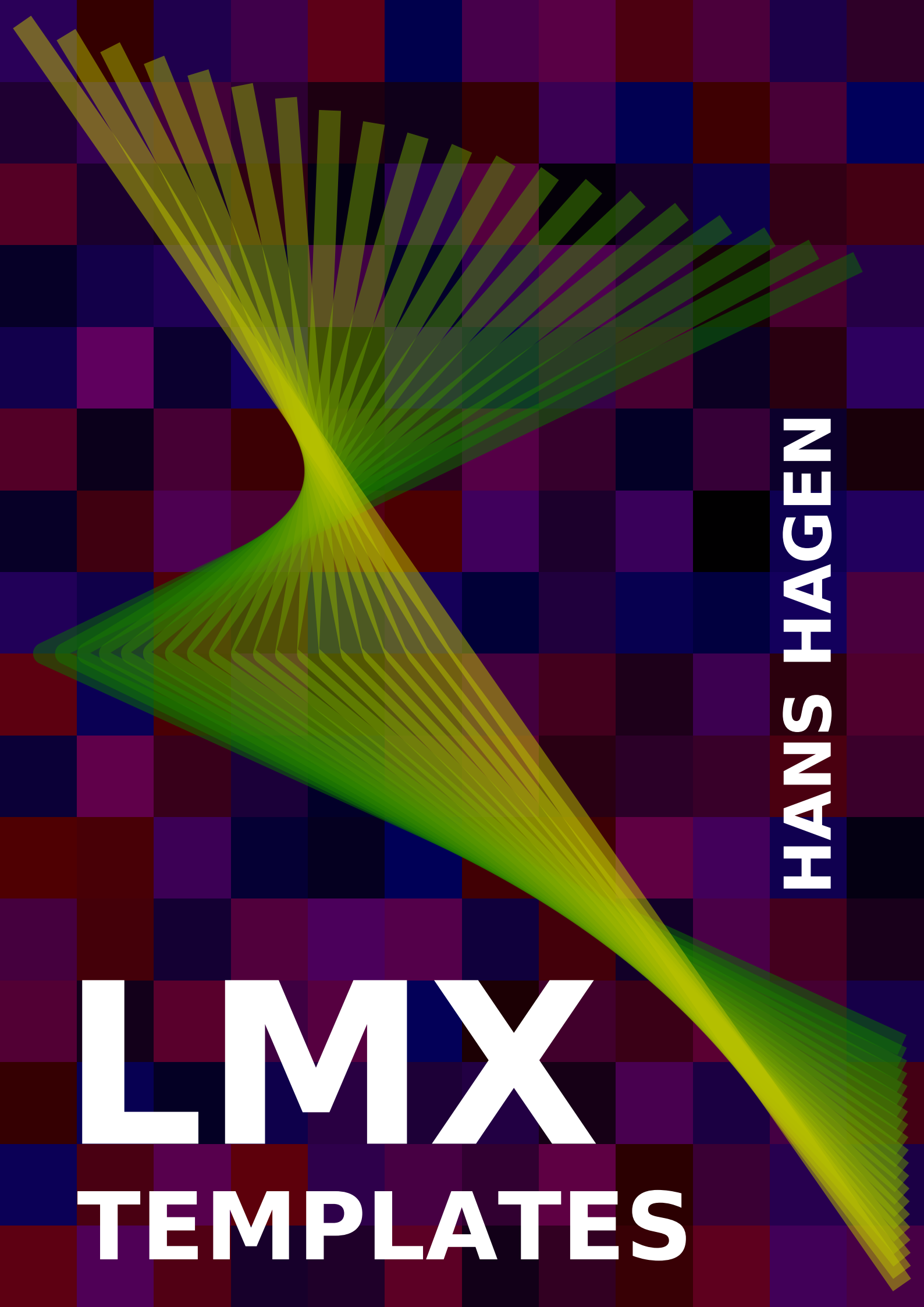
HANS HAGEN
LMX
TEMPLATES

Contents

1 Introduction

This manual is not finished yet. The main reason is that what is described here is an afternoon experiment resulting in a dozen lines of Lua *glue code that builds upon an already existing mechanism. When users like this, I will extend the basic* lmx *handler to suit the* TEX *end better. There will also be also support for cache based and in–document templates.)*

The acronym `lmx` stands for document that are a mix of Lua and xml and is just the three letters `xml` reversed. Such documents showed up pretty soon in MkIV while I was exploring ways to present debugging and error information to users using xml. As a consequence this is one of the older mechanisms available, although I doubt if users start looking for it when they start using ConTEXt.

Anyhow, because we also use lmx for populating web pages, at some point I wondered if using the same approach for TEX files made sense. I'm still not sure about it but who knows where this ends up. Currently code is shared but in the future we might end up with a variant that adds some more flexibility.

2 How it works

First of all, using this mechanism involves yet another kind of `mk`, so now we have:

mkii	The old version of ConTEXt, using pdfTEX,XETEX, etc.
mkiv	The new version of ConTEXt, using LuaTEX.
mkvi	Similar to MkIV but with named macro parameters.
mkix	A MkIV file mixed with Lua wrapped in xml processing instructions.
mkxi	Similar to MkIX but with named macro parameters.

The nice thing about sticking to wrapping in angle brackets is that it plays nice with syntax highlighting. Before we show an example of such a mix, we first point out that loading (and thereby conversion) happens automatically. A `mkix` or `mkxi` file is just a regular ConTEXt file. In the test suite there is a demo file that can be included like this:

```
\input lmxlike-001.mkxi
```

This file is loaded and converted on the fly. No caching takes place, but in due time we can use the cache built into the lmx handlers if needed. The template itself can be fed with variables in the `document` namespace:

```
\starttext
```

```
  \startluacode
     document.variables.text = "set"
  \stopluacode

  \input lmxlike-001.mkxi

\stoptext
```

Instead of a special suffix, you can also force conversion with the `macros` directive:

```
% macros=mkix
```

Part of the mentioned looks as follows:

```
\bTABLE
  <?lua for i=1,40 do ?>
    \bTR
      <?lua for j=1,5 do ?>
        \bTD
          cell (<?lua inject(i) ?>,<?lua inject(j)?>)
          is <?lua inject(variables.text or "unset") ?>
        \eTD
      <?lua end ?>
    \eTR
  <?lua end ?>
\eTABLE
```

The `<?lua ... ?>` command is conceptually different from (say) `\ctxlua` in the sense that the later executes some Lua code at that spot, while in a template a Lua function is constructed out of the whole that gets executed. In fact, we use Lua to construct an input file.

For the moment we only mention the predefined `inject` command. There are some more but they make more sense for xml and html and in due time this will be decoupled so that we can have dedicated helpers. Even the xml and html part is somewhat in flux.

The previous example can of course also be done differently. It's a matter of taste and usage what method gets used:

```
\startluacode
  context.bTABLE()
    for i=1,40 do
      context.bTR()
        for j=1,5 do
          context.bTD()
            context("cell (%s,%s) is %s",i,j,document.variables.text or "unset")
          context.eTD()
        end
```

```
      context.eTR()
    end
  context.eTABLE()
\stopluacode
```

The difference between a MkIX and MkXI file is the same as between a MkIV and MkVI file: the way macros can be defined:

```
\def\testmacro#one#two{[#one,#two]}

\testmacro{1}{2}
\testmacro{one}{two}
\testmacro{second}{first}
```

In practice one will seldom need macro definitions in a template file but the possibility is provided.

Colofon

author Hans Hagen, PRAGMA ADE, Hasselt NL
version March 22, 2013
website www.pragma-ade.nl – www.contextgarden.net