lua-font-config

Clea F. Rees*

v0.2†

Abstract

An opentype font configuration package for LuaLaTeX 2ε. The aim is to provide an alternative to fontspec which is both faster and more powerful, by doing as much processing as possible in Lua. To avoid loading fonts unnecessarily, the data used is corrected during compilation and the results cached for future use.

This package is experimental.

1 Requirements

The package requires LuaLaTeX 2ε. It will not work with any other engine or format.

Unicode maths is supported *via* lua-unicode-math. It will **not** work with unicode-math.

2 Usage

The user interface is currently somewhat basic[1] .

Currently documents using exclusively Latin scripts with no changes of the base font families are reasonably well supported *via* key-value based configuration. This enables the choice of font family for the standard families — i.e. roman/serif, sans, typewriter/mono and (Unicode) maths (main, script and calligraphic) — and configuration of additional fonts, again by family. In addition to this, scaling, 'fake bold', rendering mode, language choice and script selection may be configured for each family. A reasonable[2] variety of opentype font features have key-value support.

*Many thanks to everyone who provided feedback on version 0.1. I'd particularly like to thank Max Chernoff who is primarily responsible for cleaning up the ConTeXt code, made many other suggestions and demonstrated remarkable patience in the face of my ineptitude.

†SVN: 12180 2026/11/02

[1] v0.1: '**extremely** basic'.

[2] Where 'reasonable' means I have some idea what they do and some, possibly merely theoretical, ability to test them. This is naturally a quite unreasonable definition of 'reasonable' with a clear Western European bias. Input from users of non-Latin scripts (initially for mono-script documents) would be particularly welcome.

Alternatively or additionally, feature specifications may be passed more-or-less directly to the font loader. This enables any loader-supported specification to used and is faster, but requires knowledge of the syntax and opentype tag names.

That said, the automated configuration is already quite extensive. By default, the package tries to configure font families which are as rich as possible, insofar as the fonts support this. This includes

- multiple widths and shapes, insofar as NFSS supports them;
- italic, oblique, upright italic, reverse italic, reverse oblique;
- small-caps, italic small-caps, oblique small-caps[3];
- optical sizes.

Of course, this depends on the quality of the data the fonts provide. It is not always good or, even, vaguely accurate. If things do not fit, some (currently minimal) effort is made to accommodate them as distinct font families, so they are still available at document level.

At present, this is still a bit chaotic and probably not that easy to use, as the results are 'hidden' from you (in, of course, plain sight). Even so, you should typically need to do much less to access the fonts you want than you might otherwise expect.

The advantage of this is that if a font has several widths, you can easily adjust the one used for bold by simply redefining the relevant default. This is not very user friendly yet, but it is now a bit better than not completely terrible.

Simple usage looks like this:

```
\usepackage{lua-font-config}
\fontconfig{%
  rm = texgyrepagella,
  sf = texgyreheros,
  tt = texgyrecursor,
  math = texgyrescholamath,
}
```

`\fontconfig` {⟨*key-val list*⟩}

Specify font configuration.

2.1 Keys

Top-level keys are described here. Keys for use in key-value lists are described in section 2.2.

`rm` (*key*) = ⟨*family name*⟩

`rm` (*key*) = {⟨*key-val list*⟩}

[3] This should work whether small-caps are in separate fonts (e.g. Latin Modern) or implemented by the smcp feature (e.g. TEX-Gyre Pagella).

Bug tracker: https://codeberg.org/cfr/nfssext/issues | Code: https://codeberg.org/cfr/nfssext | Mirror: https://github.com/cfr42/nfssext

Specify family to use as the main document font, typically a serif or 'roman' family.

Available keys are listed below.

`sf` (*key*) = ⟨*family name*⟩

`sf` (*key*) = {⟨*key-val list*⟩}

Specify family to use as the default sans family, typically a sans-serif.

Available keys are listed below.

`tt` (*key*) = ⟨*family name*⟩

`tt` (*key*) = {⟨*key-val list*⟩}

Specify family to use as the default 'typewriter' family, typically monowidth.

Available keys are listed below.

`family` (*key*) = ⟨*family name*⟩

`family` (*key*) = {⟨*key-val list*⟩}

Specify an additional text family. This will create a macro which can be used to switch to the requested family[4].

Available keys are listed below.

`math` (*key*) = ⟨*family name*⟩

`math` (*key*) = ⟨*key-val list*⟩

Specify the family to use for typesetting Unicode mathematics. This sets the fonts used for symbols, script and calligraphic alphabets. It does not change the fonts used to typeset operators, since these are taken from text fonts and configured separately.

Available keys are listed below.

This option uses lua-unicode-math. unicode-math is **not** supported. If you wish to use unicode-math, do not use this option. Instead, load unicode-math and use `\setmathfont` as usual.

`debug` (*key*) = {⟨*key-val list*⟩}

This can be used to enable debugging and/or modify the logging level. It is primarily intended for use in debugging the package and unlikely to be of use in production. However, it may be useful the use this key if reporting a bug which cannot be reproduced e.g. which is platform specific or depends on system fonts.

2.2 Keys for specifying font families

All keys described in section 2.1 accept either a simple family name, as shown in the example above, or a braced key-value list. For example,

[4] Usually you will want to specify which macro, since otherwise you must figure out the family name the code uses, which may or may not match the one requested. See section 2.2.4.

```
\usepackage{lua-font-config}
\fontconfig{%
  rm = {%
    name = texgyrepagella,
    fea  = {script=ltn;+tlig;+pnum},
  },
  sf = {%
    name  = texgyreheros,
    scale = 1.2,
    fea   = {script=cyrl;},
  },
  tt = {%
    name  = texgyrecursor,
    fea   = {+zero;},
    %% for completeness: false is anyway the default
    force = false,
  },
}
```

This section describes these 'inner' keys.

2.2.1 Common keys

These keys are accepted by all of `rm`, `sf`, `tt`, `family` and `math`.

name (*key*) = ⟨*lookup name*⟩

The 'name' to lookup. This may, but need not be, the name of a font family. Shorter, less specific names will configure more families; longer, more specific names will configure fewer.

For example, `name = texgyrepagellax`, will configure fewer than `name = texgyrepagella`, since the latter string matches both families, while the former does not.

The name is lower-cased prior to lookup. The data used to generate actual font families uses a cleaned up version of the given name or, sometimes, a variation, depending on the data returned from the database.

For example, `tex gyre PaGellA` yields the same results as `texgyrepagella`.

Numbers are likewise removed[5].

force (*key*) = `true`|`false`

Boolean which determines whether cached data is to be regenerated on the basis of a new search of the font-loader's database.

[5] This is problematic for fonts which depend on digits in their names to distinguish them e.g. Venturis ADF has four distinct 'titling' fonts, whose names differ by a single character, which is 1, 2, 3 or 4. The algorithm used to sort the fonts into NFSS families currently gets confused by this, leading to misleading warnings about 'apparent duplicate' fonts. (Something is indeed awry, but they are not duplicates. Rather what should be four families appears to be only one.)

Bug tracker: https://codeberg.org/cfr/nfssext/issues | Code: https://codeberg.org/cfr/nfssext | Mirror: https://github.com/cfr42/nfssext

Note that this doesn't regenerate the loader's database itself. It simply zaps data stored by this module for one or more font families.

`scale` (*key*) = ⟨*factor*⟩

Optional factor by which the font should be scaled e.g. `20` or `0.8`. Ignored if a family includes distinct optical sizes.

2.2.2 Text-mode keys

These keys are accepted in all key-value specifications except for the top-level `math` key used to configure Unicode maths. They are used to configure font 'features' of various sorts. These include opentype features provided by the fonts, but also some additions provided by the font-loader or in other ways.

There are currently two keys in this category. `fea` is a relatively low-level interface. It is fast, powerful and requires familiarity with opentype font features and the syntax supported by luaotfload. `features` is a mid-level interface which aims to provide a more user-friendly wrapper around `fea`.

If you are not sure which to use, use `features`.

The two keys may be used to configure the same family. If so, consistent options are combined; in case of conflicts, `features` takes precedence over `fea`. This means you can use the more-user friendly interface for most things and still add any unsupported specialist setup, if necessary.

`features` (*key*) = {⟨*key-value list*⟩}

Optional list of font features **for text fonts only**.

Note that this interface is *highly* experimental: it may, and very likely will, change[6].

The default is equivalent to `features={ script=dflt, mode=node, language=dflt, ligatures=tex }`. Unless provided in separate fonts, small-caps are enabled, if available, for upright, italic and oblique small-caps.

Features apply to *all* fonts in the family specified by `name`, so you should not specify features you do not wish applied to all of them.

Currently, nine keys are supported. Where multiple values are listed, any combination of values is permitted in any order, but **inconsistent values should be avoided**.

A ⟨*key list*⟩ requires a comma-separated list of keys *without values*:

{⟨*key1*⟩,⟨*key2*⟩,…}

[6] It is almost certain that it excludes things people need; it may also include things they don't. Moreover, the groups of options are somewhat arbitrary, as are the keys' names. It needs significant user input to determine what works best here, as fontspec's framework is quite different.

At some point, it also needs to be extended to permit per-font configuration e.g. so that fake bold can be applied to create an additional weight, such as the `bf` series for a family which lacks bold. And it should also be possible to specify different default weights and widths for the stand `md` and `bf` series e.g. to use light and demi-bold, rather than regular and bold. Right now, none of this is possible *via* the key-value interface, though the latter is straightforward to do for users familiar with the roles of macros such as `\mddefault` and `\bfdefault`, at least in simple cases.

Bug tracker: https://codeberg.org/cfr/nfssext/issues | Code: https://codeberg.org/cfr/nfssext | Mirror: https://github.com/cfr42/nfssext

A ⟨*key-value list*⟩ requires a comma-separated list of key-values: {⟨*key1*⟩=⟨*value1*⟩,⟨*key2*⟩=⟨*value2*⟩,…}

`alternates` (*key*) = {⟨*key/key-value list*⟩}

⟨*key/key-value list*⟩ should be a comma-separated list of one or more of the following valueless keys

`contextual`	enable standard contextual alternates
`nocontextual`	disable them
`contextualswash`	enable contextual 'swash'
`linefinal`	enable end-of-line forms
`wordfinal`	enable end-of-word forms
`justification`	enable forms designed to help justify text
`swash`	enable 'swash'

and/or one or more of the following key-value pairs

`annotation`	=⟨*integer*⟩	enable 'annotation' form ⟨*integer*⟩
`charactervariant`	=⟨*integer i*⟩:⟨*integer j*⟩	enable the *i*th character mapping, selecting the *j*th variant
`stylistic`	=⟨*integer*⟩	enable 'stylistic' form ⟨*integer*⟩
`sset`	=⟨*integer*⟩	enable the set of 'stylistic' forms ⟨*integer*⟩

`embolden` (*key*) = ⟨*factor*⟩

Add 'fake bold' to make the family heavier. Note that this applies to all fonts in the family; it is not intended to add fake bold to a family which lacks one.

`figures` (*key*) = {⟨*key list*⟩}

⟨*key list*⟩ should be a comma-separated list of one or more of the following valueless keys

`lining`	use lining digits
`osf|oldstyle`	use oldstyle / hanging digits
`prop|proportional`	use proportional digits
`tab|tabular`	use tabular digits
`zero`	use the slashed zero

`forms` (*key*) = {⟨*key list*⟩}

⟨*key list*⟩ should be a comma-separated list of one or more of the following valueless keys

`capstopetite`	map ABC to 'petite' caps
`capstosc`	map ABC to small-caps
`case`	enable forms designed for use in capitalised contexts
`nocase`	disable them
`hist|historical`	enable 'historical' forms of characters
`random`	characters vary pseudo-randomly
`smalltopetite`	map abc to 'petite' caps
`titling`	enable forms intended for use in titles
`unicase`	map ABCabc to letters in a single-cased alphabet

`language` (*key*) = ⟨*language id*⟩

⟨*language id*⟩ must currently be a valid identifier, as recognised by luaotfload. `otfinfo -l` ⟨*path to font file*⟩ may be used to determine identifiers supported by the font.

`ligatures` (*key*) = {⟨*key list*⟩}

⟨*key list*⟩ should be a comma-separated list of one or more of the following valueless keys

`contextual`	enable 'contextual' ligatures
`nocontextual`	disable them
`discretionary`	enable discretionary ligatures
`hist|historic`	enable historical ligatures
`required`	enable required ligatures
`norequired`	disable them
`standard`	enable standard ('common') ligatures
`nostandard`	disable them
`tex`	enable TEX ligatures e.g. `--`
`notex`	disable them

`mode` (*key*) = `base|node|harf|harfbuzz`

Set the rendering mode.

`script` (*key*) = ⟨*script id*⟩

⟨*script id*⟩ must currently be a valid identifier, as recognised by luaotfload. `otfinfo -s` ⟨*path to font file*⟩ may be used to determine identifiers supported by the font.

`spacing` (*key*) = {⟨*key list*⟩}

⟨*key list*⟩ should be a comma-separated list of one or more of the following valueless keys

`capsspacing`	enable spacing designed for capitalised text
`kern`	enable kerning
`nokern`	disable kerning

`fea` (*key*) = {⟨*opentype feature spec.*⟩}

Optional list of font features **for text fonts only**.

Note that the specification is passed to the font loader with minimal processing, so you must use 'raw' feature specifications. *If you do not know what these are, use the `features` key rather than this one; if you need an unsupported feature, file an issue on the bug tracker (link below).*

The default is equivalent to `fea={mode=node;scrpt=dflt;language=dflt;+tlig}`. Unless provided in separate fonts, `+smcp` is added, if available, for upright, italic and oblique small-caps.

Features apply to *all* fonts configured as a result of the lookup, so you should not specify features you do not wish applied to all fonts to which the specification applies.

2.2.3 Default text family keys

These keys are accepted by `rm`, `sf` and `tt` i.e. when specifying the default text families for the document.

These keys all do the same thing, so you may use your preferred syntactic sweetener.

`math` (*key*) `=true|false`

`maths` (*key*) Should the current specification also be used for the corresponding text family in maths?

For example, if you are configuring the main document family with `rm`, should that family also be used for operators, `\mathrm` etc.?

The default is equivalent to `math = true`.

`nomath` (*key*) Equivalent to `math = false`.

`nomaths` (*key*)

2.2.4 Additional text family keys

`command` (*key*) `=\command`

Use `\command` to switch to the font family, rather than creating a command from the family name.

By default the command to switch to the font family depends on the discovered family name i.e. on the result of searching the database for the specified family. In many cases, this will be the value passed to the `name` key, but often it will not and sometimes this may not be the command name you want anyway. `command` allows you to override it.

For example, the following would setup `\playfair` to switch to Playfair Display, rather than the default choice of `\playfairdisplay`.

```
\usepackage{lua-font-config}
\fontconfig{%
  family = {% setup an additional non-default family
    name = Playfair Display,
    command = \playfair,
  },
}
```

3 Compatibility

3.1 ltx-talk

Load with `\RequirePackage` *before* loading the class. The package emulates the class settings by default, so the main benefit of loading it is speed. If you wish to change the fonts, use `\fontconfig` after the class is loaded[7]

[7] If this turns out to be something people want to do, I can adjust things to avoid configuring fonts twice. For now, this is just a good way for me to test the code and avoid the pain of watching fontspec load New Computer Modern.

4 Examples

Figures 1 to 3 demonstrate simple use of the package to configure the main document font, a Unicode maths font (*via* lua-unicode-math and a complete font configuration (roman/serif, sans, typewriter and maths). These examples use the default configurations for text and maths fonts, so the configurations are extremely simple.

Figures 4 to 6 demonstrate slightly more complex setups.

Figure 7 demonstrates how to use lua-font-config to speed up the compilation of presentations created with ltx-talk.

example-lmr -- fig. 1

```
\documentclass{article}
\usepackage{lua-font-config}
\fontconfig{rm = latinmodernroman}
\begin{document}
This is Latin Modern Roman.
It features \textit{italics} and \textsl{oblique}.
\textsc{Small-caps are implemented as distinct fonts, as are
↪  \textsl{oblique small-caps}.}
{\fontshape{ui}\selectfont Upright italics are less commonly used.}

\normalfont\bfseries
This is the bold expanded weight.
It features \textit{italics} and \textsl{oblique}.

\normalfont\fontseries{db}\selectfont
The ‘demi-bold’ is of regular width, with \textsl{oblique}.

\normalfont
\Huge
Where optical sizes are available, these provide higher quality
↪  output than simple scaling.

\LARGE
Where optical sizes are available, these provide higher quality
↪  output than simple scaling.

\Large
Where optical sizes are available, these provide higher quality
↪  output than simple scaling.

\large
Where optical sizes are available, these provide higher quality
↪  output than simple scaling.

\normalsize
Where optical sizes are available, these provide higher quality
↪  output than simple scaling.
```

```
\small
Where optical sizes are available, these provide higher quality
↪  output than simple scaling.

\footnotesize
Where optical sizes are available, these provide higher quality
↪  output than simple scaling.

\scriptsize
Where optical sizes are available, these provide higher quality
↪  output than simple scaling.

\tiny
Where optical sizes are available, these provide higher quality
↪  output than simple scaling.

\normalsize
\footnotesize
This is footnote-sized Latin Modern Roman.
It features \textit{italics} and \textsl{oblique}.
\textsc{Small-caps are implemented as distinct fonts, as are
↪  \textsl{oblique small-caps}.}
{\fontshape{ui}\selectfont Upright italics is less commonly used.}

\normalfont\bfseries
This is the bold expanded weight.
It features \textit{italics} and \textsl{oblique}.

\normalfont\fontseries{db}\selectfont
The 'demi-bold' is of regular width, with upright and
↪  \textsl{oblique} shapes.

\normalfont
The output should look very much as if 'lua-font-config' had not
↪  been loaded!
\end{document}
```

example-ncm-math -- fig. 2

```
\documentclass{article}
\usepackage{lua-font-config}
\fontconfig{%
  rm = newcomputermodern,
  sf = newcomputermodernsans,
  tt = newcomputermodernmono,
  math = newcomputermodernmath,
}
\parindent=0pt
\begin{document}
\[
    \sin^2\theta + \cos^2\theta = 1
\]
```

This is Latin Modern Roman. It features *italics* and *oblique*. Small-caps are implemented as distinct fonts, as are *oblique small-caps*. Upright italics are less commonly used.
This is the bold expanded weight. It features *italics* and *oblique*.
The 'demi-bold' is of regular width, with *oblique*.

Where optical sizes are available, these provide higher quality output than simple scaling.

Where optical sizes are available, these provide higher quality output than simple scaling.

Where optical sizes are available, these provide higher quality output than simple scaling.

Where optical sizes are available, these provide higher quality output than simple scaling.

Where optical sizes are available, these provide higher quality output than simple scaling.

Where optical sizes are available, these provide higher quality output than simple scaling.

Where optical sizes are available, these provide higher quality output than simple scaling.

Where optical sizes are available, these provide higher quality output than simple scaling.

Where optical sizes are available, these provide higher quality output than simple scaling.

This is footnote-sized Latin Modern Roman. It features *italics* and *oblique*. Small-caps are implemented as distinct fonts, as are *oblique small-caps*. Upright italics is less commonly used.
This is the bold expanded weight. It features *italics* and *oblique*.
The 'demi-bold' is of regular width, with upright and *oblique* shapes.
The output should look very much as if 'lua-font-config' had not been loaded!

Figure 1: `example-lmr`

Bug tracker: https://codeberg.org/cfr/nfssext/issues | Code: https://codeberg.org/cfr/nfssext | Mirror: https://github.com/cfr42/nfssext

$$\sin^2\theta + \cos^2\theta = 1$$

xxx

$$\mathbf{\sin^2\theta + \cos^2\theta = 1}$$

xxx

Figure 2: example-ncm-math

```
$\mathsf{x} \mathtt{x} \mathbf{x}$

\mathversion{bold}
\[
    \sin^2\theta + \cos^2\theta = 1
\]
$\mathsf{x} \mathtt{x} \mathbf{x}$
\end{document}
```

example-ncm-complete -- fig. 3

```
\documentclass{article}
\usepackage{lua-font-config}
\fontconfig{%
  rm    = New Computer Modern,
  sf    = New Computer Modern Sans,
  tt    = New Computer Modern Mono,
  math  = New Computer Modern Math,
}
\begin{document}
New Computer \textsl{Modern} \textit{with} \textsc{\textit{default}
↪  settings}.

\sffamily
New Computer \textsl{Modern} \textit{with} \textsc{\textit{default}
↪  settings}.

\ttfamily
New Computer \textsl{Modern} \textit{with} \textsc{\textit{default}
↪  settings}.

\normalfont
New Computer Modern Math with text fonts as above:
\[
    \sin^2\theta + \cos^2\theta = 1
\]
\[
    \mathscr{ABC abc} \mathcal{ABC abc}
\]
\mathversion{bold}
```

New Computer *Modern with* DEFAULT SETTINGS.
New Computer *Modern with* DEFAULT SETTINGS.
New Computer *Modern with* DEFAULT SETTINGS.
New Computer Modern Math with text fonts as above:

$$\sin^2\theta + \cos^2\theta = 1$$

$$\mathscr{ABC}abc\mathcal{ABC}abc$$

$$\boldsymbol{\sin^2\theta + \cos^2\theta = 1}$$

$$\boldsymbol{\mathscr{ABC}abc\mathcal{ABC}abc}$$

Figure 3: example-ncm-complete

```
\[
    \sin^2\theta + \cos^2\theta = 1
\]
\[
    \mathscr{ABC abc} \mathcal{ABC abc}
\]
\end{document}
```

Note that figs. 4 and 5 give exactly the same output; the only difference is that one uses features and the other fea to configure features for the configured families. See section 2.2.2.

example-features -- fig. 4

```
\documentclass{article}
\parindent=0pt
\usepackage{lua-font-config}
\fontconfig{%
  rm = {% serif uses oldstyle, proportional figures
    name = TeX Gyre Pagella,
    features = {%
      figures = {oldstyle, proportional},
    },
  },
  sf = {% sans uses proportional lining figures
    name = texgyreheros,
    features = {%
      figures = {lining, proportional},
    },
  },
  tt = {% typewriter uses tabular lining figures and no ligatures
    name = texgyrecursor,
    features = {%
      figures = {lining, tabular},
      ligatures = {notex, nostandard},
    },
  },
```

```
}
\begin{document}
{%
  \TeX-Gyre Pagella: 0123456789
}

{%
  \sffamily
  \TeX-Gyre Heros: 0123456789
}

{%
  \ttfamily
  \TeX-Gyre Cursor 0123456789
  -- --- outfitted officials affluently affiliate fish finials
   ↪  flagranty, first
  finalising infinite floral affinities of fish-filled fingers
}

\[
    \sin\quad
    \mathrm{rm}\quad
    \mathsf{sf}\quad
    \mathtt{tt}\quad
    \mathit{it}\quad
    \mathbf{bf}
\]
\mathversion{bold}
\[
    \sin\quad
    \mathrm{rm}\quad
    \mathsf{sf}\quad
    \mathtt{tt}\quad
    \mathit{it}\quad
    \mathbf{bf}
\]
\end{document}
```

example-fea -- fig. 5

```
\documentclass{article}
\parindent=0pt
\usepackage{lua-font-config}
\fontconfig{%
  rm = {% serif uses oldstyle, proportional figures
    name = TeX Gyre Pagella,
    fea = {+ONUM;+PNUM},
  },
  sf = {% sans uses proportional lining figures
    name = texgyreheros,
    fea = {+pnum;+lnum},
  },
```

TEX-Gyre Pagella: 0123456789
TEX-Gyre Heros: 0123456789
TEX-Gyre Cursor 0123456789 -- --- outfitted officials affluently affiliate fish finials flagranty, first finalising infinite floral affinities of fish-filled fingers

sin rm sf tt *it* **bf**

sin rm sf tt *it* **bf**

Figure 4: example-features

```
  tt = {% typewriter uses tabular lining figures and no ligatures
    name = texgyrecursor,
    fea = {-TLIG;+tnum;+lnum;-liga},
  },
}
\begin{document}
{%
  \TeX-Gyre Pagella: 0123456789
}

{%
  \sffamily
  \TeX-Gyre Heros: 0123456789
}

{%
  \ttfamily
  \TeX-Gyre Cursor 0123456789
  -- --- outfitted officials affluently affiliate fish finials
   ↪  flagranty, first
  finalising infinite floral affinities of fish-filled fingers
}

\[
    \sin\quad
    \mathrm{rm}\quad
    \mathsf{sf}\quad
    \mathtt{tt}\quad
    \mathit{it}\quad
    \mathbf{bf}
\]
\mathversion{bold}
\[
    \sin\quad
    \mathrm{rm}\quad
    \mathsf{sf}\quad
    \mathtt{tt}\quad
```

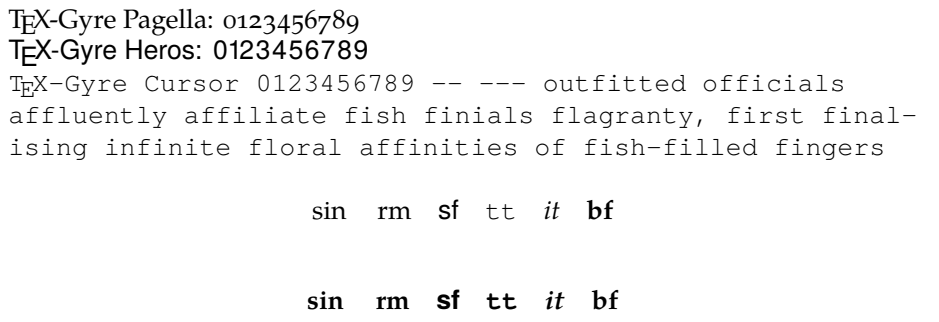


Figure 5: example-fea

```
    \mathit{it}\quad
    \mathbf{bf}
\]
\end{document}
```

Figure 6 demonstrates usage of the fam key to configure an additional family and a custom font switch to access it.

example-add -- fig. 6

```
\documentclass{article}
\usepackage{lua-font-config}
\fontconfig{%
  family  = {
    name    = Lobster Two,
    scale   = 2,
    command = \lobster,
  },
}
\begin{document}
This is some ordinary text in the default \LaTeXe{} font.

{\lobster This is Lobster Two.
  \textit{Italics but no \textsc{small-caps}} \textsc{even in
   ↪ upright}.
  We do have \textbf{bold and \textit{bold italics}}, but there is
   ↪ no,
  \textsl{oblique, so here we get italics, too.}\par
}

Back to the default.
\end{document}
```

Figure 7 demonstrates the default configuration for ltx-talk. This is approximately equivalent to the default setup if lua-font-config is not loaded. Note the way the package is loaded *before* \documentclass. If we used \usepackage after the class was loaded, we would not benefit from the faster compilation our Lua-based approach provides[8]. See section 3.1.

[8] In fact, although lua-font-config *is* faster generally and although this is likely due to greater use

This is some ordinary text in the default LaTeX 2ε font.

This is Lobster Two. *Italics but no* small-caps even in upright. We do have bold and *bold italics*, but there is no, *oblique, so here we get italics, too.*

Back to the default.

Figure 6: example-add

example-talk -- fig. 7

```
\DocumentMetadata{tagging=on,pdfstandard=ua-2,pdfversion=2.0,
↪  lang=en-GB}
\RequirePackage{lua-font-config}% MUST precede class.
\documentclass{ltx-talk}

\title{Some Title}
\author{A. U. Thor}
\institute{Pantheon, Greece}
\date{Ancient}
\begin{document}
\begin{frame}
  \maketitle
\end{frame}
\begin{frame}
  \frametitle{Frame}
  Some text\footnote{With a footnote.} on a slide.

  \textsc{Small-Caps \textsl{italics \textbf{bold}} \textbf{bold}.}

  Regular, \fontseries{k}\selectfont Book and \bfseries Bold.
  \normalfont
  \begin{equation}
    v = ut + \frac{1}{2}at^2
  \end{equation}
  \begin{equation}
    \sin^2\theta + \cos^2\theta = 1
  \end{equation}
  Script $\mathscr{ABC abc}$ and calligraphic $\mathcal{ABC abc}$.

  Bold maths:
  \mathversion{bold}
  \begin{equation}
    v = ut + \frac{1}{2}at^2
  \end{equation}
  \begin{equation}
```

of Lua, the size of the difference in this particular case suggests some additional factor is involved. However, I've not tried to confirm this or to identify the cause.

```
    \sin^2\theta + \cos^2\theta = 1
  \end{equation}
  Script $\mathscr{ABC abc}$ and calligraphic $\mathcal{ABC abc}$.
\end{frame}
\end{document}
```

5 TO DOs

These are not in any particular order, though there is some tendency for higher priority items to be earlier.

1. Conceptual
   - Abandon the whole thing since packages are not supposed to meddle with fonts?
2. NFSS
   - Support multiple font configurations e.g. for multiple languages with multiple scripts.
     - Support for babel? This!!!

       babel provides e.g. `\babelfont`, which uses `\fontspec`. Should lua-font-config provide this?
3. Interface
   - Provide a proper user interface (more similar to ConTEXt, since we're using the same (MKIV) sorting? The interface is now not too dissimilar to a LATEX-ified version of this, I suppose, except there are no goodies or typefaces and our families are a bit richer and more flexible.
   - Information for user - a bit better now, hopefully, as we use the standard mechanisms, except for debug messages.
   - Accommodate `.ttc`.
     - The basis is done, but not sure what, if any, interface is needed?
     - How to test this??
4. Loader
   - Test with `.ttf`.
     - What needs testing exactly?? Or is this all on the loader?
   - Variable fonts.
     - How to do or test??
   - Colour. Do??
     - What does 'Do??' refer to?
   - Cope with (text) symbol fonts? Not sure is that really needed?
   - Combo, fallback & all that.

Some Title

A. U. Thor
Pantheon, Greece
Ancient

Frame

Some text[1] on a slide.
SMALL-CAPS *ITALICS* ***BOLD*** **BOLD**.
Regular, Book and **Bold.**

$$v = ut + \frac{1}{2}at^2 \tag{1}$$

$$\sin^2\theta + \cos^2\theta = 1 \tag{2}$$

Script $\mathscr{ABCabc}$ and calligraphic $\mathcal{ABCabc}$.
Bold maths:

$$v = ut + \frac{1}{2}at^2 \tag{3}$$

$$\boldsymbol{\sin}^2\theta + \boldsymbol{\cos}^2\theta = 1 \tag{4}$$

Script $\mathscr{ABCabc}$ and calligraphic $\mathcal{ABCabc}$.

[1] With a footnote.

Figure 7: `example-talk`

Bug tracker: https://codeberg.org/cfr/nfssext/issues | Code: https://codeberg.org/cfr/nfssext | Mirror: https://github.com/cfr42/nfssext

5. Cache
    - Cached data should depend on db/fnt versions.
        - Or is this automatic?
        - Or maybe it shouldn't?
6. Documentation
    - Not everything implemented in terms of the UI is in the user doc.
    - The Lua code has some comments, but these don't always amount to documentation.
    - The overall infrastructure isn't really documented, except for a partially accurate ASCII database schema included inline.
    - Registered Lua functions mapped to expl3 names are not documented.
    - Use of callbacks, hooks etc. is more-or-less undocumented.
7. Code
    - Should be cleaned up – I cannot need 2,500 lines to load a font!

Unnumbered Gripe The code is too long, too complex, too clunky and too simplistic. (Yes, of course, it can be both.)

- Some remaining cleanup in the ConTeXt file, though this is much better than it was.
- Improve performance.
    - There is some duplication and too many pre-definitions for my taste.
- Possibly modify database creation to avoid correcting the data again??

8. Test suite
    - There are no unit tests (cf. nfssext-cfr, where the tests are much more focused). I don't know how to implement this for (non-public) Lua functions.

6 Implementation

There is very little here, because almost everything is done in Lua.

<*sty> <@@=lfc>

```
\NeedsTeXFormat{LaTeX2e}[2021-11-15]
\RequirePackage{svn-prov}
\ProvidesPackageSVN[\filebase.sty]{$Id: lua-font-config.dtx 12180 2026-
 11-02 02:55:44Z cfrees $}[v0.2 \revinfo][\filebase: Font configuration
 for LuaLaTeX]
% This is needed only because l3doc overwrites \fileversion and \filedate.
% Neither ltxdoc nor doc does this.
```


Bug tracker: https://codeberg.org/cfr/nfssext/issues | Code: https://codeberg.org/cfr/nfssext | Mirror: https://github.com/cfr42/nfssext

```
% But it seems better to do it this way on the basis of the general precautionary principle.
\DefineFileInfoSVN[lfc]

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% copied verbatim, excepting format from Joseph Wright's' siunitx.sty under LPPL
\@ifundefined{directlua}{%
  \PackageError{lua-font-config}{Package requires LuaLaTeX}
  {To use lua-font-config, you must compile your document with LuaLaTeX.}%
}{}
\@ifundefined{ExplLoaderFileDate}{%
  \RequirePackage{expl3}%
}{}
%% almost verbatim from siunitx.sty
\@ifl@t@r\ExplLoaderFileDate{2022-02-24}{%
}{%
  \PackageError{lua-font-config}{Support package expl3 too old}
  {%
    You need to update your installation of the bundles 'l3kernel' and
    'l3packages'.\MessageBreak
    Loading lua-font-config will abort!%
  }%
  \endinput
}
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
\ExplSyntaxOn
\prop_gput:NnV \g_msg_module_name_prop {lfc} \lfc@base
```

Messages.

Note that \msg_* are not recognised in the indexing; \msg appears instead. This is a limitation of the doc package, as far as I know.

```
\msg_new:nnnn {lfc} {needs-luatex} {
  \msg_critical_text:n {lfc} :: ~requires~LuaLaTeX~\msg_line_context:
} { \msg_see_documentation_text:n {lfc} }

\sys_if_engine_luatex:F { \msg_critical:nn {lfc} {needs-luatex} }

\msg_new:nnnn {lfc} {math-needs-lum} {
  \msg_error_text:n {lfc} :: ~math~config~requires~lua-unicode-math,~but~
  you've~loaded~unicode-math.~This~cannot~work~\msg_line_context:
} { To~configure~Unicode~maths~fonts~with~this~package,~we~need~to~load~
  lua-unicode-math,~but~unicode-math,~which~is~incompatible~has~already~
  been~loaded,~so~I'm~going~to~ignore~this~part~of~your~settings.~
  \msg_see_documentation_text:n {lfc}
}
\msg_new:nnnn {lfc} {loading} {
  \msg_info_text:n {lfc} :: ~loading~#1~for~maths~support~
  \msg_line_context:
} { \msg_see_documentation_text:n {lfc} }
```


Bug tracker: https://codeberg.org/cfr/nfssext/issues | Code: https://codeberg.org/cfr/nfssext | Mirror: https://github.com/cfr42/nfssext

\l__lfc_features_seq (*var.*)
\l__lfc_tmpa_bool (*var.*)
\l__lfc_tmpa_tl (*var.*)
\l__lfc_tmpb_tl (*var.*)

Custom ephemeral variables.

```
\bool_new:N \l__lfc_tmpa_bool
\seq_new:N \l__lfc_features_seq
\tl_new:N \l__lfc_tmpa_tl
\tl_new:N \l__lfc_tmpb_tl
```

rm (*key*)
sf (*key*)
tt (*key*)

Top-level keys for default text families.

```
\clist_map_inline:nn {rm, sf, tt} {
  \keys_define:nn {lfc} {
    #1 .code:n            = {
      \tl_clear:N \l__lfc_tmpa_tl
      \tl_clear:N \l__lfc_tmpb_tl
    \keys_set_exclude_groups:nnnN {lfc/#1} {config} {##1} \l__lfc_tmpa_tl
         \keys_set_groups:nnVN  {lfc/#1}  {config}       \l__lfc_tmpa_tl
  \l__lfc_tmpb_tl
      \keys_set_groups:nnV  {lfc/#1} {lateconfig} \l__lfc_tmpb_tl
      \bool_set_true:N \l__lfc_tmpa_bool
    },
    #1 .value_required:n  = true,
    #1 .usage:n           = preamble,
  }
```

rm/name (*key*)
rm/fea (*key*)
rm/force (*key*)
rm/scale (*key*)
rm/maths (*key*)
rm/math (*key*)
rm/nomaths (*key*)
rm/nomath (*key*)
sf/name (*key*)
sf/fea (*key*)
sf/force (*key*)
sf/scale (*key*)
sf/maths (*key*)
sf/math (*key*)
sf/nomaths (*key*)
sf/nomath (*key*)
tt/name (*key*)
tt/fea (*key*)
tt/force (*key*)
tt/scale (*key*)
tt/maths (*key*)
tt/math (*key*)
tt/nomaths (*key*)
tt/nomath (*key*)

Configure keys for default text families.

```
  \keys_define:nn {lfc/#1} {
    name      .code:n = { \use:c {__lfc_set_#1:n} {##1} },
    fea       .code:n = { \keys_set:nn {lfc/fam} {fea      = {##1}} },
    features  .code:n = { \keys_set:nn {lfc/fam} {features = {##1}} },
    force     .code:n = { \keys_set:nn {lfc/fam} {force    = {##1}} },
    scale     .code:n = { \keys_set:nn {lfc/fam} {scale    = {##1}} },
    maths     .code:n = { \__lfc_set_fam_cfg:nn   {maths}{##1} },
    math      .meta:n = { maths = {##1} },
    nomaths   .meta:n = { maths = false },
    nomath    .meta:n = { nomaths },
    unknown   .code:n = {
      \exp_args:NnV \use:c {__lfc_set_#1:n} \l_keys_key_str
    },
    name      .value_required:n   = true,
    math      .default:n          = true,
    maths     .default:n          = true,
    nomath    .value_forbidden:n  = true,
    nomaths   .value_forbidden:n  = true,
    unknown   .value_forbidden:n  = true,
    fea       .groups:n           = config,
    features  .groups:n           = lateconfig,
    force     .groups:n           = config,
    maths     .groups:n           = config,
    math      .groups:n           = config,
    nomath    .groups:n           = config,
    nomaths   .groups:n           = config,
    scale     .groups:n           = config,
  }
```


Bug tracker: https://codeberg.org/cfr/nfssext/issues | Code: https://codeberg.org/cfr/nfssext | Mirror: https://github.com/cfr42/nfssext

```
}
```

debug (*key*)
math (*key*)
family (*key*)

Top-level keys for additional families, Unicode maths and debugging.

```
\keys_define:nn {lfc} {
  debug   .code:n           = {
    \tl_clear:N \l__lfc_tmpa_tl
    \keys_set_groups:nnnN
      {lfc/debug} {toggle} {state = true, #1} \l__lfc_tmpa_tl
    \keys_set:nV {lfc/debug} \l__lfc_tmpa_tl
  },
  debug   .usage:n          = general,
  family  .code:n           = {
    \tl_clear:N \l__lfc_tmpa_tl
    \tl_clear:N \l__lfc_tmpb_tl
    \keys_set_exclude_groups:nnnN
                      {lfc/fam} {config,lateconfig} {#1} \l__lfc_tmpa_tl
      \keys_set_groups:nnVN  {lfc/fam}  {config}         \l__lfc_tmpa_tl
  \l__lfc_tmpb_tl
    \keys_set_groups:nnV  {lfc/fam} {lateconfig}  \l__lfc_tmpb_tl
    \bool_set_true:N \l__lfc_tmpa_bool
  },
  family  .value_required:n = true,
  family  .usage:n          = general,
  math    .code:n           = {
    \IfPackageLoadedTF {unicode-math} {
      \msg_error:nn {lfc} {math-needs-lum}
    } {
      \msg_info:nnn {lfc} {loading} {lua-unicode-math}
      \RequirePackage{lua-unicode-math}
      \tl_clear:N \l__lfc_tmpa_tl
    \keys_set_exclude_groups:nnnN {lfc/math} {config} {#1} \l__lfc_tmpa_tl
      \keys_set:nV {lfc/math} \l__lfc_tmpa_tl
      \bool_set_true:N \l__lfc_tmpa_bool
    }
  },
  math    .value_required:n = true,
  math    .usage:n          = preamble,
}
```

fam (*key*)

Configure keys for families.

fam/name (*key*)
fam/fea (*key*)
fam/features (*key*)
fam/force (*key*)
fam/scale (*key*)

Configure keys for families.

```
\keys_define:nn {lfc/fam} {
  name      .code:n = { \__lfc_set_fam_name:n {#1} },
  command   .code:n = {
    \exp_args:Nne \__lfc_set_fam_cfg:nn {csname}{ \cs_to_str:N #1 }
  },
  fea       .code:n = { \__lfc_set_fam_cfg:nn {fea}{#1} },
  features  .code:n = {
    \seq_clear:N \l__lfc_features_seq
    \keys_set:nn {lfc/features} {#1}
    \seq_if_empty:NF \l__lfc_features_seq {
```


Bug tracker: https://codeberg.org/cfr/nfssext/issues | Code: https://codeberg.org/cfr/nfssext | Mirror: https://github.com/cfr42/nfssext

```
      \exp_args:Nne \__lfc_set_fam_cfg:nn {fea} {
        \seq_use:Nn \l__lfc_features_seq {;}
      }
    }
  },
  force     .code:n = { \__lfc_set_fam_cfg:nn {force}{#1} },
  scale     .code:n = { \__lfc_set_fam_cfg:nn {scale}{#1} },
  unknown   .code:n = {
    \exp_args:NV \__lfc_set_fam_name:n \l_keys_key_str
  },
  name      .value_required:n   = true,
  fea       .value_required:n   = true,
  features  .value_required:n   = true,
  force     .default:n          = true,
  scale     .value_required:n   = true,
  unknown   .value_forbidden:n  = true,
  name      .usage:n            = general,
  command   .usage:n            = general,
  fea       .usage:n            = general,
  features  .usage:n            = general,
  force     .usage:n            = general,
  scale     .usage:n            = general,
  unknown   .usage:n            = general,
  command   .groups:n           = config,
  fea       .groups:n           = config,
  features  .groups:n           = lateconfig,
  force     .groups:n           = config,
  scale     .groups:n           = config,
}
```

math/name (*key*)
math/force (*key*)
math/scale (*key*)

Configure keys for Unicode maths.

```
\IfPackageLoadedF {unicode-math} {
  \keys_define:nn {lfc/math} {
    name      .code:n = { \use:c {__lfc_set_math:n} {#1} },
    force     .code:n = { \keys_set:nn {lfc/fam} {force    = {#1}} },
    scale     .code:n = { \keys_set:nn {lfc/fam} {scale    = {#1}} },
    unknown   .code:n = {
      \exp_args:NnV \use:c {__lfc_set_math:n} \l_keys_key_str
    },
    name      .value_required:n   = true,
    force     .default:n          = true,
    scale     .value_required:n   = true,
    unknown   .value_forbidden:n  = true,
    name      .usage:n            = preamble,
    force     .usage:n            = preamble,
    scale     .usage:n            = preamble,
    unknown   .usage:n            = preamble,
    force     .groups:n           = config,
    scale     .groups:n           = config,
  }
}
```

features (*key*) Trying to be a more user-friendly wrapper around `fam/fea`.

features/alternates (*key*)

features/embolden (*key*)

features/figures (*key*)

features/forms (*key*)

features/language (*key*)

features/ligatures (*key*)

features/mode (*key*)

features/script (*key*)

features/spacing (*key*)

```
\keys_define:nn {lfc/features} {
  alternates  .code:n = {
    \__lfc_set_fam_features:nn {alternates} {#1}
  },
  embolden    .code:n = {
    %% \__lfc_set_fam_features:nn {}{embolden=#1}
    \seq_put_right:Nn \l__lfc_features_seq {embolden=#1}
  },
  figures     .code:n = {
    \__lfc_set_fam_features:nn {figures} {#1}
  },
  forms         .code:n = {
    \__lfc_set_fam_features:nn {forms} {#1}
  },
  language      .code:n = {
    %% \__lfc_set_fam_features:nn {}{language=#1}
    \seq_put_right:Nn \l__lfc_features_seq {language=#1}
  },
  ligatures     .code:n = {
    \__lfc_set_fam_features:nn {ligatures} {#1}
  },
  mode          .choices:nn = {base,node,harf}{
    %% \__lfc_set_fam_features:nn {mode}{#1}
    \seq_put_right:Ne \l__lfc_features_seq {mode=\l_keys_choice_str}
  },
  script        .code:n = {
    %% \__lfc_set_fam_features:nn {}{script=#1}
    \seq_put_right:Nn \l__lfc_features_seq {script=#1}
  },
  spacing       .code:n = {
    \__lfc_set_fam_features:nn {spacings} {#1}
  },
  alternates  .value_required:n = true,
  embolden    .value_required:n = true,
  figures     .value_required:n = true,
  forms       .value_required:n = true,
  language    .value_required:n = true,
  ligatures   .value_required:n = true,
  mode        .value_required:n = true,
  script      .value_required:n = true,
  spacing     .value_required:n = true,
}
```

debug/state (*key*) Configure keys for debugging.

debug/categories (*key*)

```
\keys_define:nn {lfc/debug} {
  categories  .code:n          = { \__lfc_debug_set_cats_scanner:n {#1} },
  state       .choice:,
  state/false .code:n              = { \__lfc_debug_set_false: },
  state/true  .code:n              = { \__lfc_debug_set_true: },
  categories  .usage:n             = general,
  state       .usage:n             = general,
```

```
  categories  .value_required:n   = true,
  state       .default:n          = true,
  state       .groups:n           = toggle,
}
```

`\fontconfig` The main (and currently(?) only) public macro.

```
\NewDocumentCommand\fontconfig { m }{
  \bool_set_false:N \l__lfc_tmpa_bool
  \keys_set:nn {lfc} {#1}
  \bool_if:NT \l__lfc_tmpa_bool {
    \__lfc_configure_doc_families:
  }
}
```

`\__lfc_set_mathrm:` (*fn.*) `\__lfc_set_mathsf:` (*fn.*) `\__lfc_set_mathtt:` (*fn.*) TODO: move to hooks if module gets support for switching the families out e.g. for language changes. Currently, the code is too simple-minded for this to arise ....

```
\cs_new_protected_nopar:Npn \__lfc_set_mathrm: {
    \SetMathAlphabet  \mathrm  {normal}  {TU}  \rmdefault  {\mdseries@rm}
  \shapedefault
  \SetMathAlphabet \mathit {normal} {TU} \rmdefault {\mdseries@rm} \it-
  default
    \SetMathAlphabet  \mathbf  {normal}  {TU}  \rmdefault  {\bfseries@rm}
  \shapedefault
    \SetMathAlphabet  \mathrm  {bold}    {TU}  \rmdefault  {\bfseries@rm}
  \shapedefault
  \SetMathAlphabet \mathit {bold}   {TU} \rmdefault {\bfseries@rm} \it-
  default
    \SetMathAlphabet  \mathbf  {bold}    {TU}  \rmdefault  {\bfseries@rm}
  \shapedefault
 \SetSymbolFont{operators}{normal} {TU} \rmdefault {\mdseries@rm} \shapedefault
    \SetSymbolFont{operators}{bold}     {TU}  \rmdefault  {\bfseries@rm}
  \shapedefault
}
\cs_new_protected_nopar:Npn \__lfc_set_mathsf: {
    \SetMathAlphabet  \mathsf  {normal}  {TU}  \sfdefault  {\mdseries@sf}
  \shapedefault
    \SetMathAlphabet  \mathsf  {bold}    {TU}  \sfdefault  {\bfseries@sf}
  \shapedefault
}
\cs_new_protected_nopar:Npn \__lfc_set_mathtt: {
    \SetMathAlphabet  \mathtt  {normal}  {TU}  \ttdefault  {\mdseries@tt}
  \shapedefault
    \SetMathAlphabet  \mathtt  {bold}    {TU}  \ttdefault  {\bfseries@tt}
  \shapedefault
}
```

`eclare_lum_symbol_fonts:nn` (*fn.*) Maths setup function.

```
\cs_new_protected_nopar:Npn \__lfc_declare_lum_symbol_fonts:nn #1#2 {
  \DeclareSymbolFont{lummain}{TU}{#1}{m}{n}
```

```
  \SetSymbolFont{lummain}{bold}{TU}{#2}{b}{n}
}
```

Compatibility code for ltx-talk.

This should be equivalent to the default setup, modulo any differences between emulating the package and using `\setmathfont` for lua-unicode-math.

```
\hook_gput_code:nnn {class/ltx-talk/before} {.} {
  \hook_gput_code:nnn {package/mathtools/after} {.} {
    \RequirePackage{lua-unicode-math}
    \fontconfig{
      rm      = {newcomputermodernsans},
      sf      = {newcomputermodernsans},
      math    = {newcomputermodernsansmath},
    }
    \tagpdfsetup { math / mathml / luamml / load = true }
  }
}
```

The Lua modules do the actual work.

```
\lua_load_module:n {lua-font-config}
\ExplSyntaxOff
```

</sty>

Change History

v0.1

General: First public release. 20

v0.2

General: Bug fixes, code cleanup, somewhat better UI, doc improvements. 20

`fam`: Add `fea` user-interface wrapper: `features` provides a key-value interface for configuration. 23

`family`: Adjusted to support addition of `features` interface. 23

`features`: Initial `features` keys: `alternates`, `embolden`, `figures`, `forms`, `language`, `ligatures`, `node`, `script`, `spacing`. 24

`\fontconfig`: Forbid `\par` in input. 26

`\l__lfc_tmpb_tl`: Added `\l__lfc_features_seq`, `\l__lfc_tmpb_tl`. 22

Index

Numbers written in italic refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in roman refer to the code lines where the entry is used.

Symbols

`\__lfc_configure_doc_families:` (fn.) 244, 280

`\__lfc_debug_set_cats_scanner:n` (fn.) 230, 280

`\__lfc_debug_set_false:` (fn.) 232, 280

`\__lfc_debug_set_true:` (fn.) . 233, 280

`\__lfc_declare_lum_symbol_fonts:nn`

`\__lfc_set_fam_cfg:nn` (*fn.*)

`\__lfc_set_fam_features:nn` (*fn.*)

`\__lfc_set_fam_name:n` (*fn.*)

`\__lfc_set_mathrm:` (*fn.*)

`\__lfc_set_mathsf:` (*fn.*)

`\__lfc_set_mathtt:` (*fn.*)

Bug tracker: https://codeberg.org/cfr/nfssext/issues | Code: https://codeberg.org/cfr/nfssext | Mirror: https://github.com/cfr42/nfssext

(fn.) 265
__lfc_set_fam_cfg:nn (fn.) 71, 131, 133, 138, 143, 144, 280
__lfc_set_fam_features:nn (fn.) 189, 192, 196, 199, 202, 206, 209, 213, 217, 280
__lfc_set_fam_name:n (fn.) 129, 146, 280
__lfc_set_mathrm: (fn.) 247, 280
__lfc_set_mathsf: (fn.) 247, 280
__lfc_set_mathtt: (fn.) 247, 280

A

alternates (key) 6

B

\bfseries@rm . . 250, 251, 252, 253, 255
\bfseries@sf 259
\bfseries@tt 263
\bool_if:NT 243
\bool_new:N 48
\bool_set_false:N 241
\bool_set_true:N 60, 109, 122

C

\clist_map_inline:nn 52
command (key) 8
\cs_new_protected_nopar:Npn 247, 257, 261, 265
\cs_to_str:N 131

D

debug (key) 3, 94
debug/categories (key) 229
debug/state (key) 229
\DeclareSymbolFont 266

E

embolden (key) 6
\exp_args:Nne 131, 138
\exp_args:NnV 76, 173
\exp_args:NV 146

F

fam (key) 128
fam/fea (key) 128
fam/features (key) 128
fam/force (key) 128
fam/name (key) 128
fam/scale (key) 128
family (key) 3, 3, 94
fea (key) 7
features (key) 5, 187
features/alternates (key) 187
features/embolden (key) 187
features/figures (key) 187
features/forms (key) 187
features/language (key) 187
features/ligatures (key) 187
features/mode (key) 187
features/script (key) 187
features/spacing (key) 187
figures (key) 6
\fontconfig 2, 240, 272
force (key) 4
forms (key) 6
functions:
  __lfc_configure_doc_families: 244, 280
  __lfc_debug_set_cats_scanner:n 230, 280
  __lfc_debug_set_false: . 232, 280
  __lfc_debug_set_true: . . 233, 280
  __lfc_declare_lum_symbol_fonts:nn 265
  __lfc_set_fam_cfg:nn 71, 131, 133, 138, 143, 144, 280
  __lfc_set_fam_features:nn 189, 192, 196, 199, 202, 206, 209, 213, 217, 280
  __lfc_set_fam_name:n 129, 146, 280
  __lfc_set_mathrm: 247, 280
  __lfc_set_mathsf: 247, 280
  __lfc_set_mathtt: 247, 280

G

\g_msg_module_name_prop 31

H

\hook_gput_code:nnn 269, 270

I

\IfPackageLoadedF 167
\IfPackageLoadedTF 114
\itdefault 249, 252

K

keys:

  alternates 6
  command 8
  debug 3, 94
  debug/categories 229
  debug/state 229
  embolden 6
  fam 128
  fam/fea 128
  fam/features 128
  fam/force 128
  fam/name 128
  fam/scale 128

Bug tracker: https://codeberg.org/cfr/nfssext/issues | Code: https://codeberg.org/cfr/nfssext | Mirror: https://github.com/cfr42/nfssext

family 3, 3, 94
fea 7
features 5, 187
features/alternates 187
features/embolden 187
features/figures 187
features/forms 187
features/language 187
features/ligatures 187
features/mode 187
features/script 187
features/spacing 187
figures 6
force 4
forms 6
language 6
ligatures 7
math 3, 3, 8, 94
math/force 167
math/name 167
math/scale 167
maths 8
mode 7
name 4
nomath 8
nomaths 8
rm 2, 2, 52
rm/fea 65
rm/force 65
rm/math 65
rm/maths 65
rm/name 65
rm/nomath 65
rm/nomaths 65
rm/scale 65
scale 5
script 7
sf 3, 3, 52
sf/fea 65
sf/force 65
sf/math 65
sf/maths 65
sf/name 65
sf/nomath 65
sf/nomaths 65
sf/scale 65
spacing 7
tt 3, 3, 52
tt/fea 65
tt/force 65
tt/math 65
tt/maths 65
tt/name 65
tt/nomath 65
tt/nomaths 65
tt/scale 65
\keys_define:nn 53, 65, 94, 128, 168, 187, 229
\keys_set:nn 67, 68, 69, 70, 136, 170, 171, 242
\keys_set:nV 99, 121
\keys_set_exclude_groups:nnnN 57, 105, 120
\keys_set_groups:nnnN 97
\keys_set_groups:nnV 59, 108
\keys_set_groups:nnVN 58, 107

L

\l__lfc_features_seq (var.) . . . 48, 135, 137, 139, 193, 203, 210, 214
\l__lfc_tmpa_bool (var.) 48, 60, 109, 122, 241, 243
\l__lfc_tmpa_tl (var.) 48, 55, 57, 58, 96, 98, 99, 103, 106, 107, 119, 120, 121
\l__lfc_tmpb_tl (var.) 48, 56, 58, 59, 104, 107, 108
\l_keys_choice_str 210
\l_keys_key_str 76, 146, 173
language (key) 6
\lfc@base 31
ligatures (key) 7
\lua_load_module:n 280

M

math (key) 3, 3, 8, 94
math/force (key) 167
math/name (key) 167
math/scale (key) 167
\mathbf 250, 253
\mathit 249, 252
\mathrm 248, 251
maths (key) 8
\mathsf 258, 259
\mathtt 262, 263
\mdseries@rm 248, 249, 254
\mdseries@sf 258
\mdseries@tt 262
mode (key) 7
\msg_critical:nn 35
\msg_error:nn 115
\msg_info:nnn 117

N

name (key) 4
\NewDocumentCommand 240
nomath (key) 8
nomaths (key) 8

Bug tracker: https://codeberg.org/cfr/nfssext/issues | Code: https://codeberg.org/cfr/nfssext | Mirror: https://github.com/cfr42/nfssext

P

\prop_gput:NnV 31

R

rm (key) *2*, *2*, 52
rm/fea (key) 65
rm/force (key) 65
rm/math (key) 65
rm/maths (key) 65
rm/name (key) 65
rm/nomath (key) 65
rm/nomaths (key) 65
rm/scale (key) 65
\rmdefault 248, 249, 250, 251, 252, 253, 254, 255

S

scale (key) *5*
script (key) *7*
\seq_clear:N 135
\seq_if_empty:NF 137
\seq_new:N 49
\seq_put_right:Ne 210
\seq_put_right:Nn 193, 203, 214
\seq_use:Nn 139
\SetMathAlphabet ... 248, 249, 250, 251, 252, 253, 258, 259, 262, 263
\SetSymbolFont 254, 255, 267
sf (key) *3*, *3*, 52
sf/fea (key) 65
sf/force (key) 65
sf/math (key) 65
sf/maths (key) 65
sf/name (key) 65
sf/nomath (key) 65
sf/nomaths (key) 65
sf/scale (key) 65
\sfdefault 258, 259
\shapedefault 248, 250, 251, 253, 254, 255, 258, 259, 262, 263
spacing (key) *7*
\sys_if_engine_luatex:F 35

T

\tagpdfsetup 277
\tl_clear:N .. 55, 56, 96, 103, 104, 119
\tl_new:N 50, 51
tt (key) *3*, *3*, 52
tt/fea (key) 65
tt/force (key) 65
tt/math (key) 65
tt/maths (key) 65
tt/name (key) 65
tt/nomath (key) 65
tt/nomaths (key) 65
tt/scale (key) 65
\ttdefault 262, 263

U

\use:c 66, 76, 169, 173

V

variables:
  \l__lfc_features_seq 48, 135, 137, 139, 193, 203, 210, 214
  \l__lfc_tmpa_bool 48, 60, 109, 122, 241, 243
  \l__lfc_tmpa_tl 48, 55, 57, 58, 96, 98, 99, 103, 106, 107, 119, 120, 121
  \l__lfc_tmpb_tl 48, 56, 58, 59, 104, 107, 108