From MkII to MkIV

What is MkIV

- it is the next generation ConTEXt, using LuaTEX
- we also use MkIV to explore new ways and replace code
- if possible the code ends up in generic modules
- the working title of this effort is called MetaTEX
- these modules can be combined into ConTEXt MkIV
- the idea is that eventually we can also make smaller specialized subsets
- in August 2007 MkIV goes beta (alpha code is accessible for those involved in the ConTEXt development)
- we're exploring ways to make lean and mean ConTEXt distributions that run from zip files

The tools

- luatools: this is a replacement for KPSEWHICH plus a bit more; it also generates formats and runs LUATEX with bootstrap code
- mtxrun: this script starts applications (or documents or ...) and runs LUA scripts with libraries preloaded
- both luatools and mtxrun contain all relevant libraries (self-merged)
- x-ldx: we provide a documentation subsystem, comparable to the existing one but using XML
- eventually the current RUBY scripts will be replaced by embedded or companion LUA scripts that use TEXLUA as LUA engine

The files

- MkIV provides alternative code blocks, more drastic replacements than the usual engine specific drop-ins (depending on how much XeTeX diverts from normal TeX, at some point we may have MkIII code for XeTeX)
- LuaTeX specific code can be recognized by the file suffix: foo.tex, foo.mkii, foo.mkiv, foo.lua
- large runtime data collections like fonts are cached: font tables are normally about half a megabyte but sometimes they are tens of megabytes
- temporary files (including formats) end up in the temporary path
- we collect font files in fonts/data/vendor/collection (at least on our machines)

More files

- ConTEXt's buffers are now kept in memory
- auxiliary data is now moved to Lua tables
- index sorting is now done internally
- data and functions are organized in tables
- these are byte-compiled into the format
- currently (July 2007) we have 67 modules (3 megabyte bytecode)

Work done so far

- file io, reading from other resources
- error handling
- there is now a generic font feature subsytem
- we have written a first framework for more clever verbatim
- metapost conversion (prelude to integration) is LUA based
- all kind of conversions are now done in LUA
- input regimes are dealt with by LUA instead of TEX
- multipass data managed is now handled by LUA
- experimental new XML handling (a LUA based parser is ready)

Work in progress

- we will provide additional spacing models and improve existing ones
- in addition to calcmath there will be alternative input methods for math
- there wil be more intelligent font support and inline feature switching
- we will explore automatic adaption of font handling to languages and scripts
- alternative hyphenation methods will be provided, for instance using dictionaries
- MkII already supports many color models and font rendering variants but we will move this to attributes
- there will be a user friendly interface to virtual fonts

The impact

- we can get rid of quite some resources, especially font related files
- we can experiment with much simpler resource trees
- updating may come down to dropping a zip file in an update path
- different and more flexible solutions can be provided for similar problems