2149

COLLABORATORS

	TITLE : 2149		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		January 16, 2023	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

2149

1.1 Personal Fonts Maker - 4. PFM: The Project Menu

4. PFM: The Project Menu

This chapter describes the functions which can be accessed through the "Project" menu of the Personal Fonts Maker. The "Project" menu contains all commands used to read and save data to peripheral units. Other commands, which are used to create a new font, obtain information on the font status or the program, or terminate the work with the Personal Fonts Maker, are also included in this menu.

4.1 New Font

This function clears the current font and restores the standard font parameters. All character bitmaps are cleared (section 3.8). The characters are switched to the "Off" status (section 3.10). The "X Size" (section 3.4) and "Space" (section 3.5) parameters of the characters are set to "X Max", while "Kerning" (section 3.6) is set to 0 (zero).

Only the font data of the current font environment (section 3.2) is cleared. If another font is in memory, it is not modified. The memory occupied by the font's bitmaps is cleared, but not released. Section
4.2
explains how to free the associated memory.

If the displayed font has not yet been stored, the Personal Fonts Maker displays a warning message. The program always displays such messages before something which has not yet been saved is cleared. The "Proceed" gadget of the message requester must be selected to confirm the choice and clear the font data. If the "Cancel" gadget is selected the function is aborted and nothing happens. The the <Return> key and the <Esc> key on the keyboard can be pressed instead of the two gadgets, as explained in sections 1.10.11 and 1.10.15.

4.2 Free Font Memory

When the program starts, it allocates memory only for one font in the selected format (section 7.3). The other font environment does not occupy any font memory until the user enters the second font environment through the "Font" gadget (section 3.2).

The user generally does not have to bother about when the program occupies or frees some memory. These operations are done automatically by the program. In some cases, however, it may be useful to limit the occupied memory, especially if there is not enough free system memory. If both font environments have been used, but one of the two is not necessary any more, this function can be used.

This function does all the things the "New Font" function does (section

4.1
). In addition, it frees the arrays containing the values "X Size ↩
",
"Space" and "Kerning" for each character, the bitmap memory allocated by the current font environment and the whole character set data (section

4.7
). The program automatically switches to the other font ↩
environment
(section 3.2).

The font memory can be freed only if the other font environment has some memory allocated. This limitation is necessary because the program needs at least one font environment to work properly. If neither font environment had any memory allocated for the bitmap and the other font data, the user could do almost nothing with the Personal Fonts Maker. This is why the program automatically switches to the other font environment after this function is called. For the same reason, memory is allocated when the user switches to the second font environment for the first time (or after a call to this function). An error message is displayed if this function is called when the current font environment is the only one which has some allocated memory. The error message is displayed, for example, if this function is called two times consecutively, whereas the first call would leave only one font environment in working condition.

The memory freed by this function is automatically reallocated when the user re-enters the font environment as described in section 3.2.

4.3 Load PFM Font

Sections 2.4 ("Storage of Fonts") and 3.23 ("The File Requester") ought to be read for a better understanding of this section.

This function loads a font previously stored in the PFM format. If the displayed font has not yet been stored, the Personal Fonts Maker displays a warning message. The "Proceed" gadget of the message requester must be selected (or the <Return> key pressed) to confirm the choice and load a new font which will clear the data of the font currently displayed. If the "Cancel" gadget is selected, or the <Esc> key is pressed, the function is aborted. The same happens if the "Cancel" gadget of the file requester is selected.

If the font which is to be loaded has a different format than that currently selected (section 7.3), a requester appears. The requester indicates the sizes and densities currently selected and those of the specified font. Four gadgets at the bottom of the requester let the user choose how to continue. If the "Proceed" gadget is selected, the font is loaded with no modifications to its bitmaps. The characters which do not fit in the current limits are cut. If the height of the selected font is smaller than the current font height, the origin of all character coordinates remains the top left corner of the character box. Selecting "Stretch", each character of the new font is stretched (or "shrunk") to the current format, depending on the currently selected stretch mode. Section 7.10 explains how to set different stretch modes. If the "Adapt" gadget is selected, the size and densities of the current font environment (section 7.3, "Font Description") are adapted to those of the selected font. If the "Cancel" gadget is selected, the operation is aborted.

The font file contains some data regarding the horizontal and vertical sizes and densities of the characters. If these values are the same as those of the current font environment, the font is loaded normally. Otherwise, the requester mentioned above is displayed. If the "Stretch" gadget is selected, the values are used to stretch the characters in the font so that they fit in the current font environment. If the "Adapt" gadget is selected, the font description variables of the current font environment (section 7.3) are adapted to the format of the font being loaded. The font description variables can also be modified manually using the requester described in section 7.3, or can be loaded from an existing parameter file as described in section 7.1.

The "Adapt" option should be selected with care if a font to be downloaded to a printer is being loaded, as the current format should already be set to the size and density required by the printer. Most low-end impact printers are not flexible enough to accept more than two different font heights (e.g. 8 for draft and 16 for letter quality), while the majority of page printers can deal with an almost unlimited range of font sizes.

Also contained in the font file is the name of the character set which was used when the font was last saved. This is usually the set following which the font was designed (section 2.8, "Character Sets"). If the current character set is different from the set after which the characters in the newly selected font are arranged, a warning message is displayed. The message informs the user about the character set used when the font was saved, and the current set (if one has been selected). The warning message is useful because if the current character set does not match the font’s set, the default character images used by the program (e.g. in the "Default" gadget and the "Quick Character Selection" box, sections 3.7 and 3.22) could be ordered differently from the characters in the font. Even if this message is displayed, the font is loaded normally, but the user is advised to load the specified character set too.

In most cases, only one character set is used. It sometimes happens, for example, that some fonts are designed specifically for a given dot matrix printer, while other - high resolution fonts - are designed for a laser printer using a different character set. If the same font environment is used to edit the different fonts alternatively, the character sets must also be updated accordingly. To re-arrange the characters in a font after a different character set, the Amiga font format can be used as an intermediate format (section
4.5

).

The "Load PFM Font" function does not load fonts in the standard Amiga font format. The "Import Amiga Font" (section
4.5
) must be used for this
purpose.

Appendix O contains a detailed technical description of the IFF CPFM font format used by the Personal Fonts Maker.

4.4 Save PFM Font

Sections 2.4 ("Storage of Fonts") and 3.23 ("The File Requester") introduce notions which are important for a proper understanding of this function.

This function saves the font data of the current font environment to the specified file. The standard file requester is used for the selection of the file.

The function stores only "On" characters (section 3.10). "Off" characters are not saved and do not occupy any memory in the file. In particular, the undefined character (sections 2.8 and 3.10), which is always saved by the "Export Amiga Font" function (section
4.6
) is not
saved if it is "Off".

Neither the character set, nor the brush, nor any macros, nor the program parameters are saved by this function. Sections
4.9
, 5.2, 6.2
and 7.2 respectively explain how to save this data.

Any problems encountered by the program or the Amiga operating system during the save operation are signalled through specific messages. Appendix G lists all program messages. Section 14.1 contains important information on how to handle any errors which may occur during a write operation.

Fonts stored by this function cannot be read by programs which handle only the standard Amiga fonts. If the font is to be read by one of these programs, it should be saved with the "Export Amiga Font" function, described in section
4.6
.

As described in section 2.4, fonts saved in the PFM format by this function contain more information that those stored in the Amiga format. PFM fonts also occupy considerably less storage space than Amiga fonts.

Appendix O contains a detailed technical description of the IFF CPFM font format used by this function to store fonts. Programmers are encouraged to use that documentation to write their own font handling routines.

4.5 Import Amiga Font

This function loads a font stored in the Amiga font format (section 2.4, "Storage of Fonts"). A special version of the file requester, called the font requester, is used for the selection of the font file (section 3.24, "The Font Requester").

This function is very similar to the "Load PFM Font" procedure, described in section
4.3
. Most notes regarding that operation also apply for this function. This is true in particular for the character stretch and adapting options.

The characters in an Amiga font file are re-mapped according to the character set currently in use (section 2.8). If the current character set is the Amiga set, the order of the characters in the font is left almost unchanged (any characters whose code is in the range 0-31 or 128-159 are ignored). But if the character set currently used is not the Amiga set (for example, the "PC_Usa2" set), the characters of the Amiga font are automatically inserted into the appropriate positions of the new set.

The conversion to a non-Amiga character set is very important if an Amiga screen font is to be used as a point of departure for a different type of font, like a printer-downloadable font. As explained in section 2.8, printers do not use the same character set used by the Amiga. In fact, there is not even a standard among printer character sets. This is one of the cases in which the program's flexibility of being able to use different character sets becomes invaluable.

The word "import" is used to define this function because the program has to do some conversions before the characters of an Amiga font can be processed by the user. As explained above, the characters may have to be re-arranged in the font. Compared to the PFM default format, Amiga font files lack some information (section 2.4) which is either calculated by the program, or replaced with default values.

The effects of a re-mapping to a non-Amiga character set are reversed when the font is exported back to the Amiga format (section
4.6
). It should be noted that only the two functions which read and write Amiga fonts use the character set data to change the order of the characters in the font.

The undefined character (sections 2.8 and 3.10) is always the last character in a font. The character has the decimal code 256, therefore it is the 257th character, since the code of the first character is 0 (zero). The undefined character is used only in Amiga fonts, and has no equivalent in the Personal Fonts Maker character sets. The undefined character is always loaded, edited and saved as the last character in the font.

If a character which appears in the selected Amiga font has no equivalent in the current character set, then that character is not loaded. As described in section 4.10 ("Define Character Set"), this happens when a -1 code appears in that character's position of the reverse encoding vector. If no character set is selected when an Amiga font is loaded, all positions of the encoding vector will contain the default -1 value. This means that only the undefined character can be loaded.

When the program starts, it automatically loads a default character set for each font environment. But if the user frees the character set data (sections 4.2 and 4.7), or if the program cannot access the default character sets, it may happen that - as mentioned above - no character set is selected. This could lead to the rather unusual situation just described, where only the undefined character of an Amiga font is loaded. It is sufficient to select a character set (sections 4.8 and 4.10) to load the desired characters from an Amiga font.

4.6 Export Amiga Font

Two sections are closely related to the concepts described here. Section 4.4 describes the "Save PFM Font" function, which is very similar to this function. Section 4.5 explains how the "Import Amiga Font" function, which is the reverse of the function described here, works. Both sections should be read as an introduction to this section. Appendix G explains the meaning of the error messages which may be displayed.

The "Export Amiga Font" function stores the current font in the Amiga font directory of the volume specified in the font requester. As explained in section 3.24 ("The Font Requester"), a special version of the file requester is used to access Amiga font files.

The program ensures that the characters of the current font are placed in the order defined by the Amiga character set. Therefore, if the character set of the font environment is not the Amiga set (sections 2.8 "Character Sets" and 4.10 "Define Character Set"), the program changes the order of the characters as necessary. Those characters which appear in the current font, but have no equivalent in the Amiga character set (e.g. a "heart" sign), are not saved.

An Amiga font must contain at least one "On" character (section 3.10) in addition to the undefined character. The Personal Fonts Maker saves only the characters marked as "On". The undefined character is always

saved, even if it is "Off", since an Amiga font must contain data for that character (sections 2.8 and 3.10). An error message is displayed if there is not at least one "On" character which has an equivalent in the Amiga character set. This can happen if no character set is defined, as already described in section
4.5
("Import Amiga Font").

When the Personal Fonts Maker modifies an Amiga font which is currently used by other programs, those programs continue to use the old version of the font.

It is possible to create a font file which can replace the default font used by the operating system for menu texts, Workbench icons and so on. To replace the "Topaz 8" (8 x 8 character size) and "Topaz 9" (10 x 9 character size) system fonts, for example, two fonts must be created. All characters in both fonts must be eight or ten dots wide, respectively. One font must be eight dots tall, the other nine. The "Fixed Pitch" flag (section 8.14) must be set in both fonts. Once the fonts are saved in the Amiga format, they can be selected with the "FF" (FastFonts, section 1.13) program. For example, the command

```
     FF fontname.font
```

will cause the font saved with the name "fontname" to become the default system font for that font size. The ".font" suffix is automatically appended by the Personal Fonts Maker to the name written in the font requester. The ".font" suffix must be added by the user of "FF" to the name which was specified in the font requester of the Personal Fonts Maker. The suffix does not need to be appended in the font requester by the user. The "FF" program can be found on the standard Amiga version 1.3 Workbench disks. The Amiga documentation explains how to use the command. Under version 2.0 (and beyond) of the Amiga operating system, there is no need to use the "FF" program, as the selection of the system font is a standard feature.

The remaining part of this section should be particularly interesting to programmers who access Amiga fonts directly.

The following table shows how the Personal Fonts Maker defines the Amiga font data structures:

Amiga Variable	PFM Parameter	Sections
Baseline	REF3 (minus 1)	2.3, 2.7.2.14, 3.21
CharBitWidth	X Size	2.7.2.22, 3.4
CharKern	Kerning	2.7.2.10, 3.6
CharSpace	Space	2.7.2.16, 3.5
Style, Flags	ATRB	2.7.2.1, Chapter 8
XSize (font width)	X Max	2.7.2.21, 2.7.2.22, 3.4, 7.3.1
YSize	Y Size (Y Max)	2.7.2.25, 2.7.2.26, 7.3.2

When an Amiga font is imported, the "X Max" parameter is set to the width of the widest character in the Amiga font, rather than to the nominal width stored in the Amiga "X Size" parameter.

It should be noted that the third reference point of a PFM font is used to calculate the position of the Amiga font baseline. This is based on the assumption (section 3.21, "Reference Points") that the most likely reference point to be used to mark the baseline is the third from the top, whereas the first two could be used to mark the positions of the cap line and the meanline, and the fourth for the underline position. The origin of the Amiga baseline is just below the origin of the reference points, therefore the program subtracts 1 (one) to the value stored in the REF3 variable to calculate the Amiga baseline.

Chapter 8 describes the attributes which can be associated with a font. Some of these attributes also apply for Amiga fonts, as explained in that chapter.

4.7 Free Character Set

Sections 2.8 ("Character Sets"),
4.5
("Import Amiga Font") and
4.6
("Export Amiga Font") are closely related with this and the ↩
following
sections dealing with character sets. All these sections should be read for a better understanding of the topic.

This function frees all the memory occupied by the character set data of the current font environment. This includes the default images of the characters in the set and the encoding vector used to convert characters between the current character set and the Amiga set. A warning message allows the user to abort the operation if the encoding vector of the current character set has been modified, and the set has not yet been saved.

After this command is executed, the current font environment will not be able to do any operations which require a character set to be defined, until a new set is loaded or defined (sections
4.8
and
4.10
). This means
that no Amiga fonts can be loaded or saved, as the program would not know where to place (i.e. which code to assign to) the Amiga characters in the current font, and vice versa. Without the default character images, both the "Default" gadget (section 3.7) and the quick character selection window (section 3.22) require a lot of imagination and good will.

This function frees only one character set. The character set data of the other font environment is not modified. It is not necessary to free the character set to load a new character set (section
4.8
).

4.8 Load Character Set

Different character sets can be used with the Personal Fonts Maker.

This makes it possible to work with fonts from different environments (section 2.8). For example, it is possible to edit Amiga fonts, 7-bit coded national fonts (appendix E), PC printer fonts and many more fonts all with the same program.

This function can be used to load and use a character set which has previously been defined and stored. The character sets described in appendices C (IBM PC), D (Amiga) and E (national 7-bit sets) are already stored in the default "PFM_CharSets" drawer (section 1.12). A new character set can be created by the user with the "Define Character Set" and "Edit Character Set" functions (sections
4.10
and
4.11
).

When a character set file is loaded, both the encoding vector and the default images of the characters are loaded, as explained in section 2.8 ("Character Sets"). A warning message allows the user to abort the load operation if the current character set has not yet been saved. If the operation is confirmed by the user, the previous character set data of the font environment is cleared.

It should be noted that this function does not copy the default character images contained in the file to the current font. To do this, the "Edit Character Set" function (section
4.11
) must be used.

Each font environment (sections 2.6 and 3.2) can use a different character set. When a character set file is loaded, the selected character set is applied only to the current font environment. The character set must be loaded again from the other font environment if it is to be used there as well. Initially the program automatically loads one default character set for each font environment (sections 1.12 and 2.6.8), whereas the second environment's character set is initialized only if there is enough free RAM.

4.9 Save Character Set

A character set created or modified by the user can be saved with this function, so that it can be loaded and used again in the future.

The descriptions of the standard file requester (section 3.23), the warning messages (section 7.11.4, "Confirm Overwriting") and the error messages (section 14.1 and appendix G) are also valid for this function.

As explained in section 2.8 ("Character Sets"), both the encoding vector and the images of the characters in the set are saved. The current font is used to create the default images of the characters. All font data, including the reference points and style attributes, is saved. This makes it easier to modify the default character images at any time. Section
4.11
("Edit Character Set") explains how to select the proper

font format (e.g. 16 by 18 character size). An error message is displayed if the format of the font is not suitable to create the default images (appendix G).

The format of the file created by this function resembles in part that of the standard PFM font file. A character set file is also an IFF file. Like a font file, a character set file contains graphic data for the default character images. Other parts of the file format differ, since a character set contains information not needed in a font file (i.e. the encoding vector), and vice versa. Appendix O contains the technical documentation necessary to access the Personal Fonts Maker character set files through external programs.

A character set cannot be loaded by the "Load PFM Font" function. To edit the default images of the characters in the set, the character set must first be loaded with the "Load Character Set" function (section

4.8

). Then, a font with the graphic data contained in the set must be created with the "Edit Character Set "function (section

4.11

).

This function is not the exact reverse function of "Load Character Set" (section

4.8

), as this latter function does not create a font with the default character images. The reverse of this function is "Load Character Set" followed by "Edit Character Set" (section

4.11

).

Once a character set has been saved, the name of that set can also be stored as a reference with all fonts created using that character set. For this reason, a character set file should never be deleted or renamed if fonts which reference that set exist. The worst thing that may happen in this case is that these fonts need to be loaded and saved again to update their internal reference to the character set (not so terrible, after all). To prevent this from happening by mistake, the Personal Fonts Maker also stores the name of the character set inside the character set file itself. This allows the program to display a warning message (appendix G) if a program other than the Personal Fonts Maker was used to rename the character set file. If the file is renamed, for example with the AmigaDOS "Rename" command, the Personal Fonts Maker displays a warning message when that file is loaded, since the file name is different from the internal name of the character set.

4.10 Define Character Set

As described in section 2.8 ("Character Sets"), each of the two font environments can work with a different character set. A character set consists of an encoding vector and a set of default character images. The encoding vector allows the program to exchange data between Amiga fonts and fonts based on a different character set (sections

4.5

, "Import Amiga

Font" and 4.6 , "Export Amiga Font"). The function described here allows the user to modify the encoding vector. Section 4.11 ("Edit Character Set") explains how to edit the character images.

The Personal Fonts Maker uses a flexible character encoding scheme. The association between each character in a font and its Amiga character set code is not part of the font, but is described by the encoding vector which is part of the character set. The encoding vector is only used to exchange data to and from Amiga font files.

The encoding vector is defined by a 256-element array. The 257th character in a font is always the undefined character, which is never translated and therefore does not need a position in the encoding vector. The array is indexed by character code (an integer in the range 0 to 255). The elements of the array are Amiga character set codes. This means that the encoding vector is a conversion table from codes of the selected character set to Amiga character set codes.

The encoding vector allows the Personal Fonts Maker to save a font which adopts any character set selected by the user to an Amiga font file. The encoding vector is used to re-order the characters in the font so that they follow the order specified by the Amiga character set (appendix D). If a particular character in the font does not exist in the Amiga character set, then it is not translated. A -1 (minus one) must be placed in the positions of the encoding vector associated with those characters which have no Amiga equivalent. For example, the code for the "heart" sign in the "PC_Usa2" character set is 3 (three). The Amiga has no "heart" sign, therefore it would be impossible to save that character of the font to an Amiga font. For this reason, a -1 (minus one) is placed at position 3 (three) of the encoding vector. Element number 3 is the fourth element of the vector, as the count starts from 0 (zero).

The Amiga character set can be used from the beginning to create a font. One of the predefined character set files which come with the Personal Fonts Maker contains the Amiga character set. The encoding vector of that set is very simple: each element contains the same code as the position code of the element in the array, except the characters whose decimal code in the Amiga set ranges from 0 to 31 or from 128 to 159, which have no graphical equivalent (these are marked by a -1 in the encoding vector). The encoding vector of most character sets is not as repetitive. The "PC_Usa2" character set, for example, uses the code 171 for the '½' (one half) character, whereas in the Amiga set the code of that character is 189. The position 171 of the encoding vector will therefore contain the value 189. Other characters are processed in a similar way. If a character in the "PC_Usa2" set has no equivalent in the Amiga set (like the "heart" sign mentioned above), a -1 (minus one) is placed in its position of the encoding vector. Some characters have the same code in both sets, as those whose codes are in the range from 33 to 126. Those positions in the vector will contain the same codes as the array element code, as described for the Amiga characer set. Appendices C and D illustrate the differences between the IBM PC and the Amiga set.

The encoding vector just described can only be used to convert characters to the Amiga set. What if a font is to be converted from the Amiga character set? The answer is very simple. The Personal Fonts Maker creates a reverse vector, based on the data stored in the original vector. The reverse vector is used by the "Import Amiga Font" function (section

4.5
).

Some character sets contain the same character more than once, i.e. the same character has more than one code. In IBM's Code Page 850, for example, the '\§' (section) character appears both at position 21, and at position 245. When the reverse encoding vector is built by the Personal Fonts Maker, only the highest of multiple codes is used. This means that when the '\§' (section, Amiga code 167) character is loaded from an Amiga font, it is copied only to position 245 of the font environment using the set mentioned above. Multiple assignments of the same character should, however, be avoided, as the effects are not always intuitive. All codes but one should be excluded, setting the associated parameter in the encoding vector to -1.

When the program is started, it automatically loads one default character set for each font environment (sections 1.12 and 2.6.8). If a set cannot be loaded, or if the user clears the character set data (section
4.7
) without loading a new set, all positions of the encoding vector will contain a -1 (minus one). Under these conditions, the "Import Amiga Font" and "Export Amiga Font" functions will not work, as explained in sections
4.5
and
4.6
.

All default characters with an associated Amiga conversion code other than -1 (minus one) must be marked as "On". This should be done when a character set is created or modifed, and before that character set is saved. If a character has no default image (i.e. if it is "Off") it cannot have a translation code. "Off" characters are not saved with the character set data. The conversion codes associated with these characters are automatically reset to -1 (minus one) when a character set is loaded.

The requester displayed by this function allows the user to change the codes of the encoding vector, so that a character set can be created or modified. The left part of the requester contains the "Character #" gadgets, similar to those described in section 3.3. The right part contains the gadget associated with the Amiga code linked to the character displayed at the left.

Each part contains a string gadget where the user can type a character code. Two gadgets under each string gadget allow the user to decrease or increase the value displayed in the string gadget by one unit at a time. On the right of each of the two string gadgets, there is the default image of the character associated with the displayed code.

The codes of the character set being defined by the user are used in

the left string gadget. Standard Amiga character set codes, as in appendix D, are used in the right string gadget. The left default image is the image of the character having the selected code in the set defined by the user. The images on the left side can be modified by the user, as explained in section
4.11
("Edit Character Set"). However, any redefined images are displayed in the requester described here only after the character set containing these images has been selected to be the current character set. The character images on the right part are those of the Amiga "Topaz 8" font.

Using the gadgets on the left part of the requester, the user can quickly access any position in the encoding vector. During this phase, the gadgets on the right part are automatically updated by the program so that they always show which Amiga character is associated with the character on the left. The user can link a new Amiga character with the character on the left, either using the keyboard to write a new code in the right string gadget, or with the two little gadgets below the string gadget. If the Amiga set has no equivalent character, a -1 (minus one) code must be specified on the right part.

4.11 Edit Character Set

This function can be used in conjunction with "Define Character Set" to modify or create a new character set. While "Define Character Set" (section
4.10
) is used to access and modify the encoding vector of the character set, this function allows the user to edit the default images of the characters in the set. Sections 2.8 ("Character Sets"), 3.7 ("The ’Default’ gadget"), 3.22 ("Quick Character Selection"),
4.7
("Free Character Set"),
4.8
("Load Character Set"),
4.9
("Save Character Set")
and
4.10
("Define Character Set") contain additional information on this subject.

A character set contains the images of the "normal" aspect of the characters included in the set. The program uses these images to show the user which character is currently being edited, or which character appears at a certain position in the character set. An Amiga font cannot be used for this purpose, as the character set selected by the user could contain characters which are not included in the Amiga set. Similarly, a "smaller" version of the font being edited by the user cannot be used for the same purpose, as that font could be completely "empty" or in the middle of radical editing operations. For these reasons, it is important that the default images in the character set contain clear and readable default images of all characters in the set.

The default images can be edited as if they were part of a standard font created with the Personal Fonts Maker. This function automatically adjusts some font parameters to edit the character set. The same parameters could, however, also be set manually, following a few rules. The characters must be 16 dots wide and 18 dots tall. All characters must have the same width. The X/Y ratio (sections 7.3.3 and 7.3.4) should be as close as possible to the screen display proportions, so that the characters displayed on the screen will look the same as in the character editing box. The average Width:Height proportions are 65:28 in NTSC mode (640 x 200 pixels), and 62:33 in PAL mode (640 x 256 pixels). The X Dpi and Y Dpi parameters (sections 7.3.3 and 7.3.4) can be set manually by the user to these values.

This function automatically selects the proper character size and proportions. This function also copies the graphic data, the reference points and the style attributes (chapter 8) of the default character images to the current font. A warning message is displayed before an unsaved font is cleared by this new data.

The font with the default images can be edited like any other font. Characters having codes not defined by the character set should be marked as "Off".

If the character set being edited is to be used to create printer downloadable fonts, all codes used by that particular printer as control codes should be marked as "Off". This will help the users of the character set to remember that some places in the font should not be occupied by graphic characters. For example, on most printers the decimal code 27 is reserved for the "Escape" command. If the character set had a user-defined sign in that position, the user could be "tempted" to place a similar character in the font. A font with such a redefined character would probably cause some "mysterious" printer errors if downloaded.

When the character set is saved, only the data associated with the "On" characters is stored. All conversion codes associated with characters marked as "Off" are reset to the default -1 (minus one) code when the character set is loaded.

4.12 Printer Test

This section contains the two subsections "Character" and "Font", named after the menu items used to verify the printed aspect of the entire font or the current character, respectively. The content of this section covers the general aspects of both functions. Each subsection contains a description of the details specific to each function.

These functions allow the user to verify the aspect of the current font when it is downloaded and printed. The functions are also useful to check whether the specified FFDL sequences work correctly with the printer, or need any changes.

The functions work in two steps. First, they output the font data processed by the FFDL sequences. This phase is similar to the "Write Font Data/Printer" function described in sections
4.13
and

4.13.2
. After the
download, some characters are sent to the printer, as described in the following subsections. The character codes are not processed in any way by the printer driver, i.e. the same codes which are output by the Personal Fonts Maker are also received by the printer. The character set of the current font environment must be the same as that of the printer.

Sections
4.13
("Write Font Data") and 7.3 ("Font Description") explain the execution of the FFDL sequences in more detail. Section 14.2 ("Problems with Printers") contains some suggestions on what can be done if the printer does not behave as expected.

4.12.1 Character

This function causes the character currently displayed in the character editing box to be printed. The character is printed four times on the same line. It is printed three times consecutively, followed by a space and then printed again. This allows the user to see what the character looks like when it is surrounded by other characters, or isolated. Also, having the character printed three times consecutively makes it possible to judge the appearance of patterns generated by repetitions of the same character.

The FFDL sequences are executed as if the first character and the last character in the range of the characters to be output were the same character, i.e. the current character. It is not possible to print an "Off" character, nor the undefined character (decimal character code 256). Similarly, it is not possible to print a character which is out of the selected range (sections 2.7.2.7 "FFDL Variables: FRST", 2.7.2.11 "FFDL Variables: LAST" and 7.3.9 "Range").

4.12.2 Font

This function prints all "On" characters in the specified range of the current font. For this reason, it is important that the valid range specified for the FFDL sequences (section 7.3.9 "Font Description/Range") covers all "On" character that are to be tested.

Each line contains 64 characters. If the current font has more than 64 characters, or the codes of these characters fall between two or more different multiples of 64, the characters are distributed over several lines.

4.12.3 Line Feed

This command advances the paper by one line. This can be very useful for reading a text which is still under the printer head without having to access the printer controls.

4.12.4 Form Feed

This command has been designed for page printers, i.e. printers which print one sheet of paper at a time (like most laser printers), and do not eject the paper until the text reaches the end of the page, or a "Form Feed" command is received. On some printers this command causes the printer to print the content of the page buffer and eject the page. On other printers, for example impact printers with automatic sheet feeder, the command causes the current sheet of paper to be ejected and a new sheet to be introduced from the paper hopper. If continuous forms paper is used, the command positions the printer head at the beginning of the following page.

4.13 Write Font Data

This section contains the two subsections "File" and "Printer", named like the menu items used to write the font data to the printer or to a file, respectively. The content of this section covers the general aspects of both functions. Each subsection contains a description of the details specific to each function.

The functions described here execute the FFDL sequences of the current font environment, in the order described in section 7.3 ("Font Description"). The resulting output is sent to a file or directly to the printer, depending on the selected function. Sections 2.4 ("Storage of Fonts"), 2.7 ("Programming the Output Format: the Cloanto FFDL") and 7.3 ("Font Description") contain important information on this subject. Chapter 13 contains several examples using these functions.

If an error occurs during the execution of the FFDL sequences, the execution is aborted and a message is displayed. Appendix G contains a detailed explanation of all messages. The data associated with the FFDL sequences executed before the error is output normally. If the "Font Description" requester (section 7.3) is displayed after the execution of an FFDL sequence is interrupted by an error, the cursor is automatically positioned where the error occurred.

During the output, the title bar on the top of the screen displays information relative to the current execution status. The following is a sample content of the title bar:

```
Writing in progress (0-255)   #115 - Bytes written: 3452
```

The title bar displays the following information: the codes (between parentheses) of the first and the last characters through which the FFDL sequences are cycling, as specified in section 7.3.9 ("Font Description/Range"); the code of the character currently processed by the FFDL sequences, or the "EPIL" or "PROL" abbreviations to indicate that the epilogue or prologue FFDL sequence is being executed (sections 7.3.5 to 7.3.8); the last number indicates the number of bytes already output, particularly useful for checking whether the memory limit of the printer has been exceeded. This information remains displayed for a few additional seconds after the termination of the data output.

4.13.1 File

This function stores the data output by the FFDL sequences associated with the current font environment to a file. The standard file requester (section 3.23) is displayed to let the user specify the file name. If the file already exists, a warning message is displayed, unless the user has disabled this function as described in section 7.11.4 ("Confirm Overwriting"). The function can be aborted by selecting the "Cancel" gadget of either requester.

The data stored to the file is exactly the same as the data which would be sent to the printer if the user so specified. If the output data contains a printer downloadable font, the file can be sent to the printer by a word processor or by the user clicking the Workbench icon. If the user enables the Workbench icon function (section 7.8, "Icons") the icon created by the Personal Fonts Maker will contain a reference to the "PrintRawFiles" program as a default tool. This means that when the user double-clicks on the icon of the file created by this function, the "PrintRawFiles" will automatically send the content of the file to the printer. Section 12.2 explains how the "PrintRawFiles" utility works.

4.13.2 Printer

This function is useful whenever printer downloadable fonts are created with the Personal Fonts Maker. Once the download format is specified through the FFDL sequences, the program can create font data ready to be processed to the printer. Downloaded fonts are printed with the same speed and quality as the printer's internal fonts.

This function causes the data to be output to the printer. A requester is displayed before the output begins. If the "Cancel" gadget of the requester is selected the operation is aborted.

The data is not translated or processed in any way by the Amiga printer driver, therefore it does not matter which printer driver is installed. The printer receives exactly the same data that would be stored in the file if the function was called to send the data to a file. The program uses the printer driver only to direct the data flow to the printer. The user must specify through the Amiga "Preferences" program, or another program designed for this purpose, the port to which the printer is connected. If more than one printer is connected to the Amiga, the one currently selected will be used.

When the program is initially loaded it checks whether there is a memory expansion to at least 1 Mbyte of RAM. If there is enough memory, the printer driver is automatically installed from the Workbench disk to the computer's RAM. This frees the user from having to insert the Workbench disk when this function (or "Printer Test") is called. In case of memory shortage, or if a different printer driver is specified through the "Preferences" program, the memory occupied by the printer driver is freed. If the printer driver is not in RAM, it will have to be executed from the Workbench disk (or hard disk). The Amiga operating system will ask the user to insert the appropriate disk if it is not already mounted.

If the printer is switched off, or if it is not "On Line", an Amiga system message will be displayed after a few seconds. If the user selects the "Cancel" gadget of the system requester, the operation will be aborted.

The font data is cleared from the printer’s memory whenever the printer is reset or switched off. If the user wishes to preserve the font data without having to load the Personal Fonts Maker again, the data can be stored in a file.

After a font has been downloaded and selected with the appropriate printer control commands, it is ready to be used by the printer when a program sends text to it. Section 14.2 contains useful suggestions for improving the output quality and solving any problems which may occur.

4.14 Font Statistics

This function displays a requester containing general information on the status of the current font. The requester does not contain any gadgets except "OK", at the bottom-right of the requester, which can be used to remove the requester. Pressing <Esc> or <Return> has the same effect.

The requester displays different types of information. The following subsections, named after the texts on the requester, explain the meaning of the displayed data.

4.14.1 Activation Table

The top part of the requester contains a map which shows the "On/Off" status of each character in the font. The table looks like a simplified form of the "Quick Character Selection" requester (section 3.22). The order of the characters is the same, but asterisks are displayed instead of the character images. Each column contains ten asterisks. Black asterisks are associated with "On" characters, light green asterisks with "Off" characters.

4.14.2 ON Characters

This number indicates the quantity of active ("On") characters in the font. The number is the same as that of the black asterisks in the top part of the requester. The undefined character is included in this count, if it is "On".

This information can be very useful when a certain limit of "On" characters cannot be passed. For example, some printers will not accept more than 128 characters to be downloaded. If the current font is to be downloaded to such a printer, this field can be checked to prevent any inconvenience.

4.14.3 Average Width

This field contains the average width, in dots, of the characters marked as "On". This information can be useful to get a quick idea of how large the characters in the font are.

The displayed value could be used to modify the density setting for proportional characters on a word processor which is to print with the font. Section 13.6 ("Creating a Word Processor Font Size Table") explains how to obtain more detailed information on the size of the characters.

If all the characters have fixed-width, the displayed value will be the same as the character width.

4.14.4 Widest Character

The displayed value indicates the code (as in section 3.3, "Character #") of the widest "On" character in the current font. If more than one character has the same width, the value indicates the position of the character with the lowest code.

The number indicating the character code is followed by a second value, enclosed between parentheses, which gives the width, in dots, of that character. If a proportionally-spaced font whose limits are not known is loaded or imported, this value can be used to determine the minimum value of the "X Max" parameter (section 7.3.1). If the font has fixed-width characters, the second value is the same as the average width and the width of the narrowest character.

4.14.5 Narrowest Character

This field is similar to "Widest Character", except that the code and the width values of the narrowest character are displayed.

4.14.6 Font Status

The text indicates the status of the current font. One of six texts can be displayed:

"New"
"Loaded"
"Imported"
"Stretched"
"Saved"
"Modified"

The font is "New" if the program has just been loaded, or if the "New Font" (section
4.1
) was executed. When the current font is "New" the title bar contains a copyright notice instead of the name of the font.

"Loaded" indicates that the current font has been loaded from a font file in the default PFM format. The font has not yet been modified.

"Imported" is similar to "Loaded", but indicates that the current font has been imported from an Amiga font file (e.g. an Amiga screen font).

"Stretched" means that the font has just been automatically stretched by the program from one format to a different one. This can also occur when a font is read from a file, if the "Stretch" option has been selected. Section 7.10 explains how the stretch works.

The status of the current font is "Saved" if the font has just been saved in the PFM format. The status is not changed to "Saved" when a font is exported to an Amiga font file, as the Amiga font format does not contain all the information of the PFM format, and some characters in the font may not have an equivalent in the Amiga character set. The "Saved" status is used to inform the user that the current font can be safely cleared, with no loss of data.

The font becomes "Modified" as soon as the user edits a character image, defines a new position for a reference point or changes the style attribute flags. The "Modified" status replaces any other previous status. If the previous status was "New", the default "Unnamed.fnt" font name is displayed on the title bar.

4.15 Memory Information

This function displays a requester containing information on the amount of available RAM. The "OK" gadget of the requester can be selected to remove the requester.

The requester contains different numbers. The displayed values are calculated before the requester is displayed. In this way, the memory temporarily occupied to display the requester is not subtracted and counted as occupied. The unit for all numbers is the byte. Sections 1.3 ("Memory") and 2.4 ("Storage of Fonts") contain additional information on memory.

The upper part of the requester contains nine numbers, divided into three columns of three rows each. The three rows are labelled: "Chip", "Fast" and "Total" (abbreviated "Tot."). The three columns are: "Used", "Free" and "Largest".

"Chip" stands for "Chip Memory", which is the portion of the system memory which can be accessed by the Amiga special-purpose custom chips. The custom chips are used, for example, to handle graphical data, like the contents of screens and windows. If there is not enough chip memory, some operations, like the opening of a requester, may fail. Section 7.7 explains how some chip memory can be freed by closing the Workbench screen.

All the remaining RAM is called "Fast" memory. This memory is outside the range that the custom chips can access. It is called "Fast" because the custom chips cannot access this memory, therefore the microprocessor does not have to wait if a custom chip is working with the same memory (this is called "bus contention").

The Amiga’s memory is either "Chip" or "Fast", therefore the sum of the two yields the "Total" memory. The "Total" row contains the sums of each column’s "Chip" and "Fast" memory occupation numbers.

For each row (type of memory) there are three columns: "Used", "Free" and "Largest". The first number in the row indicates how much of the particular kind of memory associated with the row is already occupied. The second number tells how much memory is still available to be used by the Amiga operating system or other programs. The third number indicates the size of the largest chunk of free memory.

The "Largest" field deserves a more detailed explanation. The free memory is usually fragmented into several non-contiguous segments. For example, 200 Kbytes of free memory may consist of a single chunk of 140 Kbytes, another 50 Kbytes long, and several other smaller chunks for a total of 10 Kbytes. If a program asks the system for a block of 200 Kbytes, it will not get that memory. The available 200 Kbytes cannot be merged into one single chunk, as other programs are using the memory between the free chunks. It is possible that new Amiga operating systems, beyond the versions available at the time of writing, take advantage of the features of the more powerful microprocessors mounted in the Amiga, making the operation just described possible. Currently, the only thing that can be done by the operating system in case of memory shortage is the freeing of some libraries, devices and other resources which are not currently used.

In a multitasking environment like the Amiga, it is possible that programs other than the Personal Fonts Maker allocate and free memory. Therefore, the displayed values represent only a "snapshot" of the current situation.

The lower part of the requester contains information regarding the memory currently allocated by the program’s two font environments. This includes the bitmap memory for each font’s character images and the memory reserved for the character buffer, the brush and the "Undo" function.

The amount of memory occupied by the graphical data of the current font is independent from the quantity of "On" characters. However, the program always allocates enough memory for a full font consisting of 257 characters in the maximum allowed size. The memory occupied by the "Undo" function and the character buffer is always sufficient to contain the largest character in any of the two font environments. The amount of memory reserved for the brush depends on the size of the selected brush. The displayed values do not include the memory occupied by the program code itself or by its screens and windows.

The first two numbers, introduced by the "Font 1" and "Font 2" texts, contain the memory occupied by the graphical data of each font environment separately.

These amounts of memory, plus the memory occupied by the default images of the character set and the space, offset and character conversion vectors, are those which would be freed by the "Free Font Memory" function (section
4.2
) on each of the two font environments. The third number ("Total") is the sum of the two values, plus the memory occupied by the undo, character buffer and brush functions (sections 3.9, 3.11 and 3.17). This total is already included in the data regarding the total system memory occupied, displayed in the upper part of the requester.

4.16 Delete

The Personal Fonts Maker can also delete files. This may be used to remove old files which are not needed any more, or to rename a file, by deleting it and saving it again with the new name. Two functions can be used, each of which is described in the following subsections.

4.16.1 File

This function can delete any file accessible through the Amiga operating system. The standard file requester (section 3.23) can be used to delete the file. A warning message is displayed before the file is deleted. The operation can be aborted by selecting the "Cancel" gadget of either the file requester or the warning message.

This command can be used to delete any kind of file, regardless of its content. The function may be used freely to remove parameter files, macro files, character sets and fonts in the Personal Fonts Maker default format. The command should not, however, be used to delete Amiga fonts, as there is a special function for this purpose (section 4.16.2).

If a Workbench icon is associated with the file to be deleted, it is removed as well.

4.16.2 Amiga Font

This is a variant of the "Delete File" function, described in section 4.16.1. The font requester (section 3.24) is used instead of the more general file requester to select the Amiga font to be deleted. This is the function which must be used to delete Amiga fonts. All other files, including fonts saved in the Personal Fonts Maker format, can be deleted with the "Delete File" function (section 4.16.1).

The reason for which a special function is needed to delete Amiga fonts, is that data regarding such a font is not stored in a single file. When an Amiga font is deleted, information in other files must be updated by the Personal Fonts Maker. If other programs are used to directly delete, or in any way alter a file containing Amiga font data, the Commodore "FixFonts" program must be used to reorganize the font information structures.

An error message is displayed if a font which is stored in ROM (e.g. the default "Topaz 8" font) is selected, as such a font cannot be deleted.

4.17 About

A requester appears when this option is selected. Some information on the Personal Fonts Maker version, the copyrights and its authors is displayed. The "OK" gadget (or the <Return> or <Esc> keys) can be selected to remove the requester.

4.18 Quit

This command is used to terminate the work with the Personal Fonts Maker. A warning requester can appear to inform the user about all the pieces of data which have not yet been saved. The "Proceed" and "Cancel" gadgets of the requester can be used to confirm or cancel the command. If the choice is confirmed, the program terminates. The screen is closed, all memory and resources occupied by the program are freed.

It is good practice to always exit from all programs before switching the computer off or resetting the system (with the <Commodore> + <Amiga> + <Ctrl> keys).