Das Pfarrei-Paket*

Markus Kohm†

2023/11/22‡

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 inspiriert, eine Sammlung zu beginnen, die entsprechende Dinge leisten. Allerdings habe ich mich 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.

Ergänzt habe ich das ganze durch einige Befehle und Umgebungen, die für die Erstellung von Lied- und Gebetsheftchen für den Gottesdienst praktisch sein könnten. Jedenfalls habe ich selbst Nutzen daraus gezogen. Weiteres ist geplant.

Inhaltsverzeichnis

*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 at least you may try to call “`a5toa4 -help`” to get some useful information.

†komascript at gmx info

‡Version r37

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 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.

`a5toa4.tlu` Ich dachte mir, das könne man in ein einziges Programm gießen. Dieses heißt bei mir dann schlicht „a5toa4.tlu“. Bei optimaler Installation kann es auch als „a5toa4“ angesprochen werden.

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

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

`--booklet` `--sidebyside` 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. Voreingestellt 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.

`--overwrite` Ein Überschreiben der Ursprungsdatei wie bei Christian Justens Original-Skripten kann man aber alternativ ebenfalls 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.

`pfarrei.tlu` 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.

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.

`a5toa4` Zusammenfassung

`-s` 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.

`-b` Um aus einem Quell-PDF „`foo.pdf`“ ein Ziel-PDF „`foo-booklet.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
```

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

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

```
a5toa4 -V
```

`-h` Eine Hilfe zum Programm erhält man mit:

```
a5toa4 -h
```

2 Die LaTeX-Seite

Die LaTeX-Seite zu den Skripten aus Abschnitt 1 besteht zunächst einmal aus dem LaTeX-Dokument „`a5toa4.tex`“. Dieses lädt wiederum das LaTeX-Paket „`pfarrei.sty`“. Die eigentliche Funktionalität verbirgt sich darin.

`\AvToAiv[`⟨*Original-Datei*⟩`]`
`\OriginalFile` Die Anweisung `\AvToAiv` erledigt die Hauptarbeit für `a5toa4`. Die Voreinstellung für das optionale Argument ⟨*Original-Datei*⟩ ist `\OriginalFile`. Das Skript „`a5toa4.tlu`“ setzt dieses Makro entsprechend.

Man kann das LaTeX-Paket pfarrei aber auch per

```
\usepackage{pfarrei}
```

oder

```
\usepackage[booklet]{pfarrei}
```

direkt in seinem Dokument laden. Dann stellt es Umgebungen und Befehle für die Erstellung von Textblättern oder Textheften für Pfarrer, Lektoren und andere Mitwirkende an einem Gottesdienst oder auch für die Gemeinde bereit.

`\ifbooklet{`⟨*Dann-Code*⟩`}{`⟨*Sonst-Code*⟩`}`
Es kann Code davon abhängig ausgeführt werden, ob ein Booklet erzeugt wird oder nicht. Im Falle eines Booklets wird der ⟨*Dann-Code*⟩ ausgeführt, anderenfalls der ⟨*Sonst-Code*⟩. Dies wird auch intern, beispielsweise innerhalb von `\AvToAiv` oder innerhalb der `booklet`...`page`-Umgebungen verwendet, um die entsprechende Entscheidung durchzuführen.

`bookletfrontpage`
`\bookletfrontpagestyle` Mit der Umgebung `bookletfontpage` kann eine Titelseite erstellt werden, die nur bei der Erzeugung eines Booklets ausgegeben wird. Die Umgebung sollte immer am Anfang des Dokuments stehen, anderenfalls produziert sie eine Fehlermeldung. Dies geschieht, weil die Titelseite nun einmal die erste Seite sein sollte. Sollte der Seitenstil `empty` einmal nicht für die Titelseite gewünscht werden, kann man `\bookletfrontpagestyle` auf den Namen des gewünschten Stils umdefinieren.

`\motto{`⟨*Motto*⟩`}`
`\titlepicture{`⟨*Bild*⟩`}`
`\title{`⟨*Titel*⟩`}`
`\parish{`⟨*Gemeinde*⟩`}`
`\date{`⟨*Datum*⟩`}`

`\makebooklettitlepage` Ähnlich zu `\maketitle` bei den Standard-Klassen kann man für Booklets (aufbauend auf der oben erklärten Umgebung `bookletfrontpage` statt auf `titlepage`) mit Hilfe von `\makebooklettitlepage` eine Titelseite für ein Booklet erzeugen. Der Titel besteht aus einem Motto (oben auf der Seite), gefolgt von einem Titelbild, gefolgt von dem Titel, Angaben zur Gemeinde und zum Schluss das Datum. Es ist zu beachten, dass man `\makebooklettitlepage` *nicht* selbst in eine Umgebung packen sollte, da diese Anweisung intern die Umgebung `bookletfrontpage` verwendet. Darüber hinaus ist zu beachten, dass als Titelbild nicht einfach ein Dateinamen angegeben wird, sondern die kompletten Anweisungen, um ein Bild einzuladen oder zu erstellen. Auch muss ggf. ein Grafikpaket selbst geladen werden.

`bookletbackpage`
`bookletemptypage`
`\bookletbackpagestyle`
`\bookletemptypagestyle`

Für ein Booklet kann mit der Umgebung `bookletbackpage` außerdem eine Rückseite definiert werden. Die Definition kann an beliebiger Stelle erfolgen. Die Ausgabe findet auf einer durch vier teilbaren Seite am Ende des Dokuments statt. Dazu werden ggf. leere Seiten eingefügt. Den Inhalt der *leeren Seiten* kann ebenfalls mit einer Umgebung, nämlich `bookletemptypage`, definiert werden. Wie bereits beim Seitenstil für die Vorderseite, kann auch der Seitenstil für die *leeren Seiten* und die Rückseite umdefiniert werden.

`samedoublepage` Wenn der Inhalt dieser Umgebung auf die aktuelle Seite passt, wird er ausgegeben. Passt er nicht auf die aktuelle Seite und die aktuelle Seite ist eine Seite mit einer ungeraden Seitenzahl, dann wird zunächst ein Seitenumbruch durchgeführt. Anschließend wird der Inhalt ausgegeben. Dadurch wird vermieden, dass beispielsweise ein Lektor oder Pfarrer unnötig innerhalb eines Textes umblättern muss. Es ist zu beachten, dass innerhalb dieser Umgebung Fußnoten, Marginalien und Gleitumgebungen nicht korrekt funktionieren.

`\setupprayer{`⟨*Optionen*⟩`}`
`prayer`
`\noresponder`

Die Umgebung `prayer` ist für Gebete gedacht. Die Umgebung hat ein optionales Argument. Dieses ist identisch mit ⟨*Optionen*⟩ gilt dann aber nur lokal für diese Gebetsumgebung. Als ⟨*Optionen*⟩ kann eine Komma-separierte Liste von ⟨*Schlüssel*⟩=⟨*Wert*⟩-Optionen angegeben werden. Folgende Schlüssel werden verstanden:

`leader=`⟨***Vorbeter***⟩**:** Setzt die Voreinstellung für das optionale Argument von `\item` auf ⟨*Vorbeter*⟩.

`responder=`⟨***Antwortende(r)***⟩**:** Setzt die Voreinstellung für das optionale Argument des automatisch als Antwort eingefügten `\item` auf ⟨*Antwortende(r)*⟩.

`response=`⟨***Antwort***⟩**:** Setzt die automatisch eingefügte Antwort auf ⟨*Antwort*⟩.

Eine automatische Antwort wird nur eingefügt, wenn sowohl `responder` als auch `response` angegeben sind. Soll nur für einzelne `\item` keine automatische Antwort erstellt werden, so ist irgendwo innerhalb des entsprechenden `\item` (*nicht* jedoch innerhalb des optionalen Arguments von `\item`) oder danach ein `\noresponder` einzufügen.

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. Trotzdem hat die Verwendung einer solchen zusätzlichen Datei den Vorteil, dass man bei Bedarf lokal auch ein ganz anderes „a5toa4.tex“ speichern kann.

3.2 Das LaTeX-Paket „pfarrei“

`\ifbooklet` Wir haben eine Option, die bestimmt, ob wir ein Booklet erzeugen oder nicht. Später wird das schlicht über das Makro `\ifbooklet` festgestellt. Falls ein Booklet erzeugt werden soll wird das erste, sonst das zweite Argument ausgeführt.

```
\newcommand*{\ifbooklet}{}
\let\ifbooklet\@secondoftwo
\DeclareOption{booklet}{\let\ifbooklet\@firstoftwo}
\DeclareOption{nobooklet}{\let\ifbooklet\@secondoftwo}
```

Dann werden die Optionen ausgeführt:

```
\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}[1][\OriginalFile]{%
  \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
  \ifbooklet{%
    \includepdf[pages=-,booklet]{#1}%
  }{%
    \includepdf[pages=-,nup=2x1]{#1}%
  }%
}
```

`bookletpage` Da mehrfach eine Umgebung für einzelne Seiten benötigt wird, definieren wir dafür eine eigene Umgebung, die dann entsprechend individualisiert wird. Es ist zu beachten, dass diese Umgebung nicht als Umgebung aufgerufen werden sollte, sondern direkt `\bookletpage` und `\endbookletpage`.

```
\newenvironment*{bookletpage}{%
  \edef\reserved@a{bookletpage}%
  \ifx\@currenvir\reserved@a
    \PackageError{pfarrei}{‘bookletpage’ used as ordinary environment}{%
      Please note, that ‘bookletpage’ is a virtual environment
      only.\MessageBreak
      You should not use it directly as an environment, but use
      ‘\string\bookletpage’ at the\MessageBreak
      begin code and ‘\string\endbookletpage’ at the end code of a wrapper
      environment.\MessageBreak
      If you’ll continue, expect several additional errors%
    }%
    \let\bookletpagebox\@tempboxa
    \let\bookletpagestyle\@empty
  \else
    \expandafter\let\expandafter\bookletpagebox
    \csname \@currenvir box\endcsname
    \expandafter\let\expandafter\bookletpagestyle
    \csname \@currenvir style\endcsname
  \fi
  \edef\reserved@a{%
    \noexpand\begin{lrbox}{\bookletpagebox}
    \noexpand\begin{minipage}[t][\textheight][t]{\textwidth}%
      \begingroup
        \def\noexpand\@currenvir{\@currenvir}%
        \def\noexpand\@currenvline{\@currenvline}%
        \noexpand\parskip=\the\parskip
        \noexpand\parindent=\the\parindent
        \noexpand\parfillskip=\the\parfillskip
  }\reserved@a
}{%
```

Das Ende wird in zwei Portionen aufgeteilt:

```
  \endbookletpagebox
  \printbookletpagebox
}
```

`\endbookletpagebox` das Beenden der Erzeugung der Box für die Seite

```
\def\endbookletpagebox{%
        \par
      \endgroup
    \end{minipage}%
  \end{lrbox}%
  \global\setbox\bookletpagebox\box\bookletpagebox
}
```

`\printbookletpagebox` und die Ausgabe der Box, falls ein Booklet erzeugt wird.

```
\newcommand*\printbookletpagebox[1][\@currenvir]{%
  \ifbooklet{%
    \@ifundefined{bookletpagestyle}{}{%
      \ifx\bookletpagestyle\@empty\else\thispagestyle{\bookletpagestyle}\fi%
    }%
    \clearpage\noindent\usebox\bookletpagebox\clearpage
  }{%
    \PackageInfo{pfarrei}{`#1' not printed}%
  }%
}
```

`bookletfrontpage` Umgebung, um eine Booklet-Titelseite zu erzeugen.

`\bookletfrontpagebox` `\bookletfrontpagestyle` Es ist nur eine Seite erlaubt. Also packen wir das ganze in eine Box. Dafür brauchen wir eine solche:

```
\newsavebox\bookletfrontpagebox
\newcommand*{\bookletfrontpagestyle}{empty}
\newenvironment*{bookletfrontpage}{%
  \bookletpage
}{%
  \endbookletpagebox
  \clearpage
  \ifbooklet{%
    \ifnum\c@page>\@ne
      \PackageError{pfarrei}{Booklet front page not first page}{%
        The booklet font page should be the first page, but it seems, that it
        is\MessageBreak
        page no. \the\c@page. \space Maybe you should put it immediately after
        `\string\begin{document}.%
        Nevertheless, if you'll continue it will be printed here%
      }%
    \fi
    \printbookletpagebox
  }{%
    \PackageInfo{pfarrei}{Booklet front page ignored}%
  }%
}
```

`bookletbackpage` `\bookletbackpagebox` `\bookletbackpagestyle` Im Prinzip ist die Booklet-Rückseite fast wie bei der Titelseite, allerdings fügen wir hier so viele Leerseiten ein, dass wir wirklich auf einer durch 4 teilbaren Seite landen.

bookletemptypage
\bookletemptypagebox
\bookletemptypagestyle

Die Umgebung für die konfigurierbare *Leerseite*, besitzt keine eigene Ausgabe, sondern sammelt tatsächlich nur den Inhalt. Die Ausgabe erfolgt dann bei der Ausgabe der Booklet-Rückseite.

```
\newsavebox\bookletemptypagebox
\newcommand*{\bookletemptypagestyle}{empty}
\newenvironment*{bookletemptypage}{%
  \bookletpage
}{%
  \endbookletpagebox
}
\newsavebox\bookletbackpagebox
\newcommand*{\bookletbackpagestyle}{empty}
\newenvironment*{bookletbackpage}{%
  \bookletpage
}{%
  \endbookletpagebox
  \if@filesw\immediate\write\@mainaux{\string\printbookletbackpage}\fi
}%
```

\printbookletbackpage
\@printbookletbackpage

Die eigentliche Ausgabe der letzten Seite geschieht über einen Eintrag in der aux-Datei. Das ist zwar nicht ganz sauber (und scrlfile meckert deshalb ggf.), aber die Standard-Klasse letter macht das ähnlich für die Etiketten. Also hoffen wir einfach, dass es gut geht.

```
\newcommand*{\printbookletbackpage}{}
\newcommand*{\@printbookletbackpage}{%
  \ifbooklet{%
    \clearpage
    \let\bookletpagestyle\bookletemptypagestyle
    \ifvoid\bookletemptypagebox
      \let\bookletpagebox\strutbox
    \else
      \let\bookletpagebox\bookletemptypagebox
    \fi
    \@tempcnta=\c@page
    \divide\@tempcnta by 4
    \multiply\@tempcnta by 4
    \ifnum \@tempcnta=\c@page\else
      \advance\@tempcnta by 4
      \@whilenum \c@page<\@tempcnta\do{%
        \printbookletpagebox
      }%
    \fi
    \let\bookletpagestyle\bookletbackpagestyle
    \let\bookletpagebox\bookletbackpagebox
    \printbookletpagebox
  }{%
    \PackageInfo{pfarrei}{Booklet back page ignored}%
  }%
}
```

Das Drucken darf während des Lesens der aux-Datei in \begindocument nicht erfolgen. Daher definieren wir es erst danach entsprechend um. Dadurch ist es während des Lesens der aux-Datei in \enddocument funktionsfähig.

```
\AtBeginDocument{\let\printbookletbackpage\@printbookletbackpage}
```

\makebooklettitlepage
\motto
\@motto
\titlepicture
\@titlepicture
\title
\@title
\parish
\@parish
\date
\@date

Der Standard-Titel für Booklets besteht aus einem Motto, einem Bild, einem Titel, der Gemeinde und dem Datum. Wir gehen davon aus, dass die Klasse bereits \date, \@date, \title und \@title zur Verfügung stellt. Allerdings ergibt die Voreinstellung \today für das Datum hier wenig Sinn, weshalb die Voreinstellung in einen leeren Wert geändert wird. Den Rest machen wir komplett selbst:

```
\newcommand*{\motto}[1]{\gdef\@motto{#1}}
\newcommand*{\@motto}{}
\newcommand*{\titlepicture}[1]{\gdef\@titlepicture{#1}}
\newcommand*{\@titlepicture}{}
\providecommand*{\title}[1]{\gdef\@title{#1}}
\providecommand*{\@title}{}
\newcommand*{\parish}[1]{\gdef\@parish{#1}}
\newcommand*{\@parish}{}
\providecommand*{\date}[1]{\gdef\@date{#1}}
\def\@date{}
\newcommand*{\makebooklettitlepage}{%
  \begin{bookletfrontpage}
    \parskip.5\baselineskip
    \parindent\z@
    \parfillskip \z@ \@plus 1fil
    \centering
    \ifx\@motto\@empty\else{\Huge\@motto\par}\fi
    \vfill
    \ifx\@titlepicture\@empty\else\@titlepicture\par\fi
    \vfill
    \parskip\z@
    \Huge
    \@title\par
    \@parish\par
    \@date\par
    \ifx\@title\@empty\ifx\@parish\@empty\ifx\@date\@empty\null\fi\fi\fi
  \end{bookletfrontpage}
}
```

samedoublepage
\samedoublepage@save@hbox

Es wird etwas umständlich eine vertikale Box gespeichert,

```
\let\samedoublepage@save@hbox\hbox
\newenvironment*{samedoublepage}{%
  \par
  \let\hbox\vbox
  \begin{lrbox}{\@tempboxa}%
    \let\hbox\samedoublepage@save@hbox
}{%
  \end{lrbox}%
  \let\hbox\samedoublepage@save@hbox
```

um diese dann in Stücke zu zerschneiden, die auf eine Seite passen,

```
  \@tempdima\ifdim\pagegoal=\maxdimen\textheight
            \else\dimexpr\pagegoal-\pagetotal\fi
  \ifdim \@tempdima
        <\dimexpr\ht\@tempboxa+\dp\@tempboxa\relax
    \ifodd\c@page
```

Wobei aber nicht auf ungeraden Seiten mit nur einem Stück des Kuchens begonnen wird.

```
      \newpage
      \@tempdima\textheight
    \else \typeout{even page}%
    \fi
    \@whiledim \@tempdima
              <\dimexpr\ht\@tempboxa+\dp\@tempboxa\relax\do{%
      \splitmaxdepth\dp\strutbox
      \splittopskip\topskip
      \setbox\z@\vsplit\@tempboxa to \@tempdima
      \usebox\z@
      \newpage
      \@tempdima\textheight
    }%
  \fi
```

Und am Ende nicht vergessen, den Rest der Box auch noch auszugeben.

```
  \ifvoid\@tempboxa\else\usebox\@tempboxa\fi
}
```

`prayer`
`\setupprayer`

Die Gebetsumgebung wird mit Hilfe von keyval definiert. Sie kann außer mit dem optionalen Argument auch jederzeit mit `\setupprayer` konfiguriert werden.

`\prayer@responder`
`\prayer@response`
`\prayer@leader`

Dafür werden mehrere interne Makros benötigt, um die gewünschten Werte aufzunehmen.

```
\RequirePackage{keyval}
\define@key{pfarrei.prayer}{response}{\def\prayer@response{#1}}
\define@key{pfarrei.prayer}{responder}{\def\prayer@responder{#1:}}
\define@key{pfarrei.prayer}{leader}{\def\prayer@leader{#1:}}
\newcommand*{\prayer@responder}{}
\newcommand*{\prayer@response}{}
\newcommand*{\prayer@leader}{}
\newcommand*{\setupprayer}{%
  \setkeys{pfarrei.prayer}%
}
```

`\ifprayer@firstitem`
`prayer@response@item`

Da die Antwort beim ersten `\item` noch nicht ausgegeben werden darf, sondern nur nach den nachfolgenden, muss die Information, ob es das erste `\item` ist, über einen Schalter gespeichert werden.

```
\newif\ifprayer@firstitem
\newcommand*{\prayer@response@item}{%
  \ifprayer@firstitem\else
```

```
    \ifx\prayer@responder\@empty\else
      \ifx\prayer@response\@empty\else
        \prayer@save@item[\prayer@responder] \prayer@response
      \fi
    \fi
  \fi
}
```

\prayer@item
\prayer@save@item

Die \item-Anweisung von prayer ist etwas anders als von anderen Listen-Umgebungen. Sie baut jedoch auf der Originaldefinition auf. Daher muss die Originaldefinition gespeichert und entsprechend erweitert verwendet werden.

```
\newcommand*{\prayer@item}[1][\prayer@leader]{%
  \prayer@response@item
  \prayer@firstitemfalse
  \prayer@save@item[{#1}]%
}
```

\noresponder

Das Abschalten der nächsten automatischen Antwort geschieht einfach, indem so getan wird, als wäre das nächste \item wieder das erste.

```
\newenvironment*{prayer}[1][]{%
  \begin{description}
    \begingroup
      \def\@currenvir{prayer}%
      \setupprayer{#1}%
      \let\prayer@save@item\item
      \let\item\prayer@item
      \prayer@firstitemtrue
      \let\noresponder\prayer@firstitemtrue
}{%
      \prayer@response@item
    \endgroup
  \end{description}
}
```

4 Implementierung der Skripten

4.1 Der kleine Wrapper „a5toa4.tlu“

```
-- $Id: pfarrei.dtx 37 2023-11-22 18:54:26Z ps $

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

4.2 Das Haupt-Skript „pfarrei.tlu“

```
local version_number = string.sub( '$Revision: 37 $', 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',
   ['--sidebyside'] = 'sidebyside',
   ['-o']           = 'overwrite',
   ['--overwrite']  = 'overwrite',
   ['-d']           = 'debug',
   ['--debug']      = 'debug',
}

-- detect action options and do action
local i = 1
local action
while arg[i] do
   action = action_opts[arg[i]]
   i = i+1
   if     action == 'help' then
      print( arg[0]..action_version );
      print( 'Usage: ' .. arg[0] .. ' <action option>' )
      print( '       ' .. arg[0] .. ' [<processing options>] <PDF file> ...\n' )
      print( action_help );
      os.exit( 0 );
   elseif action == 'version' then
      print( arg[0] .. action_version );
      os.exit( 0 );
```

```
298    end
299 end
300
301 -- process options and parameters
302 local booklet = false
303 local overwrite = false
304 local debug = false
305 i = 1
306 while arg[i] do
307    action = processing_opts[arg[i]]
308    if     action == 'booklet' then booklet = true
309    elseif action == 'sidebyside' then booklet = false
310    elseif action == 'overwrite' then overwrite = true
311    elseif action == 'debug' then debug = true
312    elseif action == nil then
313       -- build the temporary tex file
314       local tmpdir = os.tmpdir("pfarrei.XXXXXX" )
315       local tmpfile = string.match( arg[i], '.*/(.*)$') or arg[i]
316       -- pdflatex's -output-directory search for source pdf works with path specification but i
317       -- when simple file name in the current working directory is provided, we need to provide
318       local local_source=''
319       if tmpfile == arg[i] then local_source = '../' end
320       local basename = string.match( tmpfile,'(.*)%.[^.]*$') or tmpfile
321       tmpfile = tmpdir..'/'..basename..'.tex'
322       local file = assert( io.open( tmpfile, 'w' ) )
323       if booklet then assert( file:write("\\PassOptionsToPackage{booklet}{pfarrei}\n") ) end
324       assert( file:write("\\def\\OriginalFile{"..local_source,arg[i],"}\n") )
325       assert( file:write("\\input{a5toa4.tex}\n") )
326       assert( file:flush() )
327       file:close()
328       -- call pdflatex
329       assert( os.execute( 'pdflatex -interaction=batchmode -output-directory='..tmpdir..' '..tm
330       -- copy the resulting pdf file
331       local srcfile = assert( io.open( tmpdir..'/'..basename..'.pdf', 'rb' ) )
332       if overwrite
333       then
334          tmpfile = arg[i]
335       else
336          tmpfile = string.match( arg[i], '(.*)%.[^.]*$' ) or arg[i]
337          if booklet
338          then
339             tmpfile = tmpfile..'-booklet.pdf'
340          else
341             tmpfile = tmpfile..'-sidebyside.pdf'
342          end
343       end
344       local destfile = assert( io.open( tmpfile, 'wb' ) )
345       local buffer
346       while true do
347          buffer = srcfile:read(8388608)
```

```
348         if buffer==nil then break end
349         assert( destfile:write(buffer) )
350      end
351      assert( destfile:close() )
352      srcfile:close()
353      if debug
354      then
355         print('DEBUG: Temporary files in: '..tmpdir);
356      else
357         tmpfile=tmpdir..'/'..basename
358         os.remove( tmpfile..'.aux' )
359         os.remove( tmpfile..'.tex' )
360         os.remove( tmpfile..'.log' )
361         os.remove( tmpfile..'.pdf' )
362         os.remove( tmpdir )
363      end
364      overwrite = false
365   end
366   i=i+1
367 end
```