Das Pfarrei-Paket*

Markus Kohm

2013/03/10

Zusammenfassung

In „Die TEXnische Kömdie“, Ausgabe 1/2013 hat Christian Justen über seinen Einsatz von LATEX im Pfarrdienst berichtet. Einige der von ihm verwendeten bash-Scripte und seine Schilderungen dazu haben mich dazu inspiriert, eine Sammlung zu beginnen, die entsprechende Dinge leisten. Allerdings habe ich mich dazu entschlossen, keine bash-Script zu verwenden, sondern die Funktionalität derselben in lua-Scripte für die Ausführung mit texlua zu realisieren, da texlua inzwischen Bestandteil aller TEX-Distributionen ist. Somit wurde das ganze besser vom Betriebssystem unabhängig.

Inhaltsverzeichnis

1 Die Skripte-Seite

In besagtem Artikel wurden zwei Skripte, „a5toa4“ und „a5toa4bogen“ vorgestellt. Beide erzeugen aus PDF-Dateien, die im A5-Format vorliegen, neue PDF-Dateien im A4-quer-Format.

Das erste, „a5toa4", arrangiert die A5-Seiten dabei so, dass jeweils zwei aufeinander folgende A5-Seiten nebeneinandern ausgegeben werden. Das ist vor allem

*Sorry if you don't understand German, but currently there's only a German manual, because the idea of this is from a German article at “Die TEXnische Komödie”. But you may try to call “`a5toa4 -help`” to get some useful information.

dann praktisch, wenn man nur einseitig druckt und die Seiten später geteilt werden sollen. Im folgenden wird diese Ausgabe *side-by-side* genannt.

Das zweite, „a5toa4bogen“, erzeugt hingegen ein sogenanntes Booklet. Dabei werden die Seiten so angeordnet, dass nach dem beidseitigen Druck, der ganze Stapel nur noch in der Mitte geknickt werden muss, um eine Art Heft zu erhalten. Daher wird für diese Ausgabe im Weiteren die Bezeichnung *booklet* verwendet.

Ich dacht mir, das könne man in ein einziges Programm gießen. Dieses heißt bei mir dann schlicht „a5toa4.tlu“ bzw. bei optimaler Installation kann es zumindest unter Linux und OSX auch als „a5toa4“ angesprochen werden.

Ruft man das Programm mit der Option „`-V`“ oder „`-version`“ auf, so gibt es lediglich eine Versionsinformation aus.

Beim Aufruf mit Option „`-h`“ oder „`-help`“ wird hingegen eine ausführliche Hilfe zu den Aufrufmöglichkeiten und den Optionen ausgegeben.

Man kann es aber auch mit einer Reihe von Durchführungsoptionen und Namen von PDF-Dateien aufrufen. In diesem Fall wird entweder eine booklet- oder eine side-by-side-Ausgabe erzeugt. Die Wahl, ob eine booklet- oder eine side-by-side-Ausgabe erfolgen soll, erfolgt über die Optionen „`-b`“ oder „`-booklet`“ für booklet bzw. „`-s`“ oder „`-sidebyside`“ für side-by-side. Voreingstellt ist die side-by-side-Ausgabe.

Für die Durchführung wurde auch ein Verbesserungsvorschlag von Christian Justen selbst aufgegriffen. Bei ihm haben die Skripte mit einer Zwischendateien mit dem festen Namen „`cj.tmp`“ gearbeitet. Im Extramfall konnte dies dazu führen, dass Dateien unerwünscht überschrieben und so vernichtet wurden.

In meiner Fassung als texlua-Skript wird mit einem temporären Verzeichnis im aktuellen Arbeitsverzeichnis gearbeitet. Darin wird eine temporäre LaTeX-Datei erzeugt und alle Ausgabedateien eines PDFLaTeX-Laufs abgelegt. Das PDF-Ergebnis des PDFLaTeX-Laufs wird dann wieder in das noch immer aktuelle Arbeitsverzeichnis kopiert. Im Normalfall wird dabei der Basisname (also ohne Extension) der Ursprungsdatei je nachdem, was erzeugt wurde, entweder um „`-booklet.pdf`“ oder „`-sidebyside.pdf`“ ergänzt. Auch dies ist eine Abweichung von Christian Justens Original-Skripten, bei denen immer die Ursprungsdatei vom Ergebnis überschrieben wurde. Dieses Verhalten ist aber ebenfalls zu erreichen, indem man Option „`-o`“ oder „`-overwrite`“ angibt. Nach erfolgreichem PDFLaTeX-Lauf und dem Kopieren des Ergebnisses ins aktuelle Arbeitsverzeichnis, wird das temporäre Verzeichnis samt Inhalt wieder gelöscht.

Eine weitere Änderung bei mir betrifft die Tatsache, dass man für die Verarbeitung mehrerer Dateien nicht mehrere Aufrufe benötigt, man kann auch in einem Aufruf beliebig oft Optionen und Dateien hintereinander anfügen, die dann nacheinander verarbeitet werden. So würde beispielsweise mit dem Aufruf:

```
a5toa4 -b foo.pdf -s bar.pdf -o bum.pdf
```

sowohl „`foo-booklet.pdf`“ als auch „`bar-sidebyside.pdf`“ erzeugt und anschließend „`bum.pdf`“ durch eine side-by-side-Fassung von sich selbst überschrieben werden.

Es ist zu beachten, dass nach jeder erzeugten Datei die Überschreibeinstellung

von Option „-o“ oder „-overwrite“ aus Sicherheitsgründen wieder aufgehoben wird. Will man also mehrere Dateien nacheinander bearbeiten und bei mehreren davon soll die Ursprungsdatei überschrieben werden, so ist vor jeder dieser Dateien die Option erneut zu setzen.

Tatsächlich ist „a5toa4.tlu“ aber nur ein Wrapper für „pfarrei.tlu“. Das wurde deshalb so gemacht, damit das eigentliche Programm leichter durch Versionen in TEXMFLOCAL oder TEXMFHOME ersetzt werden kann, ohne dass jedes Mal das *Binary* ersetzt werden muss. Unter Windows muss man allerdings erst einmal dafür sorgen, dass das Skript als Binary aufgerufen wird.[1]

Eine letzte kleine Änderung meiner Fassung betrifft die Anforderungen an das Seitenformat der Quelldateien. Im Original wurde mit der pdfpages Option noautoscale gearbeitet. Damit wurden die Seiten der Quelldatei nicht an das Seitenformat der Zieldatei angepasst. War die Quelldatei also nicht im Format A5, sondern beispielsweise A6, dann wurden zwei A6-Seiten auf einer A4-quer-Seite platziert. War die Quelldatei im Format A4, passten ihre Seiten nicht einmal auf die A4-quer-Seite. Ich habe diese Option daher weggelassen. Nun werden die Seiten der Quelldatei automatisch ins A5-Format gebracht, bevor sie auf der A4-quer-Seite platziert werden.

Zusammenfassung

Um aus einem Quell-PDF „foo.pdf“ ein Ziel-PDF „foo-sidebyside.pdf“ zu erzeugen, bei dem die Seiten des Quell-PDF aufeinander folgend, nebeneinander auf einer A4-quer-Seite platziert sind, verwendet man:

```
a5toa4 -s foo.pdf
```

Soll die Ziel-PDF hingegen die Quell-PDF überschreiben, so gibt vor dem Dateinamen zusätzlich Option „-o“ an.

Um aus einem Quell-PDF „foo.pdf“ ein Ziel-PDF „foo-sidebyside.pdf“ zu erzeugen, bei dem die Seiten des Quell-PDF so auf einer A4-quer-Seite platziert sind, dass durch Falten in der Mitte ein Heft entsteht, verwendet man:

```
a5toa4 -b foo.pdf
```

Soll die Ziel-PDF hingegen die Quell-PDF überschreiben, so gibt vor dem Dateinamen zusätzlich Option „-o“ an.

Informationen über die verwendete Version von „a5toa4“ erhält man mit:

```
a5toa4 --version
```

Eine Hilfe zum Programm erhält man mit:

```
a5toa4 --help
```

[1] Hierfür wird noch ein Windows-Batch-Jongleur gesucht.

2 Die LaTeX-Seite

Die LaTeX-Seite zu den Skripten aus Abschnitt 1 besteht zunächst einmal aus dem LaTeX-Dokument „`a5ona4.tex`“. Dieses lädt wiederum das LaTeX-Paket „`pfarrei.sty`“. Die eigentliche Funktionalität verbirgt sich darin. Derzeit gibt es fast nur eine Anweisung `\AvToAiv`. Diese erledigt die Hauptarbeit, wobei die erwartet, dass die Original-PDF-Datei, die verarbeitet werden soll, im Makro `\OriginalFile` abgelegt ist. Das Skript sorgt dafür, dass dies geschieht. Darüber hinaus fällt die Entscheidung, ob eine booklet- oder eine side-by-side-Ausgabe erfolgen soll, schlicht über die Frage, ob das Makro `\Booklet` für LaTeX definiert ist oder nicht.

Eine weitere Funktionalität stellt „`pfarrei`.sty“ derzeit noch nicht bereit. Hier sollen allerdings in naher Zukunft einige Dinge implementiert werden, die Christian Justen in seinem Artikel ebenfalls erwähnt hat.

3 Implementierung der LaTeX-Seite

3.1 Das LaTeX-Dokument „`a5toa4.tex`“

Das ist wirklich geradezu trivial:

```
\documentclass[a4paper,landscape]{article}
\usepackage{pfarrei}
\begin{document}
\AvToAiv
\end{document}
```

Das war es schon.

3.2 Das LaTeX-Paket „`pfarrei`“

Als erstes werden die Optionen ausgeführt (obwohl es noch keine gibt):

```
\ProcessOptions*
```

Dann benötigen wir ein paar Pakete:

```
\RequirePackage{ifpdf}
\RequirePackage{pdfpages}
```

`\AvToAiv` Der Name des Makros kommt von `a5toa4`. Es muss sichergestellt sein, dass dies im PDF-Modus verwendet wird.

```
\newcommand*{\AvToAiv}{%
  \ifpdf\else
    \PackageError{pfarrei}{PDF mode needed}{%
      a5toa4 needs the direct PDF mode.\MessageBreak
      Usually this may be activated using either pdflatex, lualatex or
      xelatex.%
    }%
    \input{x.tex}
  \fi
```

```
  \@ifundefined{Booklet}{%
    \includepdf[pages=-,nup=2x1]{\OriginalFile}%
  }{%
    \includepdf[pages=-,booklet]{\OriginalFile}%
  }%
}
```

4 Implementierung der Skripten

4.1 Der kleine Wrapper „a5toa4.tlu“

```
-- $Id: pfarrei.dtx 20 2013-03-10 14:19:06Z mjk $

kpse.set_program_name(arg[-1], 'a5toa4')
require('pfarrei')
```

4.2 Das Haupt-Skript „pfarrei.tlu“

```
local version_number = string.sub( '$Revision: 20 $', 12, -2 )
local action_version = ' r' .. version_number .. '\n' .. [[

Copyright (c) 2013 Markus Kohm.
License: lppl 1.3c or later. See <http://www.latex-project.org/lppl.txt>.
]]
local action_help = [[
action options:

  -h, --help            Print this help message.
  -V, --version         Print the version information.

processing options:
  -b, --booklet         Generate a booklet instead of only two pages side by
                        side onto one page.  The whole booklet will be one
                        signature.
  -s, --sidebyside      Generate only two pages side by side onto one page
                        instead of a booklet.
  -o, --overwrite       Write the output to the <PDF file> instead of appending
                        "-sidebyside.pdf" or "--booklet.pdf" to the basename
                        of <PDF file>
]]
local action_opts = {
   ['-h']           = 'help',
   ['--help']       = 'help',
   ['-V']           = 'version',
   ['--version']    = 'version',
}
local processing_opts = {
   ['-b']           = 'booklet',
   ['--booklet']    = 'booklet',
   ['-s']           = 'sidebyside',
```

```
60    ['--sidebyside'] = 'sidebyside',
61    ['-o']           = 'overwrite',
62    ['--overwrite']  = 'overwrite'
63 }
64
65 -- detect action options and do action
66 local i = 1
67 local action
68 while arg[i] do
69    action = action_opts[arg[i]]
70    i = i+1
71    if     action == 'help' then
72       print( arg[0]..action_version );
73       print( 'Usage: ' .. arg[0] .. ' <action option>' )
74       print( '       ' .. arg[0] .. ' [<processing options>] <PDF file> ...\n' )
75       print( action_help );
76       os.exit( 0 );
77    elseif action == 'version' then
78       print( arg[0] .. action_version );
79       os.exit( 0 );
80    end
81 end
82
83 -- process options and parameters
84 local booklet = false
85 local overwrite = false
86 i = 1
87 while arg[i] do
88    action = processing_opts[arg[i]]
89    if     action == 'booklet' then booklet = true
90    elseif action == 'sidebyside' then booklet = false
91    elseif action == 'overwrite' then overwrite = true
92    elseif action == nil then
93       -- build the temporary tex file
94       local tmpdir = os.tmpdir("pfarrei.XXXXXX" )
95       local tmpfile = string.match( arg[i], '.*/(.*)$') or arg[i]
96       local basename = string.match( tmpfile,'(.*)%.[^.]*$') or tmpfile
97       tmpfile = tmpdir..'/'..basename..'.tex'
98       local file = assert( io.open( tmpfile, 'w' ) )
99       if booklet then assert( file:write("\\def\\Booklet{}\n") ) end
100      assert( file:write("\\def\\OriginalFile{",arg[i],"}\n") )
101      assert( file:write("\\input{a5toa4.tex}\n") )
102      assert( file:flush() )
103      file:close()
104      -- call pdflatex
105      assert( os.execute( 'pdflatex -interaction=batchmode -output-directory='..tmpdir..' '.
106      -- copy the resulting pdf file
107      local srcfile = assert( io.open( tmpdir..'/'..basename..'.pdf', 'rb' ) )
108      if overwrite
109      then
```

```
110         tmpfile = arg[i]
111      else
112         tmpfile = string.match( arg[i], '(.*)%.[^.]*$' ) or arg[i]
113         if booklet
114         then
115            tmpfile = tmpfile..'-booklet.pdf'
116         else
117            tmpfile = tmpfile..'-sidebyside.pdf'
118         end
119      end
120      local destfile = assert( io.open( tmpfile, 'wb' ) )
121      local buffer
122      while true do
123         buffer = srcfile:read(8388608)
124         if buffer==nil then break end
125         assert( destfile:write(buffer) )
126      end
127      assert( destfile:close() )
128      srcfile:close()
129      tmpfile=tmpdir..'/'..basename
130      os.remove( tmpfile..'.aux' )
131      os.remove( tmpfile..'.tex' )
132      os.remove( tmpfile..'.log' )
133      os.remove( tmpfile..'.pdf' )
134      os.remove( tmpdir )
135      overwrite = false
136   end
137   i=i+1
138 end
```

Index

Numbers written in italic refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in roman refer to the code lines where the entry is used.