0054

COLLABORATORS

	TITLE : 0054	

ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		August 8, 2022	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

0054

1.1 Personal Fonts Maker - 7. PFM: The Preferences Menu

7. PFM: The Preferences Menu

All the functions described in this chapter deal with program and font parameters. Section 2.6 should be carefully read for a better understanding of the sections contained here. The menu is called "Preferences" because the functions which can be accessed through this menu allow the user to custom-tailor the program to the most different needs and tastes.

7.1 Load Preferences

As already explained in section 2.6 ("Program and Font Parameters"), all parameters of the Personal Fonts Maker can be saved to a file.

The standard file requester, described in section 3.23, is used to select the file to be loaded. By default, the drawer where the program loads and saves parameter files is "PFM_Prefs", in the "PFM" volume.

As described in section 2.6, the Personal Fonts Maker has two font environments. Each of the two has a full set of font parameters, completely independent of the other environment. The program parameters, however, are shared by the two environments. When a parameter file is loaded, it affects the program parameters used by both font environments, but only the font parameters of the current font environment. If a parameter file contains some font parameters which are to be used from both environments, that file must be loaded two times: once from each font environment. The "StartupF1.prf" and "StartupF2.prf" files in the "PFM:PFM_Prefs" drawer contain the initial settings for the two environments. The Personal Fonts Maker automatically tries to load these two parameter files when the program is started.

A parameter file can contain the user interface preferences, the FFDL sequences for a particular printer, format information for the dynamic view screen or just a nice screen colour setting. All these parameters, and many more, can also be stored together in the same file. This is usually the case of the files containing the program's initial default parameter settings (sections 1.11 and 2.6).

The Personal Fonts Maker does not put a minimum or maximum limit to the number of parameters which can be set in a file. If the file being loaded contains only a few assignments, the program will modify only the variables referenced to in the file. No other program or font parameters are modified.

It is often useful to extract and use only some parameters from a longer parameter file. If, for example, a file contains definitions for screen colours, audio volume, font format and FFDL sequences, but only the latter are to be loaded, the variables must be "filtered". A second requester, displayed after the file requester, allows the user to select the variables which will be considered when the program reads a parameter file. The requester is called the filter requester.

The filter requester contains two groups of gadgets. On the left, the gadgets referred to program parameters, on the right those relative to font parameters. Each gadget has a four character abbreviation associated with a variable, as described in section 2.6. The gadgets can either be highlighted or not highlighted. The parameter name in highlighted gadgets is light green on a darker background. Only the parameters associated with the highlighted gadgets are read by the program. All other parameters in the file are skipped. This makes it possible to selectively load only part of the parameters in a file, without affecting the other variables.

When a gadget is selected with the mouse it changes state. A highlighted gadgets becomes non-highlighted, and vice versa. As there are more than 25 gadgets in the filter requester, the program offers some shortcuts to quickly change the state of an entire group of gadgets. When the <Cursor Left> key is pressed, all program parameter gadgets become non-highlighted. Pressing the <Cursor Right> key, the same happens for the font parameter gadgets. While the <Shift> key is held down, the two cursor keys can be used to highlight the gadgets.

The loading of a parameter file can be cancelled by selecting the "Cancel" gadget of either the file requester or the filter requester. As described in section 2.6, if an error is found in the selected parameter file all parameters in the file are ignored and an error message is displayed. This cannot happen if the parameter file was written by the Personal Fonts Maker, but is not impossible if the user typed the file using a word processor or a text editor. Appendix G explains the possible error messages.

One of the most common uses of this function is to select different FFDL sequences and font formats. The Personal Fonts Maker comes with several predefined parameter files which can be used to download fonts in different formats to the most different printers.

7.2 Save Preferences

This function is the opposite of "Load Preferences". The font parameters of the current font environment and the program parameters can be saved to a file to be loaded again when necessary.

The standard file requester (section 3.23) is used to specify where the file is to be saved, and with what name. As described in section

7.11.4
, a warning message can be displayed if a file with the same name would be overwritten by the new file.

The filter requester (section
7.1
) allows the user to determine which variables are to be saved in the selected file. Only the variables associated with the highlighted gadgets are saved. If the file is loaded again later, the parameters associated with the gadgets which were not highlighted when the file was saved are not affected, even if the same gadgets in the filter requester of the load operation are highlighted. It is important that when a parameter file is created, only the parameters which are really necessary for the purpose of that file are stored. All other parameters should be deselected with the filter requester. For example, if a file intended to describe a particular font format is saved, the colour settings should not be included, as a subsequent loading of that file could cause the user interface colours to be changed even if this was not expected. In general, it is advisable to have all gadgets highlighted only to save the initial startup parameter files.

The operation can be interrupted by selecting the "Cancel" gadget of either the file requester, the overwrite warning message or the filter requester. Sections 14.1 ("Problems with disks") and appendix G contain useful information on how to deal with problems and errors which may be encountered during the save operation.

As explained in section 2.6 ("Program and Font Parameters"), a parameter file is a plain ASCII text file. When the Personal Fonts Maker stores a parameter file, it does not put any comments in the file, and uses only a minimum of spaces and empty lines to make the file more readable (for the user). If a word processor or text editor is used to write a parameter file, and the same file is then loaded and saved again by the Personal Fonts Maker, comments, additional spaces and new lines are stripped. Lower case letters in variable names are converted to capital letters.

This function can be used to store the initial default values for all parameters used by the Personal Fonts Maker. Once the parameters are set, they can be stored in the "StartupF1.prf" and "StartupF2.prf" files, in the "PFM_Prefs" drawer of the "PFM" volume. These are the parameter files which the Personal Fonts Maker tries to load after it is started. Section 1.11, 1.12 and 2.6 contain additional information on this operation.

7.3 Font Description

This function displays a requester which allows the user to modify the font description parameters. The parameters which can be set include all font parameters, except those regarding the character set (section 4.8), the grid of the character editing box (section
7.6
) and the italic factor (section
7.4
). Font parameters differ from the remaining program

parameters since there are two sets of font parameters (one for each font environment), while the program parameters are shared by both font environments.

This requester allows the user to define not only the sizes and the proportions of the font, but also how a font in that format is to be written when the FFDL is used to output the data. Section 2.7 ("Programming the Output Format: the Cloanto FFDL") has more on this subject.

Before a font in a format other than the current one is designed or loaded, the new format should be specified through this requester. If this is not done before a font in a different format is loaded, a requester appears which allows the user to stretch the font to the current format or adapt the current font format to that of the selected font, as described in section 4.3. A similar requester (without the "Adapt" gadget) appears if the font format is modifed while a font is stored in the current font environment's memory. If, for example, the user modifies the "Y Max" parameter (section 7.3.2) while a font is being edited, the program can stretch the font conforming to the current stretch mode, if allowed by the user. Sections 4.3 ("Load PFM Font") and 7.10 ("Stretch") explain this operation in more detail.

The requester described here contains all the parameters usually necessary to define the format in which a printer downloadable font is to be designed and sent to the printer. Different font descriptions can be created for different printers or different formats of the same printer (e.g. draft vs. letter quality). These different font format descriptions can be saved, as explained in section 7.2 ("Save Preferences"), to create a library of the most used font formats. The Personal Fonts Maker comes with different font format parameter files, as described in section 12.6.

The "Font Description" requester contains different string gadgets. Section 1.9.6 ("String Gadgets") explains how to modify the text in the string gadgets. The following subsections explain the use of each gadget in detail.

7.3.1 X Max

This parameter defines the maximum width, in dots, of the characters in the font. If the characters are proportionally spaced their width can be any value from 1 to "X Max". The valid range for "X Max" is from 1 to 255. If it is necessary to have fixed pitch characters, all characters in the font should be "X Max" dots wide. Sections 4.6 ("Export Amiga Font") and 8.14 ("Fixed Pitch") contain additional information on fixed-width fonts.

The "X Max" parameter only limits the range of the "X Size" of each character. The "Space" and "Kerning" parameters (sections 3.5 and 3.6)

can be used to add additional spaces to the left and the right of the character, making the total character width larger than "X Max".

When the font description is used to define the format of printer downloadable fonts, this parameter is usually the same as the maximum character width allowed for the particular font format. This limit is printer specific. For example, some printers cannot handle characters wider than 36-37 dots in the proportionally spaced letter quality format. In other printers, the only limit is the amount of available (printer) memory. On most printers, however, characters cannot be wider than 255 dots, as the character width is stored in a single byte.

When the Personal Fonts Maker is used to save a font in the Amiga font format, this parameter is saved as the Amiga "X Size" (font width) parameter of the Amiga font. Section 4.6 ("Export Amiga Font") has more on this.

Sections 2.6.24 and 2.7.2.21 explain the use of XMAX as a font parameter and as an FFDL variable, respectively.

7.3.2 Y Max

This parameter determines the maximum height of the characters in the font. The bitmap of the characters is always tall enough to contain an image "Y Max" dots high. The value assigned to "Y Max" must be in the range from 1 to 255.

As far as the character bitmaps, the reference points and other character relative data are concerned, "Y Max" is a constant, more than a maximum value. Unlike the character width, which can be varied from character to character to modify the character's "X Size", there is no equivalent for the character height. Once the height of the font has been determined by setting "Y Max", all characters in that font will have that size. This does not mean that the actual height of each character's image will be equal to that maximum. In fact, it is even possible that not even one character in the font will reach the maximum height allowed by the font format. The value assigned to "Y Max" is often chosen to accommodate a particular printer download format, or a program's screen font format, rather than the real font size.

As described in section 1.4 ("File Names and Titles"), it is useful to append a suffix, indicating a font's size, after the font name when the font is stored. A font which fits in 24 dots, for example, could be named "Font_24.fnt". This would immediately identify the font as being downloadable to a 24-pin printer. Like the typographical point size (section 2.3), the "Y Max" parameter can become the unit after which a font is chosen.

This parameter is called "Character Height" in the documentation of some printers. When a font is saved in the Amiga format, the value contained in this variable is stored in the Amiga equivalent parameter called "Y Size".

Sections 2.6.26 and 2.7.2.25 explain the use of YMAX as a font parameter and as an FFDL variable, respectively. As described in section 2.7.2.26, this parameter can also be accessed as the YSIZ FFDL variable.

YSIZ is a copy of YMAX, existing only to form a logical pair with XSIZ.

7.3.3 X Dpi

This parameter, in conjunction with "Y Dpi", determines the ratio at which the character images in the font are to be displayed and printed.

"Dpi" means "Dots per Inch". The value should be set to the number of adjacent dots which can be printed (or displayed) in a horizontal row (one inch long) by the device generating the final font output. The resolution of a printer is usually 180, 300, 360 or 400 dpi, while an average monitor cannot display more than 100 pixels per inch. Professional printers can print at resolutions of 1200 and even 2400 or more dpi.

Since the horizontal resolution is often different from the vertical resolution, the program checks "X Dpi" and "Y Dpi" to ensure that the characters in the editing box are displayed in the same proportions as those of the final output. For example, if a printer's resolution in the selected print mode is 360 dpi horizontally, and 180 dpi vertically, the dots in the character editing box are also displayed twice as tall as they are wide. For this purpose, it would be the same to specify 360 and 180 or 2 and 1 in the "X Dpi" and "Y Dpi" string gadgets, as the ratios of 360 to 180 and 2 to 1 are the same. The display ratio of the character editing box is calculated by dividing the "X Dpi" value by "Y Dpi", in consideration of the current screen mode (e.g. PAL or NTSC).

The "Edit Character Set" function (section 4.11) automatically sets both values to the current screen resolution. This information is supplied by the Amiga operating system. In a standard (640 by 200 pixels) NTSC screen the default values are 65 and 28, respectively. In a 640 by 256 PAL-mode screen the values are 62 ("X Dpi") and 33 ("Y Dpi").

As described in section 7.10 ("Stretch") the Personal Fonts Maker uses the density information to stretch the character images between different formats. These transformations are necessary to ensure, for example, that an 'o' letter which is round in one font will not look elliptical when it is moved to another font, or when the format of the original font is modified.

Sections 2.7.2.20 and 2.7.2.24 explain how the XDPI and YDPI variables can be accessed in a FFDL sequence.

7.3.4 Y Dpi

This parameter is similar to "X Dpi", but measures the density of the dots in a vertical (rather than horizontal) column (one inch high). The Personal Fonts Maker needs both values to be correctly set, since on many output devices the vertical resolution can be different from the horizontal one. The correct display ratio of the character editing box (i.e. the dimensions of the single dots) can only be calculated if both valued are set.

Section

7.3.3
("X Dpi") has additional general information on both parameters.

7.3.5 Prologue

This and the successive three string gadgets ("ON Sequence", "OFF Sequence" and "Epilogue") are used to define the FFDL sequences which have to be used to output the current font in a user-defined format. Section 2.4 ("Storage of Fonts") contains a general description of all the possible formats in which the Personal Fonts Maker can save a font. Section 2.7 ("Programming the Output Format: the Cloanto FFDL") explains how to write an output format description using the Font Format Description Language. Sections 4.12 ("Printer Test") and 4.13 ("Write Font Data") describe the functions which use FFDL sequences to output the font data.

To output a font's data in a format described through the Font Format Description Language, the Personal Fonts Maker goes through three different steps. First, it outputs the data associated with the "Prologue" FFDL sequence. Then, for every character in the specified range (section

7.3.9
) it executes either the "ON Sequence" (section
7.3.6
) or the "OFF Sequence" (section
7.3.7
), depending on the character's status. Finally, the program executes the FFDL sequence specified in "Epilogue" (section

7.3.8
).

In the case of a printer's font download format description the "Prologue" sequence usually contains at least the control codes to select a printer ROM font/mode (e.g. fixed-pitch draft, or proportional letter quality) and copy the ROM font to the printer's RAM (see "Downloaded Printer Fonts", section 2.5), where some (or all) characters will be overwritten by those downloaded by the Personal Fonts Maker.

On some printers a command must be issued (at the beginning of the prologue) to specify how much memory is to be reserved for downloaded fonts and in which order one or more fonts are to occupy that memory. For example, on the NEC Pinwriter Plus series of printers, the FFDL sequence "FS \W (0)" can be used to set the printer so that two fonts with a maximum of 128 characters (7-bit codes) can be downloaded, while "FS \W (1)" will reserve the entire memory for a full 256-character font. On other printers similar operations can be performed by setting one or more dip-switches as appropriate (also see section 14.2, "Problems with Printers").

Other printers, like the HP LaserJet series and compatibles, require a substantially more complex prologue, specifying the name of the font and several other font attributes. This is explained better in the documentation of the printer being used.

The parameter files which come with the Personal Fonts Maker (section 12.6) contain font descriptions for different printer download formats and other useful applications. The files can be loaded and examined for a better understanding of the Font Format Description Language.

7.3.6 ON Sequence

This FFDL sequence is executed for every character in the specified range which is "On".

In the case of a printer download format description, this FFDL sequence outputs all the data relative to the character currently processed. Section 13.4 ("Downloading a Font to the Printer") contains more information on this subject.

7.3.7 OFF Sequence

This is the FFDL sequence which the program executes for every "Off" character in the selected range.

This sequence is usually not defined (i.e. the string gadget is left empty) in a printer download format description, as the purpose of the "On/Off" status (section 3.10) is to separate the characters which must be downloaded from those which are not. "Off" characters are generally not downloaded.

This sequence can be useful, for example, when the Font Format Description Language is used to generate a textual description of a font. Section 13.6 ("Creating a Word Processor Font Size Table") describes such an application. Another application is the output of "dummy" characters having zero-width, if it is not possible to simply skip "Off" characters.

7.3.8 Epilogue

This FFDL sequence is executed only once, after all other sequences. In the case of a font download, the commands which instruct the printer to use the font which has just been downloaded are placed here.

7.3.9 Range

The "On" and "Off" sequences can be repeated for an entire range of characters in a font. The "Repeat Sequences from # / to #" string gadgets allow the user to specify the codes of the first and the last characters through which the program has to cycle.

The second code must be equal to or higher than the first code (if no negative codes with special meaning are used). The valid range is 0 to 255. The undefined character (code 256) cannot be accessed. The program automatically replaces an out-of-range code with the closest valid value. The Personal Fonts Maker can also reverse two codes if necessary (e.g. RNGE 33 126 is the same as RNGE 126 33).

Three negative values are also accepted, but have a special meaning if they are used to define the limits of the range. -1 (minus one) means "The First 'On' Character" if it appears in the first string gadget and "The Last 'On' Character" if it is written in the second gadget. -2 (minus two) is interpreted as "The First 'Off' Character" if it appears in the first string gadget and "The Last 'Off' Character" if it is written in the second gadget. At least one "On" or "Off" character must exist for the -1 and -2 codes, respectively, to work properly. If this is not the case, the output function ("Printer Test" or "Write Font Data", sections 4.12 and 4.13) will fail. -3 (minus three) stands for "The Current Character", which is the character which was displayed in the character editing box before the execution of the FFDL sequences began. It should be noted that these negative values displayed in the string gadgets are not converted to a positive character code by the program, but are only interpreted in a special way during the execution of the FFDL sequences. When the requester is displayed again after the execution of the sequences, the negative values will still be there.

For every character in the selected range, the appropriate FFDL sequence is executed to output the data (more in section 2.7.2, "FFDL Variables"). For every "On" character in the range the "On" sequence is executed, while the "Off" sequence is executed when an "Off" character is encountered. The characters are always processed starting from the character in the range having the lowest code, and increasing the value by one unit for every cycle.

As explained before, the Personal Fonts Maker performs a validity check on the values used to define the range. Sometimes, however, a valid range may cause an error when the FFDL sequences are executed. For example, if a range from -1 to 100 is specified, meaning "from the first 'On' character to the character whose code is 100", but if the first "On" character is 120, or no such character exists, the sequence cannot be executed. If the output of the font data is requested ("Printer Test", section 4.12 and "Write Font Data", section 4.13) in a similar condition, an error message is displayed and the title bar contains information in a format similar to the following:

```
Writing in progress (120-100)      - Bytes written: 0
Writing in progress (?-100)        - Bytes written: 0
```

If the range delimits the characters to be downloaded to a printer, some printer-dependent rules should be applied. Most printers will not work properly if characters with codes equal to control codes (e.g. ESC, decimal code 27) are downloaded. This does not normally happen, as non-graphical characters are usually marked as "Off", but it is a good practice to always let the range start from the code 32 (Space Character) and end at 126 (seven bit codes, where 127 is a control code) or 255 (eight bit codes). The beginning of the range should be lowered as appropriate if any redefined characters have codes smaller than 32. Some printers will not accept the download of characters having the 8th bit set (i.e. codes greater than 127). In this case, the range should not exceed the 127th character, and the Printer Driver Modifier (chapters 9 to 11) should be used if any characters in the texts to be printed have a code greater than 127.

7.4 Italic Factor

This parameter determines how much the "Italicize" brush function (section 5.6) should slant the image to the right. Each font environment may have a different "Italic Factor" parameter.

When the "Italicize" function is executed, the program starts counting from the bottom of the brush image. After the number of lines determined through this parameter is counted, all remaining (upper) lines are moved to the right by one dot position and the count restarts.

The possible values range from 1 to 255. Lower values yield more slanted images. A value of one, for example, causes a vertical line to be slanted by 45 degrees if the horizontal and the vertical densities are the same. The "Italicize" function modifies the image only if the number of lines which make up the brush is greater than the "Italic Factor" value.

To obtain optically equivalent levels of italic characters among different font formats, each format’s display ratio must be considered to set the italic factor.

7.5 Coordinates

This parameter determines whether (and how) some information regarding the current dot-position in the character editing box is to be displayed on the title bar. The status of the parameter can be modified by selecting one of the two subitems, called "Start 0:0" and "Start 1:1". The modes associated with these two menu subitems are described in the following subsections.

Either one or none of the two subitems can be selected. When a subitem is selected, a checkmark appears on the left of the text, as described in section 1.9.8 ("Menus"), and the associated coordinates display mode is activated.

To display the coordinates, the Personal Fonts Maker reserves a small part on the right of the title bar. Two values are displayed in this little rectangle, indicating respectively the horizontal ("X") and the vertical ("Y") position in the character editing box, starting from the top left of the box. When the mouse pointer is not over the character editing box, no value is displayed.

While a mouse button is held down to perform an operation in the editing box, the displayed values are relative coordinates, indicating the distance from the dot which was under the pointer when the mouse button was pressed. In this mode, a ’+’ (plus) or ’-’ (minus) sign is displayed before each number. A negative horizontal position indicates that the current position is to the left of the starting dot. A negative vertical position indicates that the current position is over that dot.

Some macro commands (chapter 6, "Macros") are followed by coordinates indicating a position in the character editing box. Whenever one of these commands (e.g. MOVE) is executed, the Personal Fonts Maker updates the coordinates, just as if the function was called manually by the user.

The coordinate display mode can be terminated by clicking the box containing the coordinates with the mouse, or deselecting the menu subitem

associated with the display mode, as described in section 1.9.8 ("Menus").

The following two subsections describe the differences between the two coordinates display modes. The sections have the same name as the menu subitems they refer to.

7.5.1 Start 0:0

When this mode is selected, the coordinates are displayed starting from the origin as 0:0. This means that the values displayed for the horizontal and vertical positions will be zero when the current position is over the origin. This is also valid for relative coordinates.

The coordinates are displayed on the title bar introduced by lower case 'x' and 'y' letters. Upper case letters are used if the origin is 1:1 ("Start 1:1" option).

7.5.2 Start 1:1

In this mode, the coordinates are displayed counting from 1:1 as the origin. The origin itself is on the same position as in the "Start 0:0" mode, but is addressed differently. The value zero is never used, neither for vertical nor horizontal position values, neither in the relative nor in the absolute coordinates mode. When the current position is over the origin, the two values are set to one. The coordinates on the title bar are introduced by capital 'X' and 'Y' letters, to distinguish them from coordinates having 0:0 as their point of origin (introduced by lower case letters).

Relative coordinates also start from +1:+1. The dot at the top left of the +1:+1 position (the origin) is -1:-1.

This mode affects only the way coordinates are displayed on the title bar. Coordinate values used in macros are always interpreted in the same way, as described in chapter 6 ("Macros").

7.6 Grid

The character editing box, described in depth in section 3.1, displays an enlarged copy of the current character's image. This parameter determines how each dot which makes up the character image is to be separated from the dots surrounding it.

Two modes can be selected and deselected by the user. These are described in detail in the following two subsections. If no mode is selected, adjacent dots in the character editing box are not separated. This can be useful to see what a character will look like when it is printed or displayed. The size of the character editing box may be reduced (section 3.13, "The 'Smaller' Gadget") to bring the size of the displayed character down towards the size of the printed character.

7.6.1 Lines

When this mode is selected, a grid of lines is displayed over the entire character editing box. The resulting pattern creates a small outlined box for each dot. The lines are displayed in dark green if the character is "On", light green otherwise.

7.6.2 Dots

This mode is similar to the "Lines" mode, only that the lines which make up the grid have the same colour as the background of the character editing box. The grid is invisible when the box is empty, but causes adjacent (coloured) dots to be separated by a thin line. This makes the individual dots easily recognizable.

7.7 Workbench

The Personal Fonts Maker lallows the user to close the Workbench screen (normally used by the Amiga operating system to display disk icons, Shell and CLI windows and any windows opened by other programs) manually.

The Workbench screen is public, and can be closed only if it does not contain other windows than those used to display the contents of disks (or other volumes) and drawers. Any other windows are linked to different programs, which must be terminated to close the respective windows.

More than 40 Kbytes of RAM are freed when a PAL-size Workbench screen of 256 lines, 640 pixels wide, with four colours (two bit planes) is closed. The Personal Fonts Maker automatically tries to close the Workbench screen if it does not find a memory expansion on a 512 Kbyte Amiga system. This is a feature of the memory save mode. As described in section 1.11, this mode can be activated manually by holding the <F1> key down while the program is starting.

7.7.1 Open

This function tries to open the Workbench screen, if it is not already open. If there is not enough chip memory to open the screen, an error message is displayed (appendix G).

7.7.2 Closed

When this command is selected, the program tries to close the Workbench screen. If the operation is impossible (as described in the introduction to this section), an error message is displayed.

7.8 Icons

When a file is written by the Personal Fonts Maker, a Workbench icon (section 1.9.7) can automatically be saved with it. An icon is a particular kind of gadget, displayed by the Workbench program, which makes a file immeditately recognizable through a graphical image. Icons can be used to select, load, rename, copy or delete the associated file (or

drawer, or volume) using the mouse.

This parameter allows the user to decide whether (and how) a graphic Workbench icon is to be linked with and saved together with the file. The following subsections explain the three possible choices in detail.

7.8.1 No

A small disadvantage of icons is that they contain graphical data, which requires some - though not a lot of - memory. The Personal Fonts Maker's data file icons occupy less than 488 bytes of memory. For the more technically interested users, this means that an icon occupies only one disk sector on any Amiga filing system (a sector in the original non-FFS Amiga filing system can store 488 bytes of data), plus one sector for the directory entry. More than 800 of these icons can fit on a standard 880 Kbyte disk.

If the disk space is to be highly optimized, or if icons are never used, this option can be selected to save files without an associated icon.

7.8.2 Yes (Default Font Icon)

When this option is selected, the Personal Fonts Maker automatically associates a Workbench icon with each file which is written. Different icon images are used to distinguish between the files saved by the program. Font files, character set files, printer download files, brush files, macro files and parameter files all have a different icon. The icon images are designed to make the different file types immediately recognizable from the icon.

Icons are never saved when a font is stored in the Amiga font format.

7.8.3 Yes (Character Font Icon)

This option causes the program to save files just like in the "Default Font Icon" mode, with one difference regarding font files. An image of the character displayed in the character editing box is used instead of the default icon image. This makes different fonts saved in the Personal Fonts Maker font file format distinguishable through the icon.

If, for example a font is saved when the 'B' letter of that font is displayed in the character editing box, the image of that letter is used to create the file's icon.

7.9 Joined Fonts

As described in section 3.2 ("The 'Font' Gadget"), it is possible to work on two fonts at the same time. Very often, this implies that data is to be moved between characters having the same code in both fonts.

If this option is not selected, the user, when moving to the other font, would find the current character still the same as when that font

was last edited. The selection of a new current character in one font does not make the character with the same code in the other font the current character of that font. The current characters of the two fonts may have different codes.

With this option activated, both font environments always have the same current character. This is extremely useful if both fonts use the same character sets (i.e. characters in the same positions have the same default image) and the two fonts are being compared, or one font is being used as the basis for a new font. This is the default mode of the Personal Fonts Maker, and works both during manual operations and when macros are executed.

7.10 Stretch

The Personal Fonts Maker can deal with fonts in different sizes and resolutions (section 7.3, "Font Description"). In some cases, when a character's image or an entire font are to be loaded or copied to a font environment having a different format, the program can automatically stretch the character images. This can be done when a font is loaded, when an Amiga font is imported and when the format of the current format is modified by setting the parameters with the requester or loading a parameter file. The stretch functions are also used to calculate the value which replaces a '+' sign written in the "X Size", "Space" and "Kerning" gadgets, as described in sections 3.4, 3.5 and 3.6.

The user can modify the way images are stretched, as described in detail in the following subsections.

7.10.1 Proportional X

The Personal Fonts Maker can employ two different techniques to stretch images. These affect the way fonts are adapted when they are converted into a new format, the stretching of an image with the "Paste from Buffer" function (sections 3.12 and 7.10.3) and the value which automatically replaces a '+' (plus) sign written in the "X Size", "Space" and "Kerning" string gadgets (sections 3.4 to 3.6).

"Proportional X" means that the aspect of the image, i.e. the X/Y ratio, remains unchanged after the stretching. A character stretched with this technique may look smaller or larger, but not taller or wider.

The amount by which an image is to be enlarged or reduced is calculated by the difference in the heights of the source and destination formats. The image width is modified consequently, taking into account any differences between the densities of the original and the new format. This conversion can have one drawback: after an image has been stretched, the new character width may exceed the maximum width ("X Max", section 7.3.1) defined by the target format. In this case, some columns of dots on the

right of the image have to be cut. To avoid this, unless the "Maximum X" mode (section
7.10.2
) is selected, the "X Max" parameter of the target format should be set sufficiently high so that the characters fit in the new format without having to be cut.

7.10.2 Maximum X

This mode, as opposed to "Proportional X", optimizes the use of the available horizontal space, but does not necessarily maintain the original proportions. The height is always the maximum ("Y Max") defined by the format. This technique stretches the image without modifying the ratio of the image width ("X Size") to the maximum width ("X Max").

For example, if a character 14 dots wide ("X Size") has to be converted from a 20 ("X Max") by 10 ("Y Max") font with an X/Y ratio of 2 to 1 (two horizontal units equal one vertical unit, i.e. the dots are taller than they are wide) to a new format of 30 x 40 with a ratio of 1 : 1, the character will become 28 dots wide with the "Proportional X" technique (therefore maintaining the original aspect). If the "Maximum X" technique is chosen instead, the character is stretched to a width of 21 dots (21 is 70% of 30, just as 14 was 70% of 20). In both cases, the height of the new image is the maximum possible height ("Y Max"), i.e. 40 dots, versus the 10 dots of the original format.

7.10.3 Recall & Stretch

The same techniques used to stretch an entire font can be employed by the program to adapt the character stored in the buffer (section 3.11, 3.12) to the format of the current font environment. This may be necessary after a character copied from one font is pasted into the other font, having a different format.

If the "Recall & Stretch" mode is selected, the Personal Fonts Maker stretches the image stored in the character buffer before it is pasted to a font having a format different than the font from which the character was copied. If this function is disabled by the user, or if it is not needed because the data in the buffer has the same format as the current font environment, the image is copied as it is stored in the buffer, dot by dot.

7.11 File Requester

This group of parameters allows the user to control some aspects of the file requester (section 3.23). Each parameter is associated with a menu subitem which can be selected and deselected with the mouse (section 1.9.8, "Menus"). The following subsections explain each parameter in detail.

7.11.1 Expand Path

This parameter determines whether the program can modify the content of

the "Path" string gadget of the file requester.

If the option is enabled (as it is by default), the Personal Fonts Maker transforms device names (e.g. "DF0") into volume names (e.g. "PFM Disk"). This facilitates the access of files stored on disks (or other media) which are moved from one drive to another. Also, logical names (like "PFM") are expanded to the full paths (e.g. "Work:Programs/PFM") originally assigned to them. This makes it easier for the user to understand the position of each file. The file requester's directory lists can be stored more efficiently if this option is enabled (multiple occurrences of the same list are always recognized).

If the option is disabled, the program never modifies the names which are written in the "Path" string gadget. This may be useful if logical names are used on purpose, for example if the "PFM" logical name is used to access files in an installation-independent manner. This is necessary if the path name has to be stored in a parameter file which could be loaded by another user on a different computer. It may not be possible for a full path valid on one system, like "Work:Programs/PFM", to be interpreted on another computer, where the Personal Fonts Maker is stored in a different drawer. In this case, logical names must be used. Sections 1.12 ("Environment Variables and Standard Drawers") and 12.4 ("AskAssign") explain how to work with logical names.

7.11.2 List Icons

As explained in sections 1.9.7 ("The User Interface/Icons") and 7.8 ("Preferences/Icons"), the Amiga environment and the Personal Fonts Maker in particular have the ability to associate graphical icons to files and drawers. An icon is stored in a separate file, having the same name as the file plus the ".info" suffix. The file containing the icon data for the "Oberon_24.fnt" font file, for example, would be "Oberon_24.fnt.info". Versions of the Amiga operating system beyond 1.3 may also use ".icon" files to store additional information on the icons contained in drawers.

Icons are useful when their images appear in the Workbench environment, but having the names of all ".info" and ".icon" files displayed in the list box is, under normal circumstances, a waste of space. If all files had icons, the list box would have to contain at least twice as many names. By default, the Personal Fonts Maker does not show icon files in the list box, but this parameter can be set to include ".info" and ".icon" files as well.

The so-called "dot-files" are never inserted in the list box. These are all the files whose name begins with a '.' sign (ASCII decimal code 46). By convention, these files contain system and program environment information not to be accessed by the user.

7.11.3 Double-Click

When this option is selected, the names displayed in the list box of the file requester can be selected by double-clicking the left mouse button. The requester disappears as if the "Proceed" gadget had been

selected. The speed at which the button must be pressed can be set through the Amiga Preferences, as described in the Amiga documentation.

If the option is disabled, the names can still be selected with the mouse, but double-clicks are not interpreted in any special way.

This parameter also affects the response of some macro requesters to double-clicks, as described in chapter 6.

7.11.4 Confirm Overwriting

This parameter affects only the requesters used to define a name for a file to be written by the Personal Fonts Maker. If this parameter is set, a second requester is displayed if the file already exists and would be overwritten by the operation being executed.

The warning requester contains two gadgets: "Proceed" and "Cancel". If "Cancel" is selected, the function is aborted, and no file is written.

7.12 Language

Some versions of the Personal Fonts Maker come with user interface texts in different languages. The "Language" menu item has one subitem for each available language. With this option, it is possible to select the language used to display all menus, gadgets and program texts.

This parameter does not set the keyboard language, which must be selected as described in section 1.10.1 ("The SetMap Command").

7.13 Audio

The audio of the Personal Fonts Maker consists of the sounds which are emitted when different functions are selected. The audio is stereophonic. This means that all sound signals are emitted more or less differently on two audio channels.

Two knobs, similar to the ones used to modify the colours (section

7.14

), can be used separately to set the volume of error- and cue- ↩ sound.

The cue signals are sounds emitted when a key is hit, or a gadget or menu item is selected. All other sounds are error (or warning) signals.

A number on the right of each knob indicates the volume level. This can range from 0 to 64. By default, the volume of error sounds is higher than that used for other sounds.

As explained in sections 1.11 ("Loading the Personal Fonts Maker") and 1.13 ("Installing the Personal Fonts Maker"), the audio only works if the program can activate the CloantoAudio unit. The Personal Fonts Maker may have the audio unused as a default to save memory on systems with 512 Kbytes of RAM and no memory expansion. Section 1.11 describes the memory save mode in more depth.

7.14 Colour Bias

The default screen opened by the Personal Fonts Maker uses eight different colours. With this function, all user interface colours, including those of the mouse pointer, can be changed. In this documentation, the colours used by the program in certain circumstances are often mentioned. The colours which are described are, of course, the standard default colours, and can become meaningless if modified by the user.

The Personal Fonts Maker uses different levels of shadows and other visual effects to improve the general "look and feel" of the user interface. This requires that each colour be in a given relationship with all other colours. For example, it would look rather odd if a blue object on a red surface had a yellow shadow. For this reason, it is not possible to modify each colour separately from the others, but only the entire colour environment as a whole. The three knobs displayed in the requester described here allow the user to select about 30.000 variations on the default colour pattern.

Each colour displayed on the screen is composed of different amounts of red, green and blue mixed together. Each of these three basic components is measured in a scale from 0 (none) to 15 (maximum). Black, for example, has the three levels set to 0, while the colour yellow has red and green set to 15, and blue to 0.

The eight colours (the default quantity) of the Personal Fonts Maker have different levels of red, green and blue. Each of the three knobs can be moved while it is selected with the mouse, as explained in section 1.9.5. The three knobs labelled Red, Green and Blue can be used to add or subtract a value to the levels of red, green and blue of all eight program colours.

The number on the right of each knob indicates the current setting of each knob. The value can range from -15 (minus fifteen) to 15. This value is added to the levels of the components of the three basic colours which make up all user interface colours. The program makes sure that these levels do not under- or overflow, becoming smaller than 0 or greater than 15. One of the user interface colours is defined as black and cannot be changed. A value of 15 for all three knobs, for example, would set the red, green and blue levels of all program colours (except black) to 15 (the maximum), leaving a black and white display, while the entire user interface would become black setting the three levels to -15.

The "Default" gadget can be used to clear the three bias values, restoring the program’s default values. The default colour settings can also be restored at any time by pressing the <Help> key (even when the "Colour Bias" requester is not displayed). Any variations to the colour settings can be confirmed by selecting "Proceed", while "Cancel" restores the colours used before the requester was displayed.