TTFLIB

COLLABORATORS

	TITLE : TTFLIB	

ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		August 24, 2022	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

TTFLIB

1.1 main

```
                ttf.library version 0.6.2

A truetype font engine for Amiga computers.


                 Overview

                 Credits/Legal

                 Requirements

                 Installation

                 Usage

                   ttfmanager

                   ttfinstall

                   ttflist

                   ftview

                 Limits

                 Recent Changes

                 Future

                 FAQ
                The most current publicly available version of this software
can be found at the following web address:

      http://home.sprynet.com/sprynet/ragriffi/

Send comments, suggestions, and bug reports to:
```

Richard Griffith
ragriffi@sprynet.com

1.2 ttf.library Overview

Overview

ttf.library is a truetype compatible font engine for Amiga OS.
It functions in a manner compatible with the outline font engine
standard established by Commodore with the bullet.library engine
for compugraphic format fonts. This means that Amiga applications
which use normal system fonts should now be able to use truetype
format fonts.

1.3 Credits and Legal Issues

Credits and Notes

The ttf.library and related programs were made possible by
the outstanding achievement known as the FreeType Project.
For more information, visit http://www.freetype.org/

Also, Amish S. Dave's type1.library (which does for postscript
type1 fonts what ttf.library does for truetype) served as an
invaluable inspiration and an initial guide for the rather
poorly documented amiga outline font engine format.

Legal

I am not a lawyer, nor do I want to be, so expect plain
english here.

First, this library comes with NO WARRANTY.

ttf.library is free, think of it as my gift to the faithful.
I do not restict its use and/or distribution, however, some
of the technology used is covered by a separate license, see
the
FreeType license
for details. I do ask that if you
distribute it, please don't try and call it your own work. I
know better, you know better, and I'm sure karma will get you.
If you sell it, be advised you will have stiff competition,
as it will continue to be available free. (As a side note, no
I don't think all software should be free. I make my living as
a programmer, consultant, general geek type, so I do place a
great value on software. This one, though, I feel should be
free. So there...)

1.4 FreeType Project LicenseSystem Requirements

The FreeType Project LICENSE

Introduction:

The FreeType Project is distributed in several archive packages; some of them may contain, in addition to the FreeType font engine, various tools and contributions which rely on, or relate to, the FreeType Project.

This license applies to all files found in such packages, and which do not fall under their own explicit license. The license affects thus the FreeType font engine, the test programs, documentation and makefiles, at the very least.

This license was inspired by the BSD, Artistic and IJG (Independent JPEG Group) licenses, which all encourage inclusion and use of free software in commercial and freeware products alike. As a consequence, its main points are that:

- We don't promise that this software works. However, we'll be interested in any kind of bug reports. ("as is" distribution)
- You can use this software for whatever you want, in parts or full form, without having to pay us. ("royalty-free" usage)
- You may not pretend that you wrote this software. If you use it, or only parts of it, in a program, you must acknowledge somewhere in your documentation that you've used the FreeType code. ("credits").

We specifically permit and encourage the inclusion of this software, with or without modifications, in commercial products, provided that all warranty or liability claims are assumed by the product vendor.

Legal Terms:

0.Definitions:

Throughout this license, the terms "package", "FreeType Project", and "FreeType archive" refer to the set of files originally distributed by the authors (David Turner, Robert Wilhelm, and Werner Lemberg) as the "FreeType project", be they named as alpha, beta or final release.

"You" refers to the licensee, or person using the project, where "using" is a generic term including compiling the project's source code as well as linking it to form a "program" or "executable". This program is referred to as "a program using the FreeType engine".

This license applies to all files distributed in the original FreeType archive, including all source code, binaries and documentation, unless otherwise stated in the file in its original, unmodified form as distributed in the original archive. If you are unsure whether or not a particular file is covered by this license, you must contact us to verify this.

1. No Warranty:

THE FREETYPE ARCHIVE IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT WILL ANY OF THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY DAMAGES CAUSED BY THE USE OR THE INABILITY TO USE, OF THE FREETYPE PROJECT.

As you have not signed this license, you are not required to accept it. However, as the FreeType project is copyrighted material, only this license, or another one contracted with the authors, grants you the right to use, distribute, and modify it. Therefore, by using, distributing, or modifying the FreeType project, you indicate that you understand and accept all the terms of this license.

2. Redistribution:

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistribution of source code must retain this license file (LICENSE.TXT) unaltered; any additions, deletions or changes to the original files must be clearly indicated in accompanying documentation. All copyright notices must be preserved in all copies of source files.

- Redistribution in binary form must provide a disclaimer that states that the software is based in part of the work of the FreeType team in the distribution documentation. We also encourage you to put an URL to the FreeType web page in your documentation, though this isn't mandatory.

These conditions apply to any software derived from or based on the FreeType code, not just the unmodified files. If you use our work, you must acknowledge us. However, no fee need be paid to

```
  us.


3. Advertising:

  The names of  FreeType's authors and contributors may  not be used
  to endorse or promote  products derived from this software without
  specific prior written permission.

  We suggest,  but do not require, that  you use one or  more of the
  following phrases to refer  to this software in your documentation
  or advertising  materials: "FreeType Project",  "FreeType Engine",
  "FreeType library", or "FreeType Distribution".


4. Contacts:

  There are two mailing lists related to FreeType:

    o freetype@lists.lrz-muenchen.de

      Discusses general use and applications of FreeType, as well as
      future and  wanted additions to the  library and distribution.
      If  you're looking  for support,  start  in this  list if  you
      haven't found anything to help you in the documentation.


    o freetype-devel@lists.lrz-muenchen.de

      Discusses bugs,  as well  as engine internals,  design issues,
      specific licenses, porting, etc.


    o http://www.physiol.med.tu-muenchen.de/~robert/freetype.html

      Holds the current  FreeType web page, which will  allow you to
      download  our  latest  development  version  and  read  online
      documentation.


  You can also contact us individually at:

    David Turner      <turner@email.enst.fr>
    Robert Wilhelm    <robert@physiol.med.tu-muenchen.de>
    Werner Lemberg    <wl@gnu.org>


                        --- The End ---
```

1.5 System Requirements

```
Requirements

OS3.0 or higher
68030 or higher processor
```

It would be possible to make a plain 68000 version, but so far
no one has asked.

1.6 Installation

Installation

Install-ttflib is a standard installer script, just double
click the icon.

If you prefer a manual installation, here is a breakdown
of the primary components:

File	Installation instructions
ttf.library	copy to your LIBS: drawer
ttfmanager	copy to your favorite tool drawer (i.e. sys:utilities)
ttfinstall	" " " " " " (optional)
ttflist	" " " " " " (optional)
ftview	" " " " " " (optional)

1.7 Using ttf.library

Usage

The following tools are available:

ttfmanager
make truetype fonts available to system

ttfinstall
alternate (CLI) tool to install fonts

ttflist
list names and other details from font files

ftview
preview a font

1.8 Using ttfmanager

ttfmanager

Please note. This is the first release of ttfmanager. It is
not complete, but it provides a Workbench environment that is
as capable as
ttfinstall
. Look for future improvements.

To use a truetype font, it must first be installed to some part
of the system FONTS: drawer. Installation is a bit different than
with the compugraphic intellifont program. First, outlines (the
actual truetype font files) are NOT copied to the font directory.
Instead only the ".font" and ".otag" files are created in the
selected directory, and those reference the outlines at the same
locations from which they were installed. This makes it much
easier to use large font libraries directly from CD-ROM, for
example. The second major difference is that the fonts may be
installed to any drawer, not just a FONTS: component. (Note that
to USE a font in many programs, the drawer must be assigned
to FONTS:, but this allows you to selectively add the required
font path components as needed.) This allows for the rotating
font assignments that are practically required with large font
collections. (Image how long it would take to load the standard
system font requester if you had 6000 fonts actually available!)

To install truetype fonts, enter a source directory (where the
ttf files are located) in the source directory box. Clicking
the 'set' button to the right will bring up a standard file
requester to allow you to choose the directory. A list of the
internal names of all installable truetype fonts in the source
directory will be generated in the Available Fonts list.

Set the Destination Drawer in a similar fashion.

To install a single font, click on it in the Available Fonts
list. If you desire, you can change the name in the Font Name
box. Then click the 'Install' button.

To install all of the fonts in the Available Fonts list, click
on the 'Install All' button. All fonts installed this way
will get default names and options.

Please note that ttfmanager will OVERWRITE the .font and .otag
files if they already exist, thus installing two fonts with
the name 'CoolNewFont' will result in only one available font.

1.9 Using ttfinstall

```
                ttfinstall

Note:
                ttfmanager
                 is now available and will perform font
installation from the Workbench environment. This tool is
included for those who want CLI and scripting power. For
an overview of font installation, including differences
from the system intellifont program, see
                ttfmanager
                .
```

The command line for installing a truetype font looks like

the following:

```
  ttfinstall fontfile.ttf sys:fonts [EXACT | TYPO | DESIGN]
```

Note that 'fontfile.ttf' is the actual file to be installed, and 'sys:fonts' can be any existing directory. In addition, a file name pattern can be specified for 'fontfile.ttf', allowing the installation of several fonts at once. For example:

```
  ttfinstall cd0:funky/#?.ttf work:ttfonts
```

Would install all the font found in the cd0:funky drawer to the ttfonts drawer on volume work:

One optional switche, either EXACT, TYPO, or DESIGN may be specified. These control the source of the font metrics used to calculate the size ratios needed to map truetype point sizes to the Amiga style point specifictions.

EXACT, the default, examines each glyph that will be used, to determine the ascent and decent extremes. This is more acurate than any other method, but noticeably slower.

TYPO, which was the method used in version 0.5 and below, uses the precalculated Typo values from the OS/2 table in the font.

DESIGN uses values within the font header which express the designers intentions, not neccessarily the actual values.

As these options suggest, there are many ways to get these values, and none of the precalculated values is correct in every font. diskfont.library expects glyph bitmaps to be limited to the declared point size in height. If this size is exceeded, the result is usually baseline shifting from glyph to glyph, or clipped extremes on one or more glyphs. The default setting should be correct, but the other options are made available for those who like to tinker.

Once a truetype font is installed on the fonts: path, it should be possible to select that font from any decent font requester.

Please note that ttfinstall will OVERWRITE the .font and .otag files if they already exist, thus installing two fonts with the name 'CoolNewFont' will result in only one available font. It is possible to rename both the .font and .otag files once they have been created, so multiple fonts with the same internal name can be used if you work at it. ttfmanager has an option to change the name when installing individual fonts.

1.10 Using ttflist

ttflist

ttflist works a lot like the normal system 'list' command, but

```
is designed to display readable names along with the file name.
Also, it will optionally provide a great deal of perhaps useful
information from a font file. The command line would look like:

  ttflist [pattern] [all] [verbose] [encode]

where:

 pattern  is an optional amigados file pattern or
          drawer name. By default it is "#?.(ttf|ttc)"

 all      is an optional switch to list matching drawer
          contents as well. Note that the pattern specified
          must match the drawer names to be searched. When
          all is specified the default pattern is "#?".

 verbose  gives a lengthy, and probably ignorable, report for
          each font found.

 encode   gives a verbose report and displays the results of
          encoding mapping all 256 amiga characters against
          all encoding tables in the font file.

(verbose and encode are mainly for the developers benefit ;)

example:  ttflist cd0: all
```

1.11 Using ftview

```
ftview

ftview is a truetype font display program from the freetype
project test suite, with very minimal 'amigaizing'. Its
command line format is:

  ftview [-g] [-r res] pointsize fontfile ...

where

 -g        is an optional grey scale (smoothing) rendering.
           Smoothing looks very wonderful, but is not available
           through ttf.library. Also note that ftview opens on
           the default public screen, and if that screen does
           not have sufficient available or matching pen colors,
           it may look worse. The -g option uses five distinct
           grey levels.

 -r res    specifies an optional resolution to use in rendering
           the qlyphs of the font. 'res' is in Dots Per Inch.
           Note this doesn't change the display resolution
           only the value used for font rendering.

 pointsize is a required argument specifying the point size
           to display.
```

```
 fontfile  is one or more truetype font files to view

Example: ftview 30 flyingp.ttf

While the font is displayed, options to change many rendering
characteristics are available by keystroke or standard amiga
menu selections.
```

1.12 Limits and known problems

```
Limits, warnings

This software has had minimal testing. Its performance on
the development machine has been constantly monitored with
enforcer and mungwall, however, it is still unproven. Also,
it relies VERY heavily on code which I did not write, and
have not examined in great detail. (Although, I must note
that the FreeType code seems very solid!) Please use caution,
and use at your own risk.

Several feature of the bullet library standard have not yet
been implemented (rotation, shearing, kerning pairs) as they
are not required for proper font operation via diskfont.library
Programs expecting to use advanced (or undocumented) features
of the bullet.library may not work as expected with the
ttf.library.
```

1.13 Recent Change History

```
Changes in v0.6.2

  Only the ttf.library has changed in this version.

  The quest for perfection continues - the character advance
  calculations are virtually identical to that common M$ OS
  engine. An issue with non-glyph widths (i.e. the space
  character will be fixed in the next installer version.

  Source of mystery crashes (of other tasks) found and fixed.

Changes in v0.6

  Re-examined the character advance calculations to improve
  spacing at smaller sizes.

  Modified the font install routines to calculate the extremes
  of the ascent and descent by default. Problem fonts (baseline
  shifts or clipped glyphs) should be re-installed with the
  new ttfmanager or ttfinstall version.

  Added a fake version number to ttf.library for WordWorth.
```

```
Changes in v0.5

  Added ttfmanager and Install-ttflib.
  Re-examined linkages, often resulting in smaller file sizes.

Changes in v0.4

  WidthLists are now supported.

  Subtle changes to returns of non-existent glyphs to be more
  compatible with bullet, and make smaller (memory wise) fonts
  when called by diskfont.library.

  ttfinstall - correct advance width problem with monospaced fonts

Changes in v0.3

  Sets many more tags in otag file to make some programs much
  happier (FinalWriter, to name one.) As a result, ALL FONTS
  installed with v0.2 or earlier MUST BE RE-INSTALLED. Sorry,
  but the results are usually worth it.

  Mimic bullet.library in handling of glyphless codes, such as
  a space.

  Widths are calculated more appropriately, resulting in better
  display under many conditions.

  Several other peculiar, but un-enumerated, bugs were squashed.
```

1.14 The future of ttf.library

```
Future

  Many improvements in the works for ttfmanager.

  Alternate character set capability. Although many truetype
  fonts contain many glyphs outside the realm of plain Latin1,
  ttf.library must use a (normally) unicode encoding table
  from the font to obtain the appropriate glyph, and the only
  point of reference is the normal ECMA-Latin1 that is native
  to the Amiga. To adapt to non-latin1 locales, it is common
  to establish new meanings for the single byte sized system
  character set. I am considering an optional table that
  could be used by the library to specify a 16 bit unicode
  meaning to each of the 8 bit system codes, allowing the use
  of any subset of the font's available glyphs. If you would
  have a need for such an enhancement, please let me know.
```

1.15 Frequently Asked Question

FAQ

Can we make bitmaps, please?

The number one requested enhancement is for bitmap font generation. It is comming. In the meantime, a font editor that uses diskfont.library to load a font will do the job. Also, Bitline by Georg Steger holds some promise.

How about an 060/PPC version

Once the development of the library has slowed down, support for additional processors is planned. As it is, I spend more time on packaging than development, and additional variations would add to that. A PPC creates some additional concerns, but it is being considered.

Why are the fonts so tiny? 8 point soandso.font looks bad. On a PC I can use 7 point arial, but it is just little squiggles on my amiga. Etc, etc.

A quick lesson of fonts and terms is in order. In more traditional typography, a point is a standard unit of measure, approximately 1/72 of an inch. A point size refers to the size of an imaginary box surrounding the letter 'M', know as the em square. The size of this box in 1/72 inch units is the point size. When Windoze defined its version of point size, it invented a new concept, the virtual display inch (I'm not kidding!) which is "approximately 30 to 40 percent larger" If you consider that the declared resolution of a typical Win display is 96DPI square, you'll find that the math works out to the em square point measurement is approximately the pixel height count of the letter M.

The Amiga however uses 'points' to specify the entire height of the font, not just the letter M. There is no provision, through diskfont.library for a glyph to fall outside of a 'point' high bitmap. As a consequence, a box which will hold the 'M' won't hold a 'Ç' or an 'È' or any other characters. The needed room must come from somewhere, and this means that the sizes must be scaled to fit. Since ttf.library is at the bottom of the chain, the apparently inflated point size must be specified. As a result, requests for 8 point glyphs will usually result in total garbage, as 3-4 'points' get used for any needed ascenders and descenders, leaving little to reflect the characters. 10 is sometimes recognizeable, and 12 starts to be useful for most fonts.

To match an Amiga point size exactly to those of another platform, examine the highest ascender, lowest decender and the letter 'M' on the other platform. The letter 'M' should be approximately the point size pixels tall. Add

the additional pixel heights of the high and low extremes to that point height, and you should have a close value for a corresponding Amiga point size.

Why does the font look different in ftview?
Why does this look good in ftview, and bad as a font?

The glyphs shown in ftview and those displayed using ttf.library should be identical, if called with the same parameters, which is not usually (or should I say, easily) the case. Anything that calls upon ttf.library must specify a resolution (dpi) for both x and y dimensions. When diskfont.library does the call, it picks its own value to try to translate the point size to the Amiga form, which is basically a pixel height. So, in example, asking for Arial/18 in a font requester, asks for 18 point glyphs at a DPI of 66 by 66. On the otherhand, asking for 'ftview 18 arial' gives 18 point glyphs at 96x96 DPI, it's default, and a typical VGA resolution on a PC. Since truetype hinting can change based on many factors, such as resolution and point size, it can actually be an entirely different code sequence run to do the hinting in these two cases. The hinting is actually coded in the font, not in the engine, so if the font designer/writing program doesn't produce good code to apply to the needed resolution, ugly glyphs result.

In this case, using 'ftview -r 66 18 arial.ttf' produces identical glyphs. I qualified this as 'in THIS case' because one of the things which can happen is that the dpi and/or point size requested can vary especially when the design of the font stretches beyond the EM square (which truetype fonts are free to do.) There is a ratio (OT_YSizeFactor) in the .otag file, which is set by ttfinstall to help diskfont.library adjust things so that the glyphs will fit in the Amiga defined point (pixel) dimensions. This means that the exact request to the rendering engine can vary from font to font. The good side of this is that it cleans up numerous errors that would otherwise result from glyphs which don't quite fit in the space available. Bad news, it makes it harder to match the ftview display.

What can I do to make my browser look like a PC?

You can match the font sizes used by default in MSIE or Mozilla on a Winbox. There will be differences in the layout by the browser, but the letters in the words could be made to match in most ways. Version 0.6 has had several tweaks to more closely match the Win way of spacing fonts. The biggest problem is in finding the correct sizes to specify, which are presented below. You will need to obtain the TimeNewRoman and CourierNew font families. I leave all legal, ethical

and tactical aspects of this to the individual.

Matching fonts/sizes are as follows:

```
    H1          TimesNewRomanBold/35
    H2          TimesNewRomanBold/27
    H3          TimesNewRomanBold/20
    H4          TimesNewRomanBold/18
    H5          TimesNewRomanBold/14
    H6          TimesNewRomanBold/11

    Small       TimesNewRoman/14
    Normal      TimesNewRoman/18
    Big         TimesNewRoman/20

    FONT SIZE=1 TimesNewRoman/10
    FONT SIZE=2 TimesNewRoman/14
    FONT SIZE=3 TimesNewRoman/18
    FONT SIZE=4 TimesNewRoman/20
    FONT SIZE=5 TimesNewRoman/27
    FONT SIZE=6 TimesNewRoman/35
    FONT SIZE=7 TimesNewRoman/53

    Preformated CourierNew/14
```

Can’t open ttf.library version 2

This is a bug in WordWorth. It has no reason to specify a library version at all when opening the library. This is likely a holdover from a very early bullet.library, but it is still WRONG! In the meantime, I have faked the version in the library header to pass this test. Version 0.6 appears as version 10.6 to the system. Any future versions will continue to be offset by 10, as long as required.