

NAME

autoinst – wrapper around the LCDF TypeTools, for installing OpenType fonts in LaTeX.

SYNOPSIS

autoinst [*options*] **font(s)**

DESCRIPTION

Eddie Kohler’s *TypeTools*, especially *otftotfm*, are superb tools for installing OpenType fonts in LaTeX, but their use (even in automatic mode) is complicated: they need many long command lines and don’t generate the *fd* and *sty* files LaTeX needs. **autoinst** simplifies the font installation process with *otftotfm* by generating and executing all command lines and by creating and installing all *fd* and *sty* files.

Given a family of font files (in either *otf* or *ttf* format), **autoinst** will create several LaTeX font families:

- Four text families (with lining and oldstyle digits, in tabular and proportional variants), each with the following shapes:

<i>n</i>	Roman text
<i>sc</i>	Small caps
<i>nw</i>	“Upright swash”; usually normal text with some extra “oldstyle” ligatures, such as ct, sp and st.
<i>tl</i>	Titling shape. Meant for all-caps text only (even though it sometimes contains lowercase glyphs as well), where letterspacing and the positioning of punctuation characters have been adjusted to suit all-caps text. This shape is generated only for the families with lining digits, since all-caps text doesn’t mix with old-style digits.
<i>it</i>	Italic (or oblique) text
<i>scit</i>	Italic small caps
<i>sw</i>	Swash
<i>tlit</i>	Italic titling
- For each text family: a family of TS1–encoded symbol fonts, in roman and italic shapes.
- Four families with superiors, inferiors, numerators and denominators, in roman and italic shapes.
- An ornament family, in roman and italic shapes.

Of course, if the fonts don’t contain oldstyle digits, small caps etc., the corresponding shapes and families are not created. Furthermore, the creation of most families and shapes can be controlled by command-line options (see “COMMAND-LINE OPTIONS” below).

The generated font families are named *<FontFamily>-<Suffix>*, where *<Suffix>* is one of

<i>LF</i>	proportional (i.e., figures have varying widths) lining figures
<i>TLF</i>	tabular (i.e., all figures have the same width) lining figures
<i>OsF</i>	proportional oldstyle figures
<i>TOsF</i>	tabular oldstyle figures

<i>Sup</i>	superior characters (many fonts have only an incomplete set of superiors: digits, punctuation and the letters <i>abdeilmnorst</i> ; normal forms will be used for the other characters)
<i>Inf</i>	inferior characters; usually only digits and punctuation, normal forms for the other characters
<i>Orn</i>	ornaments
<i>Numr</i>	numerators
<i>Dnom</i>	denominators

The generated fonts are named `<FontName>-<suffix>-<shape>-<enc>`, where `<suffix>` is the same as above (but in lowercase), `<shape>` is either empty, “sc”, “swash” or “titling”, and `<enc>` is the encoding. A typical name in this scheme would be “MinionPro-Regular-osf-sc-ly1”.

On the choice of text encoding

By default, **autoinst** generates text fonts with OT1, T1 and LY1 encodings, and the generated style files use LY1 as the default text encoding. LY1 has been chosen over T1 because it has some empty slots to accomodate the additional ligatures and alternate glyphs provided by many OpenType fonts. Different encodings can be selected using the `-encoding` command-line option (see “COMMAND-LINE OPTIONS” below).

Using the fonts in your LaTeX documents

autoinst generates a style file for using the font in LaTeX documents, named `<FontFamily>.sty`. This style file also takes care of loading the *fontenc* and *textcomp* packages, if necessary. To use the font, simply put `\usepackage{<FontFamily>}` in the preamble of your document.

This style file defines a number of options:

lining, oldstyle, tabular, proportional

Choose which digits will be used for the text fonts. The defaults are “oldstyle” and “proportional” (if available).

ultrablack, ultrabold, heavy, extrablack, black, extrabold, demibold, semibold, bold

Choose the weight that LaTeX will use for the “bold” weight (i.e., the value of `\bfdefault`).

light, medium, regular

Choose the weight that LaTeX will use for the “regular” weight (i.e., the value of `\mddefault`).

scaled=<scale>

Scale the font by a factor of `<scale>`. For example: to increase the size of the font by 5%, use the command `\usepackage[scaled=1.05]{<FontFamily>}`.

This option is only available when the *xkeyval* package is found in your TeX installation.

The style file will also try to load the *fontaxes* package, which gives easy access to various font shapes and styles. It is available from CTAN (<http://www.ctan.org/tex-archive/macros/latex/contrib/fontaxes>). Using the machinery set up by *fontaxes*, the generated style file also defines a number of commands (which take the text to be typeset as argument) and declarations (which don’t take arguments, but affect all text up to the end of the current group) of its own:

DECLARATION	COMMAND	SHORT FORM OF COMMAND
<code>\tlshape</code>	<code>\texttitling</code>	<code>\textttl</code>
<code>\sufigures</code>	<code>\textsuperior</code>	<code>\textsu</code>
<code>\infigures</code>	<code>\textinferior</code>	<code>\textin</code>

In addition, the `\swshape` and `\textsw` commands are redefined to place swash on the secondary shape axis (*fontaxes* places it on the primary shape axis); this makes these commands behave properly when nested, so that `\swshape\upshape` will give upright swash.

There are no commands for accessing the numerator and denominator fonts; these can be selected using *fontaxes*' standard commands, e.g., `\fontfigurestyle{numerator}\selectfont`.

The style file also provides a command `\ornament{<number>}`, where *<number>* is a number from 0 to the total number of ornaments minus one. Ornaments are always typeset using the current family, series and shape. A list of all ornaments in a font can be created by running LaTeX on the file *nfssfont.tex* (part of a standard LaTeX installation) and supplying the name of the ornament font.

To access the ornaments, **autoinst** creates a font-specific encoding file *<FontFamily>_orn.enc*, but only if that file doesn't yet exist. This means you can provide your own encoding vector, e.g. if your fonts use non-standard glyph names for ornaments.

These commands are only generated for shapes and number styles that actually exist; no commands are generated for shapes and styles that don't exist, or whose generation has been turned off. Please note that these commands are built on top of *fontaxes*; if that package cannot be found, you're limited to using the lower-level commands from standard NFSS (`\fontfamily`, `\fontseries`, `\fontshape` etc.).

Using multiple font families in one document

If you want to use more than one font family in a document, be aware that style files generated by versions of **autoinst** older than 2009 are incompatible with those generated by newer versions.

NFSS codes

NFSS identifies fonts by a combination of family, series (weight plus width), shape and size. **autoinst** parses the output of `otfinfo --info` to determine these parameters. When this fails (e.g., because the font family contains unusual widths or weights), **autoinst** ends up with different fonts having the *same* values for the font parameters, which means that these fonts cannot be used in NFSS. **autoinst** will then split the font family into multiple subfamilies (based on the font's "Subfamily" value) and try again. (Since many font families overdo the "Subfamily" by making each font a separate subfamily, this strategy is only used as a last resort.)

If this proliferation of font families is unwanted, either rerun **autoinst** on a smaller set of fonts (omitting the ones that failed to parse correctly) or else add the missing widths, weights and shapes to the tables `%FD_WIDTH`, `%FD_WEIGHT` and `%FD_SHAPE`, near the top of the source code. Please also send a bug report (see AUTHOR below).

autoinst maps widths, weights and shapes to NFSS codes using the following tables. These are based both on the standard *Fontname* scheme and on the tables in Philipp Lehman's *Font Installation Guide*, but some changes had to be made to avoid name clashes in font families with many different widths and weights.

WEIGHT		WIDTH	
Thin	t	Ultra Compressed	up
Ultra Light	ul	Extra Compressed	ep
Extra Light	el	Compressed, Compact	p
Light	l	Compact	p
Book	[1]	Ultra Condensed	uc
Regular	[1]	Extra Condensed	ec
Medium	mb	Condensed	c
Demibold	db	Narrow	n
Semibold	sb	Semicondensed	sc
Bold	b	Regular	[1]
Extra Bold	eb	Semiextended	sx
Ultra	ub	Extended	x
Ultra Bold	ub	Expanded	e
Black	k	Wide	w
Extra Black	ek		
Ultra Black	uk		
Heavy	h	SHAPE	
Poster	r		
		Roman, Upright	n [2]
		Italic	it
		Cursive, Kursiv	it
		Oblique, Slanted	it [3]
		Incline(d)	it [3]

Notes:

- [1] When *both* weight and width are empty, the “series” attribute becomes “m”.
- [2] Adobe Silention Pro contains two “Roman” shapes (“RomanI” and “RomanII”); the first of these is mapped to “n”, the second one to “it”.
- [3] Mapping the “Slanted”, “Oblique” or “Inclined” shape to “it” instead of “sl” simplifies **autoinst**. Since font families with both italic and slanted shapes do – to the best of my knowledge – not exist (apart from Computer Modern, of course), this shouldn’t cause problems in real life.

A note for MiKTeX users

Automatically installing the fonts into a suitable TEXMF tree (as **autoinst** does by default) requires a TeX-installation that uses the *kpathsea* library; with TeX distributions that implement their own directory searching (such as MiKTeX), **autoinst** will complain that it cannot find the *kpsewhich* program and install all generated files into subdirectories of the current directory. If you use such a TeX distribution, you should either move these files to their correct destinations by hand, or use the *-target* option (see “COMMAND-LINE OPTIONS” below) to specify a TEXMF tree.

Also, some OpenType fonts lead to *pl* and *vpl* files that are too big for MiKTeX’s *pltoif* and *vptovf*; the versions that come with W32TeX (<http://www.w32tex.org>) and TeXLive (<http://tug.org/texlive>) don’t have this problem.

COMMAND-LINE OPTIONS

You may use either one or two dashes before options, and option names may be shortened to a unique prefix (e.g., **-enc** instead of **-encoding**).

-encoding=encoding[,encoding]

Use the specified encodings for the text fonts. The default is “OT1,T1,LY1”. For each encoding, a file *<encoding>.enc* (in all *lowercase*) should be somewhere where *otftotfm* can find it. Suitable encoding files for OT1, T1/TS1 and LY1 come with **autoinst**. (Note that these files are called *fontools_ot1.enc* etc. to avoid name clashes with other packages; the “fontools_” prefix doesn’t need to be specified.)

Multiple text encodings can be specified as a comma-separated list: **-encoding=OT1,T1**. The encodings are passed to *fontenc* in the order specified, so the last one will be the default text encoding.

-sanserif

Install the font as a sanserif font, accessed via `\sffamily` and `\textsf`. Note that the generated style file redefines `\familydefault`, so including it will still make this font the default text font.

-typewriter

Install the font as a typewriter font, accessed via `\ttfamily` and `\texttt`. Note that the generated style file redefines `\familydefault`, so including it will still make this font the default text font.

-ts1

-nots1

Turn the creation of TS1-encoded fonts on or off. The default is **-ts1** if the text encodings (see *-encoding* above) include T1, **-nots1** otherwise.

-smallcaps

-nosmallcaps

Turn the creation of small caps fonts on or off. The default is **-smallcaps**.

-swash

-noswash

Turn the creation of swash fonts on or off. The default is **-swash**.

-titling

-notitling

Turn the creation of titling fonts on or off. The default is **-titling**.

-superiors

-nosuperiors

Turn the creation of fonts with superior characters on or off. The default is **-superiors**.

-inferiors

-noinferiors

Turn the creation of fonts with inferior digits on or off. The default is **-noinferiors**.

-fractions

-nofractions

Turn the creation of fonts with numerators and denominators on or off. The default is **-nofractions**.

–ornaments**–noornaments**

Turn the creation of ornament fonts on or off. The default is **–ornaments**.

–target=DIRECTORY

Install all generated files into the TEXMF tree at *DIRECTORY*.

By default, **autoinst** searches your \$TEXMFLOCAL and \$TEXMFHOME paths and installs all files into subdirectories of the first writable TEXMF tree it finds (or into subdirectories of the current directory, if no writable directory is found).

Note: this option gets overridden by the **–manual** option!

–manual

Manual mode. By default, **autoinst** immediately executes all *otftotfm* command lines it generates; with the **–manual** option, these commands are instead written to a file *autoinst.bat*. Also, the generated *otftotfm* command lines specify the **–pl** option (which tells *otftotfm* to generate readable/editable *pl* and *vpl* files instead of the default *tfm* and *vf* files) and leave out the **–automatic** option (which tells *otftotfm* to leave all generated files in the current directory, rather than install them into your TEXMF tree).

When using this option, you should run *pltoft* and *vptovf* after executing all commands, to convert the *pl* and *vf* files to *tfm* and *vf* format.

Note: this option overrides the **–target** option!

–verbose

Verbose mode; print detailed info about what **autoinst** thinks it’s doing.

–extra=text

Pass *text* as options to *otftotfm*. To prevent *text* from accidentally being interpreted as options to **autoinst**, it should be properly quoted.

–figurekern**–nofigurekern**

Some fonts provide kerning pairs for tabular figures. This is probably unwanted (e.g., numbers in tables won’t line up exactly). The option **–nofigurekern** adds extra **–ligkern** options to the command lines for *otftotfm* to suppress such kerns (but only for the “TLF” and “TOsF” families). Since this leads to very long command lines (it adds one hundred such options) and the problem affects only few fonts, the default is **–figurekern**.

SEE ALSO

Eddie Kohler’s TypeTools (<http://www.lcdf.org/type>).

Perl is pre-installed on most Linux and Unix systems; on Windows, try ActiveState’s ActivePerl (available from <http://www.activestate.com>) or Strawberry Perl (<http://strawberryperl.com>).

XeTeX (<http://www.tug.org/xetex>) and LuaTeX (<http://www.luatex.org>) are TeX extensions that can use any font (including both flavours of OpenType) without TeX-specific support files.

John Owens’ *otfinst* (available from CTAN) is another wrapper around *otftotfm*, and may work when **autoinst** doesn’t.

The MinionPro for LaTeX project (<http://developer.berlios.de/projects/minionpro/>, also on CTAN) developed very complete support files for Minion Pro (including math), but the project appears to have been abandoned; these files don’t work with the latest versions of Minion Pro.

AUTHOR

Marc Penninga <marcpenninga@gmail.com>

When sending a bug report, please give as much relevant information as possible; this includes at least (but may not be limited to) the output from running **autoinst** with the `-verbose` option. Please make sure that this output includes all (if any) error messages.

COPYRIGHT

Copyright (C) 2005–2012 Marc Penninga.

LICENSE

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 2 of the License, or (at your option) any later version. A copy of the GNU General Public License is included with **autoinst**; see the file *GPLv2.txt*.

DISCLAIMER

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

RECENT CHANGES

(See the source code for the rest of the story.)

- 2012–03–06 Implemented the “splitting the font family into multiple subfamilies” emergency strategy when font info parsing fails. Added recognition for a number of unusual widths, weights and shapes.
- 2012–02–29 Fixed a bug in the font parsing code, where possible widths, weights and shapes were tested in the wrong order; this led to “ExtraLight” fonts being recognised as “Light”. Added recognition for “Narrow” and “Wide” widths. Also added the `-(no)figurekern` command-line option.
- 2012–02–01 Reorganised the code, and fixed some bugs in the process. Added the `-target` command-line option. Made **autoinst** install the *fd* and *sty* files in the same TEXMF tree as the other generated files. Generate OT1, T1 and LY1 encoded text fonts by default. Made `-titling` a default option (instead of `-notitling`). Updated the documentation.
- 2011–06–15 Fixed the font info parsing code for some fonts that are too lazy to spell out “Italic” in full.
- 2010–04–29 The `--feature=kern` option is only used for fonts that contain either a *kern* feature or a *kern* table. Font feature selection commands in the *sty* file are only generated for shapes and figure styles that are supported by the current font, and whose generation has not been turned off using the command-line options. Fixed the font info parsing to work with the Condensed fonts in the Minion Pro family.
- 2010–04–23 Always provide the `--feature=kern` option to *otftotfm*, even if the font doesn’t have a *kern* feature; this will make *otftotfm* use the *kern* table if present. For fonts without a *liga* feature, add `--ligkern` options for the common f–ligatures to the *otftotfm* command line, so that any ligatures present in the font will still be used. Bug fix: the generated *sty* files now work for font families with names containing digits.

- 2009-04-09 Prefixed the filenames of the included encoding files with *fontools_*, to prevent name clashes with other packages.
- 2009-04-06 A small patch to the `get_orn` subroutine: it now also recognises the *bullet.xxx* ornament glyphs in Adobe Kepler Pro.