NAME

autoinst − wrapper around the LCDF TypeTools, for installing and using OpenType fonts in (La)TeX.

SYNOPSIS

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

DESCRIPTION

Eddie Kohler's *LCDF TypeTools* are superb tools for installing OpenType fonts in LaTeX, but they can be hard to use: they need many, often long, command lines and don't generate the *fd* and *sty* files LaTeX needs. **autoinst** simplifies the use of the *TypeTools* for font installation by generating and executing all commands for *otftotfm* and by creating and installing all necessary *fd* and *sty* files.

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

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

n	Roman (i.e., upright) text
it, *sl*	Italic and slanted (sometimes called oblique) text
sc	Small caps
sw	Swash
tl	Titling shape. Meant for all-caps text; letterspacing and the positioning of punctuation characters have been adjusted to suit all-caps text. (This shape is only generated for the families with lining digits, since old-style digits make no sense with all-caps text.)
scit, *scsl*	Italic and slanted small caps
nw	"Upright swash"; usually roman text with a few "oldstyle" ligatures like ct, sp and st.
tlit, *tlsl*	Italic and slanted titling text

- For each T1−encoded text family: a family of TS1−encoded symbol fonts, in roman, italic and slanted shapes.
- Families with superiors, inferiors, numerators and denominators, in roman, italic and slanted shapes.
- An ornament family, also in roman, italic and slanted shapes.

Of course, if your fonts don't contain italics, oldstyle digits, small caps etc., the corresponding shapes and families are not created. In addition, the creation of most families and shapes can be controlled by the user (see "COMMAND-LINE OPTIONS" below).

These families use the *FontPro* project's naming scheme: *<FontFamily>−<Suffix>*, where *<Suffix>* is:

LF	proportional (i.e., figures have varying widths) lining figures
TLF	tabular (i.e., all figures have the same width) lining figures
OsF	proportional oldstyle figures
TOsF	tabular oldstyle figures
Sup	superior characters (note that most fonts have only an incomplete set of superior characters: digits, some punctuation and the letters *abdeilmnorst*; normal forms are used for other characters)
Inf	inferior characters; usually only digits and some punctuation, normal forms for other characters
Orn	ornaments
Numr	numerators
Dnom	denominators

The individual 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 (also in lowercase). A typical name in this scheme would be "FiraSans−Light−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 accommodate the additional ligatures found in many OpenType fonts. Other encodings can be chosen using the *−encoding* option (see "COMMAND-LINE OPTIONS" below).

Using the fonts in your LaTeX documents

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

This style file defines a number of options:

`lining`, `oldstyle`, `tabular`, `proportional`
: Choose which figure style to use. The defaults are "oldstyle" and "proportional" (if available).

`scale=<number>`
: Scale the font by a factor of *<number>*. E.g., to increase the size of the font by 5%, use `\usepackage[scale=1.05]{<FontFamily>}`. May also be spelled `scaled`.

  This option is only available when you have the *xkeyval* package installed.

`medium`, `book`, `text`, `regular`
: Select the weight that LaTeX will use as the "regular" weight; the default is `regular`.

`heavy`, `black`, `extrabold`, `demibold`, `semibold`, `bold`
: Select the weight that LaTeX will use as the "bold" weight; the default is `bold`.

The previous two groups of options will only work if you have the *mweights* package installed.

The style file will also try to load the *fontaxes* package (available on CTAN), which gives easy access to various font shapes and styles. Using the machinery set up by *fontaxes*, the generated style file 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) to access titling, superior and inferior characters:

```
DECLARATION     COMMAND          SHORT FORM OF COMMAND

\tlshape        \texttitling     \texttl
\sufigures      \textsuperior    \textsu
\infigures      \textinferior    \textin
```

In addition, the `\swshape` and `\textsw` commands are redefined to place swash on *fontaxes*' secondary shape axis (*fontaxes* places it on the primary shape axis) to make them 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 ornament glyphs, **autoinst** creates a font-specific encoding file *<FontFamily>_orn.enc*, but only if that file doesn't yet exist in the current directory. This is a deliberate feature that allows you to provide your own encoding vector, e.g. if your fonts use non-standard glyph names for ornaments.

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

NFSS codes

***NOTE**: this functionality was almost completely rewritten in release 2019−03−13. Older versions tried to directly map all fonts to short NFSS codes, but often had to invent non-standard codes in order to deal with the many different weights and widths that occur in the wild. These non-standard NFSS codes used by older versions of **autoinst** will no longer work for fonts installed with newer versions; for those you'll have to either use the long names or stick to the standard NFSS codes. This change mainly concerns very light or very heavy weights and very condensed widths; for more moderate weights and widths, existing code will probably continue to work.*

NFSS identifies fonts by a combination of family, series (the concatenation of weight and width), shape and size. **autoinst** parses the font's metadata (more precisely: the output of `otfinfo --info`) to determine these parameters. When this fails (usually because the font family contains uncommon weights, widths or shapes), **autoinst** ends up with different fonts having the *same* values for these font parameters; such fonts cannot be used in NFSS, since there's no way distinguish them. Whenever **autoinst** detects such a situation, it will quit with a detailed error message. If that happens, either rerun **autoinst** on a smaller set of fonts or add the missing widths, weights and shapes to the tables `%NFSS_WIDTH`, `%NFSS_WEIGHT` and `%NFSS_SHAPE`, near the top of the source code. Please also send a bug report (see AUTHOR below).

The mapping of shapes to NFSS codes is done using the following table:

```
SHAPE                              CODE
--------------------------------   ----
Roman, Upright                     n
Italic, Cursive, Kursive           it
Oblique, Slant(ed), Incline(d)     sl
```

(*Exception:* Adobe Silentium Pro contains two Roman shapes; we map the first of these to ''n'', for the second one we [ab]use the ''it'' code as this family doesn't contain an Italic shape.)

The mapping of weights and widths to NFSS code is a more complex, two-step proces. In the first step, all fonts are assigned a ''series'' name that is simply the concatenation of its full weight and width (expanding any abbreviations and converting to lowercase). So a font with ''Cond'' width and ''Ultra'' weight will be known as ''ultrablackcondensed''.

In the second step, **autoinst** tries to map all combinations of NFSS codes (ul, el, l, sl, m, sb, b, eb and ub for weights; uc, ec, c, sc, m, sx, x, ex and ux for widths) to actual fonts. Of course, not all 81 combinations of these NFSS weights and widths will map to existing fonts; and conversely it may not be possible to assign every existing font a unique code in a sane way (especially on the weight axis, some font families offer more choices or finer granularity than NFSS's codes can handle; e.g., Fira Sans contains fifteen(!) different weights, including an additional ''Medium'' weight between Regular and Semibold).

This mapping between NFSS codes and actual fonts is based on a few principles:

Usefulness. As many of the most commonly used NFSS codes as possible should point to actual fonts.
Exactness. Exact matches always win: if the font family contains a Condensed Semibold font, that's what the ''sbc'' code will map to.
Sanity. A code like ''sb'' will always map to something semi-boldish. If there's no Semibold font it might map to Demibold or Medium, but never to Black. If there is no close match, the NFSS code will simply not be used.
Well-ordering. The mapping respects the ordering that is inherent in the NFSS codes, so ''sb'' will be heavier than ''m'' and lighter than ''b''.
Uniqueness. No two NFSS codes will map to the same font (with the exception of ''bx''; since this is so ubiquitous in Latex, **autoinst** will treat it as a synonym for ''b'' if there is no BoldExtended font).

These rules should ensure that the standard NFSS codes (and high-level commands such as `\bfseries`, which are built on top of these codes) will ''just work''. To access specific weights or widths, use the `\fontseries` command with the full series name (i.e., `\fontseries{demibold}\selectfont`).

To see exactly which NFSS codes map to which fonts, please refer to the generated *fd* files.

A note for MiKTeX users

Automatically installing the fonts into a suitable TEXMF tree (as **autoinst** tries to do 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 contain so many kerning pairs that the resulting *pl* and *vpl* files are too big for MiKTeX's *pltotf* and *vptovf*; the versions that come with W32TeX (*http://www.w32tex.org*) and TeXLive (*http://tug.org/texlive*) don't seem to have this problem.

COMMAND-LINE OPTIONS

autoinst tries hard to do The Right Thing (TM) by default, so you usually won't really need these options; but most aspects of its operation can be fine-tuned if you want to.

You may use either one or two dashes before options, and option names may be shortened to a unique prefix (e.g., **−encoding** may be abbreviated to **−enc** or even **−en**, but **−e** is ambiguous (it may mean either **−encoding** or **−extra**)).

−dryrun
: Don't generate any output files; only parse the input fonts and create *autoinst.log* showing which fonts would have been generated.

−encoding=*encoding[,encoding]*
: Generate the specified encoding(s) for the text fonts. Multiple text encodings may be specified as a comma-separated list: `-encoding=OT1,T1,LY1` (without spaces!). The generated style file passes these encodings to *fontenc* in the specified order, so the last one will become the default text encoding for your document.

  The default choice of encodings 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, LY1, LGR and T2A/B/C come with **autoinst**. (These files are called *fontools_ot1.enc* etc. to avoid name clashes with other packages; the "fontools_" prefix may be omitted.)

−ts1 / −nots1
: Control the creation of TS1−encoded fonts. The default is **−ts1** if the text encodings (see *−encoding* above) include T1, **−nots1** otherwise.

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

−lining / −nolining
: Control the creation of fonts with lining figures. The default is **−lining**.

−oldstyle / −nooldstyle
: Control the creation of fonts with oldstyle figures. The default is **−oldstyle**.

−proportional / −noproportional
: Control the creation of fonts with proportional figures. The default is **−proportional**.

−tabular / −notabular
: Control the creation of fonts with tabular figures. The default is **−tabular**.

−smallcaps / −nosmallcaps
: Control the creation of small caps fonts. The default is **−smallcaps**.

–swash / –noswash
Control the creation of swash fonts. The default is **–swash**.

–titling / –notitling
Control the creation of titling fonts. The default is **–titling**.

–superiors / –nosuperiors
Control the creation of fonts with superior characters. The default is **–superiors**.

–inferiors=[sinf | subs | dnom]
The OpenType standard defines several kinds of digits that might be used as inferiors or subscripts: ‘‘Scientific Inferiors’’ (OpenType feature ‘‘sinf’’), ‘‘Subscripts’’ (‘‘subs’’) and ‘‘Denominators’’ (‘‘dnom’’). This option allows the user to determine which of these styles **autoinst** should use for the inferior characters. The default is not to create fonts with inferior characters.

Note that many fonts contain only one (or even none) of these types of inferior characters. If you specify a style of inferiors that isn't present in the font, ***autoinst*** *silently falls back to its default of not creating fonts with inferiors; it doesn't try to substitute one of the other features.*

–fractions / –nofractions
Control the creation of fonts with numerators and denominators. The default is **–nofractions**.

–ornaments / –noornaments
Control the creation of ornament fonts. The default is **–ornaments**.

–defaultlining / –defaultoldstyle
–defaulttabular / –defaultproportional
Tell **autoinst** which figure style is the current font family's default (i.e., which figures you get when you don't specify any OpenType features).

Don't use these options unless you are certain you need them! They are only needed for fonts that don't provide OpenType features for their default figure style; and even in that case, **autoinst**'s default values (**–defaultlining** and **–defaulttabular**) are usually correct.

–nofigurekern
Some fonts provide kerning pairs for tabular figures. This is very probably not what you want (e.g., numbers in tables won't line up exactly). This option adds extra *––ligkern* options to the commands for *otftotfm* to suppress such kerns. Note that this option leads to very long commands (it adds one hundred *––ligkern* options), which may cause problems on some systems.

–mergewidths / –nomergewidths
Some font families put Condensed, Narrow, Extended etc. fonts in separate families; this option tells **autoinst** to merge those separate families into the main family. The default is **–nomergewidths**.

–extra=*text*
Append *text* as extra options to the command lines for *otftotfm*. To prevent *text* from accidentily being interpreted as options to **autoinst**, it should be properly quoted.

–manual
Manual mode. By default, **autoinst** immediately executes all *otftotfm* commands it generates; with the **–manual** option, these commands are instead written to a file *autoinst.bat*. Furthermore it adds the *––pl* option (which tells *otftotfm* to generate human readable/editable *pl* and *vpl* files instead of the default *tfm* and *vf* files) and omits the *––automatic* option (which causes *otftotfm* to leave all generated files in the current directory, rather than install them into your TEXMF tree). Manual mode is meant to enable tweaking the generated commands and post-processing the generated files.

When using this option, run *pltotf* and *vptovf* after executing the commands (to convert the *pl* and *vf* files to *tfm* and *vf* format) and move all generated files to their proper destinations.

All following options are only meaningful in automatic mode, and hence ignored in manual mode:

−target=*DIRECTORY*
: Install all generated files into the TEXMF tree at *DIRECTORY*. This option allows the user to override **autoinst**'s default behaviour, which is to search the `$TEXMFLOCAL` and `$TEXMFHOME` paths and install 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).

−vendor=*VENDOR*
−typeface=*TYPEFACE*
: These options are equivalent to *otftotfm*'s *−−vendor* and *−−typeface* options: they change the ''vendor'' and ''typeface'' parts of the names of the subdirectories in the TEXMF tree where generated files will be stored. The default values are ''lcdftools'' and the font's FontFamily name.

  Note that these options change *only* directory names, not the names of any generated files.

−updmap / −noupdmap
: Control whether or not *updmap* is called after the last call to *otftotfm*. The default is **−updmap**.

SEE ALSO

Eddie Kohler's **TypeTools** (*http://www.lcdf.org/type*).

Perl can be obtained from *http://www.perl.org*; it is included in most Linux distributions. For Windows, try ActivePerl (*http://www.activestate.com*) or Strawberry Perl (*http://strawberryperl.com*).

XeTeX (*http://www.tug.org/xetex*) and **LuaTeX** (*http://www.luatex.org*) are Unicode-aware TeX engines that can use OpenType fonts directly, without any (La)TeX−specific support files.

The **FontPro** project (*https://github.com/sebschub/FontPro*) offers very complete LaTeX support (even for typesetting maths) for Adobe's Minion Pro, Myriad Pro and Cronos Pro font families.

AUTHOR

Marc Penninga (*marcpenninga@gmail.com*)

When sending a bug report, please give as much relevant information as possible. If you see any error messages (either from **autoinst** itself, from the *LCDF TypeTools*, from Perl or from the OS), include these *verbatim*; don't paraphrase.

COPYRIGHT

LICENSE

DISCLAIMER

RECENT CHANGES

(See the source for the full story, all the way back to 2005.)

2019−03−13 Overhauled the mapping of fonts (more specifically of weights and widths; the mapping of shapes didn't change) to NFSS codes. Instead of inventing our own codes to deal with every possible weight and width out there, we now create ''long'' codes based on the names in the font metadata. Then we add ''ssub'' rules to the *fd* files to map the standard NFSS codes to our fancy names (see the section **NFSS codes**; based on discussions with Frank Mittelbach and Bob Tennent).

2018−08−10 Added encoding files for LGR and T2A/B/C to *fontools*.