0O0O

COLLABORATORS			
	TITLE : 0O0O		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		August 8, 2022	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

0O0O

1.1 Cloanto - The Kara Collection - ColorType

Cloanto ColorType

ColorType has been acclaimed as the best bitmapped fonts editor for the Amiga by enthusiasts and professionals worldwide. It is the leading tool for handling fonts in up to 256 colors (no size limit, RTG compatible). Its Rexx interface enables integration with other packages and makes external functions directly accessible from the tool bar. Special functions for anti-aliasing (even using an Amiga vector font as a point of departure), right-to-left fonts, texturing and image processing (in combination with Personal Paint) are provided, among others. This makes it an ideal companion to video applications, and a must for titling.

1. First Steps

ColorType is launched with a double-click on its Workbench icon, on any Amiga with 1 Mbyte of free RAM (ideally 1 or 2 Mbytes of Chip RAM). Color fonts are supported from version 2 (or higher) of the Amiga operating system.

It is very easy to explore ColorType by choosing from the menus and gadgets, but it remains recommended to make a backup copy of any data or software before experimenting with functions which write data.

Because ColorType is a result of Personal Paint development applied to fonts, many users of Personal Paint will be familiar with the tool bar and the main functions of ColorType. For more information, please refer to the documentation of Personal Paint . The Personal Fonts Maker package, which provides additional features for black and white fonts (such as printer downloading), contains more extended documentation on fonts.

Amiga fonts are loaded and saved with the Load Amiga Font and Save Amiga Font commands in the Project menu. The format of the font (size, number of colors) can be changed from the Font Format requester, which is displayed from the Settings menu. The color palette is edited by pressing <p> - equivalent to selecting Edit in the "Color/Palette" menu.

ColorType automatically flushes unused font data from RAM if the system

has less than 100 Kbytes of contiguous Chip RAM (otherwise, the Amiga operating system would tend to accumulate font information for each font opened).

The Show Font command in the Project menu opens a screen with all characters in the font (excluding "Off" characters). The screen uses vertical autoscroll (moving the mouse to the top or bottom of the screen scrolls the page), if necessary, and it is closed by pressing any key or mouse button.

Most gadgets on the tool bar react differently depending on whether they are selected with the left or the right mouse button (which in most cases activates the Tool Settings requester). The drawing tools can be selected in the upper or lower part to activate different effects (e.g. rectangular outlines or full boxes).

There are a few special gadgets which are not normally found in a painting program. The top five gadgets are:

- Font
- Character
- X Size
- Space
- Kerning

All five can be selected with the left mouse button to increase the value of the setting, and with the right button to decrease it.

The Font gadget allows the user to select the font environment. ColorType has two working environments which can contain independent fonts. Or one environment could contain a font, and the other a large picture used as a storage area for clips.

The Character gadget allows the user to move forwards and backwards in the font. Characters can also be selected by clicking on "??", which activates the quick character selection window. Amiga fonts can store up to 257 characters (ranging from 0 to 256). The character code 256 is reserved for the undefined character, which is generally used by fonts stored in the Amiga font format.

The horizontal size of the character can be controlled by the X Size value. This size, which corresponds to the size of the grid, can be made smaller or larger. The minimum size is 1 (one column of dots), while the grid can be as large as that defined by the Font Format settings. This parameter does not need to include additional spaces at the left or right of the character. The Space and Kerning parameters should be used for this purpose. Unused characters are disabled with the On/Off gadget.

The Space parameter defines the logical horizontal size of the character, starting from the beginning of the bitmap. The value associated with this parameter is usually slightly higher than the X Size parameter. This parameter affects the placement of the following character.

The Kerning parameter (also called Left Offset) determines the starting position of the character when it is typed or printed. The parameter specifies an offset from the position immediately following the last character which was printed or displayed. This value is normally left to

0. Negative values are allowed, and are used for example in certain handwriting effects.

The Space and Kerning parameters do not affect the bitmap of the character. Instead, they control the horizontal spacing between the characters when several characters are grouped together to print, display or otherwise output text.

The Rexx gadget (between the Airbrush and the Brush tools) can be selected with the right mouse button to select a new Rexx script, or with the left button to repeat a previously selected action. ColorType comes with a set of macros which can easily be explored with the About function in the Macro Selection selection requester. The ColorType Advanced Features section explains macros and the Rexx interface in more detail.

The two gadgets which appear under the Brush tool are the Memory tool and the Chop tool. Memory can be selected with the left button to copy the current character (and spacing information) to an internal buffer, and with the right button to paste it. Chop narrows the character grid, cutting off empty columns. Chop can be selected with the right mouse button to center the character without removing empty columns. The other gadgets move and flip the character.

The little arrows between the tool bar and the character editing box indicate up to four reference points, which are useful when designing new fonts. These can be set by selecting a position with the left mouse button in the reference point area and by pressing a function key between <F1> and <F4>.

The box to the left of the Grid Color tool indicates whether the font has a remappable foreground color. Some applications can change this color to change the font’s appearance in a particular environment. To define the color, just click on the box and then pick a color from the palette. To disable this feature, click on the box and then click anywhere outside the palette.

Amiga fonts are normally accessed through the "FONTS:" assigned path. The ColorType package includes the "AssignFonts" utility to reassign "FONTS:". This utility can be used, for example, to reassign the path from "SYS:Fonts" to "ColorType:Fonts" and vice versa. The font requester of ColorType provides this capability directly, but other programs may require a tool such as AssignFonts.

1.1 Tutorial 1: How to Create a Color Font in Minutes

The "Fonts" drawer includes the color font "TenMinutes", which you can design in a few minutes following these instructions.

Select Load Amiga Font from the Project menu. From the list of fonts which are included with the Amiga, choose CGTimes. This is a vector font which can easily be scaled to any size. Type "55s" in the string gadget to the right of the gadget with the font name. The "s" means that the system should scale the font to a size of 55, rather than picking the font with the closest size (from the pre-scaled bitmaps which are already available). Press the "Bold A" gadget to make the font bold. If you do not have this font, pick any black and white font with a comparable size. Bold

fonts are preferred in this example, because the characters have to be filled.

Select Font Format from the Settings menu. Move the Colors slider to 8. The font now has 8 colors. Select "Color/Palette/Load" and load the "TenMinutes.col" palette which is included with ColorType.

Try to set "Settings/Grid/Dots" to see if you prefer this option, which affects how the characters are displayed in the editing box.

On the tool bar, click on the Fill tool with the right mouse button. This displays the Fill Area Parameters requester. Make sure the Fill Type cycle gadget indicates "Gradient". Click on the gadget with the circle until it shows four arrows inside the circle (Shape gradient type). Click on the other gadget (to its left) so that it shows an arrow with four boxes (Manual gradient mode). Without leaving the requester, click on the second color in the tool bar palette area (the pale yellow box, to the right of the black box) with the left mouse button. Select the last color (brown) with the right mouse button. Press Proceed.

Now the fill tool can be applied to the characters of the font. Click (with the left mouse button) anywhere on each character. If you select the "Settings/Dynamic View" option you will be able to see how the changes affect a sample of text.

You’ve created a color font!

1.2 Tutorial 2: How to Create an Anti-Aliased Font

This example shows you how to anti-alias a black and white font, where intermediate shades of gray are used to create the illusion of a higher resolution. This effect is frequently used in the video field, and can be very useful when working with a paint program, to preserve the quality of small text.

The program option used in this example is called Color Average Resize. Activate it from the "Settings/Stretch" menu.

In general, to create an anti-aliased font you need to load an Amiga (possibly vector) font, add intermediate shades of gray or color to the palette and then reduce the font size ("Settings/Font Format" menu). The more colors you add, the better the result, and the smaller you can make the font with respect to the original.

In this example we will load a font twice the size of the anti-aliased font. Following the instructions of the previous lesson, load CGTimes 50, or any other font of similar size. This font does not need to be bold (as in the previous example).

From the Font Format requester (Settings menu) set the number of colors to 4.

Now display the Edit Palette requester (by pressing <p>, or from the "Colors/Palette" menu) and arrange the four colors so that white appears first and black last. The other two colors should be intermediate shades of gray. If you did not change the default program options, this palette

can be created by swapping the first color with the third, and the second with the fourth. To create the intermediate grays, click on the first color (white), then select either one of the Spread gadgets (marked with I or II) and click on the last color (black).

Now you should have light gray characters on a white background. We want them to be black on white. In the palette under the tool bar, select the black box with the left mouse button to make it the foreground color. Then select the light gray box with the right button to make it the background color. In the Color menu, select Change Background to Foreground. Now the font characters should be black on white.

Display the Font Format requester again. Whatever values are displayed in the Font size gadgets, divide them by two. For example, if the values are 44 and 50, replace them with 22 and 25, respectively. Select Proceed.

Before converting the font, ColorType displays a requester indicating the new size. Leave the checkmark on Stretch and select Proceed.

You have now created an anti-aliased font!

2. ColorType Advanced Features

2.1 The Macro Selection Requester

When the Macro tool is selected with the right mouse button (or with the left button, with no previously selected macro), the Macro Selection Requester appears.

The central area of the requester displays the available macros, which are associated to Rexx programs stored in ColorType's "Rexx" directory, or elsewhere, if so specified in the program settings. Keyboard shortcuts are shown to the right of the macro names.

Four gadgets appear in the upper part of the requester: New, Edit, Delete and About. New and Edit display the Edit Macro requester to respectively define a new macro definition or edit an existing one. Delete removes a macro entry from the list. About displays instructions and other information on the selected macro, if the associated Rexx program contains such data (appearing anywhere in the program, delimited by "/**" and "*/"). More information about the format of the information displayed by the About function appears in the description of the WORDWRAP option of ColorType's RequestNotify Rexx command.

The Execute, Exit and Cancel gadgets on the bottom of the requester respectively execute the selected macro, or exit the requester leaving any changes to the macros as they are, or cancel all changes and exit the requester.

2.2 The Edit Macro Requester

ColorType macros are defined by two sets of information: a Rexx program and the data specified in the Edit Macro requester. The latter is entirely optional, but it is useful to associate an extended name and a shortcut to the Rexx file, and also to access Rexx programs which may be stored in

directories other than "ColorType/Rexx".

The Name field allows the user to define a name other than the file name. If no name is defined, the file name (without its suffix) appears in the list of the Macro Selection requester. The use of different arguments (see below) makes it possible to have several macros appearing with different names, but referring to the same Rexx program.

The Shortcut field is used to associate a keyboard shortcut to the macro. The same format of menu and other shortcuts applies (as used in "UIText.xxx" files).

The File field specifies the full path and file name of the macro. By default, ColorType macros are stored in the "ColorType/Rexx" directory. This field makes it possible to include macros associated to Rexx programs stored elsewhere, for example in "PPaint/Rexx". The Set gadget displays a file requester, which can be used to define the File information.

The Arguments field allows the user to define an optional string which the software passes to the Rexx program upon execution. This string may contain some special codes, introduced by the "%" sign:

%p = Name of the Rexx port

%n = Macro name (as defined in the Name field)

%b = Mouse button pressed (typically in combination with %x and %y):
1 = left, 3 = right. The middle mouse button is not used,
as it is associated to the program's "image drag" mode.

%x = X-coordinate

%y = Y-coordinate

%w = Width (relative to %x)

%h = Height (relative to %y)

When a macro is executed, ColorType detects the presence of any %p, %x, %y, %w and %h codes and waits for the user to mark a point or a rectangular region (only if %w or %h are found in the argument string). If the macro is operating in rectangular mode (if %w or %h are used in the argument), then the macro is executed as soon as the mouse button used to define the rectangle is released. Otherwise, the macro is executed immediately. If only %p, %x, or %y are used (that is, no %w or %h), then the macro is executed as soon as the mouse button is pressed (without waiting for the button to be released). This allows the macro to do some work based on the position and the timing of further user actions. In such a case, the LockGUI command should never be used in the Rexx program before the required input is received.

When the Rexx program receives the argument string, all "%" codes have already been replaced by ColorType with the associated information. In the same way as arguments are normally passed to Rexx programs by the Amiga Shell, ColorType passes all arguments as a single string (ARG(1)). It is then up to the Rexx program to parse the string (PARSE ARG <template>). The macros which come with ColorType include examples of argument

parsing.

In the program settings files, the MACRO keyword is used to set the various parameters found in the Edit Macro requester, in the following format:

```
MACRO = "Name", "Shortcut", "File", "Arguments"
```

For example:

```
MACRO = "Antialias Font", "Ctrl-a",
        "ColorType:Rexx/AntiAliasFont.ctrx", "%p"

MACRO = "Help - Rexx", "Ctrl-h",
        "ColorType:Rexx/Help.ctrx", "%p Command GUI \"%n\""

MACRO = "Test", "", "ColorType:Rexx/Test.ctrx", "%x %y %p"
```

2.3 The Rexx Interface

ColorType has a powerful ARexx (Amiga Rexx) interface which allows the program to be controlled by macros as well as by third party software. All font manipulation commands have a corresponding Rexx command. In addition, ColorType includes a set of generic commands for sophisticated user interaction.

The name of the first Rexx port is "COLORTYPE". Additional ports are named "COLORTYPE_2", "COLORTYPE_3", etc.

2.3.1 Input/Output Environment

ColorType includes many commands to accept input and display output using the default ColorType screen and user interface. However, Rexx also has standard input and output commands (PULL and SAY) which are designed to work in text terminal mode (such as the Amiga Shell).

If ColorType is run from the Shell, then the default Rexx input/output also occurs in the same Shell environment. It should be noted that in this case, running ColorType in deferred mode with "Run" or an equivalent command makes data input from the Shell impossible.

If instead ColorType is launched from the Workbench, then a default console is assigned to the Rexx environment. If the Rexx program has an icon which includes a "CONSOLE=..." Tool Type, then that environment is used (if the AUTO switch is used, then the console window is opened only when necessary).

For debugging purposes, it may be useful to combine Workbench execution with the "TCO" and "TCC" Shell commands to respectively activate and terminate the Global Tracing Console.

2.3.2 Syntax conventions

All Rexx commands are case-insensitive.

Keywords can optionally be followed by the "=" sign if the assignment is not made inside quotes. For example,

```
FILE="Ram Disk:Test"
```

is the same as

```
FILE "Ram Disk:Test"
```

but

```
"FILE=Ram Disk:Test"
```

is parsed as a single argument (keywords within quotes are not recognized if there is other text within the quotes).

The use of keywords is optional, unless the sequence of the arguments is changed, or the referenced argument is not predictable, as for example in:

```
LoadImage FILE "Preview" PREVIEW
```

Keywords are especially useful to specify only a few arguments, leaving the others as default:

```
RequestResponse PROMPT "This is a message"
```

whereas the same command with the arguments identified by their position would be:

```
RequestResponse "Title" "This is a message" "Proceed" "Cancel"
```

2.3.3 Parameters and Options

In this documentations, the term "parameter" is used when a value (string or numerical) is passed to the Rexx command, whereas an "option" is a single keyword (such as QUIET), which activates a certain feature.

For documentation purposes only, parameters and options are followed by a slash ("/") and by a capital letter to indicate the type of parameter, and are separated by commas. This format is often referred to as "command templates". However, the slash/letter combinations and the commas are not part of the Rexx commands.

The template specifications described here are identical to the standard templates used in AmigaDOS commands.

`<parameter>/A` The parameter must Always be included in order for the Rexx command to work.

`<option>/S` The option simply works as a Switch. If the keyword is included, the option is on, if it is not included, the option is off.

`<value>/N` Indicates that a Numerical parameter (rather

than text) is expected. Hexadecimal (base 16) numbers should be preceded by "$" or by "0x".

<parameter>/M Multiple parameters are accepted.

<string>/F The string must be the Final parameter of the command line. The entire ending part of the command line is taken as the desired string. Double quotes are never necessary (even if the string contains spaces).

2.3.4 Return Values

Functions which return a result use the RESULT Rexx variable. It should be noted that string results (for example file paths) are returned delimited by quotes (e.g. "ColorType:fonts").

The success, warning and failure status of the last command is stored in the RC Rexx variable.

Rexx programs may use the RETURN and EXIT commands to return a string upon termination. ColorType automatically displays such result values. For example:

```
RETURN 'Bye'
```

terminates the Rexx program and displays a requester with the following message:

"Macro returned: Bye"

2.3.5 Warning and Error Codes

The following numerical codes are returned in the RC Rexx variable.

Code	Description
0	Normal result or positive selection (OK/Proceed)
5	Operation cancelled by user
30	Unknown command
31	Required argument(s) missing
32	Too many arguments
33	Incompatible options
34	Generic error
35	Error during file I/O
36	File could not be opened
37	Not enough memory
38	File format not recognized (or decryption failure)
39	The file does not contain the required data
40	Error in file data
41	Unable to load this type of image
42	Decryption impossible
43	Warning: partial load
44	Printer device cannot be opened
45	Error during printer output

```
46        Unknown format
47        Unknown format option
48        Unknown setting
49        Numeric argument required
50        Value out of range / illegal value
51        Reserved for Personal Paint
52        Reserved for Personal Paint
53        User brush needed
54        Character not found
55        Font not found
56        Action not possible on OFF characters
```

3. Rexx Command Reference

The following sections explain the use of all Rexx commands supported by ColorType. A simpler list with all commands is returned by ColorType in the RESULT variable when the "Help" Rexx command is issued.

These three options (template: <option>/S) are common to all Rexx commands:

FORCE	This option causes all requesters which ask the user to make a choice to be skipped and return a default result to the program.
QUIET	This option disables all error and warning messages.
NOPROGRESS	This option disables all progress requesters.

3.1 Center

This command centers the current character (if there are empty columns to its left and/or right). The horizontal size of the character remains unchanged.

3.2 Chop

This command removes empty columns of pixels to the left or right of the current character, if any.

3.3 Clear

This command clears the current character, using the current background color.

3.4 Copy

This command copies the current character (and spacing information) to the internal buffer. This is equivalent to selecting the Memory tool with the left mouse button.

3.5 DeleteFont PATH NAME SIZE/N STYLE

This command deletes the specified font from the storage system and also removes the font data from RAM, if present.

If the NAME and SIZE arguments are not specified, a font requester is displayed. If the PATH argument is not specified, then the default system path for fonts ("FONTS:") is used. The NAME argument is case sensitive ("Topaz" is not the same as "topaz"). The ".font" suffix is optional ("topaz.font" is the same as "topaz").

3.6 DynamicView TEXT/A

This command changes the text in the Dynamic View window.

3.7 FlipHoriz, FlipVert

These commands are equivalent to the Flip Horizontal and Flip Vertical tools.

3.8 FontExists PATH NAME/A SIZE/N/A STYLE EXACT/S

This command verifies whether the specified font exists or not. The result, stored in the RC variable, is either RC_OK or RC_NOFONT.

If the EXACT option is specified, then the font attributes are also considered. Otherwise only the font name and size are checked.

For example:

```
FontExists NAME 'topaz' SIZE 8 EXACT        returns RC_NOFONT,

FontExists NAME 'topaz' SIZE 8 STYLE 'of'   returns RC_OK, and

FontExists NAME 'topaz' SIZE 8              returns RC_OK.
```

The NAME attribute works as explained for DeleteFont.

3.9 Get SETTING/A

This commands returns (in the RESULT variable) the value of the specified setting.

For example:

```
Get DVIEW
```

returns 1 if Dynamic View is active, 0 otherwise.

3.10 GetActivation CNUM/N

This commands sets the RESULT variable to 0 or 1 depending on the On/Off

activation status of the current character, or the character specified in the CNUM parameter, if present.

3.11 GetAttr ASCII/S

This command returns the font attributes of the current font. For example, a (decimal) result of 16384 (hexadecimal 0x4000) would indicate that the font is designed for right to left use.

The following are the standard definitions for Amiga font attribute flags (ColorType uses only those marked by an asterisk):

Attribute	Code		Comment
Italic	0x0001	*	
Bold	0x0002	*	Excludes Light
Light	0x0004		Excludes Bold
Underline	0x0008	*	
Outline	0x0010		
Shadow	0x0020		
Superscript	0x0040		Excludes Subscript
Subscript	0x0080		Excludes Superscript
Enlarged	0x0100	*	Excludes Condensed
Condensed	0x0200		Excludes Enlarged
Reverse	0x0400		
Serif	0x0800		
Draft	0x1000		
Fixed Pitch	0x2000	*	
Right to Left	0x4000	*	
Landscape	0x8000		

If the ASCII option is specified, then the result is formatted as a standard size+attributes string (e.g. "8 of").

3.12 GetChar

This command returns the position value (0-256) of the current character.

3.13 GetEnv

This command returns the value of the current environment (0 or 1).

3.14 GetKern CNUM/N

This command returns the Kerning value of either the current character, or the character specified in the optional CNUM argument.

3.15 GetPen BKG/S FRG/S GRID/S REMAP/S

This command returns the palette position number (0-255) associated to one of the following arguments:

```
   BKG    Background color
   FRG    Foreground color
   GRID   Grid color
   REMAP  Foreground remapping color (-1 = no color set)
```

3.16 GetRefPoint REF1/S REF2/S REF3/S REF4/S

This command returns the position of the specified reference point.

3.17 GetSpace CNUM/N

This command returns the Space value associated to the current character, or the character specified in the optional CNUM argument.

3.18 GetXSize CNUM/N

This command returns the X Size value associated to the current character, or the character specified in the optional CNUM argument.

3.19 GetZoom

This command returns the current "zoom" level, which corresponds to the sizes of the dots on the screen, in pixels. For example, a value of 1 indicates "no zoom", and a value of 20 indicates a high degree of magnification.

3.20 Help KEY COMMAND/S IOFORMAT/S

This command returns a table with all Rexx commands (if the COMMAND argument is specified) or all available input/output modules and their parameters and options (IOFORMAT argument).

If the KEY argument is used, then information is only provided for the item specified.

3.21 IsBlank

This command returns 1 if the current character is empty (entirely in the background color), or 0 otherwise.

3.22 ListFonts PATH

This command returns the full list of fonts available in the specified path (or in "FONTS:", if no PATH argument is specified).

Each lines contains information on one font, plus standard size and attribute data. For example:

```
   "topaz" 8 of
   "topaz" 9 oetf
```

```
"topaz" 11 f
"Diamond" 12
"Diamond" 20
"Emerald" 17
"Emerald" 20
"Garnet" 16
"Garnet" 9
"Personal" 8
```

3.23 LoadFont PATH NAME SIZE/N STYLE STRETCH/S NOSTRETCH/S

This command loads the specified font. If either the NAME or the SIZE argument are missing, then the font requester is displayed.

Different combinations of options including FORCE can be used to select a particular font format without displaying the "Format mismatch" format selection requester.

The "FORCE STRETCH" options can be used to scale the font to the current format.

The "FORCE NOSTRETCH" options, instead, load the font leaving its graphic data unchanged, but applying the current font format values.

If instead only the FORCE option is used, then the font format of the specified font is used.

If the "s" STYLE parameter is used, then the font is always loaded in the exact size indicated, resizing it if necessary (this is done by the system font handling routines). Otherwise, if the exact size is not present, the closest size would be loaded instead.

For example:

```
Load NAME 'topaz' SIZE 7          Loads Topaz 8

Load NAME 'topaz' SIZE 7 STYLE s  Stretches Topaz 8 to Topaz 7
```

3.24 LoadImage FILE FORMAT PREVIEW/S STRETCH/S NOSTRETCH/S OPTIONS/M/F

This command loads an image into the current character. If the FILE argument is not specified, then a file requester is displayed.

The PREVIEW option enables the preview screen (HAM or 24-bit, depending on the graphics system).

The STRETCH and NOSTRETCH options work as described for LoadFont.

The OPTIONS file format argument(s) can only appear at the end of the list of arguments.

3.25 LoadPalette FILE

This command loads the palette contained in the specified file. If no FILE argument is specified, ColorType displays a file requester.

3.26 LockGUI

This command locks all processing of user input by ColorType, except for requester input. Mouse moves and keyboard actions do not produce any effect on the main program window. The user can exit this mode during the execution of the macro by pressing the <Esc> key.

3.27 NewFont

This is equivalent to the New Font menu item.

3.28 Paste

This command pastes the character and spacing information stored in the internal buffer to the current character. This is equivalent to selecting the Memory tool with the right mouse button.

3.29 Request TITLE/A DESCR/A RESIZE/S KEEPCOL/S NOVSPACE/S

This is a very powerful and flexible command which allows Rexx programs to easily create complex requesters on ColorType's screen.

The TITLE argument sets the title of the requester.

The DESCR parameter describes the objects contained in the requester.

The RESIZE option allows the requester to be resized horizontally and vertically. Vertical resizing is activated only if the requester contains LIST objects.

The KEEPCOL option prevents the screen colors from being changed. Otherwise, if the contrast of the available colors is determined to be poor, ColorType may automatically change the screen colors while the requester is displayed.

The NOVSPACE option suppresses the automatic creation of empty lines between objects.

The DESCR Argument

This is a single argument containing a concatenation of requester object description strings. Double quotes ('"') are necessary to delimit the substrings only if these contain spaces. The text associated to each object may contain an underscore character to specify a keyboard shortcut (for example, "_Proceed"). Double quotes ('"') are preserved as such if they are preceded by the backslash character ("\"). The backslash used alone must be indicated as a double-backslash ("\\").

In the following examples, angle brackets are used to represent numerical

fields. The brackets are not part of the object definition.

The following is a list of requester objects which can be specified in the DESCR argument:

Text String Gadget

```
STRING = "Gadget Label", <Maximum Number of Characters>, "Initial Text"
```

The "Gadget Label" is the text which appears to the left of the gadget and introduces the gadget. It may contain a keyboard shortcut.

Numerical String Gadget (Integers)

```
INTSTR = "Gadget Label", <Minimum Value>, <Maximum Value>,
         <Initial Value>
```

This object defines a standard numerical string gadget, with automatic range validation and display of an initial value. Only integer numbers can be set.

Numerical String Gadget (Fractions)

```
INT10000STR = "Gadget Label", <Minimum Value>, <Maximum Value>,
              <Initial Value>
```

While INSTR is used only for integers, INT10000STR allows the user to edit fractions with up to four decimal digits. The initial value and the result value are expressed as integers equivalent to the fraction multiplied by 10 000.

Cycle Gadget

```
CYCLE = "Gadget Label", <Total Positions>, <Initial Position>,
        "Position 0 Text", "Position 1 Text" [...]
```

The Position values start from 0.

Check Box

```
CHECK = "Gadget Label", <Initial Status>
```

The Initial Status may be 0 (not checked) or 1 (checked).

Slider (Integer Values)

```
SLIDE = "Gadget Label", <Minimum Value>, <Maximum Value>,
        <Initial Value>
```

This is a standard slider, used to set values in the specified range, with a predetermined initial position.

Slider (Number of Colors)

```
COLSLIDE = "Gadget Label", <Minimum Value>, <Maximum Value>,
           <Initial Value>
```

This is a special slider which displays only the values which are a power of two: 2, 4, 8, 16, 32, 64, 128 or 256. The minimum, maximum, initial and result values is expressed in bitplanes: 1, 2, 3, 4, 5, 6, 7 or 8.

Scrolling List

```
LIST = "Gadget Label", <Total Entries>, <Initial Selection>,
       <Horizontal Size>, <Vertical Size>,
       "Entry 0 Text", "Entry 1 Text" [...]
```

This is a standard Amiga scrolling list, with an optional title (Gadget Label). The Horizontal Size is expressed in characters (if a proportional font is used, its "M" character is used as a reference). The Vertical Size is expressed in lines (number of entries to be visible at the same time).

Text

```
TEXT = "Text"
```

This is not a gadget, but a plain, centered text appearing in the requester.

Separator Line

```
SEPARATOR
```

This displays a horizontal separator line.

Empty Lines

```
VSPACE = <Number of Lines>
```

This adds a separator made of empty lines. The actual number of screen lines is recalculated in proportion to the THICKY program setting (bit-shift: lines<<=THICKY), which represents the ratio of vertical user interface items.

Action Gadgets (Bottom of Requester)

```
ACTION = "Gadget Text"
```

These are the standard gadgets which appear at the bottom of the requester. The default "Proceed" and "Cancel" gadgets are automatically positioned at the bottom of the requester if no ACTION objects are used.

```
The following Gadget Texts are associated to standard texts in the current
user interface language of the software:

   ACTION = PROCEED   displays a "Proceed" gadget,

   ACTION = CANCEL    displays a "Cancel" gadget, and

   ACTION = OK        displays an "OK" gadget.


Object Examples

STRING = "Enter _text:", 256, "Text"

INTSTR = "Enter _number:", -10, 10, 5

INT10000STR = "Enter _number:", -100000, 100000, 5000

CYCLE = "Cycle it:", 2, 0, First, Second

CHECK = "Check it:", 0

SLIDE = "Slider:", -10, 10, 5

COLSLIDE = "Colors:", 1, 8, 4

LIST = "", 3, 0, 30, 10, "First", "Second", "Third"

TEXT = "This is a text"

SEPARATOR

VSPACE = 2

ACTION = "Action"

ACTION = PROCEED

ACTION = CANCEL

ACTION = OK


Command Examples

Request '"Processing" "LIST = _Effect:, 2, 0, 20, 3, Rise, Emboss"'

Request KEEPCOL RESIZE '"Test Requester" ' ||,
'"STRING = ""Enter text:"", 256, ""Text"" ' ||,
'LIST = , 4, 0, 20, 3, First, Second, Third, Fourth ' ||,
'LIST = , 4, 0, 20, 3, First, Second, Third, Fourth ' ||,
'LIST = , 4, 0, 20, 3, First, Second, Third, Fourth ' ||,
'TEXT = Text"'


Note: In the above example, the "||" string concatenation operator and the
line separator comma are used. When a line ends with a comma that is not
```

included within a string token, ARexx combines that line with the following line and treats both as one clause. The "" (double quote used twice) is used to include a single quote in a quote-delimited string. More information can be found in the Amiga manuals and in the ARexx Manual published in the Personal Suite CD-ROM.

Return Values in RESULT

The RESULT variable indicates the position (starting from 1) of the selected action gadget at the bottom of the requester. For example:

```
Request '"Test Requester" "TEXT = ""Make your choice"" ' ||,
        'ACTION = A ACTION = B ACTION = C ACTION = D ACTION = E "'

IF RC = 0 THEN SAY RESULT    /* 1 = "A", 2 = "B", 3 = "C"; ... */
```

The RESULT.# (where "#" indicates the position of the object in the requester) variables are used to return the values of the individual objects in the requester. Simple texts and separator objects do not occupy any positions in this sequence.

For example:

```
Request '"Test Requester" ' ||,
'"STRING = ""Enter text:"", 256, ""Text"" ' ||,
'INTSTR = ""Enter number:"", -10, 10, 5 "'

IF RC = 0 THEN
   SAY '  Text entered:' RESULT.1
   SAY 'Number entered:' RESULT.2
END
```

Return Values in RC

The RC variable normally contains RC_CANCEL (5) if an "ACTION=CANCEL" gadget was selected, or RC_OK (0) otherwise. Other values can indicate various other error conditions, such as insufficient memory.

3.30 RequestFile TITLE/A PATH FILE PROC CANC SAVEMODE/S FORMATS/S

This command opens a file requester and returns the selection.

The PATH and FILE arguments may be used to set initial values for the Path and File settings.

The PROC and CANC strings can be used to change the default Proceed and Cancel gadget texts.

If the SAVEMODE option is specified, then a Save requester is displayed instead of a Load requester. This includes the file overwriting warning.

If the FORMATS option is specified, then the list of file formats is included in the requester.

These are examples of possible results (stored in RESULT):

```
"Work:Pictures/Moon.png"

"Ram Disk:T/test" FORMAT "ILBM" OPTIONS "COMPR=1" "SCRFMT=1"
```

3.31 RequestFont TITLE/A PATH NAME SIZE/N STYLE PROC CANC SAVEMODE/S

This command opens a font requester and returns the selection.

PATH, NAME, SIZE and STYLE are used to set the initial font values (like PATH and FILE in the file requester).

The PROC, CANC and SAVEMODE options work as explained for the file requester.

These are examples of possible results (stored in RESULT):

```
"Workbench:fonts" "topaz" 8 of

"ColorType:fonts" "MyTimes" 50
```

3.32 RequestNotify TITLE PROMPT/A OK SCROLL/S WORDWRAP/S WRAPCHECK/S

This command opens a simple notification requester containing a text and an "OK" gadget.

The TITLE and PROMPT arguments are used to respectively set the (optional) title and the text to be displayed.

If the SCROLL option is specified, then the text is displayed in a scrolling and resizable window. This is especially useful for displaying long texts. By default, the text is displayed in a non-proportional font (such as Topaz), without word wrap and with vertical and horizontal scroll bars.

The WORDWRAP option works together with SCROLL, and determines whether the text should be word-wrapped by the software. Word-wrapping adds line feeds and converts single LF (line feed) characters and surrounding spaces to a single space as necessary to display the text in the available horizontal size. To separate paragraphs, two LF characters should be used. To avoid tables from being reformatted, the special "NBSP" (non-break space, decimal code 160, keyboard <Alt+Space>) character can be used at the beginning of those lines which should be preserved in the original format.

The WRAPCHECK option also works together with SCROLL, and allows the user to switch WORDWRAP on and off as desired. For this purpose, a check box is added to the requester. WRAPCHECK and WORDWRAP can be used in combination to determine the initial formatting setting.

3.33 RequestPath TITLE/A PATH PROC CANC

This command opens a Path requester and returns the path selection. This is identical to RequestFile, only that the file field is not used.

These are examples of possible results (stored in RESULT):

```
"Work:FancyFonts"

"Ram Disk:test dir/subdir"
```

3.34 RequestResponse TITLE PROMPT/A PROC CANC

This command opens a simple requester which displays a text and allows the user to select Proceed or Cancel-type actions. Accordingly, the RC variable contains RC_OK or RC_CANCEL. The PROC and CANC options can be used to change the default gadget texts.

3.35 SaveDView FILE/A FORMAT OPTIONS/M/F

This command saves the graphical contents of the Dynamic View window to the specified file, using the same arguments available for SaveImage.

3.36 SaveFont PATH NAME

This command saves the current font. If the PATH argument is not specified, then the font is saved to "FONTS:". If the NAME argument is missing, a font requester is opened.

3.37 SaveFontImage FILE/A FORMAT OPTIONS/M/F

This command creates an image equivalent to that created by the Show Font command, and saves it to the specified file. The same arguments available for SaveImage are used.

3.38 SaveImage FILE FORMAT OPTIONS/M/F

This command saves the current character image to the specified file in the specified format. If the file is not specified, a file requester appears.

The FORMAT and OPTIONS arguments allow the user to select a specific file format and to set the specific options of that format. If no file format is specified, either the original (read) format is used, or the IFF-ILBM format is used (if the image was not loaded, but created new on-screen).

3.39 SavePalette FILE

This command saves the current palette as an IFF palette file.

3.40 ScreenToBack

This command brings the program screen to the back of all screens.

3.41 ScreenToFront

This command brings the program screen to the front of all screens.

3.42 Set STRETCH/S PARAMS/M/F

This command sets one or more program settings. It is equivalent to loading settings from a settings file.

The "FORCE STRETCH" options can be used to rescale the font to a new format without further confirmation requesters.

Since both ColorType and the Amiga Rexx interface use quotes to pass string arguments, two levels of quotes (single and double quotes) must be used. For example:

```
Set 'FORCE "XMAX=20" "YMAX=40" "COLORS=8"'
Set 'FORCE STRETCH "XMAX=60" "YMAX=60"'
```

3.43 SetActivation STATUS/N/A

This command changes the current On/Off activation status of the character.

3.44 SetAttr ATTR/N/A SET/S CLEAR/S CHANGE/S

This command changes one or more font attributes. If no option is specified, then the new ATTR value simply replaces the previous setting.

The SET, CLEAR and CHANGE options are used to selectively change only the bits which are active in the ATTR value, leaving the other settings unchanged. SET activates the settings (logical "OR" function), CLEAR clears the settings (logical "AND on NOT ATTR") and CHANGE "toggles" the settings (logical "XOR").

3.45 SetChar CHAR/N PREV/S NEXT/S PREVON/S NEXTON/S FIRSTON/S LASTON/S

This command makes a specific character the current character. The range for the CHAR argument is 0-256, where 256 is the undefined character. Other selection methods include:

PREV	Previous character
NEXT	Next character
PREVON	Previous On character
NEXTON	Next On character
FIRSTON	First On character
LASTON	Last On character

3.46 SetEnv ENV/N/A

This command selects one of the two environments (0 or 1).

3.47 SetKern KERN/N/A

This command sets the character Kerning value.

3.48 SetPen BKG/N FRG/N GRID/N REMAP/N

This command sets one or more colors to specific palette entries.

BKG	Background color
FRG	Foreground color
GRID	Grid color
REMAP	Foreground remapping color (-1 = no color set)

3.49 SetRefPoint REF1/N REF2/N REF3/N REF4/N

This command sets the position of one or more reference points. Reference point 3 is the font baseline.

3.50 SetSpace SPACE/N/A

This command sets the character Space value.

3.51 SetXSize XSIZE/N/A

This command sets the character X Size value.

3.52 SetZoom DOTHEIGHT/N BEST/S

This command sets the "zoom" level, which corresponds to the number of video pixels (vertically) used to render a single dot on the screen. A value, for example, of 20 indicates that each dot of a character should be 20 pixels tall, which means it is considerably magnified. A value of 1 indicates no magnification. This considerably improves performance on slower systems, so for example it can be set to speed up the execution of a macro which has to rapidly process all the characters.

The BEST option indicates that the software should automatically determine the largest possible dot size that keeps the current character within the screen boundaries.

3.53 ShiftDown, ShiftLeft, ShiftRight, ShiftUp

These commands are equivalent to the four Move buttons on the tool bar, which move the character image by one pixel towards the specified

direction.

3.54 UnlockGUI

This command clears the effect of the LockGUI command, reenabling user input in the main screen.

3.55 Version PROGRAM/S REXX/S

This command returns the version of the program or that of the Rexx interface. By default, the program version is returned.