Missing files in the installation

The reference

Very early in the development and distribution of ConTEXt we came up with what got called the minimal installation. At that time disk space and bandwidth was limited and keeping a complete distribution around some 100 megabyte sounded like a good idea. The current minimal installation is larger but mainly because we have different versions and also include the documentation. There are also some more and larger font resources but still it has a much smaller footprint than alternatives. To the best of our knowledge the distribution has free only software and resources and compresses to some 130 MB in 2025, including the source of LuaMetaTEX. It might however support some non free resources, like fonts that users buy or viewers that users use, but that's not a dependency.

Derived work

When a subset of ConTEXt is installed and some components are omitted one can argue that it is a derived work, if only because users can get surprised when something doesn't run. We (MS & HH) found out that when testing a pre-test of TEXLive 2025 some math test documents didn't compile after we installed the Lucida Bright fonts: the typescript file that defined the typeface using the filenames was not seen as free enough, or more precisely, as it refered to a commercial component, it could not be included according to policies. There is not much we can do about it other than trying to signal such a missing resource during a run. But can we predict what qualifies as unacceptable? We'll try. We take the omission of the typescript as reference and assume consistency. When we looked at what gets installed in cases other than ConTEXt, we saw plenty of similar cases that just got accepted but leave that to the readers puzzlement.

Fonts

For TEX a font is just an abstraction: the engine only needs a few properties and once the typesetting is done it moves parts of the font resource to the backend. In that respect it acts like any other typesetting program. The assumption has always been that the result (most likely a pdf file) is an accepted medium but in the perspective of derived work distributers we might be wrong here.

In a TEX setup there can be all kind of font resources that we discuss in general terms. Other macro packages might use different or additional resources.

TEX font metrics (`.tfm`) This is a binary file that contains font metrics, glyph dimensions, ligature building steps, kerning tables, math variant sequences and math extensible recipes. There are many such files in an official TEX distribution and the naming scheme obscures a bit if they are for commercial fonts. We don't need them in the MkIV and LMTX but it's a nice exercise (or game) to find them.

Virtual font (`.vf`) When present this file is used by the backend to compose characters from one or more fonts. From a distribution point of view they can refer to commercial fonts which makes them a candidate for removal. They are mostly used by eight bit engines and like tfm files they often have names that have to be interpreted by splitting the eight characters that make up the name so it's easy to have commercial bound ones that get unnoticed. The ConTEXt distribution doesn't need such files.

OpenType (`.otf`) These files pack glyphs and related data in one file. It looks like refering to them makes the refering file a candidate for removal. Interesting is that when commercial fonts are refered

to by name instead if file, such a file can escape removal. In fact, users can define fonts in their documents using names and then don't need definition files at all. However, this is kind of unreliable so depending on that is also introducing issues. Does one expect a font vendor to also collect and distribute whatever extra TEX needs?

TrueType (`.ttf`) See OpenType files.

TrueType container (`.ttc`) See OpenType files.

Adobe font metrics (`.afm`) These are test files with metric, ligature and kerning information. As they can serve as the basis for tfm files one expects those used to be also present as they can be considered source files. The ConTEXt approach to commercial (or any third party) fonts has always been to use this file as basis (reference) and for MkII we generate(d) the tfm files from those.

Type1 font data (`.pfb`) These are resources that are either free or non-free. Because the auxiliary files that are needed to make them (like additional data files for font editors) are not distributed they are not open.

Backend driver font mapping (`.map`) Here tfm filenames get mapped onto real filenames so here we can have some commercial font support creeping in. We can also define map entries from the TEX end so then TEX files become candidates for removal. However, it looks like map files are often not seen by the filters.

Backend driver font encoding (`.enc`) These files map glyph indices to glyph names that the backend can resolve. As far as I know these are, although they resemble PostScript, typical to TEX backends and I suppose that these vectors are free even when they come from commercial entities.

ConTEXt typescripts (`.mkiv`) & `.mkxl`) The typescript files refer to fonts either by fontname or by filename and both can concern a commercial font. Interesting that this is just a reference. In that sense it is not different from a tfm file with an obscure name. However, in the past obscure names could just make them end up in distributions, which more visible names in these TEX files make them unacceptable. In the past these obscure names got mapped to real (commercial) ones in map files but those were seen as databases and therefore okay. However, in the case of ConTEXt we use typescripts and curiously in MkII we can also map from there using primitives, so one can argue that a MkII typescript is a database and acceptable

ConTEXt Lua font goodies (`.lfg`) Where typescripts that define a commercial font, often combined with public fonts into a style, are seen as unacceptable, goodie files seem to pass the test. So one way out is to use a symbolic name and remap that in a goodie file, just like we remap design sizes there. It's of course a cheat but one that exposes the kind of arbitrary approach to this issue.

Various There can be other resources, for instance that set up expansion and protrusion. However we don't do that explicitly so there is no commercial font stuff here. When we took a quick look at the policies it seemed to be an ingored area.

Colors

Users might need to embed color profiles, maybe even the ones that are standard and could be refered to by name. We do have some in the distribution but as this is a specialized area users can also manage that themselves. If you validate for instance pdf files you have to take this into account, otherwise (as we often do) you can just decide not to bother. For average documents and printing it matters very little.

The color definitions that ConTEXt comes with are public or our own and we don't care about the commercial ones. You can easily define a spot and/or process color in a document style and no one except you will see it.

Graphics

Including graphics is very much related to artistic copyright. We'd love to include some more but don't like the idea of for instance permitting a user to adapt a cartoon. Of course with machine learning applications (aka ai) abusing anything to ones liking this whole discussion has become irrelevant but maybe in the end it will result in a bit more protection for distributed free graphics. It all just has to backfire huge onto the open and free software community first.

Patterns

Hyphenation patterns are a bit black magick: not all are made from resources that are public. So what does that make the patterns? What are the exact parameters used to tune them? Can it be replicated? Let's stick to saying that they sound more free and open than they often are. We just ship them and assume it's okay. Very few people have a clue what they are anyway.

Backend

Here we arrived at the most complicated issue. In MkII we support several backends but because we use an abstraction layer the core functionality is isolated. This makes it easy to remove for instance support for dvipsone (the PostScript driver that we used) and Acrobat (because it needs a commercial converter). However, removing these for ConTEXt would also mean removing them for other macro packages and if they have a more integrated approach in might render them unusable so maybe one looks away from it. Anyway, the regular installation is now LMTX so there this is not an issue: we produce pdf directly and don't need additional software.

But it doesn't end there. Right from the start, and still, much pdf functionality is only supported by commercial software. That means that in principle it should be removed from those distributions that dislike that (viewer) dependency. You can think of multi media support (which evolved over years), named actions, widgets, tagging, etc. And what is actually the threshold for at some point including support? It is a bit like “One should use or do this or that.” while when one needed it first the ‘this’ and ‘that’ were nowhere to be seen.

Even more interesting is what this does with development: TEX macro packages could always support the latest greatest features but if the code will not be included in mainstream distributions development makes little sense. Here tagging is a good example: why develop something that depends on commercial software and then not being able to distribute it which also makes it untested? In fact, being cutting edge and adaptive in retrospect makes little sense; who cares what publishers want if it puts an extra (demotivating) burden on development.

Keep in mind that the standard is not really open and free either. Older versions where available, newer ones are paywalled but one can now get a version as a reward for giving away some personal information. It definitely wasn't officially open before 2024 so in retrospect no or little pdf support should have been shipped in these distributions. Also, before that, the need for reverse engineering the format or pdf files generated by the official commercial tools could also be a reason for dropping everything.

So, to summarize: we might need to identify what features are commercially driven and isolate them. Till that has been done it might mean that the whole macro package can be dropped because it can't function without a backend.

Indicators

So what do the TEXLive (and other) folk have to look for? The next concerns ConTEXt but similar criteria apply to other macro packages. Don't bother us with discussions.

In `colors/icc` there are some color profiles. We have no clue if they relate to commerce but at least they seem to be free.

In `context/data/scite` we have generic lexer files but also some configuration files. The editor is open source and free but there is a version for os-x that is paid for, so that might mean removal of those specific in a mac installer. There are also files for Visual Studio Code that then need to be dropped.

We don't know what documents in `doc/context` violate the rules. Some documentation shows examples that use commercial fonts. Those fonts are not in the distribution, so when these manuals are processed from the sources in the distribution they either use a replacement or they render in the document font. Of course still present references to commercial fonts can be an issue but so can be hyperlinks to non-free documents or articles.

Some examples in `doc/context/examples` embed JavaScript and some in `doc/context/presentations` use JavaScript, optional content layers and maybe even tagging or widgets. When produced and distributed before there were open source and/or free tools that could handle that these documents might qualify for removal.

Maybe some help files, as in `doc/context/scripts`, have a css definition that sets up prefered fonts on a system so they then become candidates for removal by refering to a possible commercial system font.

There are some screendumps used in manuals (and therefore in the source tree) that show results in non-free or non-open viewers that users don't have on their system so again they qualify. The same is true for some example data files that refer to books, articles, music etc. that has to be bought.

Various documents and source files that deal with typesetting mathematics refer to Cambria as reference font and that one not being free makes these files debatable. The same is also true when files refer to programs for symbolic (math) computing, large language models, etc.

We're not sure if the (FontForge) Adobe cidmaps that we ship are okay with distributions that are strict.

With regards to the backend, we ignore MkII here, there might be snippets (media, widgets, JavaScript, Adobe specific features, tagging) in the `lpdf-*` files that make the whole backend unacceptable in which case one should just drop ConTEXt completely (and maybe explain to potential users why).

Typescripts and goodie files with `cambria`, `koeielettersot`, `lucida`, `minion`, `adobegaramond`, `buy`, `cow`, are candidates. I don't know about those that support `bhai`, `shobhika`, `bengali`, `devanagari`, `gujarat`, `indic`, `kannada`, `malayalam`, `tamil`, `telugu`, etc. Be our guest.

The filenames that match `mathdesign`, `informal`, `hvmath` `mathtimes` and `md*` are for MkIV only so when we go LMTX only we might no longer ship them anyway.

The only font that we ship resources for that has restrictions is the ‘koeieletters’ font based on drawings by Duane.

It must be noted that when ConTEXt is installed in TEXLive also some other stuff gets installed as side effect of packaging and we have no clue if anything in that will violate the rules.

There are a few styles and scripts that support `pfsense`, `evohome` etc. (rendering statistics and such) so again that sounds something commercial is supported. But is that different from a style for a specific scientific publisher?

Several of the `s-*` and `m-*` styles can contain examples of usage with non free or commercial (math) fonts.

Some of the `pdf-*` files that deal with validation can contain snippets that might as well be considered tricky, certainly in the perspective of the pre 2023 commercial validation market.

The LuaMetaTEX source is also in the distribution which in our opinion not only guarantees that users can compile the engine but also that it guarantees a more longer term perspective. Removing them kind of violates the idea that one should always distribute the source. Removal also makes the distribution non referential because we don't know what engine is used. The source doesn't rely on code outside that source tree. Anyway, if needed, one can always install the reference distribution alongside.

Approach

What can we do about all this? First of all we don't see it as our problem so basically we can ignore it. Let those who distribute deal with it, also because policies can differ. All we need to care about is users. So, for instance we can issue a warning when a critical component is not present. We can mark files as being potentially unacceptable by some distributors. We can just omit files but there we see no candidates so that won't happen. Even more drastic (and also more work) is to to split some functionality, most likely in backend drivers. It's up for debate.

All of the above said: we think that there is nothing ‘non-free’ in the distribution. There is some support for non-free and/ore non-open resources (like fonts) and viewers but we don't ship those. So, in the end you can as well ignore everything we said, what is what we do ourselves.

Support

You can get support at:

maillist	`ntg-context@ntg.nl / http://www.ntg.nl/mailman/listinfo/ntg-context`
webpage	`http://www.pragma-ade.nl / http://context.aanhet.net`
archive	`https://github.com/contextgarden`
wiki	`http://contextgarden.net`