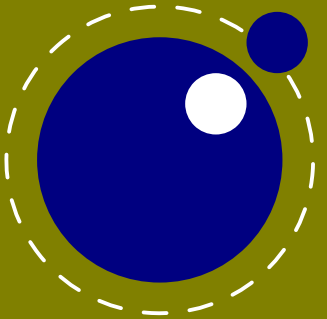
Lean and mean

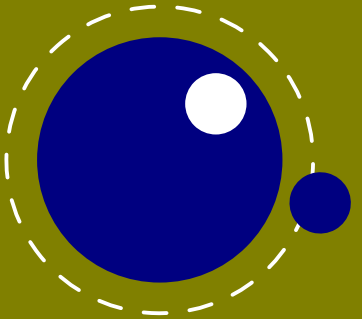
LuaMetaTEX

ConTEXt meeting, September 2019

Hans & Alan

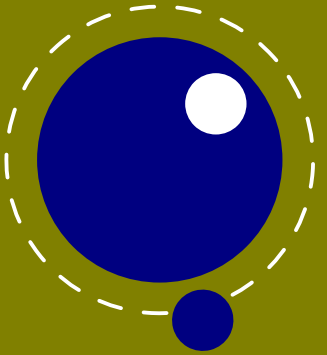


- **interferences:** ConTEXt, plain TEX and LATEX all have different demands (we want to experiment and move on and users pick up fast)
- **complexity:** the source tree is way too complex as is the build (we only need LuaTEX)
- **distributions:** no one can guarantee stability for ConTEXt (being a minor player but often a bit ahead)
- **annoyances:** experimental codes leads to usage outside ConTEXt and that triggers complaints
- **motivation:** running into folks who love to stress “huge bugs” and “much instability” wastes energy
- **arguments:** I got tired of “you need to support this because . . .” blabla
- **nagging:** like “the manual . . .” is becoming too tiresome, so best keep experiments within the ConTEXt bubble



LuaMetaTEX

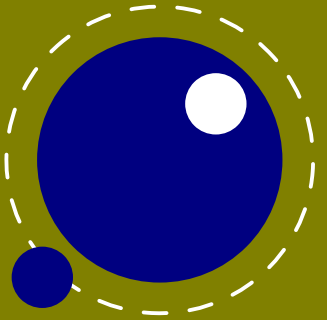
How it became



LuaMetaTEX

What it is

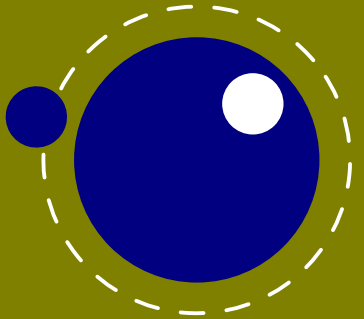
- **simplification:** we don't need all what is currently in the LuaTEX engine as we don't use it
- **source:** there is much less of it and we can get rid of web artifacts
- **compilation:** there was much more going on than was needed and only a few knew those details
- **consistency:** to guarantee consistency with ConTEXt the source code will be part of the source distribution (once I'm satisfied)
- **marketing:** this way the relation with ConTEXt and its user base is more clear
- **playground:** we can move forward and experiment without the danger of running into problems with non ConTEXt users: "use it at your own risk"
- **possibilities:** playing a bit more with the bits and pieces that are reponsible for most (interfering) issues, like the the (asynchronous) page builder



LuaMetaTEX Implications

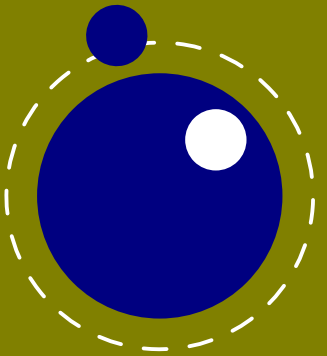
- **binary:** there is only one relatively small binary needed (that does all things needed)
- **code base:** there comes an extra source tree, but it's small (compresses to around 2 MB)
- **user control:** if needed users can compile the program so we're self contained
- **future safe:** we can move forward and improve
- **modern:** a code base with the latest LuaTEX, mplib and Lua
- **side effect:** we drop LuaJIT as it doesn't keep up (and benefits are too small)
- **design:** we have a better separation between the Knuthian front- and output format driven backend
- **independent:** there is no dependency on external libraries, we keep all we need in the code base (we only use a few small third party libraries)

- **hobyism** we don't need to carry the burden of everything (unless paid for it's only fun and users that drives development)
- **convenience:** the faster compilation makes reworking and experimenting reasonable
- **stepwise:** I take my time an do string stepswise because things should not break without fast recovery
- **feelgood:** this all fits well into the good old TEX extension model
- **eventually:** when proven useful we can always push code upstream into LuaTEX



LuaMetaTEX

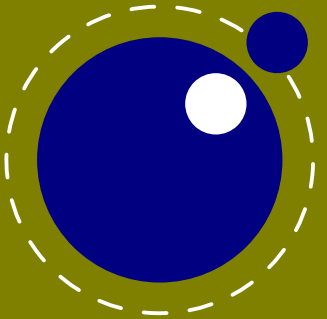
A few notes



- **original:** the starting point is LuaTEX, original web code, already cweb code
- **stability:** after a initial stage LuaTEX was stepwise extended till version one a few years ago
- **frozen:** there were only a few changes after that but no real conceptual ones
- **engine:** what is now called LuaMetaTEX is a reworked code base
- **graphics:** also mplib has been reworked a bit and some extensions were added
- **libraries:** there are a few extra (small) helper libs, but all in the source tree
- **pplib:** we already use the next version of pplib
- **pruning:** and best of all, quite some not used code could go

LuaMetaTEX

Bits and pieces



LuaMetaTEX

Some details

- **source tree:** the code base has been regrouped, globals became more local (work in progress), header files were added
- **source files:** there is hardly any font related code, languages were kept, and the backend code is dropped: show files
- **libraries:** a few libs were added and dropped: show some
- **cmake:** compilation is different: work in progress
- **mkxl:** there are new files in ConTEXt: `driv`, `lpdf`, `.mkxl` and expect more
- **binary:** there is only one stub for all

during presentation: show the source tree as well as the binary directory