XFree86 Font De−uglification HOWTO

Hal Burgiss

hal@foobox.net

v2.04c, 30 October 2002

How to improve X Window fonts. Various tips for improving font handling for XFree86, including sections on font servers, TrueType fonts, Mozilla, and related topics.

Table of Contents

Table of Contents

1. Introduction

An often heard complaint is the poor default fonts and font settings of X as implemented by many Linux distributions. Some programs use fixed width default fonts when a variable width font would be more appropriate. Other programs use fonts that are so small as to be practically unreadable. Many of the fonts that are bundled with XFree86 are not of the same quality as found on some other platforms. XFree86 does come with a halfway decent courier font, but its Times and Helvetica fonts are simple bitmap fonts that pixelize when scaled. This is changing for the better recently, but a default Linux desktop still often needs some tweaking to get the best fonts possible.

This HOWTO attempts to show how to adjust various font settings, install new fonts, and a few other things that should greatly improve the appearance and readability of fonts on the X Window Desktop. This is done by adjusting the `FontPath` in the `XF86Config` file, by adding switches to X server command line in **startx** or **xdm** (and variants), by adding new fonts, and by making sure TrueType fonts are installed and available. TrueType does indeed make a huge difference in many applications.

Comments, corrections, additions and critiques are always welcome. You can reach the author at <`hal@foobox.net`>. Contributions are also welcomed. Especially anyone who really stays current with KDE and/or GNOME issues! (So I guess nobody uses these since the silence is deafening?)

1.1. Conventions Used in this Document

- "X" and "XFree86" are (incorrectly) used here interchangably. The overwhelming percentage of Linux users indeed use XFree86.org's X implementation, but there are other implentations as well. The information here probably does not apply to those others in most cases.
- Where examples of commands are used, a "#" character is used to denote where typically the command would be run as the root user. A "$" is used where typically a non−root user would be executing the command.
- The examples use `/usr/local/share/fonts/ttfonts` as our TrueType font directory. There is no magic to this location, and could conceivably just as well be in any number of other locations. Some distros may have a default location for TrueType fonts, and you may want to use that instead.
- References to "xfs" are to the xfs as packaged by Red Hat (and some other distros) for versions 6.x and later. This differs significantly in some respects from the stock XFree86 xfs.
- References to "Netscape" are to the entire suite of programs from Netscape: Communicator, Navigator, Messenger, etc. And for all intents and purposes, font configuration in Mozilla is very similar (but generally looks better!).
- 'XF86Config' is the X configuration file. This has changed to 'XF86Config−4' for XFree86 v4.x. For the most part, we'll just use 'XF86Config' here.

  Also, while some aspects of XFree86 4.x configuration are the same as 3.3.x, there are some significant differences. We'll only highlight the differences. So unless noted otherwise, any comments or examples should apply to both 3.3.x and 4.x versions.
- File system layout varies somewhat from distribution to distribution. It is impossible to stay on top of every conceivable variation of who keeps which files where. So take the examples here with a grain of salt if the PATHs don't seem to match your system.

1.2. Change Log and What's New

- 2.04: October 30, 2002. Include info on TrueType "hinting", and enabling in freetype sources (thanks Andreas Oesterhelt!). A few new useful links, and minor cleanups.

  If this keeps up, this document will have to be renamed the *Font Beautification HOWTO* ;−).
- 2.03: October 17, 2002. Some brief notes on Red Hat 8.0 differences/newness, notably the changes that accompany Xft2. More on the distinction of core X fonts and Xft (and then Xft2).
- 2.01: October 03, 2002. Update links for obtaining the MS web fonts (TrueType fonts of choice). There are two Sourceforge projects with these fonts available, including. http://sourceforge.net/projects/font−tool/.

  Add section to Mozilla configuration to include info on Xft and anti−aliasing.
- 2.00: September 12, 2002. Wow, MS has taken down their webfonts page :(Not good news. Installing TrueType is a key ingredient to better living through nice fonts.

  Major re−organization by moving the font server section to the appendix, since any recent distro is already installing these, or they are not any longer needed.

  Some clarifications on gdkxft and font anti−aliasing. Finally, add Aristotle Pagaltzis's perl script for converting `fonts.dir` to `fonts.alias`. Include interesting example of using anti−aliased TrueType fonts in **xterm**. Note that GNOME 2 is out. Fix the usual assorted broken links.

  Clarifications on `dpi` settings, and add tip about unbalanced dpi causing major trouble with fonts.
- 1.95: February 11, 2002. A few corrections. Removed the section on Fonts in KDE since this has to have changed, and I don't know anything about KDE (does anyone want to help here?). Added a brief section on gdkxft, which adds anti−aliasing support for GTK+ 1.2 applications.
- 1.9: November 5, 2001. A few new links and some minor catch ups only.
- 1.8: June 25, 2001: Included a new section on Anti−aliasing and Xft from Danny Tholen <`obiwan@mailmij.org`>. Many thanks on this not so well documented subject. Also, Sebastiano Vigna's neat little package for downloading and installing MS webfonts: http://freshmeat.net/webFonts4Linux. A few other odds and ends.
- 1.70: April 18, 2001: Added links for converting Mac TrueType Fonts (thanks to Karl A. Krueger), links to Unicode TrueType fonts (thanks to Tzafrir Cohen for suggestions and URLs), and added a section on anti−aliasing with X 4.0.2 (or greater). Also, included a reference to cabextract, a utility that is now available for extracting Win32 Fonts (among other things) from a Window's "cab" archive.
- 1.60: March 21, 2001: A few very minor changes. Most notable news is anti−aliasing support now in XFree 4.02 (referenced in the Notes section only). Chinese translation URL added.

1.3. New Versions

The pre−release versions of this document can be found at http://feenix.burgiss.net/ldp/fdu/index.html.

1.4. Copyright

Unless otherwise stated, Linux HOWTO documents are copyrighted by their respective authors. Linux HOWTO documents may be reproduced and distributed in whole or in part, in any medium physical or electronic, as long as this copyright notice is retained on all copies. Commercial redistribution is allowed and encouraged; however, the author would like to be notified of any such distributions.

All translations, derivative works, or aggregate works incorporating any Linux HOWTO documents must be covered under this copyright notice. That is, you may not produce a derivative work from a HOWTO and impose additional restrictions on its distribution. Exceptions to these rules may be granted under certain conditions; please contact the Linux HOWTO coordinator for more information.

In short, we wish to promote dissemination of this information through as many channels as possible. However, we do wish to retain copyright on the HOWTO documents, and would very much like to be notified of any plans to redistribute the HOWTOs, this one in particular! Web page authors are free to link to this HOWTO without restriction, though the author would appreciate an email informing him of this, just so he can boost his ego by knowing who else reads and links to this document.

Many of the terms mentioned in this document are trade names. Unless otherwise stated, all trademarks are property of their respective owners.

1.5. Credits

```
  Original author: Doug Holland.
  Email: meldroc@frii.com
  WWW: http://www.frii.com/~meldroc/



 Updated and maintained by Hal Burgiss.
  Email: hal@foobox.net
  WWW: http://feenix.burgiss.net/ldp/fdu/
```

Special thanks go to:

- The developers of the XFree86 Project, for all the hard work and time they have given. Also, Juliusz Chroboczek for his work with xfsft, and XFree86 4.x to help bring TrueType to the hungry masses. And Keith Packard for his anti−aliasing, and other work. This is not to slight the many, many other XFree86 volunteers.
- Font wizard Kristin Aanestad, whose legwork and insight on much of the xfs, TrueType, Netscape, and especially, the fonts.alias sections are much appreciated. More from Kristin at Some Linux for Beginners on a wide range of topics.
- Danny Tholen <obiwan@mailmij.org> is responsible for the nice Xft section, and examples.
- The folks at comp.os.linux.x who gave me a hand in figuring all of this out in the first place.
- The Linux community in general who made all of this possible in the first place. Especially those who have offered suggestions and comments that help to make this HOWTO a better resource. Keep those cards and letters coming ;−)

1.6. Translations

Chinese: http://www.linux.org.tw/CLDP/mini/FDU.html by Yu−Chia Chang.

2. X Server Configuration

There are a few simple configuration tweaks that will help X do its job better.

2.1. Setting The FontPath

The first place to look for curing font problems is the XF86Config file. /usr/X11/lib/X11/XF86Config or /etc/X11/XF86Config are the common locations. (This may be XF86Config−4 for XFree86 4.x.) If you haven't guessed already, the most important part of this file relating to fonts is the FontPath. Before we get into that, this would be a good time to check the other parts of your X configuration. Bad monitor settings can be even more of a headache than bad fonts, so make sure your refresh rate is as high as your monitor can handle (85 Hz is great, 75 Hz is OK, 60 Hz is painful.)

Use your favorite text editor and edit XF86Config. Near the top of the file in the "Files" section, you should see something vaguely like this:

```
     FontPath        "/usr/X11R6/lib/X11/fonts/misc/"
     FontPath        "/usr/X11R6/lib/X11/fonts/Type1/"
     FontPath        "/usr/X11R6/lib/X11/fonts/Speedo/"
     FontPath        "/usr/X11R6/lib/X11/fonts/75dpi/"
     FontPath        "/usr/X11R6/lib/X11/fonts/100dpi/"
```

This much should be the same, or at least similar, for both XFree86 3.x and 4.x. The FontPath tells X where to find the fonts it uses to render text on your display. Order is important −− when an X application asks X to render some text, the X server usually has some leeway to choose the font that is used. The X server then goes through the FontPath and grabs the first font it sees that matches the X client's criteria, and then displays it. (Note that Red Hat's xfs for versions 6.x and later has a different way of setting the FontPath. See the Section A.2 below for more on xfs.)

If the 100dpi fonts are not listed, they probably did not get installed for whatever reason, so you may want install them now. Default installations may put 75dpi fonts before the 100dpi fonts. If you have a high resolution display (1024x768 or higher), this means very tiny fonts. If this is the case, the first tweak you'll use is to switch the 75dpi and 100dpi FontPath lines:

```
     FontPath        "/usr/X11R6/lib/X11/fonts/misc/"
     FontPath        "/usr/X11R6/lib/X11/fonts/Type1/"
     FontPath        "/usr/X11R6/lib/X11/fonts/Speedo/"
     FontPath        "/usr/X11R6/lib/X11/fonts/100dpi/"
     FontPath        "/usr/X11R6/lib/X11/fonts/75dpi/"
```

Next, specify that you prefer to use unscaled bitmap fonts. If you've ever used Netscape or any other program that displays titles using big fonts, you might have noticed that those fonts were pixelized. This is ugly and needs to be corrected. So add :unscaled to the ends of the misc, 100dpi and 75dpi fonts. You can even use both unscaled and scaled fonts if you want, just put the unscaled FontPath lines first to tell X you prefer unscaled fonts if possible:

```
    FontPath        "/usr/X11R6/lib/X11/fonts/misc:unscaled"
    FontPath        "/usr/X11R6/lib/X11/fonts/100dpi:unscaled"
    FontPath        "/usr/X11R6/lib/X11/fonts/75dpi:unscaled"
    FontPath        "/usr/X11R6/lib/X11/fonts/Type1"
    FontPath        "/usr/X11R6/lib/X11/fonts/Speedo"
    FontPath        "/usr/X11R6/lib/X11/fonts/misc"
    FontPath        "/usr/X11R6/lib/X11/fonts/100dpi"
    FontPath        "/usr/X11R6/lib/X11/fonts/75dpi"
```

After making these changes, restart X (and your font server, if installed). Doesn't the desktop look better already?

2.2. X Server Command Line Options

The next thing you might need to do is adjust the command line options for the X server. It is important that the `dpi` resolution be accurate. Run this command:

```
   $ xdpyinfo | grep resolution resolution: 111x111 dots per inch
```

If this does not look to match what your monitor can do, or if it is unbalanced (e.g `98x95`), you will need to use the `−dpi` switch which specifies the display resolution in dots per inch. As a lot of systems use high resolution displays these days, chances are they'll be working at 100 dpi, or better. While an unbalanced setting reportedly can really play havoc with some fonts.

If you start X from the console command prompt, type:

```
$ startx −− −dpi 100 −depth 16 # v4.x syntax
```

Or these options can be stored in `~/.xserverrc`. See the **startx** and **xinit** man pages for more on this.

If you use xdm (or friends such as gdm) for graphical logins, you'll want to edit your `/usr/X11/lib/X11/xdm/Xservers` file (or possibly `/etc/X11/xdm/Xservers`) which will have the command line for the Xserver in it. Mine has the line:

```
:0 local /usr/X11R6/bin/X −dpi 100 −gamma 1.6
```

You want to specify a value that is compatible with your monitor's output.

More information is in the **X**, **Xserver**, **xdm**, **xinit,** and **startx** man pages.

3. TrueType Fonts

Historically, the Unix world relied on Type 1 fonts for high quality scalable fonts. Linux supports Type 1 quite well, both for printing and for screen output. But, Type 1 never was widely adopted by web designers, and on other platforms. TrueType, due to its association with Windows, is the preferred web font. And XFree86 seems to render TrueType a little better.

You won't find many decent TrueType fonts included with any distribution, however. The reason is that there are not many quality TrueType fonts available under a suitable license at this time. In fact, many of the "free" ones, are rather poor. Many distributions are including some TrueType fonts, and also including tools for automating the process of adding TrueType fonts from external sources. See if that is an option for you. This will be easiest route. You will definitely want to add some quality TrueType fonts, one way or another.

Because the boys in Redmond are very concerned with the appearance of their software (as opposed to the internal workings ;), they built TrueType font support into Windows. And of course no big surprise, but they got the idea from Apple. In fact, TrueType is a registered trademark of Apple Computer, Inc. Windows 9x, NT, 2K and nearly every other incarnation of Windows comes with Arial, Times New Roman, and Courier New, which are roughly equivalent to Helvetica, Times and Courier. TrueType fonts are scalable, so they look good in large sizes, and they are well hinted, so they are readable at small sizes. Many windows applications come with dozens more TrueType fonts. And let's face it, those MS and Apple fonts are, generally speaking, just plain better than the freely available ones with a suitable license. Don't microwave your Windows CD yet, you'll want to get those quality fonts first!

3.1. Making TrueType Fonts Available

Let's start with the fonts first. Any TrueType font included with the various MS Windows incarnations should work. Don't forget word processors and other apps that may include their own fonts. MacOS fonts will only work if converted to a usable format. (See the links section for converter packages.) There are also some 'free' TrueType fonts available for download if you have already nuked that CD (see links section).

Many distributions are now bundling tools for automating the process of including quality TrueType fonts. SuSE, Debian, and Mandrake do (Red Hat 7.x does not at this time). See what packages you might have for this as this will be the most painless way to go. Essentially, these tools help migrate fonts from a Windows installation, or download those available from Microsoft [note 08/15/02 MS has recently removed these fonts!] , and then handle the installation and configuration all in one neat utility. If you do have such a utility, the below information may not be necessary!

In order to use TrueType, the fonts will have to be always accessible to X. This means they will have to be on a filesystem that is *always* mounted. This can conceivably be a Windows partition on a dual boot system. Alternately, the fonts can be copied to Linux. First **su** to root:

```
 # su -
 # mkdir -p /usr/local/share/fonts/ttfonts
```

Now, change to the new font directory:

```
  # cd  /usr/local/share/fonts/ttfonts
```

Then, add the fonts to this directory, either by copying them from your Windows system:

```
 # cp /mnt/<path_to_fonts>/*ttf  .
```

or by downloading those available directly from Microsoft: http://www.microsoft.com/typography/fontpack/default.htm. Note 08/17/02: Microsoft has recently *discontinued this page*. At this time, the following pages (may!) still have these fonts available (or google search for them):

http://nuoriso.hel.fi/__files/ms_truetype_core_fonts_for_the_web/Win95−98−NT/index.html
http://web.archive.org/web/20000420210719/http://www.microsoft.com/opentype/fontpack/default.htm

These can be unarchived under Linux with **cabextract**, which can be found http://www.kyz.uklinux.net/cabextract.php3. This would now seem to be the best way to go at this time.

A slick solution to this from Sebastiano Vigna is his http://freshmeat.net/webFonts4Linux, which automates the downloading, extracting and installation of the Microsoft fonts all in one neat package. A utility designed primarily for Red Hat can be found: http://sourceforge.net/projects/font−tool/, which includes all the core MS web fonts, plus relevant system configuration. There is a tarball, as well as RPMs (both require cabextract).

You can also get an RPM of WebFonts that contains some of the MS Web fonts from ftp://ftp.rpmfind.net/linux/contrib/noarch/noarch/webfonts−1−3.noarch.rpm. This has enough basic fonts to keep Mozilla and other web browsers happy. Something similar for Debian is http://packages.debian.org/unstable/graphics/msttcorefonts.html. This does not include the actual fonts, but facilitates the installation.

If doing it yourself, you will also have to include the new TrueType directory(s) in the X server's fontpath. So with your text editor of choice add the line(s) as appropriate:

```
     FontPath        "/usr/local/share/fonts/ttfonts"
     FontPath        "/usr/X11R6/lib/X11/fonts/misc:unscaled"
     FontPath        "/usr/X11R6/lib/X11/fonts/100dpi:unscaled"
     FontPath        "/usr/X11R6/lib/X11/fonts/75dpi:unscaled"
     FontPath        "/usr/X11R6/lib/X11/fonts/Type1"
     FontPath        "/usr/X11R6/lib/X11/fonts/Speedo"
     FontPath        "/usr/X11R6/lib/X11/fonts/misc"
     FontPath        "/usr/X11R6/lib/X11/fonts/100dpi"
     FontPath        "/usr/X11R6/lib/X11/fonts/75dpi"
```

This configuration is for "core X font" support. For additional configuration relating to the new Xft rendering engine, see the sections below on XFT and anti−aliasing.

3.2. TrueType Hinting

"Hinting" is a TrueType specific feature, that is generally considered to be a useful technique that improves the appearance of TrueType fonts. Unfortunately, there are some licensing and patent issues involved with this, and it is disabled by default in the freetype sources! And also quite likely that if you are using vendor supplied binaries, it is disabled there as well.

To enable this feature, the sources need to be rebuilt. Look for `include/freetype/config/ftoption.h` in the freetype source tree, and then search for:

```
#undef TT_CONFIG_OPTION_BYTECODE_INTERPRETER
```

And very simply, make this small change:

```
#define TT_CONFIG_OPTION_BYTECODE_INTERPRETER
```

Red Hat users can rebuild the src.rpm by toggling one setting at the top, and accomplish the same thing:

```
%define without_bytecode_interpreter    1
```

And change to:

```
%define without_bytecode_interpreter    0
```

Other vendors may have a similar, easy−to−use mechanism.

Then rebuild and install the finished binaries. Be sure to restart X as well since the freetype code is already loaded into memory by X.

Note that "hinting", "anti−aliasing", and "sub−pixel rendering" are separate concepts (see the section on Xft below for more). Again, this is not a cure−all for "ugly" fonts, but one more piece in the puzzle of font "beautification".

3.3. Configuration

This section pertains to installing and configuring TrueType fonts for any "freetype" based font renderer. This would include the `freetype` font module from XFree86−4.x, Red Hat's `xfs`, and the `xfsft` font server. Again, if you have used a distro supplied tool for migrating TrueType fonts, it is likely this has been done for you already. The steps described below would only be necessary for manual font installation.

There is still more work to be done before we can actually use any TrueType fonts. First, all font files must have lower case names. Secondly, they shouldn't have embedded spaces. And then, we will need to create a couple of configuration files to make things go.

Su to root, and change to the directory where the TrueType fonts are.

```
 # su −
 # cd /usr/local/share/fonts/ttfonts
```

If there are any upper case font names, you can use the following script to convert all names to lower case:

```
#!/bin/sh
#
## -------- convert upper to lower case ---------

ls * | while read f
 do
   if [ -f $f ]; then
     if [ "$f" != "`echo \"$f\" | tr A-Z a-z`" ]; then
      #Note that 'This' will overwrite 'this'!
      mv -iv "$f" "`echo \"$f\" | tr A-Z a-z`"
     fi
   fi
 done

## eof
```

Note the punctuation −− the backquotes are important! Remove any spaces from font names too. Once the TrueType fonts are properly installed, you must create both `fonts.dir` and `fonts.scale` files. The following commands do this:

```
# ttmkfdir -o fonts.scale
# mkfontdir
```

If you don't have **ttmkfdir** installed, check your distribution's repository, or it can be downloaded from: http://www.joerg−pommnitz.de/TrueType/ttmkfdir.tar.gz. This is necessary!

As of Red Hat 7.1, the above commands are run from the xfs init script. So restarting xfs (**/etc/rc.d/init.d/xfs restart**) will accomplish the same thing for Red Hat users. Other distros may have similar shortcuts.

You should now have `fonts.dir` and `fonts.scale` files in your TrueType font directory. **ttmkfdir** is in the `Freetype` RPM for Red Hat users, and must be run before **mkfontdir**. With Debian based distros, there is a similar utility called **mkttfdir**, and is in the `fttools` Deb package. Though this apparently does not generate as many encodings as **ttmkfdir**. These commands may not always report errors, so verify that they were created and are not empty files:

```
$ ls -l fonts.*
 -rw-r--r--  1 root   root    11657 Aug 17 10:31 fonts.dir
 -rw-r--r--  1 root   root    11657 Aug 17 10:31 fonts.scale
```

If you encounter any problems, try **ttmkfdir** with the **− m** switch. This will discard bad characters from the font file. Specify a number such as 50 or l00 (**ttmkfdir −m 50**). The files themselves are text files. Have a look:

```
$ less  fonts.dir
```

```
  114
  webdings.ttf  -microsoft-Webdings-medium-r-normal--0-0-0-0-p-0-microsoft-symb
  verdanaz.ttf  -microsoft-Verdana-bold-i-normal--0-0-0-0-p-0-ascii-0
  verdanaz.ttf  -microsoft-Verdana-bold-i-normal--0-0-0-0-p-0-fcd8859-15
  verdanaz.ttf  -microsoft-Verdana-bold-i-normal--0-0-0-0-p-0-iso8859-15
  verdanaz.ttf  -microsoft-Verdana-bold-i-normal--0-0-0-0-p-0-iso8859-9
  verdanaz.ttf  -microsoft-Verdana-bold-i-normal--0-0-0-0-p-0-iso8859-1
 [...]
```

If **ttmkfdir** is persistently giving problems by not generating a proper output file, there may be one or more "bad" fonts (ie fonts it can't handle). In that case, just start with a few common ones, like Arial and Verdana. If this works, then add a few at a time.

Now be sure the new fonts are included in the `FontPath`. And either restart X (Ctrl−Alt−BS), or the font server (if using one). You could also try refreshing the `FontPath`:

```
 # xset fp rehash
```

Red Hat 6.x/7.x users can update the FontPath and xfs:

```
 # chkfontpath --add /usr/local/share/fonts/ttfonts
 # /etc/rc.d/init.d/xfs restart
```

You should now be in business. You can check which fonts are available to X:

```
$ xlsfonts | less
```

or check them out further with **xfontsel**, or **gfontsel**. If they are visible to **xlsfonts**, then they are available to X and vice versa. If they are not there, try restarting X with Ctrl−Alt−BS.

This gets you as far as X knows about your new fonts. Individual applications will need to be configured to use them. GNOME and KDE will require additional steps as well (see the respective docs). You will also want to configure Xft (see below), if using XFree86 v4.x or later.

3.4. Font Servers

Historically, font servers were used to serve fonts over a network. Font resources could then reside on one host, and clients could access them as needed. But, the developers have enhanced these to include features such as the ability to render TrueType fonts. XFree86 4.x *does have built in support* for TrueType (see Section 4), making a font server not a necessity, though some distros default to using a font server for other reasons now.

XFree86 3.x does not come with built in TrueType support, so you'll have to add it yourself if you are using a 3.x version. This will mean installing a supplemental font server that does support TrueType. And, of course, installing the fonts themselves (see above). See the Appendix for font server installation, and related tips.

Any recent distro will have one or more font servers included, and the important configuration should be done by the installation program.

3.5. The fonts.alias File

`fonts.alias` is yet another font configuration file that can be used to tweak how fonts are handled. Like `fonts.scale` and `fonts.dir`, `fonts.alias` must be in the same directory as the fonts you are aliasing. It is not mandatory however, but does solve certain potential problems. Here is an example from the first line of`/usr/X11R6/lib/X11/fonts/misc/fonts.alias` on a Red Hat system:

```
 fixed    −misc−fixed−medium−r−semicondensed−−13−120−75−75−c−60−iso8859−1
```

`fixed` is the 'alias' here. Any time this is requested, we actually get the font definition from the second column. Font too small? Just change the definition. (Warning: this is a critical file, at least on Red Hat.) The same principle applies to all fonts, including TrueType. In fact, if you don't have TrueType, you could conceivably use this trick to have a comparable Type 1, or other, font aliased as a TrueType.

`fonts.alias` is important for some applications that don't handle the data provided by `fonts.scale` well. Most notably here is Netscape. Without a `fonts.alias` you will find that Netscape will only show point sizes of 0 and 12 available. `fonts.alias` fixes this. You might also find that if you a specify another size with the `scalable font` option under `Preferences`, Netscape will not remember this setting. Annoying! This is also fixed. So we really need this file. Sample excerpt from a `fonts.scale`:

```
 arial.ttf   −monotype−Arial−medium−r−normal−−0−0−0−0−p−0−ascii−0
 arial.ttf   −monotype−Arial−medium−r−normal−−0−0−0−0−p−0−fcd8859−15
 arial.ttf   −monotype−Arial−medium−r−normal−−0−0−0−0−p−0−iso8859−15
 arial.ttf   −monotype−Arial−medium−r−normal−−0−0−0−0−p−0−iso8859−1
```

These are scalable so we don't get any predefined point sizes. We will need to create our `fonts.alias` something like this excerpt for Arial:

```
 −monotype−Arial−medium−r−normal−−6−60−0−0−p−0−iso8859−1 \
             −monotype−Arial−medium−r−normal−−9−90−75−75−p−0−iso8859−1

 −monotype−Arial−medium−r−normal−−7−70−0−0−p−0−iso8859−1 \
             −monotype−Arial−medium−r−normal−−9−90−75−75−p−0−iso8859−1

 −monotype−Arial−medium−r−normal−−8−80−0−0−p−0−iso8859−1 \
             −monotype−Arial−medium−r−normal−−10−100−75−75−p−0−iso8859−1

 −monotype−Arial−medium−r−normal−−9−90−0−0−p−0−iso8859−1 \
             −monotype−Arial−medium−r−normal−−11−110−75−75−p−0−iso8859−1

 −monotype−Arial−medium−r−normal−−10−100−0−0−p−0−iso8859−1 \
             −monotype−Arial−medium−r−normal−−12−120−75−75−p−0−iso8859−1
```

```
-monotype-Arial-medium-r-normal--11-110-0-0-p-0-iso8859-1 \
            -monotype-Arial-medium-r-normal--12-120-75-75-p-0-iso8859-1

-monotype-Arial-medium-r-normal--12-120-0-0-p-0-iso8859-1 \
            -monotype-Arial-medium-r-normal--12-120-75-75-p-0-iso8859-1

-monotype-Arial-medium-r-normal--13-130-0-0-p-0-iso8859-1 \
            -monotype-Arial-medium-r-normal--13-130-75-75-p-0-iso8859-1

-monotype-Arial-medium-r-normal--14-140-0-0-p-0-iso8859-1 \
            -monotype-Arial-medium-r-normal--14-140-75-75-p-0-iso8859-1

-monotype-Arial-medium-r-normal--15-150-0-0-p-0-iso8859-1 \
            -monotype-Arial-medium-r-normal--15-150-75-75-p-0-iso8859-1

-monotype-Arial-medium-r-normal--18-180-0-0-p-0-iso8859-1 \
           -monotype-Arial-medium-r-normal--18-180-75-75-p-0-iso8859-1

-monotype-Arial-medium-r-normal--24-240-0-0-p-0-iso8859-1 \
           -monotype-Arial-medium-r-normal--24-240-75-75-p-0-iso8859-1
```

(*Please note that I have split each line for readability*. There should be two columns all on one line, without the "\", and separated by at least one space.) This will keep Netscape happy. Also, if font names should have embedded spaces, then you should enclose the filename in quotes. You might also note the `pointsize` discrepancy between the first and second columns of the first few rows. The first column of the first entry has a '6', whereas this is aliased to a '9' in the second column, and thus '9' point. This is by design and is an excellent way to overcome the Netscape 'damn tiny fonts' syndrome. Adjust to suit your tastes, resolution, and eyesight.

This file can be created manually with a text editor, or conceivably with some fancy sed or awk scripting. There is an excellent discussion of this file, and other font related topics at Kristin Aanestad's site at http://home.c2i.net/dark/linux.html. There is also a link to a python script which can reportedly automatically generate a `fonts.alias` file at this same site. A perl version of this script is re−printed in the Appendix. Thanks to Kristin whose work and insight was the inspiration for this section!

Another potential use of `fonts.alias` would be to map one font to something quite different. Say you don't have TrueType fonts, and didn't want to install Microsoft's. You could alias nice, scalable Type 1 fonts to a TrueType. That way when the system (or some web page) wants a TrueType, you'd get something of comparable quality instead of bitmap that doesn't scale well.

Note that with XFree86 4.0.2 and greater, there are new font handling mechanisms available via the Xft extensions. Font aliasing is done in Xft's own configuration file: `XftConfig`. This is the preferred method where anti−aliasing, and the other new rendering features are desired. See the Anti−aliasing Section for more on this and de−mystification. This is only true where the application (i.e. the toolkit, e.g QT) itself supports the new extensions! At this time, not all do (yet).

4. XFree86 4.x

XFree86 4.0 introduced native support for TrueType fonts, along with other new features. The enhanced font support is based on xfsft from Juliusz Chroboczek, which in turn is based on the FreeType font library originally from Mark Leisher, so the configuration is similar to xfsft and Red Hat's patched xfs which use the same rendering engine. As of 4.0.2, XFree86 begins to support anti−aliasing which is a technique for smoothing font outlines (see section below).

The `FontPath` is still in `XF86Config`, as always. For Red Hat 6/7 using a stock XFree86 4.x (i.e. NOT the Red Hat 7.x supplied version), this will mean moving the Red Hat xfs `FontPath` from `/etc/X11/fs/config` back to `XF86Config`. A separate font server is no longer needed just for TrueType support. You may disable it, unless it is needed to serve fonts to other clients in a network environment. Or you may want to still use it since there may be a very small performance gain in some situations (at the cost of more memory utilized). See the section below for Red Hat 7.x specific configuration issues.

```
 Section "Files"
  FontPath "/usr/X11R6/lib/X11/fonts/misc:unscaled"
  FontPath "/usr/X11R6/lib/X11/fonts/100dpi:unscaled"
  FontPath "/usr/X11R6/lib/X11/fonts/75dpi:unscaled"
  FontPath "/usr/X11R6/lib/X11/fonts/misc"
  FontPath "/usr/X11R6/lib/X11/fonts/Type1"
  FontPath "/usr/X11R6/lib/X11/fonts/Speedo"
  FontPath "/usr/share/fonts/default/Type1"
  FontPath "/usr/local/share/fonts/ttfonts"
  FontPath "/usr/X11R6/lib/X11/fonts/100dpi"
  FontPath "/usr/X11R6/lib/X11/fonts/75dpi"
 EndSection
```

In order to use TrueType, you must also specify which font `module` the X server should be using in the "Module" section:

```
 Section "Module"
  Load  "freetype"
  Load  "speedo"
  Load  "type1"
  <load other modules....>
 EndSection
```

Note that there can be only one 'Module' section, so include any other modules here as well. Again, this is the kind of configuration that your distro should be doing during installation, and may not be necessary, unless you are upgrading, or want to make manual changes yourself.

You also must have a `fonts.scale` and `fonts.dir` file for each TrueType font directory, just like for xfsft and Red Hat's xfs (see Appendix). **ttmkfdir**, http://www.joerg−pommnitz.de/TrueType/ttmkfdir.tar.gz, will come in handy for `fonts.scale` if it is not already installed. See the Configuration section above for more details and examples.

`xtt` is another available TrueType module that is best known for supporting ideographic (Oriental) type fonts. You can use either, but only one at a time.

Most X server command line options are still the same as previous versions of X:

```
$ startx −− −dpi 100
```

4.1. Xft and Core Fonts

As mentioned, the new font handling improvements are the result of the development work behind Xft. Xft is an on−going effort to improve and modernize X's font handling. The end result will be radically different from the traditional X font handling (known as "core X fonts").

What we have now is essentially two different systems: one that has been around since the dawn of time (or even before :), and one that is radically new and *still evolving*! I emphasize this, since the "still evolving" part is likely to cause some confusion until the dust settles.

The initial release of Xft v1.0 with XFree86 4.0.2 is controlled by its configuration file, `XftConfig`. This changes with later versions, where the font configuration is handled by the **fontconfig** library, with its main configuration typically located `/etc/fonts/fonts.conf`. Of course, additional development has brought other improvements as well. For more on the Xft and Fontconfig, see the Fontconfig home page, http://fontconfig.org/.

What we, as end users, will notice and appreciate most, is the anti−aliasing support and other rendering enhancements we get from Xft. Though this is not the only benefit.

4.2. Anti−aliasing

Anti−aliasing is a technique for producing even smoother, crisper looking fonts by subtly softening the sharp, jagged edges. This has long been a feature of Apple and Microsoft font rendering, and is now making it's way into X via the X Rendering Extension specification. The new extensions provide other benefits as well. Distributions that support anti−aliasing with their stable/official versions are now being released.

That is the good news. The bad news is that not all drivers support anti−aliasing yet. This is a moving target, so you will have to dig around to find whether your chipset is supported or not. The recently released 4.2 should have near universal support. More not−so−good news is that not all applications are actually taking advantage of this yet. We will have to wait for the various toolkits (TK, Xaw, etc) to catch up. KDE is an exception, and has had strong AA support via the QT toolkit. The recently released GNOME−2, has AA support now too.

Anti−aliasing is not a cure−all, but is another piece of the font puzzle. Some point sizes may look better if not anti−aliased, and some systems just may not handle the various rendering enhancements as well as others.

If you are reading this long after the publication date (October 2002), hopefully most of these shortcomings will have been overcome. All hardware will eventually be supported, mainstream distros will have shipped releases that include the new extensions, and they will be enabled by default. Many apps will look better since they will be "AA" aware, and we won't have to jump through any configuration hoops just to make it work. In the meantime, read on ...

4.2.1. Requirements

Minimum requirements for Anti−Aliasing:

- XFree86 4.0.2 or later.
- You graphic card's driver has to support anti−aliasing. If 4.0.2 (or greater) is already installed, you can get this information direct from the driver with **xdpyinfo**. Run this and look for "Number of Extensions:". If this lists "RENDER" among them, then you should be good to go. If not, well, it isn't going to work, and you will have to wait for an updated driver.
- The Freetype2 library available from http://www.freetype.org, and also now bundled with XFree86. XFree needs to be linked against this, so install and build first if building from scratch. Your distro should have Freetype packages as well. Just make sure it is `freetype−2`.
- TrueType fonts are best for display purposes. Type 1 is also good, but may not render quite as well. See above sections.
- For KDE users, KDE supports anti−aliasing as of 2.x. This will require QT−2.3.0 or later, and built with Xft support. A nice font HOWTO from Troll Tech for KDE and QT can be found: http://trolls.troll.no/lars/fonts/qt−fonts−HOWTO.html.

  GNOME 1.x does not support anti−aliasing. GNOME 2.0 has recently been released and does have native support for anti−aliasing.
- Applications that "know" about anti−aliasing. Not necessarily at the individual application level, but the libraries and toolkits (GTK, TK, etc.) that the application are built against, must be able to use the new features. At this time, not all do. KDE/QT is first out of the box and has good support. Also, **xterm** (yes, **xterm!**) supports the new extensions. An interesting example scavenged off usenet:

```
    From torvalds@penguin.transmeta.com Sun Apr 28 02:09:28 2002
    From: torvalds@penguin.transmeta.com (Linus Torvalds)
    Newsgroups: comp.os.linux.x
    Subject: Re: X font recommendation
    Organization: Transmeta Corporation
    Lines: 24
    Message−ID: <aa4ijq$259$1@penguin.transmeta.com>
    Cache−Post−Path: palladium.transmeta.com!unknown@penguin.transmeta.com
    Date: Tue, 23 Apr 2002 21:10:18 +0000 (UTC)
    X−Trace: news.sjc.globix.net 1019596253 63.209.4.196 (Tue, 23 Apr 2002 14:10:53 PDT)
    NNTP−Posting−Date: Tue, 23 Apr 2002 14:10:53 PDT

    In article <aa492r$7hc5u$2@ID−99293.news.dfncis.de>,
    William Park  <opengeometry@NOSPAM.yahoo.ca> wrote:
    >Peter Karlsson <root@localhost.localdomain> wrote:
    >> What's wrong with "fixed"? :−)
    >>
    >> xterm −sb −sl 10000 −fg yellow −bg black −geometry 120x40 −fn fixed −bc
    >
    >This will load the default 6x13 fonts, aka.
    >    −misc−fixed−medium−r−semicondensed−−13−120−75−75−c−60−iso8859−1
    >A bit too small, though it has correct proportions.  If there is bigger
    >size (8x16, 10x20, 12x24) of that fonts, then that would be ideal.

    If you like anti−aliased fonts, the command line I prefer happens to be

         xterm −geometry 100x40 −fa andale:size=11:charwidth=6
    I think that font is from the MS free fontpack.

    In order for it to work for you, you need to have the anti−aliasing
    truetype fonts working well, and probably a good monitor (and some
    people _hate_ that rounded look from anti−aliasing, and find it fuzzy
    even then).

              Linus
```

You might have to experiment with the `size` and `charwidth` values to get reasonable results.

- The new rendering extensions configuration file, `XftConfig`, must be configured for the fonts you want to use. Note that as of Xft2, this changes to `/etc/fonts/fonts.conf`.
- The new extensions supplant much of what we have been doing with font servers like xfs. `fonts.alias` and similar configuration files, for instance, are not used for fonts that are being controlled by the new extensions. Essentially, this gives us two separate font handling engines: the traditional, "core" font engine, and the new, Xft engine. The core engine is still used in some situations.

4.2.2. Installation

Keith Packard has a very brief summary of the steps required for building, installing and configuring from source at http://www.xfree86.org/~keithp/render/aafont.txt. No need to reprint it here.

Newer distro releases are likely to have the foundation support for anti−aliasing available now. Red Hat, for instance, has it available as of Red Hat 7.1.

To verify the necessary components, first make sure the "freetype" module (and any others) are loaded. Check the X server output:

```
 (II) LoadModule: "freetype"
 (II) Loading /usr/X11R6/lib/modules/fonts/libfreetype.a
 (II) Module freetype: vendor="The XFree86 Project"
        compiled for 4.0.3, module version = 1.1.9
        Module class: XFree86 Font Renderer
        ABI class: XFree86 Font Renderer, version 0.2
 (II) Loading font FreeType
```

Then verify if the "RENDER" extension is available, either check with **xdpyinfo**, or check the X server log, typically `/var/log/XFree86.0.log`:

```
 (II) Initializing built-in extension MIT−SHM
 (II) Initializing built-in extension XInputExtension
 (II) Initializing built-in extension XTEST
 (II) Initializing built-in extension XKEYBOARD
 (II) Initializing built-in extension LBX
 (II) Initializing built-in extension XC−APPGROUP
 (II) Initializing built-in extension SECURITY
 (II) Initializing built-in extension XINERAMA
```

```
(II) Initializing built−in extension XFree86−Bigfont
(II) Initializing built−in extension RENDER
```

If "RENDER" is there, anti−aliasing and the other advanced rendering extensions should be available. If not, the system should gracefully fall back to the core X fonts engine.

4.2.3. Xft Configuration (XftConfig)

By Danny Tholen <obiwan@mailmij.org>

Xft is an interface to the freetype rasterizer written by Keith Packard, member of the XFree86 Project, Inc. It allows applications to use fonts from the new X render extension using a unified font naming scheme. In /etc/X11/XftConfig (or /usr/X11R6/lib/X11/XftConfig) you will find a configuration file which can be adapted to suit your personal taste. In this section I will explain the syntax and demonstrate some things you can do with this file.

The following information is based on 4.0.3. 4.1 is just released, and there may be a few new wrinkles not touched on here. Future versions are sure to bring even more radical changes.

Note that Xft2 makes radical changes to Xft configuration. XftConfig is obsoleted and replaced by the new fontconfig libraries. Red Hat 8.x users should read the Red Hat 8.0 Differences section first. Many of the principles described still apply, but the configuration is radically different. FYI.

4.2.3.1. XftConfig Structure

The basic structure revolves around a 'pattern'. A pattern is a set of name/value−list pairs, each value−list contains one or more typed values. A certain application requests a font, for example:

```
family: "Arial"
size: 12
encoding: "iso8859−1"
```

A size 12 arial font in latin−1 encoding. The Xft extension will now try to patch this pattern to all of the fonts available in the system. And selecting the one with the best score. Before the matching is done Xft looks in XftConfig. The requested pattern can here be extended before use. An example is:

```
match any family == "Arial" edit antialias = true;
```

This will enable anti−aliasing for all fonts of the family Arial.

Also, the X server is queried to list all of its fonts; the XLFD contains just enough information to match fonts roughly.

Here's a list of attributes used in matching fonts (in priority order, this may not be up to date anymore!):

```
foundry         font foundry (string, like "monotype")
encoding        font encoding (string, like "iso8859−1")
spacing         font spacing (integers or proportional (0), mono (100),
               charcell (110))
bold           is the font bold? (boolean)
italic         is the font italic? (boolean)
antialias      is the font anti−aliased? (boolean)
family         font family (string)
size           font size (double)
style          font style (string, like "Bold Italic")
slant          font slant (roman, italic, oblique)
weight         font weight ( integers or light, medium (100), demi−bold,
               bold, black)
rasterizer     not yet used (probably "TrueType", "Type1", ...)
outline        are outlines available? (boolean)
```

4.2.3.2. XftConfig Syntax

- **dir**

  Adds a directory to the list of places Xft will look for fonts. There is no particular order implied by the list; Xft treats all fonts about the same.

- **include and includeif**

  Cause Xft to load more configuration parameters from the indicated file. "includeif" doesn't elicit a complaint if the file doesn't exist. If the file name begins with a "~" character, it refers to a path relative to the home directory of the user. This is useful for user−specific configurations.

- **match edit**

  If a pattern from an application matches the pattern after "match", it is edited with the instructions in `edit`. The pattern match is done as follows:

  ```
    match qual FIELD−NAME COMPARE CONSTANT
  ```

  where qual is either `any` (matches one specific font) or `all` (matches all fonts). An example:

  ```
   match all foundry==¨monotype¨
  ```

  which will match (and edit) all fonts belonging to the foundry "monotype".

  ```
   match any family==¨arial¨
  ```

  will match (and edit) one specific font with the family name "arial".

  `FIELD−NAME` is any one of the properties found in the above section Structure, or additionally:

  ```
   pixelsize        font size in pixels (integer)
   charspace        character space (integer)
   minspace         minimal spacing (integer)
  ```

```
rgba                 color hinting (string ¨rgb¨ or ¨bgr¨ and
                     vertical hinting ¨vrgb¨ ¨vbgr¨), aka sub−pixel hinting
xlfd                 x server font (string, type xlsfonts to
                     see a list of your xlfd strings)
file                 the font file (string)
core                 use X core fonts? (boolean)
render               use render fonts? (boolean)
index                I have no idea what this does:)
scalable             is the font scalable (boolean)
scale                scale the font (integer)
charwidth            character width (integer)
charheight           character height (integer)
matrix               no idea (not really at least)
```

`COMPARE` can be `<`, `>`, or `==`.

`CONSTANT` is the value of the field−name in the appropriate type (see above section Structure).

You can use multiple matches before you use the "edit" statement:

```
edit FIELD−NAME ASSIGN EXPR SEMI
```

Where `ASSIGN` can be one of `=`, `+=` or `=+`. With `=`, the matching value in the pattern will be replaced by the given expression. `+=` or `=+` will prepend/append a new value to the list of values for the indicated field.

`EXPR` sets the `FIELD−NAME` to a value.

`SEMI` is a semicolon (`;`). You can use multiple instructions, separated by a semicolon. See below for examples if this is confusing.

4.2.3.3. XftConfig Examples

And now I´ll try to list a few useful configurations and explain them. Note that it is configured for my system, and I may use different fonts than you, so try to adapt the examples to your own needs.

1. How do I make fonts available to Xft?

   List your Type 1 and TrueType font directories with "dir". On my system (Mandrake 7.2) this becomes:

   ```
   dir "/usr/X11R6/lib/X11/fonts/Type1"
   dir "/usr/X11R6/lib/X11/fonts/drakfont"
   ```

2. How do I use a user specific XftConfig file?

   Put an `.xftconfig` file in your user directory and add:

```
includeif   "~/.xftconfig"
```

to your standard `XftConfig`. This will enable a user specific configuration file, but it will not complain if there is no such file.

3. How do I make aliases for my fonts?

   I noted that my KDE console asks for "mono" fonts when it is looking for a fixed font. "console" is used when I select "linux" in the font menu of the KDE konsole. Therefore, I used two aliases for fonts which are also named "fixed":

   ```
   match any family == "fixed"     edit family =+ "mono";
   match any family == "console"   edit family =+ "mono";
   ```

4. Anti−aliasing my fonts are blurry and makes me dizzy!

   Although there is a big fuzz around AA in X, good fonts actually look better if they are not anti−aliased. The anti−aliasing blurs the fonts by adding gray pixels to the edges, and this may strain your eyes if you looking at them for a long time. (Your eyes will try to get the fonts sharper, which of course is not working because they are blurred;) However, for very small fonts, anti−aliasing may increase the readability of the fonts, because with sharp edges, there are too little pixels available for your mind to figure out what it means. And for bigger fonts, the edges become very jagged when not anti−aliased, so here you also might want to have aliased fonts. Of course you can also turn off the anti−aliasing for specific fonts. In other operating systems, most truetype fonts are not anti−aliased between 8 and 12 pixels, while only large Type1 fonts are anti−aliased.

   Use the following in your `XftConfig` to anti−alias only fonts of specific sizes:

   ```
   match
         any size > 8
         any size < 15
   edit
         antialias = false;
   ```

5. My fixed fonts do not appear or look _very_ wrong in the KDE konsole or similar programs!

   I noted that somehow a lot of fixed font do not tell Xft that they are fixed, and thus, mono spaced. Therefore only a part of the font is displayed. We can manually set the spacing for these fonts (this assumes you have fixed aliased with mono as in question 3 above):

   ```
   match
         any family == "mono"
   edit
         spacing = mono;
   ```

6. My Symbol, Webdings, etc. fonts do not show up!

   For some reason some (symbol) fonts are not correctly recognized, and Xft will show your default font, or a font which has the closest match (which is generally not what you mean at all). For Adobe Symbol and MS−webdings I did the following to get them working:

```
match
      any family == "webdings"
edit
      antialias = false;
      encoding += "glyphs−fontspecific";


match
      any family == "symbol"
edit
      antialias = false;
      encoding += "glyphs−fontspecific";
```

A useful way of figuring out these things is to activate debugging with:

```
export XFT_DEBUG=1024
```

This will generate a lot of output, especially if you have many fonts, because it lists the properties and scores of every font available. You can also use other values. For a nice summary of what happens (requested font, XftConfig substitutions, X server additions and the finally matched font), you can use XFT_DEBUG=2.

7. Why do my KDE programs start now soooo slooow?

   The Xft mechanism in XFree prior to 4.1 had to parse the `XftConfig` file each time a program was started. And the info of all these fonts had to be re−read. As of X 4.1.0, a cache is used and starting applications using Xft is much faster. Especially if you have many fonts this can be very useful. So, upgrading XFree86, and related packages, is a good idea.
8. I have a LCD screen on my laptop, can I use sub−pixel hinting instead of normal anti−aliasing?

   Yes you can. Sub−pixel hinting uses colors instead of gray pixels to do the AA. I do not have a LCD screen so I do not have any idea of how it looks but you can play with the "rgba" setting. Try:

   ```
   match edit rgba=bgr;
   ```

   or use rgb if you have a different type of monitor. For vertical AA you can try vbgr and vbgr.
9. My fonts still look bad!

   Good quality fonts are needed to start with. If you do not have some good TrueType fonts, it is worth it to go and look for them on the Internet. Other reasons why your fonts still look bad can be because of your build of freetype2. Snapshots versions before 2.0.2 were compiled with an option that had some patent issues. Therefore, the standard 2.0.2 and 2.0.3 compiles without this option. To fix this, download the freetype2 source rpm and change in include/freetype/config/ftoption.h line 314:

   ```
    #undef TT_CONFIG_OPTION_BYTECODE_INTERPRETER
   ```

   to:

```
        #define TT_CONFIG_OPTION_BYTECODE_INTERPRETER
```

> and rebuild with this modified source. See the freetype2 `README` file for details. Adobe Courier looks terrible on my system, so I made an alias so that Lucida console is displayed instead. If anyone can get it to display nicely I would appreciate knowing about it.

This is my `XftConfig`:

```
#
# XftConfig
#
# By: Danny Tholen
#
# Use with Type1 and TrueType fonts
#

dir "/usr/X11R6/lib/X11/fonts/Type1"
dir "/usr/X11R6/lib/X11/fonts/drakfont"
dir "/usr/share/fonts/default/Type1"

#
# alias 'fixed' and 'console' for 'mono'
# (some programs ask for 'mono' if they mean 'fixed';)
#
match any family == "fixed"        edit family =+ "mono";
match any family == "console"      edit family =+ "mono";


#
#Check users config file
#
includeif     "~/.xftconfig"


#
# Use TrueType fonts for defaults
# Danny: disabled
#match any family == "serif"       edit family += "Times New Roman";
#match any family == "sans"        edit family += "Verdana";


#
# Use lucida console as default fixed type font
# and set the spacing of "mono" to 100, this
# fixes broken fonts which are fixed, but do not
# set mono−spacing.
match
      any family == "mono"
edit
      family += "lucida console";
      spacing = 100;


#
# Alias between XLFD families and font file family name, prefer local fonts
#
match any family == "Charter"           edit family += "Bitstream Charter";
match any family == "Bitstream Charter" edit family =+ "Charter";

match any family == "Lucidux Serif"     edit family += "LuciduxSerif";
match any family == "LuciduxSerif"      edit family =+ "Lucidux Serif";

match any family == "Lucidux Sans"      edit family += "LuciduxSans";
match any family == "LuciduxSans"       edit family =+ "Lucidux Sans";

match any family == "Lucidux Mono"      edit family += "LuciduxMono";
match any family == "LuciduxMono"       edit family =+ "Lucidux Mono";


#
# TrueType font aliases
#
match any family == "Comic Sans"        edit family += "Comic Sans MS";
match any family == "Comic Sans MS"     edit family =+ "Comic Sans";
match any family == "Trebuchet"         edit family += "Trebuchet MS";
match any family == "Trebuchet MS"      edit family =+ "Trebuchet";
match any family == "Monotype"          edit family =+ "Monotype.com";
match any family == "Andale Mono"       edit family += "Monotype.com";
match any family == "Monotype.com"      edit family =+ "Andale Mono";


# Danny:
# set the AA for different fonts
#
# most TT fonts do not need to be aliased between
# 8 and 15 points, although this might be a matter of taste.
match
      any size > 8
      any size < 15
edit
      antialias = false;


# Danny: Courier looks terrible, and I
# cannot get most characters to fit nicely
# in their space. So I use courier 10 pitch
match
      any family == "courier"
edit
      family += "courier 10 pitch";
```

```
# these are symbols, and for some reason this needs to be added!:

match
      any family == "webdings"
edit
      antialias = false;
      encoding += "glyphs−fontspecific";

match
      any family == "symbol"
edit
      antialias = false;
      encoding += "glyphs−fontspecific";

match
      any family == "Standard Symbols L"
edit
      antialias = false;
      encoding += "glyphs−fontspecific";

match
      any family == "dingbats"
edit
      antialias = false;
      encoding += "glyphs−fontspecific";

match
      any family == "Cursor"
edit
      antialias = false;
      encoding += "glyphs−fontspecific";


# maybe arial looks better like this?:
match
      any family == "Arial"
      any size > 7
      any size < 15
edit
      antialias = false;


# end
```

4.2.4. GTK and GNOME

As mentioned above, KDE and QT do have solid anti−aliasing support with recent releases. GNOME−2 has just recently been released as of this writing (as of Aug 2002), and does have support for anti−aliasing, though is not in widespread use (yet).

There is also the gdkxft project available at http://sourceforge.net/projects/gdkxft/. This will add anti−aliasing support to GTK+ 1.2 applications (only!). This is a stand−alone library and not a patch. It will only effect GTK+ 1.x widgets (many of them but not all). There are some limitations, but mostly it works as advertized. Read the included `README` closely, and follow the instructions. An initial installation provides a good starting point. I added some font families to `/etc/gdkxft.conf` and I used some of Danny's suggestions above, and it seems to work mostly. It's a bit of a kludge, but quite a nice kludge ;−) Note, sadly this project seems to be no longer maintained.

There is also a couple of ways to get this working with Mozilla! On the same page above is a replacement library that can be substituted for the Mozilla library of the same name. Just a drop in replacement. But, this did not work here (on Red Hat 7.2), it segfaulted. But building Mozilla from source with the patch listed, did indeed work nicely! This step requires gdkxft to be working as well. So it's a two step process to get Mozilla working, but worth the effort if you want the best looking pages IMHO. There are also development versions of Mozilla available from ftp://ftp.mozilla.org/pub/mozilla/nightly/experimental/xft/ that have Xft/TrueType support (see the section on Mozilla below). Native Mozilla Xft support is probably working its way into mainstream releases. [Note: This should happen with Mozilla 1.2.]

4.2.5. Afterword

You've gone through all the steps, and verified that the "RENDER" extension is available, but you don't perceive a difference? Well, maybe the applications themselves just aren't there yet, and are not able to utilize these new features. Netscape, for instance, is not able to take advantage of anti−aliasing.

So how to know what does and does not make use of anti−aliasing? A quick test is to use something like xmag, or gimp, and enlarge the text considerably. Look for diagonal lines, and if they are clearly stair−stepped with no softened edges. If so, then while anti−aliasing is technically available, it is not being used. With anti−aliasing you should see gradients instead of well defined sharp edges. You can compare this with xterm and "AA":

```
$ xterm −fa charter −fs 14
```

You can also turn on Xft debugging:

```
 $ export XFT_DEBUG=2
```

Then start apps from that same terminal. This will tell if anti−aliasing is active for each font as its processed. This also gives some insight into how Xft understands font names and other details.

One final point: anti−aliasing and TrueType are completely separate issues. One does not depend on the other, though both together can can enhance appearance significantly. Especially, with good quality TrueType fonts! But any font can potentially be anti−aliased.

4.3. Red Hat 7.x and 8.x Differences

4.3.1. Red Hat 7.x

Red Hat 7.0 introduced some changes to X configuration over previous Red Hat versions. It is also different from the stock XFree86 configuration as addressed above. Notable differences:

- Both XFree86 3.3.6 and 4.x are included. If upgrading you may wind up with 3.3.6. The X configuration file is `XF86Config` for 3.3.6 and `XF86Config−4` for 4.x. Of course, you'll need to know which is which for editing and configuration purposes.
- xfs is still handling all font duties. A default Red Hat 7.x installation does not use the 'modules' section of XF86Config−4 for font handling. Instead it relies on xfs, which has this capability built in. This is different from a stock installation of XFree86 4.x where the X server does all the font work −− including TrueType.
- The socket for xfs is "unix/:7100" with RH 7.x, as opposed to "unix/:−1" in previous versions (i.e. Red Hat 6.x).
- As of Red Hat 7.1, the xfs init script actually runs **mkfontdir** and **ttmkfdir** on font directories known to xfs. So this step is not necessary when new fonts are added. Just restart xfs.

4.3.2. Red Hat 8.0

Red Hat 8.0 still includes XFree86 4.2, but it includes some significant changes! It is not clear at this time whether these are running changes to 4.2, or Red Hat has ported changes that are scheduled to appear in the soon to be released XFree86 4.3. Noteworthy changes:

- XFree86 3.3.6 is no longer bundled.
- `XF86Config−4` is deprecated in favor of `XF86Config` (X still uses the former if available though).
- The next generation of Xft is now used: Xft2! This means significant changes to font configuration, i.e `XftConfig` is no longer used in favor of `/etc/fonts/font.conf`, which is now an xml document (with much of the same functionality). See man fontconfig, and fc−cache.
- The preferred location for fonts is now `/usr/share/fonts/` and `~/.fonts/`. The system will automatically recognize, and include, any fonts (including TrueType) in these locations. See `/etc/fonts/fonts.conf`.
- Red Hat 8.0 of course includes GNOME−2, which has native anti−aliasing support for GTK apps.
- The xfs font server is still being used.

5. Adjusting Fonts in Specific Applications

5.1. Netscape

Note that Netscape 4.x should be considered obsolete these days. There are much better choices including Mozilla, Galeon and Konqueror. This section is being maintained for historical reasons; for those users who may have to use older Netscapes for one reason or another; and because it may have relevance to other applications.

Let's face it, Netscape is an important application in Linux. We all use it, and we all need it, so let's look at it specifically for a minute. An out of the box Netscape installation is prone to the font problems we've discussed −− large fonts that get pixelized, splotchy looking fonts, fonts so small they are unreadable. In short, ugly. Maybe this is why you are here?

Hopefully, at this point you have followed the above suggestions. These steps can help greatly. TrueType font availability is almost a necessity. Many web pages specify font families −− like Arial −− that are not typically available to Linux users. This is bad design, but having some of the basic TrueType fonts available will help greatly in overcoming the short−sightedness of some designers. Microsoft −− can't live with 'em, can't live without 'em.

Assuming you have TrueType working, from the Netscape menu select `Edit -> Preferences -> Fonts`. Open the `Variable Width Font` droplist on the right side of the window. Your TrueTypes should be there along with other fonts. Choose which ever one suits your fancy as the default. Check the `Allow Scaling` checkbox too. If the available point sizes are 0 and 12, you can go down and, and enter your desired point size in the box to the right and click on the `OK` button. The down−side to this is that Netscape will not remember these settings, and you will have to do this each time you start Netscape. *Unless* −− you have `fonts.alias` set up already. Then this will solve these problems. See Section 3.5 for more on `fonts.alias`.

You might consider experimenting with some `~/.Xdefaults` (or perhaps it's`~/.Xresources` on your system) settings too:

```
        Netscape*DocumentFonts.sizeIncrement: 10
        Netscape*documentFonts.maximumPoints: 240
        Netscape*documentFonts.xResolution*iso−8859−1: 120
        Netscape*documentFonts.yResolution*iso−8859−1: 120
```

The 'sizeIncrement' controls how much of a jump Netscape makes when different 'basefont' sizes are specified ala:

`<basefont size=7>`

for instance. The default is '20', which is a pretty good jump. Changing this can help Netscape from scaling to too large and too small of a font. The x and y resolutions are roughly equivalent to 'dpi' settings. Any random number within reason can be used here. Experiment. Note: Mozilla does not use this kind of configuration!

Then run:

```
$ xrdb −merge ~/.Xdefaults
```

(or `.Xresources` as the case may be) and restart Netscape. There are many settings that can be tweaked or altered this way. Look at the `Netscape.ad` (app defaults) file that should be included with Netscape packages.

If this approach does not get the job done as far as the 'tiny fonts' problem in Netscape, then see the `fonts.alias` section above. You can really fine tune many things with this approach.

5.2. Mozilla

Mozilla configuration should be roughly the same in many respects, but it does not use the Xresources type X configuration. You might find, however, that Mozilla does a much better job of handling fonts, and pages will look better overall. Highly recommended! The only caveat is, it seems to need a fairly fast system. It may be pretty sluggish on older hardware.

Also, some user preferences can be stored in "`user.js`" and "`unix.js`". Not to be confused with "`prefs.js`". `user.js` will likely have to be created by hand. Put it in whatever `.mozilla` sub−directory you find `prefs.js` in (this is not a consistent location, but typically ends like `*.slt`). Attempt to set a minimum font size:

```
// Don't ever show me a font smaller than this: some samples.
user_pref("font.min−size.variable.", 12);
user_pref("font.min−size.variable.x−western", 12);
user_pref("font.min−size.fixed.x−western", 12);
```

Other customizations can be made in `userChrome.css` and `userContent.css`. Again, both should be placed in the same directory as `prefs.js`. These files control much of the "UI" (the skin outside the browser window).

More info is available from the Mozilla developers: http://www.mozilla.org/unix/customizing.html.

It is also possbible to have anti−aliasing of fonts with Mozilla now. This might be a bit of work to get going at this time (Aug 2002), but it is possible. See the next section.

5.2.1. Mozilla with Xft

As mentioned above, the Xft extensions that give us anti−aliasing, and improved font handling, are slowly being rolled into Mozilla development. Such packages are often labeled as "TrueType" enabled. The reasoning being that they include native support for TrueType from within Mozilla. Note, that seems to be a bit of misnomer since properly installed TrueType fonts work quite well, even with ancient versions of Mozilla (though no anti−aliasing). The difference being that Mozilla does some of its own TrueType work.

There are many builds of Mozilla available, and it is not so easy to know which have Xft support. There are tips for knowing if your installed version does have this support, and then how to enable it here: http://www.mozilla.org/projects/fonts/unix/enabling_truetype.html. Worse comes to worse, you can download experimental versions from: ftp://ftp.mozilla.org/pub/mozilla/nightly/experimental/xft/.

For whatever reason, Mozilla has its own Xft configuration, that is independent of other system components. You must turn it on, and configure it (unless your vendor has done this already)! I know what you are saying, "one more place to configure fonts, sigh ...".

The above page explains rather tersely how to do this. The highlights:

Requirements:

- XFree86 4.x with Xft support.
- FreeType2 libraries.
- Mozilla with Xft support.
- TrueType fonts.

Configuration:

You will need to hand edit the relevant `unix.js` file, which is typically installed as `/usr/lib/mozilla−*/defaults/pref/unix.js`, or a similar location.

You will need to enable `FreeType2` (and possibly define the version), and then list the directories that contain your TrueType fonts (each listing must be uniquely identified, see example). Sample excerpt:

```
// TrueType //////////////////////////////////////////
pref("font.FreeType2.enable", true);
pref("font.freetype2.shared−library", "libfreetype.so.6");

// if libfreetype was built without hinting compiled in
// it is best to leave hinting off. try it both ways to see.
pref("font.FreeType2.autohinted", true);
pref("font.FreeType2.unhinted", false);

// below a certian pixel size anti−aliased fonts produce poor results
pref("font.antialias.min",        10);
pref("font.embedded_bitmaps.max", 1000000);
pref("font.scale.tt_bitmap.dark_text.min", 64);
pref("font.scale.tt_bitmap.dark_text.gain", "0.8");

// sample prefs for TrueType font dirs
//pref("font.directory.truetype.1", "/u/sam/tt_font");
//pref("font.directory.truetype.2", "/u/sam/other/tt_font");
pref("font.directory.truetype.1", "/usr/share/fonts/truetype");
```

The `//` characters are comments. Then restart Mozilla. You should see it processing each font in the directories you specified (if started from the command line). You then need to go into the Mozilla font configuration (Edit −> Preferences −> Appearance −> Fonts), and select the appropriate fonts these must be the ones that are listed with *the first letter as upper cased*:

```
 Afga−monotype−arial−iso8859−1
```

This should now give you TrueType fonts *with* anti−aliasing! Quite nice!

You should experiment with the "hinting" preferences. Having hinting enabled made a significant improvement here.

6. Odds and Ends

6.1. Notes

- Unfortunately there is no unified font handling system for Linux. You will have to configure each individual program so you can use TrueType, Type 1 or fonts that pique your fancy. And each program may well have its own way of doing this so you will have to RTFM. Desktop Environments like GNOME and KDE may provide much of this functionality however for apps that are under their control.
- Most GUI apps should be able to use TrueType, and Type 1 fonts too. Wordperfect for Linux, however, cannot use TrueType. (See the links section below for more on Wordperfect.) Text editors, terminal programs and the like need fixed width fonts, and do not play well with TrueType or other proportional fonts.
- Though not discussed here, Type 1 fonts provide many of the same benefits as TrueType and are historically better supported in the Unix world, especially for printing. XFree86 still seems to render TrueType somewhat better than Type 1. You likely have many of these installed already. Unfortunately however, Type 1 are not a web standard like TrueType. But they are suitable for many other purposes. They are where it's at for printing. See ghostscript for more on this.
- While it is possible to specify a default point size for the xfs font server, very few applications will actually use this value.
- Abiword comes with a suite of fonts, called 'Abisuite'. Apparently, some of these fonts have the same names as some of the well known MS TrueType fonts: Arial, etc. And apparently, these are not as good quality. And because of the way X searches for fonts, it may find these first and use these, even if the 'real' ones are installed and may be the preferred choice. The solution is to either make sure your preferred fonts come first in the FontPath or maybe to uninstall 'Abisuite' if it's not needed.
- The new Xft rendering extensions of XFree86 4.x will mostly supplant similar features as provided by xfs, and older XFree86 extensions. For instance, font aliasing should be done in `XftConfig` if the new extensions are being used. This would only be true where the application is built against a toolkit (like QT or GTK) that supports the new extensions. This is still not universally supported.

6.2. Links

- XFree86 Project, the guys and gals who do an incredible amount of work to give us a killer GUI environment. An overview of XFree86 fonts: XFree86 4.x. And an explanation of Xft: http://www.keithp.com/~keithp/render/Xft.tutorial.
- The Fontconfig home page is http://fontconfig.org, for all the latest news on Xft and fontconfig (the future of X font handling).
- The Video Timings HOWTO, the ins and outs of getting the most from your monitor. (Applicable only to XFree86 v.3.x for the most part.)
- Font HOWTO Many good tips for installing fonts and for applications such as StarOffice, Applixware, Wordperfect, Ghostscript, TeX/LaTeX.
- A TrueType HOWTO, good tips for printing, and a few application specific tips.
- xfsft Homepage, TrueType font support for X. This is the origin of the "freetype" font module for XFree86 4.x, and Red Hat's xfs. Good site, and good links to other information related to fonts and TrueType.
- Some Linux for Beginners. Great font site, and other Linux topics. Covers many of the topics discussed here in more detail. Some font and other tips for Mozilla: http://home.c2i.net/dark/My_Mozilla_FAQ.html.

- Two guides specifically for Debian from Paul D. Smith: http://www.paulandlesley.org/linux/xfree4_tt.html and for 3.3.x: http://www.paulandlesley.org/linux/debian_tt.html.
- X−TrueType Homepage, and yet another TrueType Font server, especially good for Japanese, Chinese and Korean character sets. [Note: link is bad 08/18/02.]
- Tips on font size problems from Netscape.
- Wordperfect for Linux −− Fonts and Printers by Rod Smith, the author of *Using Corel Wordperfect 8 for Linux* from Que. Excellent information on Wordperfect and where TrueType fits in.
- http://sourceforge.net/projects/font−tool/, contains the core MS TrueTyep fonts suitable for web browsing. http://packages.debian.org/unstable/graphics/msttcorefonts.html is a similar package for Debian.
- Sebastiano Vigna's http://gongolo.usr.dsi.unimi.it/~vigna/webFonts4Linux/ automates the downloading, extracting and installation of the Microsoft fonts all in one neat package.
- Freeware Connection −− Free Fonts Sites lots of links to lots of sites.
- Bitstream's Geometric Slabserif TrueType Font.
- Two converters for converting a Mac Font "suitcase" to a *nix compatible font: http://www.macinsearch.com/infomac2/font/util/tt−converter−15.html and http://www.netmagic.net/~evan/shareware/#TTFontConvert
- The Unicode HOWTO: http://linuxdoc.org/HOWTO/Unicode−HOWTO.html
- Two sources of 'free' TrueType fonts with large Unicode support are Bitstream Cyberbit, which covers Roman, Cyrillic, Greek, Hebrew, Arabic, combining diacritical marks, Chinese, Korean, Japanese, and more, and is available from ftp://ftp.netscape.com/pub/communicator/extras/fonts/windows/Cyberbit.ZIP. And Lucida Sans Unicode, which is included in IBM's JDK 1.3.0beta for Linux, and covers Roman, Cyrillic, Greek, Hebrew, combining diacritical marks. This can be downloaded from ftp://ftp.maths.tcd.ie/Linux/opt/IBMJava2−13/jre/lib/fonts/ as LucidaSansRegular.ttf and LucidaSansOblique.ttf. Thanks to Tzafrir Cohen for these references. He also has a nice page on Hebrew fonts and related topics at http://www.iglu.org.il/faq/?file=133.

A. Appendix: Font Servers

There are several font servers available that handle TrueType: xfstt, xfsft, and Red Hat's patched version of xfs based on xfsft. While these names are all too similar, these are different packages. One, or more, of these should be included with any recent Linux distribution, and you may have one installed already. If so, use which ever one your distribution is set up to use.

A.1. xfstt

One such font server is xfstt. xfstt was designed specifically with TrueType fonts in mind.

A.1.1. Installation

xfstt is *very* easy to install and configure. If it isn't already installed, you'll want to download the tarball, or check your CD. The most current version can be found at http://metalab.unc.edu/pub/Linux/X11/fonts/

Once you have the tarball, unpack it:

```
$ tar -zxvf xfstt-*tgz
```

Then build and install it. Read the `INSTALL` file for quick instructions, but it's a no brainer.

From the xfstt directory is all you have to do.

```
# make
# make install
```

Then start xfstt with:

```
# xfstt --sync              # updates xfstt's font database
# xfstt &                   # runs xfstt in the background.
```

xfstt should be started before the X server starts. Once you have this working correctly, you can add the above lines to `/etc/rc.d/rc.local`, or other suitable start up file. Then type:

```
$ xset +fp unix/:7101       # tells X about xfstt, and where to look for fonts.
```

or add:

```
    FontPath "unix/:7101"
```

to your `XF86Config` to tell X about the font server. Rerun **xfstt --sync** any time the FontPath, or contents, change.

A.1.2. Adjusting the Default Font Size

If your TrueType fonts appear to be very tiny, the following commands may help.

Add the `-dpi` switch to your X server command line (see above to do this.)

Use the `−−res` switch to tell xfstt to increase the default resolution. Use the following command line.

```
 # xfstt −−res 120
```

A.2. Red Hat's xfs

As of Red Hat Linux 6.0, Red Hat based distributions (Mandrake, etc) have included a specially patched version of xfs, the XFree86 Font Server, and patched X servers as well. Red Hat's xfs includes the xfsft patch set which in turn is built upon the FreeType Font library. Red Hat's xfs provides similar functionality to xfstt. xfs is able to serve both TrueType and Type 1 fonts, as well as legacy X fonts.

If you are using a Red Hat based distro, you should have xfs installed already. If not, it is in the `XFree86−xfs*rpm`. To make sure it runs as one of the default services, either use **ntsysv** or:

```
# chkconfig −−add xfs
```

Now xfs will start every time you boot.

A.2.1. Setting the xfs FontPath

The default Red Hat installation of xfs serves fonts via a Unix Domain Socket. We'll need to tell the X server where to look for xfs, and thus fonts. The FontPath in `/etc/X11/XF86Config` must include for Red Hat 6.x:

```
      FontPath   "unix/:−1"
```

This is changed for Red Hat 7.x to:

```
      FontPath   "unix/:7100"
```

At least for a default configurations. This is a reference to the socket where xfs is listening. You may include additional FontPaths, but these will be handled by the X server, and not xfs. A clean install of Red Hat 6/7 should have this already set up, but if you are upgrading from an older version, you may have to change this yourself!

xfs then has its own, separate FontPath stored in `/etc/X11/fs/config`. This is where it will look to find fonts. This is over and above the X server's FontPath in `XF86Config`. You can either add the new path(s) with a text editor, or use the **chkfontpath** command:

```
# chkfontpath −−add /new/font/path
```

The FontPath must exist before running **chkfontpath**. The relevant section of `/etc/X11/fs/config` should now look something like this:

```
     catalogue = /usr/X11R6/lib/X11/fonts/misc:unscaled,
                 /usr/X11R6/lib/X11/fonts/100dpi:unscaled,
                 /usr/X11R6/lib/X11/fonts/75dpi:unscaled,
                 /usr/X11R6/lib/X11/fonts/Type1,
                 /usr/X11R6/lib/X11/fonts/Speedo,
                 /usr/X11R6/lib/X11/fonts/misc,
                 /usr/X11R6/lib/X11/fonts/100dpi,
                 /usr/X11R6/lib/X11/fonts/75dpi,
                 /new/font/path
```

When adding a new FontPath for TrueType fonts, you will want to do this step after installing and preparing the fonts. See the next section.

A.2.2. Getting the Fonts Ready

Installation and configuration is the same as for other "freetype" based font renderers (e.g. XFree86−4.x freetype module). See the above Configuration section for details. Actually, you should just be able to install the fonts, include the new font directory to the `FontPath` with the **chkfontpath** utility, and then reload xfs:

```
      # service xfs reload
```

A.3. xfsft

xfsft is a TrueType solution from Juliusz Chroboczek. xfsft is based on the FreeType font library as developed by Mark Leisher and others. It is essentially is a patch for XFree86's xfs and related libraries −− xfs + ft. Red Hat's xfs is essentially xfsft with a few minor modifications. Also, XFree86 4.x includes the `freetype` font module which is also the result of Juliusz's work, and is one of the TrueType solutions available for XFree86 4.x.

Building xfsft requires having at least some of the XFree86 source available, in addition to xfsft itself, so this is not for the faint of heart. Instructions for building and configuring xfsft are in the tarball, so I won't go into details here. They are pretty straight forward. There are links to binaries available at the xfsft home page (see above).

Note that you *must* also create `fonts.scale` and `fonts.dir` files for xfsft. `fonts.scale` can be created manually (ugh!), or with the **ttmkfdir** utility. This is not included with xfsft but you can get it here: http://www.joerg−pommnitz.de/TrueType/ttmkfdir.tar.gz, or probably on many Linux archives sites too. Red Hat has this as part of the `Freetype` RPM. And for Debian it is called **mkttfdir** and is in the `fttools` package.

You will also need a configuration file. Here is a sample:

```
     −−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−

     clone−self = off
     use−syslog = off

     client−limit = 20

     catalogue = /usr/local/share/font/ttfonts

     error−file = /home/jec/fonts/xfs.errors

     # in decipoints
     default−point−size = 120

     # x,y
     default−resolutions = 100,100,75,75

     −−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−
```

You can then run start xfsft:

```
# xfs −port 7100 −config /path/to/your/config/file &
```

You can then add xfsft to the X server's `FontPath`:

```
$ xset +fp tcp/localhost:7100
```

If all goes well, you could then add this `FontPath` to `XF86Config`.

B. Appendix: `fonts.dir` to `fonts.alias`

Thanks to Aristotle Pagaltzis for providing this perl version of a script to convert a `fonts.scale` file to `fonts.dir`.

```
#!/usr/bin/perl −w

# This utility is a direct translation right down to the comments
# from mkfontalias.py written by Roman Sulzhyk to assist people
# in de−uglification of their true type fonts. It creates a
# fonts.alias file from the fonts.dir file found in the directory.

# It supports all of the Python script's features:
#
# − maps the following MS webfonts:
#       Verdana, Tahoma, Impact, Arial Black,
#       Comic Sans MS, Georgia, Trebuchet MS
#
# − cheats by mapping 6, 7 and 8pt to 9pt fonts
#

# (c)2002 Aristotle Pagaltzis, licensed under the GPL
# http://www.gnu.org/copyleft/gpl.html

use strict;

my $infile = "fonts.dir";
my $outfile = "fonts.alias";

my @res = (75, 75);

my %cheat_map = (6 => 9, 7 => 9, 8 => 9);

my @font_sizes = (6..16, 18, 24);

my %font_map = (
        "Arial" => "Arial",
        "Times New Roman" => "Times New Roman",
        "Verdana" => "Verdana",
        "Tahoma" => "Tahoma",
        "Impact" => "Impact",
        "Arial Black" => "Arial Black",
        "Comic Sans MS" => "Comic Sans MS",
        "Georgia" => "Georgia",
        "Trebuchet MS" => "Trebuchet MS",
        "Courier New" => "Courier New"
);

# Read in the fonts.
open(THEFILE, "<", $infile) or die "Cannot read $infile: $! − are you sure you are in the fonts d
my @fontdir = splice @{[ <THEFILE> ]}, 1; # Strip the first line
close THEFILE;

# Now, create the output
my @aliases;
foreach (@fontdir) {
        # Get rid of the first entry, but mind that other may have
        # spaces in them
        my $font = join(" ", splice @{[ split ]}, 1) or die "Cannot parse $infile line: $_\n";
        my @entries = split "−", $font;

        die "Invalid font: $font\n" unless @entries == 15;

        my $mapped_font = $font_map{$entries[2]} or next;

        # Create a bunch of aliases, for each size
        for my $size (@font_sizes) {
                # Do the "cheating" − fallback to size if not in the cheat map
                my $real_size = $cheat_map{$size} || $size;

                # Add the entry to the aliases
                push @aliases, join(
                        " ",
                        join("−", @entries[0..6], $real_size, $real_size * 10, @entries[9..14]),
                        join("−", @entries[0..1], $mapped_font, @entries[3..6], $size, $size * 10
                );
        }
}

# Boast
print "Created ", scalar(@aliases), " aliases\n";

# Backup the existing file
if(−e $outfile) {
        my $bakfile = "$outfile.bak";
        my $errormsg = "Cannot backup $outfile to $bakfile:";
        die "$errormsg file exists\n" if −e $bakfile;
        rename $outfile, $bakfile or die "$errormsg $!\n"
}

# Ok, write the file
open(THEFILE, ">", $outfile) or die "Cannot open $outfile for writing: $!\n";
print THEFILE map "$_\n", @aliases;
close THEFILE;

print "Wrote ", scalar(@aliases), " aliases to $outfile\n";
```