDER

NAME

RENDER - render the fractal.

SYNOPSIS

"RENDER" "FORCE/S"

FUNCTION

Renders the fractal depending on the actual settings.

INPUTS

FORCE:

If specified, the fractal will be rendered without any questions.

RETURNS

RESULT:

Elapsed time for the rendering in seconds.

RC:

error-level, 0 if successfull.

RC2:

error-code, 0 if successfull.

NOTES

-

SEE ALSO

-

EXAMPLE

"RENDER" "FORCE"

BUGS

-

1.33 arexx-command UNDO

NAME

UNDO - undo the last changes.

SYNOPSIS

"UNDO"

FUNCTION

Resets the last changes to the render-settings.

INPUTS

-

RETURNS

RC:

error-level, 0 if successfull.

RC2:

error-code, 0 if successfull.

NOTES

This command only sets the parameters. If it should take effect to your display, you have to call the command RENDER afterwards.

SEE ALSO

-

EXAMPLE

"UNDO"

BUGS

-

1.34 arexx-command SET_FRAME

NAME

SET_FRAME - set the zoom-frame.

SYNOPSIS

"SET_FRAME" "STEM/A"

FUNCTION

Set the coordinates of the zoom-frame.

INPUTS

STEM:

Name of the stem-variable, which contains all needed data.

Following extensions are defined:

.LEFT: Left corner of the frame-box.

.TOP: Top corner of the frame-box.

.RIGHT: Right corner of the frame-box..

.BOTTOM: Bottom corner of the frame-box.

RETURNS

RC:

error-level, 0 if successfull.

RC2:

error-code, 0 if successfull.

NOTES

Please check the screen-dimensions before you call this command. Maybe your desired frame is bigger than the screen. Use GET_SCREENATTRS for it.

The right-coordinate must be higher than the left-coordinate.

The bottom-coordinate must be higher than the top-coordinate.

SEE ALSO

GET_FRAME

EXAMPLE

FRAME.LEFT = 10

FRAME.TOP = 20

FRAME.RIGHT = 200

FRAME.BOTTOM = 300

"SET_FRAME" "FRAME"

IF RC THEN SAY "Can’t set frame:" ErrorText(RC2)

BUGS

-

1.35 arexx-command GET_FRAME

NAME

GET_FRAME - get the zoom-frame.

SYNOPSIS

"GET_FRAME" "STEM/A"

FUNCTION

Get the coordinates of the zoom-frame.

INPUTS

STEM:

Name of the stem-variable, which will contain the desired coordinates after calling this command.

Following extensions are defined:

.LEFT: Left corner of the frame-box.

.TOP: Top corner of the frame-box.

.RIGHT: Right corner of the frame-box..

.BOTTOM: Bottom corner of the frame-box.

RETURNS

RC:

error-level, 0 if successfull.

RC2:

error-code, 0 if successfull.

NOTES

-

SEE ALSO

SET_FRAME

EXAMPLE

"GET_FRAME" "FRAME"

IF RC THEN SAY "Can’t get frame:" ErrorText(RC2)

ELSE DO

SAY "Left:" FRAME.LEFT;

SAY "Top:" FRAME.TOP;

SAY "Right:" FRAME.RIGHT;

SAY "Bottom:" FRAME.BOTTOM;

END

BUGS

-

1.36 arexx-command ZOOM

NAME

ZOOM - zoom into the picture.

SYNOPSIS

"ZOOM" "LEFT/A/N,TOP/A/N,WIDTH/A/N,HEIGHT/A/N,REPEATS/N,FAST/S,STEPS/N"

FUNCTION

Zoom into the picture according to the given dimensions.

INPUTS

LEFT:

Left edge of the zoom-frame.

TOP:

Top edge of the zoom-frame.

WIDTH:

Width edge of the zoom-frame.

HEIGHT:

Height edge of the zoom-frame.

REPEATS:

Number of repeats.

FAST:

If specified, the zoom is rendered as fast as possible, this means no fading.

STEPS:

Defines, in how many steps the zoom is divided (Sorry, not supported yet).

RETURNS

RESULT:

Elapsed time for the zoom in seconds.

RC:

error-level, 0 if successfull.

RC2:

error-code, 0 if successfull.

NOTES

-

SEE ALSO

-

EXAMPLE

"ZOOM" "30 40 400 300 2" "FAST"

IF RC THEN SAY "Can’t zoom:" ErrorText(RC2)

ELSE SAY "Elapsed time:" RESULT

BUGS

The STEPS-parameter has no effect.

1.37 arexx-command DO_COLORCYCLING

NAME

DO_COLORCYCLING - start color-cycling.

SYNOPSIS

"DO_COLORCYCLING" "FORWARD/S,BACKWARD/S"

FUNCTION

Starts the color-cycling, just like choosing the menu-item "Cycle...". To stop it, you have to click into the window.

INPUTS

FORWARD:

Colors should move in forward direction.

BACKWARD:

Colors should move in backward direction.

RETURNS

RC:

error-level, 0 if successfull.

RC2:

error-code, 0 if successfull.

NOTES

Of cource only one parameter is possible, otherwise the command fails.

SEE ALSO

-

EXAMPLE

"DO_COLORCYCLING" "FORWARD"

BUGS

-

1.38 arexx-command SET_DELAY

NAME

SET_DELAY - set the delay for color-cycling.

SYNOPSIS

"SET_DELAY"

"SET_DELAY" "SECONDS/N"

FUNCTION

Specify the number of seconds between a color-cycling-step. If this command is called without any parameter, it behaves just like choosing the menu-item "Cycle/Delay...".

INPUTS

SECONDS:

a numeric value between 0 and 200.

RETURNS

RC:

error-level, 0 if successfull.

RC2:

error-code, 0 if successfull.

NOTES

-

SEE ALSO

GET_DELAY

EXAMPLE

"SET_DELAY" "10"

IF RC THEN SAY "Can’t set delay-time:" ErrorText(RC2)

BUGS

-

1.39 arexx-command GET_DELAY

NAME

GET_DELAY - get the delay-time for color-cycling.

SYNOPSIS

"GET_DELAY" "VAR/A"

FUNCTION

Get the number of seconds between a color-cycling-step.

INPUTS

VAR:

Name of the variable, which will contain the desired delay-time after calling this command.

The result is a numeric value between 0 and 200.

RETURNS

RC:

error-level, 0 if successfull.

RC2:

error-code, 0 if successfull.

NOTES

-

SEE ALSO

SET_DELAY

EXAMPLE

"GET_DELAY" "TIME"

IF RC THEN SAY "Can't get delay-time:" ErrorText(RC2)

ELSE SAY "Delay-time:" TIME

BUGS

-

1.40 arexx-command SET_PALETTE

NAME

SET_PALETTE - set the screens palette.

SYNOPSIS

"SET_PALETTE"

"SET_PALETTE" "STEM,FAST/S"

FUNCTION

Set the screen-colors. If this command is called without any parameter, it behaves just like choosing the menu-item "Palette...".

INPUTS

STEM:

Name of the stem-variable, which contains the color-values. If a color is not defined, the color will not change. The colors must defined like this:

COLORS.x = '00RRGGBB'x /* RR = red value, GG = green value, BB = blue value. */

For example:

The name of the variable is 'COLORS'. So the first color must defined like this:

COLORS.0 = '008833BB'x

Please notify, that the value should be written as a hex-number.

FAST:

If specified, the colors are changed as fast as possible, this means no fading.

RETURNS

RC:

error-level, 0 if successfull.

RC2:

error-code, 0 if successfull.

NOTES

Please determine the number of available colors using the GET_SCREENATTRS command.

SEE ALSO

GET_PALETTE

EXAMPLE

COLORS.0 = ’008833BB’x

"SET_PALETTE" "COLORS"

IF RC THEN SAY "Can’t set palette:" ErrorText(RC2)

BUGS

-

1.41 arexx-command GET_PALETTE

NAME

GET_PALETTE - get the screens palette.

SYNOPSIS

"GET_PALETTE" "STEM/A"

FUNCTION

Get the screen-colors.

INPUTS

STEM:

Name of the stem-variable, which will contains the color-values.

The colors are defined like this:

00RRGGBB RR = red value, GG = green value, BB = blue value.

Important:

The datatype of the color-definition is a string, which contains a binary value. Because of this,

you can use this values directly for SET_PALETTE.

For example:

’GET_PALETTE’ ’COLORS’

COLORS.1 = ’00FFFFFF’x

’SET_PALETTE’ ’COLORS’

If you want to print the value on screen, you must convert the string-value to hex-value:

SAY "Color 1:" C2X(COLORS.1)

RETURNS

RC:

error-level, 0 if successfull.

RC2:

error-code, 0 if successfull.

NOTES

Please determine the number of available colors using the GET_SCREENATTRS command.

SEE ALSO

SET_PALETTE

EXAMPLE

"GET_PALETTE" "COLORS"

IF RC THEN SAY "Can’t get palette:" ErrorText(RC2)

ELSE DO

DO I = 0 TO NumColors

SAY "Color" I ||": ’" || C2X(COLORS.I) || "’x";

END I

END

BUGS

-

1.42 arexx-command SET_SCREENATTRS

NAME

SET_SCREENATTRS - set screens-attributes.

SYNOPSIS

"SET_SCREENATTRS""

"SET_SCREENATTRS" "STEM,FAST/S"

FUNCTION

Set the attributes of the screen. If this command is called without any parameter, it behaves just like choosing the menu-item "Screen mode...".

INPUTS

STEM:

Name of the stem-variable, which contains the needed data.

Following extentions are defined:

.MODE: desired screen-mode.

.WIDTH: width of the screen.

.HEIGHT: height of the screen.

.DEPTH: screen-depth.

.OSCANTYPE: definition of the desired overscan-type. Following definitions are possible:

- "TEXT"

- "STANDARD"

- "MAX"

- "VIDEO"

Default is "TEXT".

FAST:

If specified, the attributes are changed as fast as possible, this means no fading.

RETURNS

RC:

error-level, 0 if successfull.

RC2:

error-code, 0 if successfull.

NOTES

-

SEE ALSO

GET_SCREENATTRS

EXAMPLE

DIMS.MODE = '40D20001'x

DIMS.WIDTH = 640

DIMS.HEIGHT = 480

DIMS.DEPTH = 8

DIMS.OSCANTYPE = "TEXT"

"SET_SCREENATTRS" "DIMS"

IF RC THEN SAY "Can't set screen-attributes:" ErrorText(RC2)

BUGS

-

1.43 arexx-command GET_SCREENATTRS

NAME

GET_SCREENATTRS - get screens-attributes.

SYNOPSIS

"GET_SCREENATTRS" "STEM/A"

FUNCTION

Get the attributes of the screen.

INPUTS

STEM:

Name of the stem-variable, which will contain the desired data.

Following extentions are defined:

.MODE: desired screen-mode.

.WIDTH: width of the screen.

.HEIGHT: height of the screen.

.DEPTH: screen-depth.

Important:

The datatype of the screenmode-definition is a string, which contains a binary value. Because of this,

you can use this values directly for SET_SCREENATTRS.

For example:

'GET_SCREENATTRS' 'DIMS'

'SET_SCREENATTRS' 'DIMS'

If you want to print the value on screen, you must convert the string-value to hex-value:

SAY "Screen-mode:" C2X(DIMS.MODE)

RETURNS

RC:

error-level, 0 if successfull.

RC2:

error-code, 0 if successfull.

NOTES

-

SEE ALSO

SET_SCREENATTRS

EXAMPLE

"GET_SCREENATTRS" "DIMS"

IF RC THEN SAY "Can't get screen-attributes:" ErrorText(RC2)

ELSE DO

SAY "screenmode: '" || C2X(DIMS.MODE) || "'x";

SAY "width:" DIMS.WIDTH

SAY "height:" DIMS.HEIGHT

SAY "depth:" DIMS.DEPTH

END

BUGS

-

1.44 arexx-command SET_FONT

NAME

SET_FONT - set screen-font.

SYNOPSIS

"SET_FONT"

"SET_FONT" "STEM,FAST/S"

FUNCTION

Set the screen-font. If this command is called without any parameter, it behaves just like choosing the menu-item "Font settings...".

INPUTS

STEM:

Name of the stem-variable, which contains the needed data.

Following extentions are defined:

.FONTNAME: name of the desired font.

.SIZE: font size.

.STYLE: font-style. Following definitions are possible:

- "NORMAL"
- "BOLD"
- "ITALIC"
- "UNDERLINED"
- "EXTENDED"

Default is "NORMAL"

FAST:

If specified, the font is changed as fast as possible, this means no fading.

RETURNS

RC:

error-level, 0 if successfull.

RC2:

error-code, 0 if successfull.

NOTES

-

SEE ALSO

GET_FONT

EXAMPLE

```
FONT.FONTNAME = "diamond.font"
FONT.SIZE = 13
FONT.STYLE = "BOLD,ITALIC"
"SET_FONT" "FONT"
IF RC THEN SAY "Can’t set screen-font:" ErrorText(RC2)
```

BUGS

-

1.45 arexx-command GET_FONT

NAME

GET_FONT - get screen-font.

SYNOPSIS

"GET_FONT" "STEM/A"

FUNCTION

Get the screen-font definitions.

INPUTS

STEM:

Name of the stem-variable, which will contain the desired data.

Following extentions are defined:

.FONTNAME: name of the desired font.

.SIZE: font-size.

.STYLE: font-style. Following definitions are possible:

- "NORMAL"
- "BOLD"
- "ITALIC"
- "UNDERLINED"
- "EXTENDED"

RETURNS

RC:

error-level, 0 if successfull.

RC2:

error-code, 0 if successfull.

NOTES

-

SEE ALSO

SET_FONT

EXAMPLE

```
"GET_FONT" "FONT"
IF RC THEN SAY "Can't get screen-font:" ErrorText(RC2)
ELSE DO
SAY "fontname:" FONT.FONTNAME
SAY "font-size:" FONT.SIZE
SAY "font-style:" FONT.STYLE
END
```

BUGS

-

1.46 arexx-command START_TIMER

NAME

START_TIMER - start timer.

SYNOPSIS

"START_TIMER"

FUNCTION

Set timer to 0 and start the timer. Usefull for speed calculations.

INPUTS

-

RETURNS

RC:

error-level, 0 if successfull.

RC2:

error-code, 0 if successfull.

NOTES

-

SEE ALSO

STOP_TIMER

EXAMPLE

"START_TIMER"

BUGS

-

1.47 arexx-command STOP_TIMER

NAME

STOP_TIMER - stop timer.

SYNOPSIS

"STOP_TIMER"

FUNCTION

Stop timer and return the passed time since the last call of the START_TIMER-command. Usefull for speed calculations.

INPUTS

-

RETURNS

RESULT:

the passed time since the last START_TIMER-call in seconds.

RC:

error-level, 0 if successfull.

RC2:

error-code, 0 if successfull.

NOTES

-

SEE ALSO

START_TIMER

EXAMPLE

"STOP_TIMER"

SAY "needed time:" RESULT

BUGS

-

1.48 arexx-command PLAY_SOUND

NAME

PLAY_SOUND - play a sound-file.

SYNOPSIS

"PLAY_SOUND" "PATH/A,ASYNC/S,REPEAT/S"

FUNCTION

Plays a sound-file using the AmigaOS-datatypes.

INPUTS

PATH:

The path of the sound-file.

ASYNC:

If TRUE, FlashMandelWOS starts an own process for playing the sound.

REPEAT:

If TRUE, the sound will be played in an endless loop. Will be ignored, if ASYNC is not specified.

RETURNS

RC:

error-level, 0 if successfull.

RC2:

error-code, 0 if successfull.

NOTES

-

SEE ALSO

STOP_SOUND

EXAMPLE

"PLAY_SOUND" "FLASHMANDEL:Welcome.8svx"

IF RC THEN SAY "Can't play sound:" ErrorText(RC2)

BUGS

-

1.49 arexx-command STOP_SOUND

NAME

STOP_SOUND - stop sound.

SYNOPSIS

"STOP_SOUND"

FUNCTION

stop a sound started with PLAY_SOUND.

INPUTS

-

RETURNS

RC:

error-level, 0 if successfull.

RC2:

error-code, 0 if successfull.

NOTES

-

SEE ALSO

PLAY_SOUND

EXAMPLE

"STOP_SOUND"

BUGS

-

1.50 arexx-error-levels

Following error-levels are defined:

0 = OK everything is o.k.

5 = WARN just a feedback, e.g. a requester was cancelled.

10 = ERROR an error occured, e.g. a wrong parameter was defined.

20 = FATAL a serious error - program should quit.

Every command returns one of this level-codes in the variable 'RC'.

1.51 arexx-error-codes

Following error-codes are defined:

0 = everything is o.k.

01 = program not found

02 = execution halted

03 = no memory available

04 = invalid character in program

05 = unmatched quote

06 = unterminated comment

07 = clause too long

08 = unrecognized token

09 = symbol or string too long

10 = invalid message packet

11 = command string error

12 = error return from function

13 = host environment not found

14 = required library not found

15 = function not found

16 = no return value

17 = wrong number of arguments

18 = invalid argument to function

19 = invalid PROCEDURE

20 = unexpected THEN/ELSE

21 = unexpected WHEN/OTHERWISE

22 = unexpected LEAVE or ITERATE

23 = invalid statement in SELECT

24 = missing THEN clauses

25 = missing OTHERWISE

26 = missing or unexpected END

27 = symbol mismatch on END

28 = invalid DO syntax

29 = incomplete DO/IF/SELECT

30 = label not found

31 = symbol expected

32 = string or symbol expected

33 = invalid sub-keyword

34 = required keyword missing

35 = extraneous characters

36 = sub-keyword conflict

37 = invalid template

38 = invalid TRACE request

39 = uninitialized variable

40 = invalid variable name

41 = invalid expression

42 = unbalanced parentheses

43 = nesting level exceeded

44 = invalid expression result

45 = expression required

46 = boolean value not 0 or 1

47 = arithmetic conversion error

48 = invalid operand

Every command returns one of this error-codes in the variable ’RC2’.