luaotfload-tool

generate and query the Luaotfload font names database

Date: 2020-02-02
Copyright: GPL v2.0
Version: 3.12
Manual section: 1
Manual group: text processing

SYNOPSIS

luaotfload-tool [-bcDfFiIlLnpqRSuvVhw]

luaotfload-tool –update [–force] [–quiet] [–verbose] [–prefer-texmf] [–dry-run] [–formats=[+|-]EXTENSIONS] [–no-compress] [–no-strip] [–local] [–max-fonts=N]

luaotfload-tool –find=FONTNAME [–fuzzy] [–info] [–inspect] [–no-reload]

luaotfload-tool –flush-lookups
luaotfload-tool –cache=DIRECTIVE
luaotfload-tool –list=CRITERION[:VALUE] [–fields=F1,F2,…,Fn]
luaotfload-tool –bisect=DIRECTIVE
luaotfload-tool –help
luaotfload-tool –version
luaotfload-tool –show-blacklist
luaotfload-tool –diagnose=CHECK
luaotfload-tool –conf=FILE –dumpconf

DESCRIPTION

luaotfload-tool accesses the font names database that is required by the *Luaotfload* package. There are two general modes: **update** and **query**.

- **update**: update the database or rebuild it entirely;
- **query**: resolve a font name or display close matches.

OPTIONS

update mode

–update, -u Update the database; indexes new fonts.

–force, -f Force rebuilding of the database; re-indexes all fonts.

–local, -L Include font files in `$PWD`. This option will cause large parts of the database to be rebuilt. Thus it is quite inefficient. Additionally, if local font files are found, the database is prevented from being saved to disk, so the local fonts need to be parsed with every invocation of `luaotfload-tool`.

–no-reload, -n Suppress auto-updates to the database (e.g. when `--find` is passed an unknown name).

–no-compress, -c Do not filter the plain text version of the font index through gzip. Useful for debugging if your editor is built without zlib.

–prefer-texmf, -p Organize the file name database in a way so that it prefer fonts in the *TEXMF* tree over system fonts if they are installed in both.

–formats=EXTENSIONS Extensions of the font files to index. Where *EXTENSIONS* is a comma-separated list of supported file extensions (otf, ttf, ttc). If the list is prefixed with a + sign, the given list is added to the currently active one; - subtracts. Default: *otf,ttf,ttc*. Examples:

1) `--formats=-ttc,ttf` would skip TrueType fonts and font collections;
2) `--formats=otf` would scan only OpenType files;
3) `--formats=+afm` includes binary Postscript files accompanied by an AFM file.

query mode

–find=NAME Resolve a font name; this looks up <name> in the database and prints the file name it is mapped to. `--find` also understands request syntax, i.e. `--find=file:foo.otf` checks whether `foo.otf` is indexed.

–fuzzy, -F Show approximate matches to the file name if the lookup was unsuccessful (requires `--find`).

–info, -i Display basic information to a resolved font file (requires `--find`).

–inspect, -I Display detailed information by loading the font and analyzing the font table; very slow! For the meaning of the returned fields see the LuaTeX documentation. (requires `--find`).

–list=CRITERION Show entries, where *CRITERION* is one of the following:

1) the character `*`, selecting all entries;
2) a field of a database entry, for instance *version* or *format**, according to which the output will be sorted. Information in an unstripped database (see the option `--no-strip` above) is nested: Subfields of a record can be addressed using the `->` separator, e. g. `file->location`, `style->units_per_em`, or `names->sanitized->english->prefmodifiers`. NB: shell syntax requires that arguments containing `->` be properly quoted!
3) an expression of the form `field:value` to limit the output to entries whose `field` matches `value`.

For example, in order to output file names and corresponding versions, sorted by the font format:

```
./luaotfload-tool.lua --list="format" --fields="file->base,version"
```

This prints:

```
otf latinmodern-math.otf  Version 1.958
otf lmromancaps10-oblique.otf 2.004
otf lmmono8-regular.otf 2.004
otf lmmonoproplt10-bold.otf 2.004
otf lmsans10-oblique.otf  2.004
otf lmromanslant8-regular.otf 2.004
otf lmroman12-italic.otf  2.004
otf lmsansdemicond10-oblique.otf  2.004
...
```

–fields=FIELDS Comma-separated list of fields that should be printed. Information in an unstripped database (see the option `--no-strip` above) is nested: Subfields of a record can be addressed using the `->` separator, e. g. `file->location`, `style->units_per_em`, or `names->sanitized->english->subfamily`. The default is plainname,version*. (Only meaningful with `--list`.)

font and lookup caches

–flush-lookups Clear font name lookup cache (experimental).

–cache=DIRECTIVE Cache control, where *DIRECTIVE* is one of the following:

1) `purge` -> delete Lua files from cache;
2) `erase` -> delete Lua and Luc files from cache;
3) `show` -> print stats.

debugging methods

–show-blacklist, -b Show blacklisted files (not directories).

–dry-run, -D Don't load fonts when updating the database; scan directories only. (For debugging file system related issues.)

–no-strip Do not strip redundant information after building the database. Warning: this will inflate the index to about two to three times the normal size.

–max-fonts=N Process at most *N* font files, including fonts already indexed in the count.

–bisect=DIRECTIVE Bisection of the font database. This mode is intended as assistance in debugging the Luatex engine, especially when tracking memleaks or buggy fonts.

DIRECTIVE can be one of the following:

1) `run` -> Make `luaotfload-tool` respect the bisection progress when running. Combined with `--update` and possibly `--force` this will only process the files from the start up until the pivot and ignore the rest.
2) `start` -> Start bisection: create a bisection state file and initialize the low, high, and pivot indices.
3) `stop` -> Terminate the current bisection session by deleting the state file.
4) `good` | `bad` -> Mark the section processed last as "good" or "bad", respectively. The next bisection step will continue with the bad section.
5) `status` -> Print status information about the current bisection session. Hint: Use with higher verbosity settings for more output.

A bisection session is initiated by issuing the `start` directive. This sets the pivot to the middle of the list of available font files. Now run *luaotfload-tool* with the `--update` flag set as well as `--bisect=run`: only the fonts up to the pivot will be considered. If that task exhibited the issue you are tracking, then tell Luaotfload using `--bisect=bad`. The next step of `--bisect=run` will continue bisection with the part of the files below the pivot. Likewise, issue `--bisect=good` in order to continue with the fonts above the pivot, assuming the tested part of the list did not trigger the bug.

Once the culprit font is tracked down, `good` or `bad` will have no effect anymore. `run` will always end up processing the single font file that was left. Use `--bisect=stop` to clear the bisection state.

miscellaneous

–verbose=N, -v Set verbosity level to *n* or the number of repetitions of `-v`.

–quiet No verbose output (log level set to zero).

–log=CHANNEL Redirect log output (for database troubleshooting), where *CHANNEL* can be

1) `stdout` -> all output will be dumped to the terminal (default); or
2) `file` -> write to a file to the temporary directory (the name will be chosen automatically.

–version, -V Show version numbers of components as well as some basic information and exit.

–help, -h Show help message and exit.

–diagnose=CHECK Run the diagnostic procedure *CHECK*. Available procedures are:

1) `files` -> check *Luaotfload* files for modifications;
2) `permissions` -> check permissions of cache directories and files;
3) **`environment` -> print relevant** environment and kpse variables;
4) `repository` -> check the git repository for new releases,

5) `index` -> check database, display information about it.

Procedures can be chained by concatenating with commas, e.g. `--diagnose=files,permissions`. Specify `thorough` to run all checks.

–conf=FILE Read the configuration from *FILE*. See **luaotfload.conf**(%) for documentation concerning the format and available options.

–dumpconf Print the currently active configuration; the output can be saved to a file and used for bootstrapping a custom configuration files.

FILES

The font name database is usually located in the directory `texmf-var/luatex-cache/generic/names/` (`$TEXMFCACHE` as set in `texmf.cnf`) of your *TeX Live* distribution as a zlib-compressed file `luaotfload-names.lua.gz`. The experimental lookup cache will be created as `luaotfload-lookup-cache.lua` in the same directory. These Lua tables are not used directly by Luaotfload, though. Instead, they are compiled to Lua bytecode which is written to corresponding files with the extension `.luc` in the same directory. When modifying the files by hand keep in mind that only if the bytecode files are missing will Luaotfload use the plain version instead. Both kinds of files are safe to delete, at the cost of regenerating them with the next run of *LuaTeX*.

SEE ALSO

luaotfload.conf(5), **luatex**(1), **lua**(1)

- `texdoc luaotfload` to display the manual for the *Luaotfload* package
- Luaotfload development https://github.com/latex3/luaotfload
- LuaLaTeX mailing list http://tug.org/pipermail/lualatex-dev/
- LuaTeX http://luatex.org/
- ConTeXt http://wiki.contextgarden.net
- Luaotfload on CTAN http://ctan.org/pkg/luaotfload

BUGS

Tons, probably.

AUTHORS

Luaotfload was developed by the LuaLaTeX dev team (https://github.com/lualatex/). It is currently maintained by the LaTeX Project Team at https://github.com/latex3/luaotfload The fontloader code is provided by Hans Hagen of Pragma ADE, Hasselt NL (http://pragma-ade.com/).

This manual page was written by Philipp Gesang <phg@phi-gamma.net>.