This is a selection of three sets of presentations at meetings and a bunch of examples for some styles that come with ConTEXt. In MKIV some of the older styles have been dropped. There are more presentations, starting mid 1990's: some are lost, others make no sense showing (or have articles instead) and I have no time anyway to collect all of them.

We show the first four pages but you can find the complete files in the ConTEXt distribution. This file is generated automatically so it's also a snapshot. Maybe some day I find the time to add some more.
No way back

- We have passed the point of no return already years ago.
- Most, apart from say, M4, with an un-cancelled fall-back on MiK
- The code base is now completely split, with the exception of some modules.
- Some solutions are implemented in Lua with only a small wrapper at the TeX end.

Finding the balance

To get an idea

- structure: sectioning, notes, description, diagrams, synonyms
- typography: sectioning, notes, description.
Typesetting
- sometimes you learn words into speech into language
- and often it may have started and ended in the mind
- sometimes you learn words or even symbols
- and it has a life of its own, but in the end it's just a process
- and it is done with tools and systems

Available at BachoTEX

How about those typographic virtues:
do they still make sense?

Hans Hagen
EuroBachoTeX
May 2013
What we can do
we try to speed up baseline performance (in pages per second)
the nicer the interface, the slower it gets, but you seldom set something up
everything you provide gets used at some point, also in inefficient ways
wrong use of functionality can have drastic and unexpected speed penalties
identify and optimize critical routines, both at the \TeX{} and Lua end
speed matters in a edit-run-preview cycle although this is mostly perception

the interaction between
ligatures hyphenation
and kerning

the generic loader

Hans Hagen – bachoTEx 2016
Grandpa’s toolbox  
(making closets)
Dad’s hobby  
(cleaning closets)
Uncle’s friends  
(talking closets)

bachoTeX 2016

- there’s closets and closets
- take the ones you put stuff in ...
  e.g. that you buy at ikea
- you have to assemble them so you
  get out your toolbox
- this kid sits next to you wondering what that tool is
- what is natural for you to use might not be so for them
- but grandpa likes to carry over his knowledge and experiences

- but … kids get their information from the internet, not from you
- (they watch discovery channel or national geographic and know a lot)
- or they look at vloggers (no bloggers) trying to learn something
- and they keep moving on … and on
- do gp’s live long tools really make sense to them?

- grandpa also has a vlogger, he is called knuth
- gp is as locked into tex as the kids are into apps
- he’s a do-it-him-self kind of person
- will his grandkids love watching him crafting
- and hear his stories about meetings and journals

- interest

- grandpas
Technologies

As each vendor came up with something, we have to deal with a all kinds of formats. And or course, as... method is useable for the tens of thousands of skin tone combinations of families, (kissing) couples, and professions.
Variable Fonts
we're ready for them

Hans Haupin
BachoTUG 2017

A Summary

- the macro package's view: just a font but with many possible variations in shapes (width, weight, slope, ...) and therefore a lot more complex user interaction
- the engine's view: an abstraction not different from other fonts but that needs a special treatment in the backend
- the system's view: a font that should be displayed exactly like any other with outlines in the user interface

Starting point

- The OpenType 1.8 specification at the Microsoft website defined the extra tables and explains how they can be used.
- The engine's view: the abstraction not different from other fonts but that needs a special treatment in the backend.
- The system's view: a font that should be displayed exactly like any other with outlines in the user interface.

Implementation steps

- First try to render content in order to see if it's working or not. This will give you an idea of how the font is working.
- Then try to load the extra tables and figure out what these define and can be used on the system's view.
- Finally, make sure that the font can be embedded as a normal font and not as an inline image.
Why oh why

In Con TEXt we have a mechanism to apply effects to a glyph stream.

An active user on the Con TEXt mailing list wondered if that could be applied to specific fonts.

The particular interest concerned the possibility to bolden fonts.

I don't really like effects and not all fonts are suitable for it.

What are effects

Normal effects are implemented using the 'effects' mechanism, which already dates way back in MkII times and of course is also available for MkIV.

\defineeffect\[outer\]\[alternative=outer,\rulethickness=1.25pt\]\effect\starteffect\[outer\]\effect\stopeffect

History

We started using MetaPost some two decades ago and immediately went the pdf route.

We used special colors plus specials to communicate extensions, for instance cmyk colors and shades.

This mechanism was stepwise improved and extended. Some mechanisms like texts needed an extra pass.

When we moved to LuaTEX and mplib we started using pre- and postscripts to carry information with the paths.

Currently we use a bit of Lua from within mplib to communicate during the MetaPost run with Con TEXt. This permits cleaner interfaces.

Colors

\startMPcode\draw image(draw image(fillunitcirclerotated 45 withcolor "red" ;
fillunitcirclerotated165withcolor "green" ;
fillunitcirclerotated285withcolor "blue" ;
) shifted( 1.25,0) ;
)draw image(fillunitcirclerotated 45 withcolor "cyan" ;
fillunitcirclerotated165withcolor "magenta" ;
fillunitcirclerotated285withcolor "yellow" ;
) shifted( 1.25,0) ;\xsizedTextWidth;\stopMPcode
e-books
old wine in new bottles
ConTExT Meeting 2011

MathML
or math in general
ConTExT Meeting 2011

MetaPost
how we adapt
ConTExT Meeting 2011
The old way

- an MKIV-sorting is delegated to ‘#isort’ in a one-passer action
- following vectors are parallelized
- output text depends on the language
- there are less small confusions with closed characters and commands

Moving on

- in MKIV-sorting happens during the run
- we only have to deal with LuaDoc (-notification)
- output text still depends on the language
- sorting can be controlled by methods
- there is an external solution (conflicting w/characters, named languages)

Character data

<table>
<thead>
<tr>
<th>regular</th>
<th>accent</th>
<th>signage</th>
<th>hanged</th>
<th>hangled</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The new releases (and betas) are checked against a growing set of test files (Lukas mails a report)

New namespaces and helpers mostly done, but will be checked for consistency.
If so, what functionality should be provided.

Do complex column mechanisms still make sense given the move to electronic paper.

For some local applications we use simple columns.

As the moment you can do something with the result.

In one of the things you can do is package all collected so far in a page.

Of course users can still program a lot, but not all need that.

Interpreting the node list and injecting visualizers.

You need to fake them by fiddling with the width and spitting boxes.

One of the things you can do is package all collected so far in a page.

This way we don't need stubs (and avoid potential conflicts in name).

But as we already had some control over what gets shown by using layers.

It has a few extensions that are loaded on demand: extras

For instance before and after each paragraph is processed.

Nesting: how about columns inside columns.

For instance before and after each paragraph is processed.

As unboxing does not work, it is somewhat interfering.

Footnotes: page, first or last column, each column (delayed, immediate).

So I figured out a way to visualize boxes, kerns, glue, etc.

In for instance itemize we used a mixed one- and multi-column model.

In due time this will all be normalized (as most lives in modules).

In between it can trigger the so called output routine.

I wonder if anyone ever used it.

It has some features that makes it easier to integrate in services.

There is room for more (but first I want the bitlib of LUA 5.2).

We have more debugging, much shows up when writing new code.

In due time this will all be normalized (as most lives in modules).

In the above context we used some helper functions that can simplify content.

In LUATEX we can also intercept content at more places.

The 'mtxrun' script

Locates and runs scripts, has a lot of helpers preloaded.

It is in fact my MKII ancestor it is not needed for index sorting etc.

It has a few extensions that are loaded on demand: extras

It is based extras it made sense to redo it.

It looks al lot like plain distribution provides a sort of ecosystem.

Some make no sense in

Contrary to its

The basics were a rather trivial quick job as we had a lot in place already.

Some aspects were tricky , like stretch and shrink (no

The 'mtxrun' script

A regular run

context --run filename

Running from an editor

context --autopdf filename

Running from a service

context-2012-mixed-columns

context-2012-the-script

context-2012-visual-debugging

Breskens 2012 Visual debugging

The truth

Some myths

Output

TEX collects content paragraph wise.

In between it can trigger the so called output routine.

In LUATEX we can also intercept content at more places.

For instance before and after each paragraph is processed.

Some facts

• ConTEXt Meeting

• Control over what gets shown by using layers.

• Some column mechanisms are overloaded.

• Some constructs interfere so we need to compensate side effects.

• Some constructs interfere so we need to compensate side effects.

• We use rules (and leaders) to visualize properties.

• Some constructs interfere so we need to compensate side effects.

• We use rules (and leaders) to visualize properties.

• Some constructs interfere so we need to compensate side effects.

• We use rules (and leaders) to visualize properties.

• Some constructs interfere so we need to compensate side effects.

• We use rules (and leaders) to visualize properties.
We try to identify and optimize critical routines, both at the TeX and Lua end, but a math engine still has to deal with ASCII input as well and tagged with some rendering directive, often indicating a font style.

In Unicode and OpenType we have alphabets with standardized code points (but testing basics only).

In traditional TeX we have alphabets in different fonts, so we're talking switches.

Everything you provide gets used at some point, also in inefficient ways, so best is not much of a burden.

Progress or standing still

We're off better but the gaps are an anomaly.

There is recognition of math as a proper (but not standardized) script.

Math as script

Alphabets

• the shape (style) of a character determines its meaning
• but in most cases an type a is entered as ASCII, without context

although not officially a script, OpenType treats it as such, but without control

Fences

• math can be input using the TeX syntax, MathML, calculator like sequences, . . .
• but apart from . . . <mo>+</mo> <mn>1</mn> <mo>)</mo>

There is recognition of math as a proper (but not standardized) script.

Combining

• speed matters in a slow run process cycle although this is mostly perception

the more the better, as long as it gets, but you seldom set anything up to that sort of level.

everything you provide gets used at some point, also in inefficient ways, so best is not much of a burden.

Unscripting

• speed matters in a slow run process cycle although this is mostly perception

the more the better, as long as it gets, but you seldom set anything up to that sort of level.

everything you provide gets used at some point, also in inefficient ways, so best is not much of a burden.

Math as script

• math can be input using the TeX syntax, MathML, calculator like sequences, . . .
• but apart from . . . <mo>+</mo> <mn>1</mn> <mo>)</mo>
After ten years of stepwise development and experimenting, we release version 1.00 of LuaT\LaTeX\ during the 10\textsuperscript{th} Con\TeX\t meeting in the Netherlands, September 2016.

The interface is now rather stable and will not change significantly which means that one can write stable packages.

So, it’s time for a bit reflection as well as time to tell what we will be doing next.

Around 2005, after we talked a bit about it, Hartmut added the Lua scripting language to \TeX\ as an experiment.

This add-on was inspired by the Lua extension to the Scite editor that I (still) use.

One could query counter registers and box dimensions and print strings to the \TeX\ input buffer.

The Oriental \TeX\ project then made it possible to go forward and come up with a complete interface.

For this, Taco converted the code base from Pascal to C, an impressive effort.
Do you know \TeX? We use \TeX for typesetting mathematical text, but also for text that has no math. Thus, I came to the conclusion that the designer of a new system must not only be the implementer and first large-scale user; the designer should also write the first user manual. The separation of any of these four components would have hurt \TeX significantly. If I had not participated fully in all these activities, literally hundreds of improvements would never have been made, because I would never have thought of them or perceived why they were important. But a system cannot be successful if it is too strongly influenced by a single person. Once the initial design is complete and fairly robust, the real test begins as people with many different viewpoints undertake their own experiments.
Thus, I came to the conclusion that the designer of a new system must not only be the implementer and first large-scale user; the designer should also write the first user manual.

The separation of any of these four components would have hurt TEX significantly. If I had not participated fully in all these activities, literally hundreds of improvements would never have been made, because I would never have thought of them or perceived why they were important.

But a system cannot be successful if it is too strongly influenced by a single person. Once the initial design is complete and fairly robust, the real test begins as people with many different viewpoints undertake their own experiments.

We thrive in information-thick worlds because of our marvelous and everyday capacity to select, edit, single out, structure, highlight, group, pair, merge, harmonize, synthesize, focus, organize, condense, reduce, boil down, choose, categorize, simplify, classify, tabulate, abstract, outline, drill down, analyze, synthesize, categorize, examine, observe, discriminate, distinguish, isolate, screen, pigeonhole, pick over, sort, integrate, blend, inspect, filter, lump, skip, smooth, chunk, average, approximate, round, aggregate, outline, summarize, itemize, review, dip into, flip through, browse, glance into, leaf through, terse, summarize, churn, synthesize, separate, distill the wheat from the chaff and separate the sheep from the goats.
Donald Knuth has spent the past several years working on a system allowing him to control many aspects of the design of his forthcoming books—from the typesetting and layout down to the very shapes of the letters! Seldom has an author had anything remotely like this power to control the final appearance of his work. This concept was first explored in his article "The Concept of a Meta-Font". I felt compelled to make some comments to clarify certain important points that touch close to my deepest interests in artificial intelligence and esthetics, which I feel might be taken wrongly by many readers, and since they are points that are often held in somewhat mixed opinion the article is charmingly written as well. However, despite my overall enthusiasm for Knuth's idea and article, there are some points in it that I strongly disagree with.

In his article "The Concept of a Meta-Font", Knuth sets forth for the first time a new system for creating, modifying, and compiling typographic typesetting systems. The separation of any of these four components would have hurt TEX significantly. If I had not participated fully in all these activities, literally hundreds of improvements would never have been made, because I would never have thought of them or perceived why they were important. But a system cannot be successful if it is too strongly influenced by a single person. Once the initial design is complete and fairly robust, the real test begins as people with many different viewpoints undertake their own experiments.
Some Quotes
that you probably know by now

Tufts
Knuth
Reich
Zapf
Materie

Tufts
Knuth
Reich
Zapf
Materie

Tufts
Knuth
Reich
Zapf
Materie

Tufts
Knuth
Reich
Zapf
Materie

Tufts
Knuth
Reich
Zapf
Materie

Tufts
Knuth
Reich
Zapf
Materie

Tufts
Knuth
Reich
Zapf
Materie

Tufts
Knuth
Reich
Zapf
Materie

Tufts
Knuth
Reich
Zapf
Materie

Tufts
Knuth
Reich
Zapf
Materie

Tufts
Knuth
Reich
Zapf
Materie

Tufts
Knuth
Reich
Zapf
Materie

Tufts
Knuth
Reich
Zapf
Materie

Tufts
Knuth
Reich
Zapf
Materie

Tufts
Knuth
Reich
Zapf
Materie

Tufts
Knuth
Reich
Zapf
Materie
Stepwise Refinement

Female Artists

- Fiona Apple
- Tori Amos
- Kate Bush
- Heather Nova
- Alanis Morissette
- Suzanne Vega

Male Composers

- John Adams
- Steve Reich
- Louis Andriessen
- Olivier Messiaen

Stepset 1

1 Step Set 1

STEP ONE

STEP TWO

Contents

1 Step Set 1

STEP ONE

STEP TWO

We thrive in information-thick worlds because of our marvelous and everyday capacity to select, edit, single out, structure, highlight, group, pair, merge, harmonize, synthesize, focus, organize, condense, reduce, boil down, choose, categorize, catalog, classify, list, abstract, scan, look into, idealize, isolate, discriminate, distinguish, screen, pigeonhole, pick over, sort, integrate, blend, inspect, filter, lump, skip, smooth, chunk, average, approximate, cluster, aggregate, outline, summarize, itemize, review, dip into, flip through, browse, glance into, leaf through, skim, refine, enumerate, glean, synopsize, winnow the wheat from the chaff and separate the sheep from the goats.