OpBible – Technical Documentation

The code of the `opbible.opm` macro file is described here.

opbible.opm
```
\_codedecl \processbooks {OpBible: macros for creating annotated Bible}

\_message{This is OP-Bible, version <0.19 Feb 2023>}
```

1 Preparatory work

Loading packages.

opbible.opm
```
\_load[vlna]  % single-letter prepositions and splitting hyphen managed specially in Czech
\_load[mte]   % micro typographical extensions

\_namespace{opb}
```

Basic settings of TEX parameters.

opbible.opm
```
\_newdimen\.lrmargin  \.lrmargin=10mm
\_margins/2 a4 (23,27,20,20)mm

\_typosize[11/13]       % typesetting size of Bible text
\_hyperlinks\Blue\Blue  % hyperlinks activated

\_parindent=20pt
\_nopagenumbers
\_mte_enablemte         % micro typographical extensions enabled
\_vlna_singlechars {Czech}{AaIiVvOoUuSsZzKk}  % lowercase "a" added to this family

\_showboxbreadth=0
\_let\notecolor=\Red

\_def\LightGrey {\_setcmykcolor{0 0 0 .1}}
\_def\LiRed {\_setcmykcolor{0 .2 .2 0}}
```

Fonts.

opbible.opm
```
\_fontfam[lm]
\_fontfam[Heros]        % fonts for notes
\_isfile{f-biblon.opm}\_iftrue
   \_fontfam[biblon]    % fonts for Bible text
\_else
   \_let\Biblon=\LMfonts
\_fi

\_fontdef\.bookfont{\_setfontsize{at19.pt}\_bf}
\_fontdef\.chapfont{\_setfontsize{at13.pt}\_bf}
\_fontdef\.markfont{\_setfontsize{at7pt}\_rm}
\_fontdef\.captionfont{\Heros\cond\_setfontsize{at8pt}\_bf}
\_def\.headfont{\.Biblon\_setfontsize{at10pt}\_rm}
\_nsprivate \Biblon ;
```

Auxiliary macros. `\.printwarn {`⟨*text*⟩`}` prints warning. `\.sedef {`⟨*name*⟩`}{`⟨*body*⟩`}` is expanded `\sdef`. `\.myaddto {`⟨*macro-name*⟩`}{`⟨*text*⟩`}` adds ⟨*text*⟩ to `\`⟨*macro-name*⟩ globally. Moeower it defines the undefined macro by `\sdef{`⟨*macro-name*⟩`}{`⟨*text*⟩`}`.

opbible.opm
```
\_let\.printwarn=\_opwarning
\_def \.sedef #1{\_ea\_edef \_csname#1\_endcsname}
\_long\_def\.myaddto#1#2{\_ifcsname#1\_endcsname
   \_gobal\_ea\_addto\_csname#1\_endcsname{#2}\_else \_global\_sdef{#1}{#2}\_fi}
```

We prepare expandable if-macros:
`\.isspacein` ⟨*text*⟩ `\_iftrue` is true if ⟨*text*⟩ includes a space.
`\.iscolonin` ⟨*text*⟩`:\_iftrue` is true if ⟨*text*⟩ includes a colon.
`\.isdivisin` ⟨*text*⟩`-\_iftrue` is true if ⟨*text*⟩ includes a divis.

opbible.opm

```
\_def\.isspacein #1 #2\_iftrue{\_isempty{#2}\_iffalse}
\_def\.iscolonin #1:#2\_iftrue{\_isempty{#2}\_iffalse}
\_def\.isdivisin #1-#2\_iftrue{\_isempty{#2}\_iffalse}
```

2 The main loop over Bible books

The `\processbooks` macro does two loops over all marks in `\printedbooks`. The macro `\printedbooks` is a list of ⟨*a-marks*⟩ of Bible books separated by spaces and it must be defined in the main file. The `\_useit` trick is used here in order we want to add ⟨*space*⟩`{}` at the end of the expanded `\printedbooks`. The first loop body sets `\pbook!`⟨*a-mark*⟩ used for hyperlinks. The second loop body does:

- Defines `\amark` as ⟨*a-mark*⟩ (an actual mark of the book used in the text).
- Defines `\bmark` as ⟨*b-mark*⟩ (a mark of the book used in file names).
- Defines `\.btit` as the book title.
- Saves ⟨*a-mark*⟩ to the `\.currbook` macro.
- Calls `\.newbook{`⟨*a-mark*⟩`}`
- Prints title of the book to the terminal and to the log.
- Calls `\bex!<a-mark>` in order to apply the `\BookException` data.
- Inputs introduction file if it exists. The real `\input` and formatin of the introduction text is done by the `\.printintro` macro.
- Inputs format definition file if it exists. Information is saved to the TEX memory.
- Inputs notes file if it exists. The notes are saved to the TEX memory.
- Calls `\bpr!<a-mark>` in order to apply the `\BookPre` data.
- Inputs `txs` file with original text of the Bible using `\.bibleinput`, i.e. prints the text from `txs` file with notes from the TEX memory.
- Calls `\bpo!<a-mark>` in order to apply `\BookPost` data.

Note that the macros `\introfile`, `\fmtfile`, and `\notesfile` give the location of aprropriate files and these macros must be defined by the user in the main file.
Note2: each book of the Bible is processed in the group. It means that all data from notes, formats etc. are stored in the memory only temporary for processing single book. After the Book is finalized, the TEX memory is freed.

opbible.opm

```
\_def\.processbooks {\_par
   \_ifx\tmark\_undefined \_def\tmark{none}\_fi
   \.checknochapbooks
   \_useit{\_ea\.processbooksA \printedbooks} {}
   \_useit{\_ea\.processbooksB \printedbooks} {}
}
\_def\.processbooksA #1 {%
   \_if\_relax#1\_relax \_else \_sxdef{pbook!#1}{}\_ea\.processbooksA \_fi
}
\_def\.processbooksB #1 {%
   \_if\_relax#1\_relax \_else
      \_edef\amark{#1}
      \_edef\bmark{\_cs{f!#1}}
      \_edef\.btit{\_cs{btit!#1}}
      \_begingroup
         \_edef\.currbook{#1}
         \.newbook{#1}
         \_wterm{** \_cs{btit!#1} {#1} (\string\tmark: \tmark) **}
         \_cs{bex!#1}
         \_isfile{\introfile}\_iftrue \.printintro
         \_else \.printwarn{File with introduction text \introfile\_space not found}\_fi
%         \.CommentedBook{#1}
         \_isfile{\fmtfile}\_iftrue \_input{\fmtfile}
         \_else \.printwarn{File with format info \fmtfile\_space not found}\_fi
         \_isfile{\notesfile}\_iftrue \_input{\notesfile}
         \_else \.printwarn{File with notes \notesfile\_space not found}\_fi
```

```
          \_cs{bpr!#1}
          \.bibleinput{\txsfile}
          \.chapafter % material after the last chapter
          \_cs{bpo!#1}
       \_endgroup
       \_ea \.processbooksB
    \_fi
}
\_nspublic \processbooks ;
```

`\.newbook{`⟨*a-mark*⟩`}` ejects previous page, prepeares header and prints the book title.

opbible.opm

```
\_def\.newbook#1{\_vfil\_supereject
   \_let\.prelinkB=\.currbook \.chapnum=0
   \_def\.prelinkC{0}\_def\.prelinkV{0}
   \_global\_headline={\_hfil \_ea\.setheadline\_ea{\.btit}}
   \_line{\_hss\.bookfont\.btit\_hss}
   \_label[cref!#1]\_wlabel{#1}
   \_par\_nobreak\_medskip
}
```

`\.setheadline{`⟨*book-title*⟩`}` sets `\_headline`. It is re-set for each new book by `\.newbook`.
The `\bibname` can be defined by user as a name of the translating variant of the Bible. If it is not defined then it is empty by default.

opbible.opm

```
\_def\.setheadline#1{\_global\_headline={\.headfont
   \_ifodd\_pageno
       \_rlap{\_it\bibname\_hss}%
       \_hfil \_the\_pageno\_hfil
       \_hbox to\.lrmargin{\_hss\_bf#1\_ifx^\_botmark^\_else\_space \_botmark\_fi}%
       \_kern-\.lrmargin
   \_else
       \_kern-\.lrmargin
       \_hbox to\.lrmargin{\_bf#1 \_firstmark\_hss}%
       \_hfil \_the\_pageno\_hfil
       \_llap{\_hss\_it\bibname}%
   \_fi
   }
}
\_def\bibname{}
```

We want `<Fm 4>` to be a link to Fm/1:4 because it is a single-chapter book. Compare `<Gn 4>` which is a link to Gn/4:1. There is a list of single-chapter books `\nochapbooks`. User must define it. The marks of these single-chapter books are separated by spaces here. The first and the last space are added to the `\nochapbooks` macro because we need them in `\.brefBookChapter`. The `\.checknochapbooks` macro does it, moreower, it checks if the `\nochapbooks` is defined. If not, it prints warning.

opbible.opm

```
\_def\.checknochapbooks {%
   \_ifx\nochapbooks\_undefined
      \.printwarn{\_noexpand\nochapbooks (boks without chapters) undefined.}%
      \_def\nochapbooks{}%
   \_else  \_edef\nochapbooks{\_space\nochapbooks\_space}\_fi
}
```

3 Book titles

The macro `\BookTile` ⟨*a-mark*⟩ ⟨*b-mark*⟩ `{`⟨*title*⟩`}` declares titles of each Bible books. The ⟨*a-mark*⟩ is an actual book mark used in printed text. The ⟨*b-mark*⟩ can be used in file names as `\bmark`. The mapping is done here: `\def\btit!`⟨*a-mark*⟩`{`⟨*title*⟩`}`, `\def\f!`⟨*a-mark*⟩`{`⟨*b-mark*⟩`}`.

The macro is defined as `\outer` because we don't want to see obscure errors due to missing a space after ⟨*b-mark*⟩ or ⟨*a-mark*⟩.

opbible.opm

```
\_def\genbooks{}
\_def\.BookTitle #1 #2 #3{%
   \_sxdef{btit!#1}{#3}\_sxdef{f!#1}{#2}\_sxdef{fb!#2}{#1}%
   \_addto\genbooks{#2 }%
}
```

The \BookException ⟨*a-mark*⟩ {⟨*code*⟩} macro adds the ⟨*code*⟩ to the \bex!⟨*a-mark*⟩ macro. It is used in \processbooks loop in the group before files are read. You can redefine some filenames or something more special here.
Macros \BookPre ⟨*a-mark*⟩ {⟨*code*⟩} and \BookPost ⟨*a-mark*⟩ {⟨*code*⟩} are defined similarly. They add ⟨*code*⟩ to the \bpr!⟨*a-mark*⟩ and to the \bpo!⟨*a-mark*⟩ macros repectively.

opbible.opm

```
\_outer\_long\_def\.BookException #1 #2{\.myaddto{bex!#1}{#2}}
\_outer\_long\_def\.BookPre       #1 #2{\.myaddto{bpr!#1}{#2}}
\_outer\_long\_def\.BookPost      #1 #2{\.myaddto{bpo!#1}{#2}}

\_nspublic \BookTitle \BookException \BookPre \BookPost ;
```

The \ChapterPre{⟨*code*⟩} and \ChapterPost{⟨*code*⟩} inserts ⟨*code*⟩ before each chapter and after each chapter. The ⟨*code*⟩ is the same for each chapter, it does not vary depending on the Book or Chapter number.

opbible.opm

```
\_long\_def\.ChapterPre  #1{\_def\.chapbefore{#1}}
\_long\_def\.ChapterPost #1{\_def\.chapafter{#1}}

%\_outer\_def\ChapterPre  {\.ChapterPre}
%\_outer\_def\ChapterPost {\.ChapterPost} % be done at the end of this file
```

4 Actions

We create the output in two steps. First step: the data from \Note etc. are read and saved to the TEX memory. For each such data element the "action" is registered to a list of actions of the given verse. Each Bible verse has its list of actions. The second step: the Bible verses are read from a .txs file and all appropriate actions (registered to this verse) are processed before the verse text is printed. These actions can modify the selected parts of the verse text.
\alist!⟨*full-vref*⟩ is the list of actions associated with the verse ⟨*full-vref*⟩. The ⟨*full-vref*⟩ is full reference to the verse in the format ⟨*book-mark*⟩/⟨*chapter-num*⟩:⟨*verse-num*⟩
\.newaction{⟨*full-vref*⟩}{⟨*action-body*⟩} allocates new action.

opbible.opm

```
\_def\.newaction#1#2{%
   \_unless\_ifcsname alist!#1\_endcsname \_sdef{alist!#1}{}\_fi
   \_ea\_addto\_csname alist!#1\_endcsname{#2}%
}
```

A typical "action" is \.replpre. The actions are processed for each Bible verse when the verse text is saved to the \.buff macro. The \.buff macro is processed after all actions of given verse are done.
\.replpre{⟨*prefix*⟩}{⟨*text*⟩}{⟨*fail*⟩} replaces first occurrence of ⟨*text*⟩ by ⟨*prefix*⟩^⟨*text*⟩^ in \.buff macro. If the ⟨*text*⟩ is empty then ⟨*prefix*⟩^^ is inserted at the beginning of the \.buff.
If ⟨*text*⟩ does not exists then ⟨*fail*⟩ is processed. It can report failed ⟨*text*⟩ by the \.text macro.

opbible.opm

```
\_def\.replpre#1#2#3{%
   \_ifx^#2^\_def\.tmp{#1{}}\_ea\_ea\_ea\_def\_ea\_ea\_ea\.buff\_ea\_ea\_ea{\_ea\.tmp\.buff}%
   \_else
      \_def\.replpredo##1#2##2\_end{%
         \_ifx\_end##2\_end \_def\.text{#2}#3% <fail>
         \_else \.replsave ##1#1{#2}##2\_end \_fi
      }%
      \_def\.replsave##1#2\_end{\_def\.buff{##1}}%
      \_ea\.replpredo\.buff#2\_end
   \_fi
}
```

\.replprepost{⟨*text*⟩}{⟨*pre*⟩}{⟨*post*⟩}{⟨*fail*⟩} searches ⟨*text*⟩ in \.buff and adds ⟨*pre*⟩ before and ⟨*post*⟩ after the ⟨*text*⟩. If the ⟨*text*⟩ is not found then ⟨*fail*⟩ is executed. The \.replprepost is used by \fmtins (with empty ⟨*pre*⟩) because we want to insert the ⟨*post*⟩ material directly.

opbible.opm

```
\_def\.replprepost#1#2#3#4{%
   \_def\.replprepostdo##1#1##2\_end{%
      \_ifx\_end##2\_end \_def\.text{#1}#4% <fail>
      \_else \.replsave ##1#2#1#3##2\_end \_fi
   }%
   \_def\.replsave##1#1\_end{\_def\.buff{##1}}%
   \_ea\.replprepostdo\.buff#1\_end
}
```

5 The \Note macro

The first parameter of the \Note macro is ⟨*gen-vref*⟩. It is generalized reference to the Bible verse. It can be ⟨*chapter-num*⟩:⟨*verse*⟩ (the ⟨*book-mark*⟩ is appended from the \.currbook macro) or ⟨*chapter-num*⟩:⟨*verse-from*⟩-⟨*verse-to*⟩ (only ⟨*verse-from*⟩ is used for generating ⟨*gen-vref*⟩.
\.gentovref{⟨*gen-vref*⟩} expands to ⟨*full-vref*⟩.

opbible.opm

```
\_def\.gentovref#1{\.currbook/\.gentovrefA#1-\end}
\_def\.gentovrefA#1-#2\end{#1}
```

\.renumvref ⟨*full-vref*⟩_relax does re-caclulating of ⟨*full-vref*⟩ using \renum data.

opbible.opm

```
\_def\.renumvref #1/#2\_relax{#1/\_trycs{rn!\tmark!#1/#2}{#2}}
```

The ⟨*word*⟩ given as a parameter of the \Note macro (see bellow) is used as a word phrase which should be be searched in the given verse text. This parameter ⟨*word*⟩ is transformed first by expansion of \.transformword{⟨*word*⟩} to the ⟨*tword*⟩ variant and the ⟨*tword*⟩ is actually used for searching. The \.transformword{⟨*word*⟩} expands to the variant of the ⟨*word*⟩ declared by \.vdef. If not declared then it expands to the ⟨*word*⟩ itself, i.e ⟨*tword*⟩ is equal to ⟨*word*⟩ in this case.

opbible.opm

```
\_def\.transformword#1{%
   \_ifcsname v!\tmark!#1\_endcsname \_lastnamedcs
   \_else #1\_fi
}
```

\Note ⟨*gen-vref*⟩ ⟨*space*⟩ {⟨*word*⟩} ⟨*text*⟩ \par transforms ⟨*word*⟩ to the ⟨*tword*⟩ (see above), saves ⟨*text*⟩ and activates replace-action of ⟨*tword*⟩ to \.doNote{⟨*note-num*⟩}{⟨*tword*⟩} in given verse.
There is an alternative syntax \Note<gen-vref> ⟨*space*⟩ {⟨*word*⟩}={⟨*pword*⟩} ⟨*text*⟩ \par If ⟨*pword*⟩ is given then it is printed in the note instead ⟨*tword*⟩. More precisely: transformed ⟨*word*⟩ is used for searching (and it is kept in the verse unchanged) but ⟨*pword*⟩ is printed in the note.
The \ww can precede \Note. If it is true then the ⟨*word*⟩ is prepared in \.nextww and ⟨*pword*⟩ is in \.nextwwA. Otherwise, the macros \.nextww and \.nextwwA are undefined.
\Note does exactly following:

- Calculates ⟨*full-vref*⟩ using \.gentovref{⟨*genv-ref*⟩} and svese it to \.fullvref.
- If the verse number of ⟨*full-vref*⟩ is zero, we want to insert the note-text before the chapter. This is one by the \.NoteB macro.
- Allocates new ⟨*note-num*⟩, i.e. \.notenum is ⟨*note-num*⟩.
- Modifies ⟨*full-vref*⟩ if \renum was declared using \.renumvref and saves the result to \.fullvrefm.
- Uses \.nextww and \.nextwwA as ⟨*tword*⟩ and ⟨*pword*⟩ if they are defined.
- Otherwise transforms ⟨*word*⟩ to ⟨*tword*⟩ by \.transformword.
- Reads ⟨*pword*⟩ (word to be printed in the note) by \.NoteA if the alternative syntax with ={⟨*pword*⟩} is used. Else ⟨*pword*⟩ is equal to ⟨*tword*⟩. Use it only if \.nextww is undefined.
- Defines \notetext!⟨*note-num*⟩ as ⟨*text*⟩.
- Defines \noteref!⟨*note-num*⟩ as ⟨*full-vref*⟩ re-calulated by \renum.
- Defines \notepre!⟨*note-num*⟩ as numeric part of modified ⟨*full-vref*⟩. and calculates ⟨*from*⟩-⟨*to*⟩ part (if exists in ⟨*gen-vref*⟩) using \.renumlabel macro. This is printed prefix of the \Note.
- Defines \pword!⟨*note-num*⟩ as ⟨*pword*⟩,
- Does \.newaction{⟨*full-vref*⟩}{\.replpre{\.doNote{⟨*note-num*⟩}}
  {⟨*tword*⟩}{\.notefail{⟨*note-num*⟩}}}.
  This is done by \.AddNote{⟨*full-vref*⟩}{⟨*note-num*⟩}{⟨*tword*⟩}.

Note that \Note is defined as \outer in order to report correctly typical mistakes with missing empty line the text of a previous \Note.

opbible.opm

```
\_newcount\.notenum
\_def\.Note #1 #2{%
   \_edef\.fullvref{\.gentovref{#1}}%
   \_ea\.isversezero\.fullvref\_iftrue
      \_ea\.NoteB
   \_else
      \_incr\.notenum
      \_edef\.fullvrefm{\_ea\.renumvref\.fullvref\_relax}%
      \_def\.tmp{#1}\.sedef{notepre!\_the\.notenum}{\_ea\.renumlabel\.fullvrefm\_relax}%
```

```
      \_ifx\.nextww\_undefined
         {\_def\.printwarn##1{}\_xdef\.tword{\.transformword{#2}}}%
      \_else \_xdef\.tword{\.nextww}\_fi
      \_afterfi{\_isnextchar={\.NoteA}{\.NoteA={}}}%
   \_fi
}
\_def\.NoteA=#1#2% #2 separated by \par or \_par:

{%
   \_sdef{notetext!\_the\.notenum}{\_ignorespaces#2}%
   \.sedef{noteref!\_the\.notenum}{\.fullvrefm}%
   \_ifx\.nextww\_undefined
      \_ifx^#1^\_sdef{pword!\_the\.notenum\_ea}\_ea{\.tword}\_else \_sdef{pword!\_the\.notenum}{#1}\_fi
   \_else
      \_sdef{pword!\_the\.notenum\_ea}\_ea{\.nextwwA}%
      \_let\.nextww=\_undefined \_let\.nextwwA=\_undefined
   \_fi
   \.reducetword
   \_ea\.addNote\_expanded{{\.fullvrefm}{\_the\.notenum}{\.tword}}%
}
\_def\.addNote#1#2#3{%
   \_ifx^#3^% \.tword is empty
      \_edef\.tmp{\_cs{notepre!#2}}%
      \_ea \.isdivisin\.tmp-\_iftrue
         \.newaction{#1}{\.replpre{\.doNote{#2}}{}{}}%
      \_else
         \.newaction{#1}{\_addto\.prebuff{\.doCNote{#2}{}}}%
      \_fi
   \_else
      \.newaction{#1}{\.replpre{\.doNote{#2}}{#3}{\.notefail{#2}}}%
   \_fi
}
%\_outer\_def\Note{\.Note} % will be done at the end of this macro file
```

The \.NoteB ⟨*text*⟩ \par does not register any action to the verse but defines \chapnote!⟨*full-vref*⟩ as the ⟨*text*⟩. This chapter note will be printed before the chapter starts.

opbible.opm

```
\_def\.NoteB #1% #1 separated by \par or \_par

{%
   \_sdef{chapnote!\.fullvref}{\_ignorespaces#1}%
}
\_def\.isversezero#1/#2:#3\_iftrue{\_ifnum #3=0 }
```

\.renumlabel ⟨*full-vref*⟩_relax expands to the numeric part of ⟨*full-vref*⟩ and appends the `--`⟨*to*⟩ part if the \.tmp macro is in the format ⟨*chapter*⟩`:`⟨*from*⟩`-`⟨*to*⟩. The ⟨*to*⟩ part is re-calculated in order to the the number of verses between ⟨*from*⟩ and ⟨*to*⟩ be kept. If the ⟨*to*⟩ part is in the format ⟨*chapter*⟩`:`⟨*verse*⟩ then it is unchanged. The \.renumlabel macro must be expandable, so we cannot use \isinlist and we prepare special expandable macros \.isdivisin and \.iscolonin.

opbible.opm

```
\_def\.renumlabel#1/#2\_relax{#2%
   \_ea\.isdivisin\.tmp-\_iftrue --\_ea\.renumlabelA\.tmp\_relax#2\_relax \_fi
}
\_def\.renumlabelA#1:#2-#3\_relax#4:#5\_relax{%
   \.iscolonin#3:\_iftrue #3\_else \_the\_numexpr#5+#3-#2\_relax \_fi
}
```

The \Note text is processed and printed in the second step, when the `.txs` file is read. Actions are assigned to each verse and they are run before the appropriate verse is printed. And \Note action says:

```
\.replpre{\.doNote{<note-num>}}{<tword>}{\.notefail{<note-num>}}
```

It means that the ⟨*tword*⟩ is searched in the verse text and replaced by \.doNote{⟨*note-num*⟩}{⟨*tword*⟩}. If ⟨*tword*⟩ is not found then \.notefail{⟨*note-num*⟩} prints warning about it and \.doNote{⟨*note-num*⟩}{} is prefixed before the verse text.

opbible.opm

```
\_def\.notefail#1{%
   \.printwarn{\_csstring\\Note: \.currverse: The text "\_unexpanded\_ea{\.text}" not found}%
   \.replpre{\.doNote{#1}}{}{}% \Note is registered with the beginning of the verse
}
```

The \.doNote{⟨note-num⟩}{⟨tword⟩} prints the real note text in the second step, when the verse text from \.buff is processed.
The ⟨chapter⟩:⟨verse⟩ is printed from \notepre! only if it differs from previous one, i.e. from \.prevnotepre. The ⟨pword⟩ is printed with uppercase first letter by \.upcasefirst and with appended dot, but the dot is not printed if the ⟨pword⟩ ends by ? or !.

opbible.opm

```
\_def\.prevnotepre{}
\_def\.doNote#1#2{%
   \_edef\.tmpb{\_cs{notepre!#1}}%
   \.notelog{\_space\_space \_csstring\\Note \.tmpb\_space {#2}={\_cs{pword!#1}} (#1)}%
   \.noteinsert{%
      {\_bf \_ifx\.prevnotepre\.tmpb \_else \.tmpb \_enskip \_glet\.prevnotepre=\.tmpb \_fi
       \.trymakedest{n:\_cs{noteref!#1}}%
       \_edef\.tmpb{\_csname pword!#1\_endcsname}%
       \_ifx\.tmpb\_empty \_else
           \_addto\.tmpb{.}\.punctpword
           \_ea\.upcasefirst \.tmpb\_space
       \_fi
      }% end of \bf
      \_cs{notetext!#1}}%
   {\notecolor#2}%
}
\_def\_printfnotemark{}
\_def\_textindent#1{\_noindent}
```

The ⟨pword⟩ is typically all lowercase. But we want to capitalize the first letter of the ⟨pword⟩ when printing by \.upcasefirst. You can say \let\.upcasefirts=\relax if you don't want this feature.

opbible.opm

```
\_def\.upcasefirst #1{\_uppercase{#1}}
```

The dot is added to ⟨pword⟩ when it is printed. But if ⟨pword⟩ ends by ! or ? then the added dot is uggly. We have to correct it in the \.punctpword macro. Note that ⟨pword⟩ is saved to \.tmpb.

opbible.opm

```
\_def\.punctpword{\_replstring\.tmpb{!.}{!}\_replstring\.tmpb{?.}{?}}
```

When \Note has empty parameter ⟨word⟩ (i.e. ⟨tword⟩) then it is anchored to the beginning of the verse. Moreower, if there are more such Notes referenced to the same verse then we merge all such notes to single note. So \.doCNote{⟨notenum⟩} is run from \.prebuff and it only adds the text of the note to the \.Cnotetext buffer. When \.prebuff is completed then \.printCnote prints the merged note.

opbible.opm

```
\_def\.doCNote #1{%
   \_edef\.tmpb{\_csname pword!#1\_endcsname}%
   \.notelog{\_space\_space \_csstring\\Note \.tmpb\_space {}={\_cs{pword!#1}} (#1)}%
   \_edef\.prevnotepre{\_cs{notepre!#1}}%
   \_ifx\.tmpb\_empty \_else
       \_addto\.tmpb{.}\.punctpword
       \_edef\.tmpb{{\_noexpand\_bf \_ea\.upcasefirst\.tmpb\_noexpand~}}%
       \_ea\_addto \_ea\.Cnotetext \_ea{\.tmpb}%
   \_fi
   \_ea\_ea\_ea\_addto\_ea\_ea\_ea\.Cnotetext\_ea\_ea\_ea{\_csname notetext!#1\_endcsname}%
}
\_def\.printCnote{%
   \_ifx\.Cnotetext\_empty \_else
      \.noteinsert{%
         {\_bf \_ea\.nobook\.currverse\_relax \.trymakedest{n:\.currverse}} \.Cnotetext
      }%
   \_fi
}
\_def\.nobook #1/#2\_relax {#2} % only chapter:verse is printed
```

\.reducetword does nothing by default. But \megrednotes re-defines it, so all \Notes are referenced to the begining of the verse and nothing is searched. The \Notes with the same verse are merged in this case using \.doCNote.

opbible.opm

```
\_def\.reducetword{}
\_def\.mergednotes{\_def\.reducetword{\_def\.tword{}}}
\_nspublic \mergednotes ;
```

Because there is asynchronous processing of the \Note text, we have a problem when an error occurs here. We cannot reference to appropriate line where the \Note is written. So, we print the parameters of processed \Note to the log file. The user can look into this file and the last printed \Note parameters here refers probably to the \Note where the reason of the error is.
The logging is done by \.notelog{⟨text⟩}. It is \wlog by default but you can set it to \ignoreit or \wterm.

opbible.opm

```
\_let\.notelog=\_wlog
```

6 Inserting data from format files

\fmtpre {⟨gen-vref⟩}{⟨what⟩} adds ⟨what⟩ to \.fmtprebuff, i.e. at the beginning of the verse.
\ftmadd {⟨gen-vref⟩}{⟨what⟩} adds ⟨what⟩ to \.buff, i.e. at the end of the verse.
\fmtins {⟨gen-vref⟩}{⟨text⟩}{⟨what⟩} inserts ⟨what⟩ after ⟨text⟩ in the verse. If ⟨text⟩ is not found then ⟨what⟩ is inserted like \fmtpre does it
All these commands allocate new action using \.newaction.
\.addpre\macro{⟨text⟩} adds the text to the macro before its original contents.

opbible.opm

```
\_def\.fmtpre#1#2{\.newaction{\.gentovref{#1}}{\_addto\.fmtprebuff{#2}}}
\_def\.fmtpreind#1#2{\.newaction{\.gentovref{#1}}{\.addpre\.preindbuff{#2}}}
\_def\.fmtadd#1#2{\.newaction{\.gentovref{#1}}{\_addto\.buff{#2}}}
\_def\.fmtins#1#2#3{\.newaction{\.gentovref{#1}}{\.replprepost{#2}{}{#3}{\.fmtfail{#3}}}}
\_def\.fmtfail#1{\.fmtwarn\_addto\.fmtprebuff{#1}}
\_def\.fmtwarn{\.printwarn{\_string\fmtins: \.currverse: The text "\.text" not found}}
\_def\.addpre#1#2{\_ea\.addpreA \_ea{#1}{#2}#1}
\_def\.addpreA #1#2#3{\_def#3{#2#1}}

\_nspublic \fmtpre \fmtadd \fmtins ;
```

\begcenter starts the centering mode. It opens a group and does setting. User must use paired \endcenter in order to close this group. The \centeringmode status is checked by \endcenter because curious error (about # character) should be occur without this checking.

opbible.opm

```
\_newdimen\.centermargin \.centermargin=4em
\_def\.begcenter{\_par \_ifnum\_lastpenalty<10000 \_medskip \_fi
   \_bgroup
   \_def\.centeringmode{y}
   \_parindent=0pt
   \_leftskip=\.centermargin plus1fill
   \_rightskip=\_leftskip
}
\_def\.endcenter{\_par
   \_ifx\.centeringmode\_undefined
      \.printwarn{\_noexpand\endcenter ignored: no \_noexpand\begcenter precedes}
   \_else \_egroup \_medskip \_fi
}
\_nspublic \begcenter \endcenter ;
```

\ind{⟨number⟩} gives an indentaion in the poetry environment. It is used in \fmtpoetry, the \ind{⟨number⟩} is inserted typically by \fmtins or \fmtpre. It ends the current line by \par only if we are not at beginning of a verse 1.
The \spacefactor is set to 1001, this value is used by the macro \.hboxorllap: the verse number is llaped after \ind.

opbible.opm

```
\_newifi\_ifopb_firstverse

\_def\.ind#1{\_unless \_ifopb_firstverse \_par \_else \_hskip-\_parindent \_fi
   \_noindent
   \_hskip#1\_iindent \_spacefactor=1001 }
```

\fmtpoetry{⟨gen-vref⟩}{⟨fmt-data⟩} saves ⟨gen-vref⟩ to \.tmpa and runs ⟨fmt-data⟩ in recursive loop using \.fmtpoetA. The \.fmtpoetB counts the number of slashes in local recursive loop and saves the result to the _tmpnum. The \.fmtpoetC inserts desired material using \fmtprepoet or \fmtins and using \ind{_the_tmpnum}.

opbible.opm

```
\_def\.fmtpoetry#1#2{\_def\.tmpa{#1}\.fmtpoetA #2\_end}
\_def\.fmtpoetA #1/{\_def\.tmpb{#1}\_tmpnum=1 \.fmtpoetB}
\_def\.fmtpoetB #1{\_ifx/#1 \_incr\_tmpnum \_ea\.fmtpoetB \_else \_afterfi{\.fmtpoetC#1}\_fi}
\_def\.fmtpoetC #1{%
   \_expanded{\_ifx\.tmpb\_empty \_noexpand\.fmtpreind{\.tmpa}\_else
              \_noexpand\.fmtins{\.tmpa}{\.tmpb}\_fi{\_noexpand\.ind{\_the\_tmpnum}}}%
   \_ifx\_end#1 \_else \_afterfi{\.fmtpoetA#1}\_fi
}
\_nspublic \ind \fmtpoetry ;

\_def\.fmtfont#1#2#3{%
   \.newaction{\.gentovref{#1}}{\.replprepost{#2}{\bgroup#3}{\egroup}{\.fmtwarnf}}}
\_def\.fmtwarnf{\.printwarn{\_string\fmtfont: \.currverse: The text "\.text" not found}}
\_nspublic \fmtfont ;
```

7 Printing verses from .txs files

When Bible text is processed then book mark is saved to `\.currbook` and each input line is separated to the ⟨*chapter-num*⟩:⟨*verse-num*⟩ and ⟨*verse-text*⟩.
The `\.processline` ⟨*chapter*⟩:⟨*verse*⟩⟨*space*⟩⟨*verse-text*⟩`^^J` is repeatedly processed.

opbible.opm

```
\_eoldef\.processline#1{\.processverse \.currbook/#1\_end}
```

`\.processverse` ⟨*full-vref*⟩⟨*space*⟩⟨*verse-text*⟩`\_end` does

- defines `\.currverse` as ⟨*full-vref*⟩,
- prepares `\.currversenum`, `\.currversetext`, `\.currchapnum` from ⟨*full-vref*⟩,
- defines `\.buff` as ⟨*verse-text*⟩,
- processes all actions from `\alist!`⟨*full-vref*⟩,
- if `\.currchapnum` changed, prints `\.chapafter` (for previous chapter) and `\.chapbefore` (for new chapter).
- prints verse from `\.buff` using `\.printverse`

opbible.opm

```
\_newcount\.chapnum
\_def\.processverse #1 #2\_end{%
   \_xdef\.currverse{#1}%
   \.preparechapverse #1
   \_let\.prelinkV=\.currversenum
   \_gdef\.buff{#2}\_gdef\.fmtprebuff{}\_gdef\.preindbuff{}\_gdef\.prebuff{}\_gdef\.Cnotetext{}%
   \_ifx\.verseto\_empty \_csname alist!#1\_endcsname \_else
      \_fornum \.versefrom..\.verseto \_do{\_csname alist!\.currbook/\.currchapnum:##1\_endcsname}%
   \_fi
   \_ifnum\.currchapnum=\.chapnum \_else
       \_ifnum\.chapnum>1 \.chapafter \_fi
       \_let\.prelinkC=\.currchapnum \.chapnum=\.currchapnum\_relax
       \.chapbefore
       \_label[cref!\.currbook\_space\_the\.chapnum]\_wlabel{\.currbook~\_the\.chapnum}%
   \_fi
   \.printverse
}
\_def\.preparechapverse #1/#2:#3 {\_def\.currchapnum{#2}%
   \_def\.verseto{}%
   \.isdivisin #3-\_iftrue \.defversefromto #3\_end
   \_else \_def\.currversenum{#3}\_glet\.currversetext=\.currversenum
   \_fi
}
\_def\.defversefromto #1-#2\_end{%
   \_def\.versefrom{#1}\_def\.verseto{#2}%
   \_def\.currversenum{#1}\_gdef\.currversetext{#1--#2}}
```

User can do little changes in the verse text using `\cnvtext{`⟨*what*⟩`}{`⟨*replaced*⟩`}`. For example you can do `\cnvtext{[]{\bgroup\it}\cnvtext{]}{\/\egroup}` for making `[words]` in brackets printed italics.

opbible.opm

```
\_def\.prepareversetext{}
\_def\.cnvtext#1#2{\_addto\.prepareversetext{\_replstring\.buff{#1}{#2}}}
\_nspublic \cnvtext ;
```

\.printverse prints verse from \.currversenum and (possibly changed) \.buff. It prints the single raised verse number first.
\.printbeforefirst is a macro which is executed just before first verse of the chapter, after all material from \fmtpre is executed. I.e after printing a chapter name (if declared by \fmtpre).
The \.fmtprebuf includes \ind command from \fmtpoetry if the verse should be indented at its begin before the verse number. The verse number is shifted up and it is in an \hbox or it is llapped in the poetry environment, more exactly immediatelly after \ind is used. The \.hboxorllap macro does this game.

opbible.opm

```
\_def\.printverse{%
   \.fmtprebuff % material accumulated by \fmtpre
   \_ifnum\.currversenum=1 \.firstversetrue \.printbeforefirst \_fi
   \_quitvmode \_mark{\.currchapnum:\.currversetext}%
   \_ifx\.verseto\_empty \.trymakedest{v:\.currverse}%
   \_else \_fornum \.versefrom..\.verseto \_do{%
       \_wlog{xxxxx v:\.currbook/\.currchapnum:##1}\.trymakedest{v:\.currbook/\.currchapnum:##1}}%
   \_fi
   \.preindbuff
   \_raise5pt\.hboxorllap{\_unless\_ifnum\.currversenum=1 \.markfont\.currversetext\,\_fi}%
   \.firstversefalse
   \.prepareversetext
   \.prebuff\.printCnote\.buff \_space
}
\_def\.hboxorllap{\_ifnum\_spacefactor=1001 \_ea\_llap \_else \_ea\_hbox \_fi}

\_def\.printbeforefirst{%
   \_par\_nobreak \_medskip
   \.trychapnote
   \_setbox0=\_vtop{\_kern-1.5ex \_ewref\_sxdef{{ch!\.currbook/\_the\.chapnum}{\_string\.mypage}}
                   \_hbox{\_setfontsize{at50pt}\_bf\LiRed\_the\.chapnum}}
   \_dp0=0pt
   \_tmpdim=\.lrmargin
   \_advance\_tmpdim by4pt
   \_ifnum\_the\.chapnum>9 \_advance\_tmpdim by19pt \_fi
   \_ifodd\_trycs{ch!\.currbook/\_the\.chapnum}{0}
      \_moveright\_tmpdim \_line{\_hss\_box0}
   \_else \_moveleft\_tmpdim \_box0 \_fi
   \_nobreak \_vskip-\_medskipamount
   \_nobreak \_nointerlineskip \_noindent
}
```

\.printchapnote{⟨*text*⟩} implements printing the notes declared by \Note ⟨*chapnum*⟩:0. It is run using \.trychapnote only if the relevant not is declared.

opbible.opm

```
\_def\.trychapnote{%
   \_ifcsname chapnote!\.currbook/\_the\.chapnum:0\_endcsname
      \.printchapnote{\_cs{chapnote!\.currbook/\_the\.chapnum:0}}\_fi
}
\_def\.printchapnote #1{\_par
      {\_leftskip=\_parindent plus1fill \_rightskip=\_leftskip \_noindent\_it #1\_par}
      \_medskip
}
\_nspublic \printchapnote ;
```

\.chapbefore is processed before each chapter. \.chapafter is processad after each chapter. User can define values by \ChapterPre and \ChapterPost macros.

opbible.opm

```
\_def\.chapbefore{\_bigskip}  \_def\.chapafter{}
```

8 Bible references

The < will be set to active as character equivalent to the macro \.bref⟨*text*⟩>. This macro does all job with the hyperlinks. Fist of all, it scans the parts of the ⟨*text*⟩ and saves them to

- \.ltextP ... the text before a link specification (given in "...")
- \.ltextB ... the book mark followed by ~

- \.ltextC ... the chapter number followed by :
- \.ltextV ... the verse number
- \.ltextS ... sub-verse identifier (a if there is a verse 4a)
- \.ltextF ... the -- if the ⟨*from*⟩-⟨*to*⟩ format is given
- \.ltextN ... the ⟨*to*⟩ part from the ⟨*from*⟩-⟨*to*⟩ format.

All these macros above can be empty if the appropriate part of the scanned ⟨*text*⟩ is missing. The \.linkpre macro includes v if it is verse link, includes n if it is note link and g if it is gloss link. These macros will be converted due to \renum data (if needed) and printed by \.linktext.

opbible.opm

```
\_def\.linktext{\.ltextP\.ltextB\.ltextC\.ltextV\.ltextS\.ltextF\.ltextN}
\_def\.bref #1>{\_let\.brefH=\_relax \_def\.linkspec{#1}\_isnextchar"{\.brefA}{\.brefA""}#1>}
\_def\.brefA"#1"{\_def\.ltextP{#1}%
   \_isnextchar{ }{\_addto\.ltextP{~}\_afterassignment\.brefB\_let\.next= }%
      {\_isnextchar{_}{\_def\.brefH{}\_afterassignment\.brefB\_let\.next= }{\.brefB}}%
}
\_def\.brefB #1>{% #1 is link-spec
   \_def\.ltextB{}\_def\.ltextC{}\_def\.ltextF{}\_def\.ltextN{}%
   \.isspacein #1 \_iftrue
        \.iscolonin #1:\_iftrue \.brefBookChapterVerse #1>%
        \_else \.brefBookChapter #1>\_fi
   \_else \.iscolonin #1:\_iftrue \.brefChapterVerse #1>%
   \_else \.brefVerse #1>%
   \_fi\_fi
   \_def\.linkpre{v}%
   \_isnextchar n{\_def\.linkpre{n}\.brefC}%
      {\_isnextchar g{\_def\.linkpre{g}\.brefC}%
         {\_isnextchar a{\_def\.linkpre{a}\.brefC}%
             {\_isnextchar i{\_def\.linkpre{i}\.brefC}{\.brefD}}}}%
}
\_def\.brefC{\_afterassignment\.brefD \_let\.next= }

\_def\.brefBookChapterVerse #1 #2:#3>{\_def\.ltextB{#1~}\.brefChapterVerse #2:#3>}
\_def\.brefBookChapter #1 #2>{\_def\.ltextB{#1~}%
    \_isinlist\nochapbooks{ #1 }\_iftrue
       \_def\.ltextC{}\_let\.ltextCin=\.ltextnCin \_afterfi{\.brefVerse #2>}%
    \_else \_afterfi{\.brefChapter #2>}\_fi}
\_def\.brefChapterVerse #1:#2>{\_def\.ltextC{#1:}\.brefVerse #2>}
\_def\.brefVerse #1>{%
   \.isdivisin #1-\_iftrue \.brefFromTo #1>%
   \_else \.versedef#1\_relax\_fi
}
\_def\.brefChapter #1>{%
   \.isdivisin #1-\_iftrue \.brefFromTo #1>\_let\.ltextC=\.ltextV
   \_else \_def\.ltextC{#1}\_fi
   \_def\.ltextV{}\_def\.ltextS{}%
}
\_def\.brefFromTo #1-#2>{\.versedef#1\_relax\_def\.ltextF{--}\_def\.ltextN{#2}}
```

Because the verse number can be in the format 11b, we need to separate the numeric part of this and save it to \.ltextV and the rest is saved to \.ltextS. This is done by the \.versedef ⟨*verse*⟩\relax macro.

opbible.opm

```
\_def\.versedef {\_afterassignment\.versedefB \_tmpnum=0}
\_def\.versedefB #1\_relax{\_edef\.ltextV{\_the\_tmpnum}\_def\.ltextS{#1}}
```

Now, we create \.linkfspec from scanned data. It is ⟨*full-vref*⟩ used for hyperlinks. We must manage all situations of incomplete links.

opbible.opm

```
\_def\.brefD{%
   \_ifnum 0\.ltextV=0 \_def\.ltextV{}\_fi
   \_if a\.linkpre \_ifx\.ltextV\_empty \_else \_edef\.ltextC{\.ltextV:}\_def\.ltextV{}\_fi\_fi
   \_edef\.linkfspec{\_ea\.ltextBin\.ltextB~/\_ea\.ltextCin\.ltextC:/\_ea\.ltextVin\.ltextV:/}%
   \.brefL
}
\_def\.ltextBin #1~#2/{\_ifx^#1^\.prelinkB \_else #1\_immediateassignment\_def\.prelinkB{#1}\_fi/}
\_def\.ltextCin #1:#2/{\_ifx^#1^\.prelinkC \_else #1\_immediateassignment\_def\.prelinkC{#1}\_fi:}
\_def\.ltextVin #1:#2/{\_ifx^#1^\.prelinkV \_else #1\_immediateassignment\_def\.prelinkV{#1}\_fi}
\_def\.ltextnCin #1:#2/{\.prelinkC:\_immediateassignment\_let\.ltextCin=\.ltextsCin}
\_let\.ltextsCin=\.ltextCin
```

\.prelinkB is ⟨*book-mark*⟩ of last referenced book. \.prelinkC is ⟨*chapter-num*⟩ of last referenced chapter. They are used if the reference is not full. They are initialized at the beginning of books and chapters and they are changed locally in the \Note text. If the \< is used then they are re-initialized.

opbible.opm

```
\_def\<{\_let\.prelinkB=\.currbook \_let\.prelinkC=\.currchapnum \_let\.prelinkV=\.currversenum \.bref}
```

Macro \.brefL recalculates \.linkfspec and \.linktext due to \renum data and creates the link \.linkpre:\.linkfspec with the text \.linktext.
\.renumlinktext ⟨*full-vref-ori*⟩_relax⟨*full-vref-modified*⟩_relax does re-calculation of the parts of the \.linktext macro.
The \.linkfspecone solves situation when chapter is given but no verse number: we must set the verse number to 1.
If the link destination is article, then the ⟨*full-vref*⟩ has reduced format ⟨*book*⟩/⟨*chapter*⟩. If the link destination is itroduction then the ⟨*full-vref*⟩ has more reduced format: ⟨*book*⟩/.
If the book mark is declared by \vdef then the printed version of the book mark is transformed depending on the current \tmark. This is done by the the \.newlinkB macro.
\.linklog {⟨*text*⟩} macro prints logging info of the link in the format

<⟨*link-spec*⟩> = [⟨*full-vref*⟩]{⟨*printed-link*⟩}

\.linklog is \wlog by default and when \tracinglinks is set. It is \ignreit when \notracinglinks is set. You can set it to \wterm if you want.

opbible.opm

```
\_def\.brefL{%
   \_edef\.linkfspecm{\_ea\.renumvref\.linkfspec\_relax}%
   \_ifx\.linkfspec\.linkfspecm \_else
      \_ea\_ea\_ea\.renumlinktext \_ea\.linkfspec \_ea\_relax \.linkfspecm \_relax
      \_let\.linkfspec=\.linkfspecm
   \_fi
   \_ifx\.ltextV\_empty \_ifx\.ltextC\_empty \_else \_ea\.linkfspecone \.linkfspec\_end \_fi\_fi
   \_if a\.linkpre\_relax \_ea\.linkfspecarticle \.linkfspec\_end \_fi
   \_if i\.linkpre\_relax \_ea\.linkfspecintro \.linkfspec\_end \_fi
   \_ifx \.ltextB\_empty \_else \_ea \.newltextB \.ltextB \_fi
%   \message{\meaning\.linkfspec, \meaning\.ltextC, \meaning\.currchapnum}%
   \.reducelinktext
   \.linklog{\.sspace <\_unexpanded\_ea{\.linkspec}>\.linkpost = [\.linkpre:\.linkfspec]%
           {\_ifx\.brefH\_empty \.ltextP \_else \.linktext\_fi}}%
   \.ensuredest \.createlink
}
\_def\.linkfspecone #1:#2\_end {\_def\.linkfspec{#1:1}\_def\.prelinkV{1}}
\_def\.linkfspecarticle #1/#2:#3\_end {\_def\.linkfspec{#1/#2}}
\_def\.linkfspecintro #1/#2\_end {\_def\.linkfspec{#1/}}

\_def\.renumlinktext #1/#2:#3\_relax #4/#5:#6\_relax{%
   \_ifx\.ltextC\_empty \_else \_def\.ltextC{#5:}\_fi
   \_def\.ltextV{#6}%
   \_ifx\.ltextN\_empty \_else
      \_ifx\.ltextF\.ltextDD
          \_isinlist\.ltextN{:}\_iftrue
             \_ifcsname rn!\tmark!#1/\.ltextN\_endcsname \_edef\.ltextN{\_cs{rn!\tmark!#1/\.ltextN}}%
             \_fi
          \_else \_edef\.ltextN{\_the\_numexpr#6+\.ltextN-#3\_relax}\_fi
      \_else \_let\.tmp=\_ignoreit % \.ltextN is a list of verses, for example 7,9,13
          \_ea\_foreach\.ltextN,\_do ##1,{\_edef\.tmp{\.tmp,\_the\_numexpr#6+##1-#3}}%
          \_let\.ltextN=\.tmp
      \_fi
   \_fi
}
\_def\.ltextDD{--}

\_def\.newltextB #1~{\_edef\.ltextB{\_trycs{v!\tmark!#1}{#1}~}}

\_def\.sspace{\_space\_space\_space\_space}
\_def\.linkpost{\_if v\.linkpre \_else \.linkpre\_fi \_space}
```

\.reducelinktext does nothing or reduces printed link if its book is equal to the current book and if its chapter is equal to printed chapter. It is activated by \reduceref and deactivated by \noreduceref. The \re macro activates \.reducelinktext only for single \.bref.

opbible.opm

```
\_def\.reducelinktextA{%
   \_edef\.tmp{\.currbook~}%
   \_ifx\.ltextB\.tmp \_def\.ltextB{}%
      \_edef\.tmp{\_trycs{_opb_currchapnum}{?}:}%
      \_ifx\.ltextC\.tmp \_def\.ltextC{}%
   \_fi\_fi
   \_ifcsname _opb_reA\_endcsname \_let\.reducelinktext=\.reA \_fi % after \re
}
\_def\.reduceref{\_let\.reducelinktext=\_reducelinktextA}
\_def\.noreduceref{\_let\.reducelinktext=\_relax}
\.noreduceref % default

\_def\.re{\_let\.reA=\.reducelinktext \.reduceref}

\_nspublic \reduceref \noreduceref \re ;
```

\tracinglinks and \notracinglinks are defined here.

opbible.opm

```
\_def\tracinglinks{\_let\.linklog=\_wlog}
\_def\notracinglinks{\_let\.linklog=\_ignoreit}
\tracinglinks
```

\.createlink creates link only if it refers to the place of printed book because we don't want to see many warnings about unreferenced links when we try to print only selected books. It creates link `\.linkpre:\.linkfspec` with the text `\.linktext`
The link is created only if the book is to be printed, i.e. the `\pbook!`⟨*book*⟩ is defined.
\tracingouterlinks acivates logging of broken links to non-existed books. By default, these links are not logged because we assume that no whole Bible is processed but only selected books.

opbible.opm

```
\_def\.createlink{{%
   \_ifx\.brefH\_empty \_let\.linktext=\.ltextP\_fi
   \_ea\.isprintedbook\.linkfspec \_iftrue
   \_link[\.linkpre:\.linkfspec]{\_ilinkcolor}{\.linktext}%
   \_else {\_ilinkcolor\.linktext}\_fi}%
}
\_def\.isprintedbook #1/#2\_iftrue{\_ifcsname pbook!#1\_endcsname}
\_def\tracingouterlinks{\_def\.isprintedbook ##1\_iftrue{\_iftrue}}
```

We don't create destinations for all verses, notes etc. but only for those which are referenced. The macro \.ensuredest is called from \.createlink and it saves immediatelly `\sdef{`⟨*link*⟩`:`⟨*full-vref*⟩`}{}` to the special file `\jobname.xrf`. And the macro \pg saves immediatelly `\sdef{pg:`⟨*link*⟩`:`⟨*full-vref*⟩`}{??}` to this file. This `.xrf` file is read before standard `.ref` file. All link destinations save `\.Xdest{`⟨*full-vref*⟩`}` to the `.ref` file. The macro \.Xdest does nothing if `\pg:`⟨*link*⟩`:`⟨*full-vref*⟩ is not defined (from `.xrf` file). Otherwise, it is defined as a correct pageno. This result is used in the \pg macro. If \⟨*link*⟩`:`⟨*full-vref*⟩ is not defined, no link destination is crated. First TEX run creates `.ref` and `.xrf` files and does not create any hyperlink destinations. Second TEX run uses data from these files and creates correct hyperlinks and page numbers.

opbible.opm

```
\_newwrite\.xrf
\_immediate\_openout\.xrf=\_jobname.xrf
\_openref

\_def\.ensuredest{\_immediate\_write\.xrf{\_string\_sdef{\.linkpre:\.linkfspec}{}}}
\_refdecl{
   \_isfile{\_jobname.xrf}\_iftrue \_input{\_jobname.xrf}\_fi^^J
   \_def\.Xdest#1{\_ifcsname pg:#1\_endcsname \_sxdef{pg:#1}{\_ea\_usesecond\_currpage}\_fi}^^J
   \_def\.mypage{\_ea\_usesecond\_currpage}
}
\_def\.trymakedest#1{%
   \_ifcsname #1\_endcsname \_dest[#1]\_ea\_glet\_csname #1\_endcsname \_undefined \_fi
   \_ewref\.Xdest{{#1}}%
}
```

The \pg macro should be used after `<...>`, i.e. the \.linkpre and \.linkfspec are defined. We use them. And the page number is saved to the `\pg:`⟨*link*⟩`:`⟨*full-vref*⟩ macro in the second TEX run.

opbible.opm

```
\_def\.pg{%
   \_ifcsname pg:\.linkpre:\.linkfspec\_endcsname
      {\_edef\.linktext{\_cs{pg:\.linkpre:\.linkfspec}}\_let\.brefH=\_relax \.createlink}%
   \_else {\Red ??}\_fi
   \_immediate\_write\.xrf{\_string\_sdef{pg:\.linkpre:\.linkfspec}{??}}%
}
\_nspublic \pg ;
```

`\cref` if simply `\ref` with `cref!` prefix.

opbible.opm

```
\_def\.cref[#1]{\_ref[cref!#1]}

\_nspublic \cref ;
```

9 Language variants

`\variants` ⟨*number-of-variants*⟩ `{`⟨*tmark-A*⟩`}` `{`⟨*tmark-B*⟩`}` `{`⟨*tmark-C*⟩`}` ... sets `\.numvariants=`⟨*number-of-variants*⟩ and does `\def\tmarkA{`⟨*tmark-A*⟩`}` `\def\var!1{`⟨*markA*⟩`}` `\def\var!2{`⟨*tmark-B*⟩`}` `\def\var!3{`⟨*tmark-C*⟩`}` etc.

opbible.opm

```
\_newcount\.numvariants
\_def\.variants{\_tmpnum=0 \_afterassignment\.variantsA \.numvariants}
\_def\.variantsA{%
   \_ifnum\_tmpnum<\.numvariants
      \_advance\_tmpnum by1
      \_afterfi{\.variantsB{\_the\_tmpnum}}%
   \_fi
}
\_def\.variantsB#1#2{%
   \_ifnum#1=1 \_gdef\tmarkA{#2}\_sxdef{var!1}{#2}%
   \_else \_sxdef{var!#1}{#2}%
   \_fi
   \.variantsA
}
\_nspublic \variants ;
```

`\vdef` `{`⟨*phrase-A*⟩`}` `{`⟨*phrase-B*⟩`}` `{`⟨*phrase-C*⟩`}` ... does `\def\v!`⟨*tmark-B*⟩`!`⟨*phrase-A*⟩`{`⟨*phrase-B*⟩`}` `\def\v!`⟨*tmark-C*⟩`!`⟨*phrase-A*⟩`{`⟨*phrase-C*⟩`}` etc. Empty parameter is interpreted as undefined data. The internal macro `\.vdefB` implements the error message if there is too few parameters of `\vdef` and we were read next `\vdef`. The `\.sedef` used in the `\.vdefB{`⟨*number*⟩`}{`⟨*param*⟩`}` does real work and it defines (rougly sepaking):

```
If ⟨param⟩ is "   \def \v!⟨tmark⟩!⟨phrase-A⟩ {⟨previous param⟩}
else              \def \v!⟨tmark⟩!⟨phrase-A⟩ {⟨param⟩}
```

opbible.opm

```
\_def\.vdef#1{\_def\.tmp{#1}%
   \_ifcsname v!\_trycs{var!2}{}!\.tmp\_endcsname
      \.printwarn{\_noexpand\vdef used secondly for phrase {\.tmp}, ignored}\_fi
   \_tmpnum=1 \_ea\.vdefA
}
\_def\.vdefA{%
   \_ifnum\_tmpnum<\.numvariants
      \_advance\_tmpnum by1
      \_afterfi{\.vdefB{\_the\_tmpnum}}%
   \_fi
}
\_def\.vdefB#1#2{\_def\.tmpa{}%
   \_ifx\.vdef#2\_def\.tmpa{#2}\_fi
   \_ifx\.tmpa\_empty
      \_ifx^#2^\_else
         \_unless \_ifcsname v!\_cs{var!#1}!\.tmp\_endcsname
            \.sedef{v!\_cs{var!#1}!\.tmp}{\_ifx"#2\.prevcs{#1}\.tmp \_else#2\_fi}%
      \_fi\_fi
      \_ea\.vdefA
   \_else \_errmessage{\_string\vdef: too few parameters. To be read again: \_string#2}%
      \_ea\.tmpa
   \_fi
```

```
}
\_def\.prevcs #1#2{\_ifnum#1=2 #2\_else \_cs{v!\_cs{var!\_the\_numexpr#1-1\_relax}!#2}\_fi}

\_nspublic \vdef ;
```

`\x/`⟨*phrase*⟩`/` expands to `\v!`⟨*tmark*⟩`!`⟨*phrase*⟩ if such control sequence is defined else it expands simply to ⟨*phrase*⟩ using `\xA`. The ⟨*tmark*⟩ is actual value of the `\tmark` macro.
Note that if `\tmark` expands to ⟨*t-markA*⟩ (used in the `\variants` macro), then the `\v!`⟨*tmark*⟩`!`⟨*phrase*⟩ is not defined and the `\x` macro expands to the ⟨*phrase*⟩ directly.
`\xA` ⟨*phrase*⟩`/` expands to ⟨*phrase*⟩ and prints warning, if `\tmark` is not the first ⟨*t-markA*⟩.

opbible.opm

```
\_def\.x/#1/{\_trycs{v!\tmark!#1}{\.xA#1/}}
\_def\.xA#1/{#1\_ifx\tmarkA\_undefined \_else \_ifx\tmark\tmarkA \_else
   \.printwarn{\_string\x/#1/ -- this phrase is undefined by \_csstring\\vdef}%
   \_fi\_fi
}
\_nspublic \x ;
```

`\ww {`⟨*phrase-A*⟩`} {`⟨*phrase-B*⟩`} ...` has the same number of parameters as `\vdef`. They are separated by spaces. Each parameter can be in the "single form", i.e. `{`⟨*phrase-A*⟩`}` or in the "extended form", i.e. `{`⟨*phrase-A*⟩`}={`⟨*printed-A*⟩`}`. The macro searchs the correct phrase (given by the `\.varnum`) and saves it to the `\.nextww`. The `\.nextwwA` is set to `\.nextww` if there is single form of the parameter else `\.nextwwA` is ⟨*printed-A*⟩ part of the parameter in the extended form. These macros are used in the next `\Note` where they are re-set to `\undefined` meaning.

opbible.opm

```
\_def\.ww{%
   \_ifx\.varnum\_undefined \.setvarnum \_fi
   \_tmpnum=0
   \_ifx\.nextww\_undefined \_ea\.wwA
   \_else \.printwarn{Only single \_csstring\\ww must be before \_csstring\\Note}%
       \_ea\.wwB \_fi
}
\_def\.wwA#1#2 {\_advance\_tmpnum by1
   \_def\.nextww{#1}\_def\.nextwwA{#2}%
   \_ifx\.nextwwA\_empty \_let\.nextwwA=\.nextww \_else \_ea \.redefwwA #2\_end \_fi
   \_ifnum\.varnum=\_tmpnum  \_ifnum\_tmpnum<\.numvariants  \_ea\_ea\_ea \.wwB \_fi
   \_else \_ea \.wwA \_fi
}
\_def\.wwB#1 {\_advance\_tmpnum by1
   \_ifnum\_tmpnum<\.numvariants \_ea\.wwB \_fi
}
\_def\.redefwwA =#1\_end{\_def\.nextwwA{#1}}

% \_outer\_def\ww{\.ww} % will be done at the end of this macro file
```

The `\switch` macro reads a pair of parameters using `\.switchA` and processes the list of variants in `\foreach` loop. If an element from the list is equal with `\tmark` then the `#2` (saved in `\.switchD` token list) is run and next parameter pairs are read by `\.switchN`, i.e. they are ignored.
The `\Note` and `\ww` and more macros are defined as `\outer` in order to better diagnose mistakes with their parameters. But we want to skip such objects in `\switch` parameters. This is the reason why we set `\_suppressoutererror=1` during the `\switch` is processed.

opbible.opm

```
\_newtoks\.switchD
\_def\.switch {\_let\.switchN=\.switchA \_suppressoutererror=1 \.switchN}
\_long\_def\.switchA #1#2{\.switchD={#2\_let\.switchN=\.switchI}%
   \_ifx\_relax#1\_relax \_the\.switchD
   \_else \_foreach #1,\_do ##1,{\_def\tmp{##1}\.switchC}%
   \_fi
   \_futurelet\.next\.switchB
}
\_def\.switchB{\_ifx\.next\_bgroup \_ea\.switchN \_else \_suppressoutererror=0 \_fi}
\_long\_def\.switchI #1#2{\_futurelet\.next\.switchB}
\_def\.switchC{\_ifx\tmp\tmark \_the\.switchD \_fi}

\_nspublic \switch ;
```

`\.setvarnum` sets the `\.varnum` as the position number of the current language variant due to the value of `\tmark`. The `\variants` declaration must precede.

opbible.opm

```
\_def\.setvarnum{\_gdef\.varnum{0}%
   \_ifnum\.numvariants=0 \_gdef\.varnum{1}\_wlog{There is only single language variant (1)}%
   \_else
      \_tmpnum=0
      \_loop
         \_advance\_tmpnum by1
         \_ea\_ifx \_csname var!\_the\_tmpnum\_endcsname \tmark \_xdef\.varnum{\_the\_tmpnum}\_fi
         \_ifnum\_tmpnum<\.numvariants \_repeat
      \_ifnum \.varnum=0 \_errmessage{\_noexpand\tmark isn't set, \_noexpand\.setvarnum failded}%
      \_else \_wlog{Language variant set by \_string\tmark{\tmark} (\.varnum)}\_fi
   \_fi
}
```

\renum ⟨*book-mark*⟩ ⟨*chapter-num*⟩:⟨*verse-num*⟩ = ⟨*t-mark*⟩ ⟨*chap-num*⟩:⟨*from*⟩-⟨*to*⟩ does

```
\def \rn!<t-mark>!<full-vref>{<chap-num>:<from>}
\def \rn!<t-mark>!<full-vref+1>{<chap-num>:<from+1>}
\def \rn!<t-mark>!<full-vref+2>{<chap-num>:<from+2>}
... etc.
\def \rn!<t-mark>!<full-vref+n>{<chap-num>:<to>}
```

opbible.opm

```
\_def\.renum #1 #2:#3 = #4 #5:#6-#7 {%
   \_tmpnum=#3\_relax
   \_fornum #6..#7 \_do {\_sxdef{rn!#4!#1/#2:\_the\_tmpnum}{#5:##1}\_incr\_tmpnum}%
}
\_nspublic \renum ;
```

10 Inserting notes to the page

We declare new insert \.noteins used in the \output routine.

opbible.opm

```
\_newinsert \.noteins
\_skip\.noteins=\_bigskipamount  % noterule height
\_count\.noteins=500             % two columns
\_dimen\.noteins=\_maxdimen      % full page of notes allowed
```

The \.noteinsert {⟨*text*⟩} inserts its parameter to the \.noteins. We open the \insert and set basic parameters using \.noteset. Then the empty box with strut height is inserted in vertical mode (in order to consecutive notes have good baselineskip between them). Then the ⟨*text*⟩ is printed and the paragraph is finalized. The empty box with strut depth is appended after the paragraph (in order to the same reason). Final \penalty0 allows breaking between notes.

opbible.opm

```
\_def\.noteinsert #1{\_insert\.noteins{%
   \.noteset
   \_vbox to\_ht\_strutbox{}\_nobreak \_vskip-\_baselineskip
   #1\_unskip\_par \_nobreak \_vskip-\_baselineskip
   \_hbox{\_lower\_dp\_strutbox\_vbox{}}
   \_penalty0
}}
\_def\.noteset{\Heros\cond \_scalemain \_typoscale[800/800] % Heros condensed 80%
   \Black \_nobreak
   \_widowpenalty=20 \_clubpenalty=20
   \_leftskip=0pt \_rightskip=0pt \_parfillskip=0pt plus1fill
   \_parindent=0pt
   \_lineskiplimit=-3pt
   \_hsize=.5\_hsize \_advance\_hsize by-1em\_relax % two columns
   \_everypar{}
}
```

We add macros for inserting two columns of notes from \.noteins into the page. First, we add \noterule with the space given by \skip\.noteins. The \.noteins material is prefixed by \penalty0 (in order to allow the next \vsplit operation) and the \vfil is added (in order to the case when the second column is smaller than the first one). The \splittopskip is set and first \vsplit to0pt adds skip given by \splittopskip to the \.noteins. The _balancecolumns from OpTEX for splitting to two columns is used. We need to set _Ncols, _dimen0 and _box6 before running _balancecolumns. We need to insert \vskip\splittopskip because _balancecolumns supposes that the typesetting point resides at

the first baseline of the columns.
The final `\vskip` does "raggedbottom". We need to add `1filll` in order to suppress the `\vfill` from the `\end` algorithm. We add `minus6pt` because the height of two columns can be by half-line higher than the insertion algorithm excepts (in the case with odd lines before splitting to the two columns).

opbible.opm

```
\_addto\_pagecontents{%
   \_ifvoid\.noteins \_else
      \_vskip\_skip\.noteins \noterule
      \_setbox\.noteins=\_vbox{\_penalty0 \_unvbox\.noteins \_vfil}
      \_splittopskip=12pt
      \_setbox0=\_vsplit\.noteins to0pt  % adding \splittopskip to \.noteins
      \_def\_Ncols{2}
      \_dimen0=.5\_ht\.noteins \_setbox6=\_box\.noteins
      \_vskip\_splittopskip
      \_balancecolumns
   \_fi
   \_unless\_ifvoid\.botins \_unvbox\.botins
   \_else \_vskip 0pt plus1filll minus8pt \_fi
}
\_def \noterule {\_kern-3pt {\Black \_hrule width\_hsize}\_kern 2.6pt }
```

11 Inserting images and articles to the page

`\.botins` is analogue insert as `\_topins` but the material is inserted to the bottom of the page. The material is created by `\.botinsert`...`\.endbot` pair of control sequences. We use it for inserting images and articles to the page.

opbible.opm

```
\_newinsert\.botins
\_def\.botinsert{\_setbox0=\_vbox\_bgroup}
\_def\.endbot{\_par\_egroup
   \_insert\.botins{\_splittopskip=0pt \_penalty100
      \_hrule height0pt \_nobreak\_medskip\_bigskip \_unvbox0
   }%
}
\_skip\.botins=\_zoskip    % no space added when a topinsert is present
\_count\.botins=1000       % magnification factor (1 to 1)
\_dimen\.botins=\_maxdimen % no limit per page
```

`\putImage` ⟨*chatper*⟩`:`⟨*verse*⟩ `{`⟨*title*⟩`}` `[`⟨*label*⟩`]` `(`⟨*params*⟩`)` `{`⟨*image-file*⟩`}` inserts the given image to the page where the begining of the verse given by ⟨*chapter*⟩`:`⟨*verse*⟩ exists. We register a new action by `\.newaction{`⟨*full-vref*⟩`}{\.doImage{`⟨*title*⟩`}[`⟨*label*⟩`](`⟨*params*⟩`){`⟨*image-file*⟩`}}`. The `\.doImage` puts the image by `\.botinsert`...`\.endbot` pair. The `\.botTitle{`⟨*title*⟩`}[`⟨*label*⟩`]` prints the title of the image (or article or watever is put to the bottom of the page) and inserts the destination of hyperlink based on the ⟨*label*⟩, if the ⟨*label*⟩ isn't empty.

opbible.opm

```
\_def\.putImage #1 #2#3[#4]#5(#6)#7{% chap:verse {Title} [label] (params) {image-file.pdf}
   \_edef\.fullvref{\.gentovref{#1}}%
   \_edef\.fullvrefm{\_ea\.renumvref\.fullvref\_relax}%
   \_ea\.newaction\_ea{\.fullvrefm}{\.doImage{#2}[#4](#6){#7}}%
}
\_def\.doImage #1[#2](#3)#4{% {Title}[label](params){image-file.pdf}
   \.botinsert
      \.botTitle{#1}[#2]%
      \_kern3pt \_nobreak
      \_hbox{\picw=\hsize #3\inspic{#4}}%
   \.endbot
}
\_def\.botTitle#1[#2]{\_hbox{\.captionfont
   \_ifx^#2^\_else \.botDest{#1}[#2]\_fi
   \_rlap{\Grey \_vrule height1.2em depth.5em width\_hsize}\White\_kern12pt #1}%
}
\_picdir={images/}
\_def\.botDest#1[#2]{\_label[#2]\_wlabel{#1}}

\_nspublic \putImage ;
```

\putArticle ⟨*chapter*⟩:⟨*verse*⟩ {⟨*title*⟩} [⟨*label*⟩] (⟨*params*⟩) inserts an article given in the file `articles-*.tex` signed by \Article [⟨*label*⟩]. The article starts at the page where ⟨*chapter*⟩:⟨*verse*⟩ is or at the next page. The article is in two-columns style and it is divided to k two-columns parts each of them is inserted at the bottom of the next page.

We calculate the number of pages used for article text by following rules. All the two-columns parts have the same height. If there are more than one such a part, the height does not exceeds 2/3 of the page. But single two-column part can be higher.

\putArticle registers \.doArticle using \.newaction. \.doArticle is run at the beginning of given verse and creates an \.botinsert. The insert material is breakable at its beginig and between each two-column boxes created by the _balancecolumn macro.

We register a new action by `\.newaction{`⟨*full-vref*⟩`}{\.doArticle{`⟨*title*⟩`}[`⟨*label*⟩`](`⟨*params*⟩`)}`.

opbible.opm

```
\_newcount\.articlenum
\_def\.putArticle #1 #2#3[#4]#5(#6){% chap:verse {Title} [number] (params)
   \_edef\.fullvref{\.gentovref{#1}}%
   \_edef\.fullvrefm{\_ea\.renumvref\.fullvref\_relax}%
   \_ea\.newaction\_ea{\.fullvrefm}{\.doArticle{#2}[#4](#6)}%
}
\_nspublic \putArticle ;
```

The \.doArticle {⟨*Title*⟩}[⟨*label*⟩](⟨*params*⟩) inserts the article to one or more pages by the pair \.botinsert...\.endbot. The Article is printed to two columns per page, all collumns of the article is completely balanced. First, the whole text is saved to the \box0 with given column size and the number of pages is calculated in _tmpnum. Then the number of columns _Ncols is 2 times the number of calculated pages. The height of each two-columns part of the article is \dimen0. Finally we do re-boxing the output of _balancecolumns in order to reach individual columns and create pairs of them by \fornum loop. These pairs are completed to blocks with LightGrey background. These blocks divided by \break are inserted into \.botinsert.

opbible.opm

```
\_def\.doArticle#1[#2](#3){% {Title}[number](params)
   \_incr\.articlenum
   \.botinsert
      \_def\.botDest##1[##2]{\.trymakedest{a:\.currbook/##2}}
      \_parindent=12pt \_iindent=\_parindent
      \_setbox0=\_vbox{\_hsize=.458\_hsize \_emergencystretch=1em
         \_hbadness=6000 \_baselineskip=\_dimexpr\_baselineskip plus1pt
         \_def\Article[##1]{\_endinput}
         \_penalty0
         \_long\_def\.searcharticle##1\Article[#2]{}
         \_ea\.searcharticle \_input \articlefile \_relax}
      \_splittopskip=12pt
      \_setbox1=\_vsplit0 to0pt % adding \splittopskip
      \_tmpdim=\_vsize \_advance\_tmpdim by-24pt % \.botTitle height plus above/below skips
      \_ifdim 2\_tmpdim > \_ht0 \_tmpnum=1
      \_else
         \_tmpnum=\.roundexpr{\_bp{\_ht0}/\_bp{1.333\_vsize}+0.999} % number of 2/3 pages
      \_fi
      \_multiply\_tmpnum by2 % number of columns
      \_edef\_Ncols{\_the\_tmpnum}
      \_dimen0=\_expr{1/\_Ncols}\_ht0 \_setbox6=\_box0 % height of each two-columns part
      \_setbox0=\_vbox{\_balancecolumns}
      \_tmpdim=\_ht0 \_advance\_tmpdim by1.2\_baselineskip
      \_setbox0=\_vbox{\_unvbox0 \_global\_setbox2=\_lastbox}
      \_setbox0=\_hbox{\_unhbox2
          \_fornum 1..\_Ncols \_do {\_unskip \_global\_setbox1##1=\_lastbox}}
          \_fornumstep -2: \_Ncols..1 \_do {
              \_hrule height0pt\_kern5pt\_nobreak\_vfill
              \_ifnum\_Ncols=##1 \.botTitle{#1}[#2]\_else \.botTitle{}[]\_fi
              \_kern3pt \_nobreak
              \_hbox to\_hsize{%
                 \_rlap{\LightGrey \_vrule height\_tmpdim depth6pt width\_hsize}%
                 \_kern\_parindent
                 \_box1##1\_hss\_box1\_the\_numexpr##1-1
                 \_kern\_parindent
              }
              \_break
```

```
          }
   \.endbot
}
\_def\.roundexpr#1{\_ea\.roundexprA\_expanded{\_expr{#1}}\_relax}
\_def\.roundexprA#1.#2\_relax{\_ifnum#1=0 0\_else #1\_fi}
```

12 Inserting images over two pages

We can insert an image at the bottom of the page which spans from even to odd page. The macro `\insertSpanImage{`⟨*Title*⟩`} [`⟨*label*⟩`] (`⟨*params*⟩`) {`⟨*image file*⟩`}` does it. The image is placed at the bottom of the pages using following rule: if the `\insertSpanImage` occurrs at the current page c then

- if c is even and the image height fits to the current page then the image is inserted to pages c, $c+1$,
- if c is even and the image height doesn't fit to the current page then the image is inserted to pages $c+2$, $c+3$,
- if c is odd then the image is inserted to pages $c+1$, $c+2$.

The macro `\insertSpanImage` saves the image in the box `\.spanpicbox`. The `\_picwidth` of the image is calculated as 2*(`\_hsize`*⟨*inner_margin*⟩). I.e. when we put the box to the page firstly then only the left half of its size is printed.

Next, `\insertSpanImage` checks if the current page is even. If it is true and if there is sufficient space `\pagegoal-\pagetotal` at the current page, the image is inserted to the current page using the `\.startinsertSpanImage` which runs `\.insertBot` in fact. The second part of the image is printed because `\_endoutput` (processed at the end of the output routine where first part of the image is inserted) runs `\.addpicbox`. The `\.addpicbox` runs second `\.insertBot` which is printed on the next page.

If the current page is odd, then `\insertSpanImage` doesn't run `\.startinsertSpanImage` immediatelly, but `\_endouput` inserts first part of the image using `\.inspicbox` which is equal to `\.inspicboxafter` in this case. It processes `\.startinsertSpanImage` which inserts the first part of the image on the next page (even) page.

If the current page is even but the image cannot fit to the current page then the delay using `\_endoutput` is activated too. But the `\.ispicboxafter` checks that the current page is even and it does nothing in this case. Next page is ofdd, so `\.ispicboxafter` invoked by next `\_endinput` inserts the first part of the image which will be printed on the next (even) page.

opbible.opm

```
\_newbox \.spanpicbox

\_def\.insertSpanImage #1#2[#3]#4(#5)#6{%
   \.checkpicbox
   \_par \_penalty0
   \_tmpdim=\_pagewidth
   \_advance\_tmpdim by-\_hoffset
   \_global\_setbox\.spanpicbox=\_hbox{\_picwidth=2\_tmpdim \_inspic{#6}}
   \_gdef\.startinsertSpanImage {\.insertBot {#1}[#3](#5){\_copy\.spanpicbox \_kern-1.2ex}}
   \.doinsertSpanImage
}
\_def\.doinsertSpanImage{%
   \_ifodd\_pageno
      \_glet\.inspicbox=\.inspicboxafter
   \_else
      \_ifdim \_dimexpr \_pagegoal-\_pagetotal > \_dimexpr \_ht\.spanpicbox+2em \_relax
         \.startinsertSpanImage
      \_else
         \_glet\.inspicbox=\.inspicboxafter
      \_fi
   \_fi
}
\_let\.inspicbox=\_useit
\_def\.inspicboxafter #1{%
   \_ifodd\_pageno
       \.startinsertSpanImage
       \_glet\.inspicbox=\_useit
   \_fi
}
\_def \_endoutput{%
```

```
   \_ifvoid\.spanpicbox\_else \.addpicbox\_fi
   \_advancepageno
   {\_globaldefs=1 \_the\_nextpages \_nextpages={}}%
   \_ifnum\_outputpenalty>-20000 \_else\_dosupereject\_fi
}
\_def\.addpicbox{\.inspicbox{\.insertBot{}[]{}{\_moveleft\_pagewidth\_box\.spanpicbox\_kern-1.2ex}}}

\_def\.checkpicbox{%
   \_ifvoid\.spanpicbox\_else \_errmessage{Two span Image/Text at single place not allowed}\_fi
}
```

`\insertSpanText{`⟨*Title*⟩`} [`⟨*label*⟩`] (`⟨*params*⟩`) {`⟨*text*⟩`}` does the same as `\insertSpanImage`, but the ⟨*text*⟩ is inserted instead the image. The `\hsize` is locally set to the desired width of the text when ⟨*text*⟩ is processed in a `\vbox`, i.e. to 2*(`\hsize`+⟨*inner_margin*⟩).

opbible.opm

```
\_long\_def\.insertSpanText #1#2[#3]#4(#5)#6{%
   \.checkpicbox
   \_par \_penalty0
   \_tmpdim=\_pagewidth
   \_advance\_tmpdim by-\_hoffset
   \_setbox0=\_hbox to2\_tmpdim{\_hss\_vbox{\_hsize=2\_tmpdim
      \_leftskip=0pt \_rightskip=0pt \_relax \_kern3pt #6}\_hss}
   \_global\_setbox\.spanpicbox=
      \_hbox{\_rlap{\White \_vrule width\_wd0 height\_ht0 depth\_dp0}\_box0}
   \_global\_ht\.spanpicbox=\_dimexpr\_ht\.spanpicbox-3pt\_relax
   \_gdef\.startinsertSpanImage {\.insertBot {#1}[#3](#5){\_copy\.spanpicbox \_kern-1.2ex}}
   \.doinsertSpanImage
}
\_nspublic \insertSpanImage  \insertSpanText ;
```

`\putSpanImage` ⟨*chatper*⟩`:`⟨*verse*⟩ `{`⟨*title*⟩`} [`⟨*label*⟩`] (`⟨*params*⟩`) {`⟨*image-file*⟩`}` runs `\insertSpanImage` at the page where the begining of the verse given by ⟨*chapter*⟩:⟨*verse*⟩ exists. We register a new action by `\.newaction{`⟨*full-vref*⟩`}{\.doSpanImage{`⟨*title*⟩`}[`⟨*label*⟩`](`⟨*params*⟩`){`⟨*image-file*⟩`}}`.
`\putSpanText` ⟨*chatper*⟩`:`⟨*verse*⟩ `{`⟨*title*⟩`} [`⟨*label*⟩`] (`⟨*params*⟩`) {`⟨*text*⟩`}` runs `\insertSpanText` at the page where the begining of the verse given by ⟨*chapter*⟩:⟨*verse*⟩ exists. The ⟨*text*⟩ is saved to `\spant!\the\.spantxtnum` and only the name of this macro is registered by the `\.newaction`.

Note that the image/text itself is inserted at the current page c and $c+1$ or at $c+1$, $c+2$ or at $c+2$, $c+3$.

opbible.opm

```
\_newcount\.spantextnum
\_def\.putSpanImage #1 #2#3[#4]#5(#6)#7{% chap:verse {Title} [label] (params) {image-file.pdf}
   \_edef\.fullvref{\.gentovref{#1}}%
   \_edef\.fullvrefm{\_ea\.renumvref\.fullvref\_relax}%
   \_ea\.newaction\_ea{\.fullvrefm}{\.insertSpanImage{#2}[#4](#6){#7}}%
}
\_long\_def\.putSpanText #1 #2#3[#4]#5(#6)#7{% chap:verse {Title} [label] (params) {image-file.pdf}
   \_edef\.fullvref{\.gentovref{#1}}%
   \_edef\.fullvrefm{\_ea\.renumvref\.fullvref\_relax}%
   \_incr\.spantextnum
   \_global\_sdef{spant!\_the\.spantextnum}{#7}%
   \_ea\.putSpanTextA
      \_expanded{{\.fullvrefm}\_ea}\_csname spant!\_the\.spantextnum\_endcsname {#2}[#4](#6)%
}
\_def\.putSpanTextA #1#2#3[#4](#5){\.newaction{#1}{\.insertSpanText{#3}[#4](#5){#2}}}

\_nspublic \putSpanImage \putSpanText ;
```

13 Inserting citations to the page

`\putCite` ⟨*gen-vref*⟩ `{`⟨*text*⟩`}` creates a citation ⟨*text*⟩ inserted to the top of the page where the verse ⟨*gen-vref*⟩ is. We regiter a new action by `\.newaction{`⟨*full-vref*⟩`}{\dotopCite{`⟨*text*⟩`}}`.

opbible.opm

```
\_def\.putCite #1 #2{% chap:verse {text}
   \_edef\.fullvref{\.gentovref{#1}}%
   \_edef\.fullvrefm{\_ea\.renumvref\.fullvref\_relax}%
   \_ea\.newaction\_ea{\.fullvrefm}{\.dotopCite{#2}}%
}
\_nspublic \putCite ;
```

`\.dotopCite {`⟨*text*⟩`}` creates the citation text by `\topinsert`...`\endinsert` form plain TeX. We distinguish two cases: the citation on a left page and the citation on a right page. We sawe the page position using `\_ewref` to the .ref file as `\sxdef{ct!`⟨*citenum*⟩`}{\.mypage}` and we know the page position in the second TeX run and use it in the `\ifodd` condition. The typesetting parameters differ in "left" and "right" case.

opbible.opm

```
\_newcount\.citenum
\_def\.dotopCite #1{%
   \.topinsertnopar
   \_typosize[12/16]\_bi
   \_incr\.citenum
   \_ifodd \_trycs{ct!\_the\.citenum}{0}\_relax
       \_leftskip=.3\_hsize plus1fil \_parfillskip=0pt
       \_noindent
       \_rlap{\_hskip\_hsize \_kern-\_leftskip \_copy\.rqqbox}\_hfill
   \_else
       \_let\quotedby=\.quotedbyright
       \_rightskip=.3\_hsize plus 1fil
       \_noindent \_llap{\_copy\.lqqbox}%
   \_fi
   {\.printCite{#1}\_unskip}\_par
   \_ewref\_sxdef{{ct!\_the\.citenum}{\_string\.mypage}}%
%   \vskip-.3\baselineskip
   \_endinsert
}
\_def\.printCite#1{\_pdfliteral{2 Tr .15 w .9 g}#1\_pdfliteral{0 Tr 0 w 0 g}}
\_def\.printCite#1{{\Grey#1}}

\_def\.topinsertnopar{\_umidfalse \_upagefalse \_begingroup\_setbox0=\_vbox\_bgroup\_resetattrs}
```

The `\.lqqbox` and `\.rqqbox` include the graphical marks for quotations. First one is used at the left pages, second one at the right pages.
The macro `\quotedby{`⟨*author*⟩`}` puts the author of the quatation to the next line. The macro `\qutedbyright` (which is used at left pages) prints the ⟨*author*⟩ at the last line if there is sufficient space.

opbible.opm

```
\_newbox\.lqqbox
\_newbox\.rqqbox
\_setbox\.lqqbox=\_hbox{\_lower3pt\_hbox{\_setfontsize{at70pt}\_bf\LiRed„}}
\_setbox\.rqqbox=\_hbox{\_kern2pt\_lower38pt\_hbox{\_setfontsize{at70pt}\_bf\LiRed"}}
\_ht\.lqqbox=0pt \_dp\.lqqbox=0pt
\_ht\.rqqbox=0pt \_dp\.rqqbox=0pt

\_def\quotedby{\_par}
\_def\.quotedbyright#1{%
   \_unskip\_nobreak\_hfill\_penalty0\_hskip2em
   \_null\_nobreak\_hskip\_iindent\_hbox{#1}}
```

The following macros `\Cite`, `\insertCite` and `\swapCites` are used for insertion of citations to the two-cloumn printed articles. The `\Cite`⟨*label*⟩`{`⟨*text*⟩`}` simply saves the ⟨*text*⟩ to the macro `\c!`⟨*article-num*⟩`!`⟨*label*⟩. The `\insertCite`⟨*label*⟩⟨*feft-or-right*⟩ inserts the citation declared by `\Cite` ⟨*label*⟩ to the text using `\vadjust`. The variant `\left` and `\right` is processed or ignored. This depends on the parity of the current page, which is restored from `.ref` file and saved to the macro `\cp!`⟨*article-num*⟩`!`⟨*label*⟩.

opbible.opm

```
\_def\.Cite #1#2{\_sdef{c!\_the\.articlenum!#1}{#2}}
\_def\.insertCite #1#2{\_def\.citelabel{#1}%
   \_ifx\_left#2\.insertCiteleft
   \_else \_ifx#2\_right\.insertCiteright\_else
      \_errmessage{\_noexpand\insertCite#1: \_noexpand\left or \_noexpand\right expected}%
   \_fi\_fi
}
\_def\.insertCiteleft {%
   \_ifnum\.citepg=1
       \.printwarn{\_noexpand\.insertCite\.citelabel: \_noexpand\.swapCites activated}\_fi
   \_ifodd \_numexpr\_trycs{cp!\_the\.articlenum!\.citelabel}{0}+\.citepg\_relax
   \_else \.insertCitelr \_left \_fi
}
```

```
\_def\.insertCiteright{%
   \_ifodd \_numexpr\_trycs{cp!\_the\.articlenum!\.citelabel}{0}+\.citepg\_relax
   \.insertCitelr \_right \_fi
}
\_def\.insertCitelr#1{\_unskip\_vadjust{\_vbox{%
   \_ewref\_sxdef{{cp!\_the\.articlenum!\.citelabel}{\_string\.mypage}}%
   \_vskip6pt
   \_advance\_hsize by\_parindent
   \_typosize[12/16]\_bi\Grey
       \_ifx#1\_left
           \_def\quotedby{\_par\_hfill}
           \_rightskip=\_parindent plus1fil \_leftskip=0pt
           \_setbox0\_vbox{%
              \_medskip \_noindent
              \_llap{\_copy\.lqqbox}\_ignorespaces
              \.printCite{\_cs{c!\_the\.articlenum!\.citelabel}}\_medskip}%
           \_hbox{\_kern-\_parindent\_rlap{\White
              \_vrule height\_ht0 width\_hsize}\_box0}%
        \_else
           \_leftskip=\_parindent plus1fil
           \_parfillskip=0pt
           \_setbox0\_vbox{%
              \_medskip \_noindent
              \_rlap{\_hskip\_hsize\_kern-\_parindent\_copy\.rqqbox}\_hfill
              \_ignorespaces \.printCite{\_cs{c!\_the\.articlenum!\.citelabel}}\_medskip}%
           \_rlap{\_rlap{\White \_vrule height\_ht0 width\_hsize}\_box0}%
        \_fi
   \_vskip6pt
}}}
\_def\.swapCites{\_def\.citepg{1}}
\_def\.citepg{0}

\_nspublic \Cite \insertCite ;
```

Insertions into the intro text

opbible.opm

```
%% TBN page 236

\_newcount\.shapenum
\_newdimen\.ii \_newdimen\.w
\_def\.oblom #1 od #2 odsadit #3 {\_par \.ii=#1 \.w=\_hsize
   \_ifdim\.ii>\_zo \_advance\.w by-\.ii
   \_else \_advance\.w by\.ii \.ii=\_zo \_fi
   \.shapenum=1 \_tmpnum=0 \_def\.shapelist{}
   \_loop \_ifnum\.shapenum<#2 \_edef\.shapelist{\.shapelist\_zo\_hsize}%
      \_advance\.shapenum by1 \_repeat
   \_loop \_edef\.shapelist{\.shapelist\.ii\.w}%
      \_advance\_tmpnum by1 \_ifnum\_tmpnum<#3 \_repeat
   \_advance\.shapenum by#3 \_edef\.shapelist{\.shapelist\_zo\_hsize}
   \.doshape}
\_def\.doshape{\_parshape \.shapenum \.shapelist}
\_newcount\.globpar
\_ifx\_partokenset \_undefined \_def\.partoken{\par} \_else \_def\.partoken{\_par} \_fi
\_def\.doshape{\_global\.globpar=0 \_ea\_def\.partoken{\_ifhmode\.shapepar\_fi}}
\_def\.shapepar{\_prevgraf=\.globpar \_parshape\.shapenum\.shapelist
   \_endgraf \_global\.globpar=\_prevgraf
   \_ifnum \_prevgraf>\.shapenum \_ea\_let\.partoken=\_endgraf \_fi
}

\_def\.Citehereleft #1 (#2) #3{{
   \_par
           \_def\quotedby{\_par\_hfill}
           \_rightskip=\_parindent plus1fil \_leftskip=0pt
           \_setbox0\_vbox{{%
              \_typosize[12/16]\_bi\Grey
              \_hsize=.5\_hsize
              \_medskip \_noindent
              \_llap{\_copy\.lqqbox}\_ignorespaces
              \.printCite{#3}\_medskip}}%
   \_tmpdim=\_ht0 \_advance\_tmpdim by\_baselineskip
```

```
   \_xdef\.lines{\_the\_numexpr \_number\_tmpdim / \_number\_baselineskip \_relax}%
   \_nointerlineskip\_vbox to0pt{\_kern#1\_baselineskip #2
           \_hbox{\_rlap{\White
              \_kern-3mm\_vrule height\_ht0 width.5\_hsize}\_box0}%
   \_vss}}
   \_tmpdim=\_hsize \_advance\_tmpdim by-2\_leftskip
   \.oblom {.5\_tmpdim} od #1 odsadit {\.lines}
}
\_def\.Citehereright #1 (#2) #3{{
   \_par
           \_def\quotedby{\_par\_parfillskip=0pt \_hfill}
           \_leftskip=\_parindent plus1fill \_rightskip=0pt
           \_setbox0\_vbox{{%
              \_typosize[12/16]\_bi\Grey
              \_hsize=.5\_hsize
              \_vskip\_medskipamount \_rlap{\_kern\_hsize\_copy\.rqqbox}\_vskip-\_medskipamount
              \.printCite{\_noindent\_ignorespaces#3}\_medskip}}%
   \_tmpdim=\_ht0 \_advance\_tmpdim by\_baselineskip
   \_xdef\.lines{\_the\_numexpr \_number\_tmpdim / \_number\_baselineskip \_relax}%
   \_nointerlineskip\_vbox to0pt{\_kern#1\_baselineskip #2
        \_hbox to\_hsize{\_hss
           \_llap{\White \_vrule height\_ht0 width.5\_hsize \_kern-3mm}%
           \_llap{\_box0}}
   \_vss}}
   \_tmpdim=\_hsize \_advance\_tmpdim by-2\_leftskip
   \.oblom {-.5\_tmpdim} od #1 odsadit {\.lines}
}

\_def\.Citehere{\_par \_ifodd\_pageno \_ea\.Citehereright \_else \_ea\.Citehereleft \_fi}

\_nspublic \Citehere ;
```

`\insertBot {`⟨*title*⟩`} [`⟨*label*⟩`] (`⟨*params*⟩`) {`⟨*data*⟩`}` inserts a material from ⟨*data*⟩ to the bottom of the current page or next page if it is unable to fit to the current one. The material is titled by ⟨*title*⟩ and it can be referred by ⟨*label*⟩. The ⟨*params*⟩ can inclue a special setting used locally for the priting of this material.

`\putBot` ⟨*chapter*⟩`:`⟨*verse*⟩ `{`⟨*title*⟩`} [`⟨*label*⟩`] (`⟨*params*⟩`) {`⟨*data*⟩`}` behaves like `\insertBot`, but the result is printed to the bottom of the page where the verse ⟨*chapter*⟩`:`⟨*verse*⟩ is, or to the next page if the material is unable to fit to the current one.

opbible.opm

```
\_def\.insertBot #1#2[#3]#4(#5)#6{% {Title} [label] (params) {data}
   \.botinsert
      \_leftskip=0pt \_rightskip=0pt \_relax
      \.botTitle{#1}[#3]%
      \_kern3pt \_nobreak
      \_vbox{\_picwidth=\_hsize #5 #6}%
   \.endbot
}
\_def\.putBot #1 #2#3[#4]#5(#6)#7{% chap:verse {Title} [label] (params) {image-file.pdf}
   \_edef\.fullvref{\.gentovref{#1}}%
   \_edef\.fullvrefm{\_ea\.renumvref\.fullvref\_relax}%
   \_ea\.newaction\_ea{\.fullvrefm}{\.insertBot{#2}[#4](#6){#7}}%
}
\_nspublic \insertBot \putBot ;
```

`\.printintro` macro (by default) prints the itroduction of th book from the `\introfile`, prints the title "Introduction" (depending on the current language and puts all introduction text between `\.begblock` and `\.endblock`.

opbible.opm

```
\_def\.printintro{%
   \.begblock
      \_dest[i:\.currbook/]
      \.chaptit{\_mtext{intro}}%
      \_input{\introfile}
   \.endblock
}
```

Text block with grey background splittable to more pages is between `\.begblock` and `\.endblock` macros. It is used for introduction text. See also OpTeX trick 0031.

opbible.opm

```
\_newcount\.blocklevel  % nesting level of blocks
\_def\.begblock{\_par\_bgroup
   \_advance\.blocklevel by1 \_advance\_leftskip by\_iindent \_rightskip=\_leftskip
   \_medskip
   \_pdfsavepos \_ea\_wref\_ea\.Xblock\_ea{\_ea{\_the\.blocklevel}B{\_the\_pdflastypos}}
   \_nobreak \_medskip
}
\_def\.endblock{\_par\_nobreak\_medskip
   \_pdfsavepos \_ea\_wref\_ea\.Xblock\_ea{\_ea{\_the\.blocklevel}E{\_the\_pdflastypos}}
   \_medskip \_egroup
}
\_refdecl{%
   \_def\.Xblock#1#2#3{\_ifnum#1=1 \_edef\.tmp{frm:\_ea\_ignoresecond\_currpage}^^J
      \_unless\_ifcsname \.tmp \_endcsname \_sxdef{\.tmp}{}\_fi^^J
      \_sxdef{\.tmp}{\_cs{\.tmp}#2{#3}}\_fi}
}
\_newdimen\.frtop \_newdimen\.frbottom % positions of top and bottom text on the pages
\_def\.frcolor{.93 g } % light grey -- color of blocks.
\_pgbackground={%
   \_slet{_opb_tmp}{frm:\_the\_gpageno}
   \_ifx\.tmp\_undefined \_def\.tmp{}\_fi
   \.frtop=\_dimexpr \_pdfpageheight-\_voffset+\_smallskipamount\_relax
   \.frbottom=\_dimexpr\_pdfpageheight-\_voffset-\_vsize-\_medskipamount\_relax
   \_ifx\.frnext y \_edef\.tmp{B{\_number\.frtop}\.tmp}\_global\_let\.frnext n\_fi
   \_ea\.printframes \.tmp B{0}E{\_number\.frbottom}
   \_ifx\.frameslist\_empty \_else
   \_pdfliteral{q \.frcolor 1 0 0 1 0 \_bp{-\_pdfpageheight} cm \.frameslist Q}\_fi
}
\_def\.printframes B#1#2E#3{\_ifnum#1=0 \_else
   \.printframe {\_hoffset}{#3sp}{\_xhsize}{\_ifnum#1=-1 \_number\.frtop\_else#1\_fi sp-#3sp}
   \_ifx^#2^\_else \_global\_let\.frnext=y \_let\.printframes=\_relax \_fi
   \_ea\.printframes\_fi
}
\_def\.frameslist{}
\_def\.printframe #1#2#3#4{\_edef\.frameslist{\.frameslist
    \_bp{#1} \_bp{#2} \_bp{#3} \_bp{#4} re f }%
}
```

Insertions objects over pictures (maps)

`\putstext`⟨*x-pos*⟩ ⟨*y-pos*⟩ `{`⟨*text*⟩`}` behaves like `\puttext` from OpTEX, but moreover, it inserts a "white shadow" as a background of the text. It can be used as text printed over a pictures (maps etc.).
`\shadowedtext{text}` creates an `\hbox{`⟨*text*⟩`}` with "white shadow" as background.
`\shadowparameter` is a number of "transparency amount" used for "white shadows". User can re-define it but it must be done before first usage of `\putstext` or `\shadowedtext` and it is used for whole document.

opbible.opm

```
\_def\.putstext{\_ea\_ea\_ea\.putstextA\_scantwodimens}
\_def\.putstextA#1#2#3{%
   \_setbox0=\_hbox{\.shadowedtext{#3}}%
   \_dimen1=#1sp \_dimen2=#2sp \_puttextB
}
\_def\.shadowedtext#1{%
   \.insertwhiteshadowresources
   \_setbox0=\_hbox{#1}%
   \_hbox{\_tmpdim=\_ht0 \_advance\_tmpdim by\_dp0
      \_lower\_dp0\_hbox{%
         \_pdfliteral{q /trans gs 1 g
            \_fornum 1..10\_do{\_oval{\_bp{\_wd0}}{\_bp{\_tmpdim}}{2+##1/2} f } Q}}%
      \_box0}%
}
\_def\.insertwhiteshadowresources{%
   \_addextgstate{trans}{<</ca \shadowparameter>>}%
   \_glet\.insertwhiteshadowresources=\_relax
}
\def\shadowparameter{.1} % default value of "transparency"

\_nspublic \putstext \shadowedtext ;
```

`\c[`⟨*init-rot*⟩`/`⟨*step*⟩`]{`⟨*text*⟩`}` prints the ⟨*text*⟩ around a curve. Each letter or space from ⟨*text*⟩ is processed individually. The first letter is rotated by ⟨*init*⟩ degrees. Next letters are printed after ⟨*step*⟩ transformation is applied.

opbible.opm

```
\_def\.c[#1/#2]#3{% text podel krivky: \c[init-rotace/repetice]{text}
   \_pdfsave\_pdfrotate{#1}\_rlap{\_edef\.tmpb{#3}\_replstring\.tmpb{ }{{ }}\_def\.tmpa{#2}%
                \_ea\_foreach\.tmpb\_do{##1\.tmpa}}\_pdfrestore \_kern10mm
}
\_let\c=\_undefined
\_nspublic \c ;
```

`\town` ⟨*dimen*⟩ ⟨*dimen*⟩ puts a circle with given `\townparams` to the given place ⟨*dimen*⟩ ⟨*dimen*⟩. It works like `\puttext` ⟨*dimen*⟩ ⟨*dimen*⟩ `{`⟨*circle*⟩`}`.

opbible.opm

```
\_def\townparams{    % default parameters of the circle:
   \_hhkern=.8pt     % diameter of the disc
   \_lwidth=.5pt     % tickness of the outline
   \_fcolor=\Red     % color of the inner disc
   \_lcolor=\Black   % color of the outline
}
\_def\.town {\_ea\_ea\_ea\.townA\_scantwodimens}
\_def\.townA #1#2{\_setbox0=\_hbox{\_incircle[\_hhkern=0pt \_vvkern=0pt \townparams]{}}%
   \_dimen1=#1sp \_dimen2=#2sp \_puttextB
}
\_nspublic \town ;
```

14 Chiasm

The pair `\begChiasm`...`\endChiasm` defines chiasm environemnt. It behaves like `\begitems`...`\enditems`, but you can use given number of `*` which denotes the indentation level. The letters A, B, C, etc. will be prefixed automatically and when you are in the backward phase then C', B', A' are prefixed. You can try:

```
\begChiasm
* Předkové a rané zkušenosti (\<11:10-12:9>)
** Rané kontakty s ostatními národy (\<12:10-14:24>)
*** Smlouva s Bohem (\<15:1-17:27>)
** Pozdní kontakty s ostatními národy (\<18:1-21:34>)
* Potomci a smrt (\<22:1-25:18>)
\endChiasm
```

opbible.opm

```
\_def\.keepstyle{\_defaultitem=\_printitem}
\_def\.easylist{\_adef*{\.countlist}}
\_def\.aast{\.countlist}
\_def\.countlist{\_tmpnum=1 \.countlistA}
\_def\.countlistA{\_futurelet\.next\.countlistB}
\_def\.countlistB{\_ifx\.next\.aast \_ea\.countlistC\_else \_ea\.countlistD \_fi}
\_def\.countlistC#1{\_incr\_tmpnum \.countlistA}
\_def\.countlistD{%
   \_ifnum\_tmpnum>\_ilevel \_fornum \_ilevel..\_tmpnum-1 \_do{\_begitems\.easylist}\_else
   \_ifnum\_tmpnum<\_ilevel \_fornum \_tmpnum..\_ilevel-1 \_do{\_enditems}\_fi\_fi
   \_startitem}

\_def\.qq#1{{\_bf#1\_trycs{Level:\_the\_ilevel}{}}\_space\_aftergroup\.qqA}
\_def\.qqA{\_sdef{Level:\_the\_ilevel}{\_rlap{'}}}
\_def\.ChiasmNumbering{\_ea\.qq \_Uchar \_numexpr `A-1+\_ilevel\_relax\_space} % A, B, C, D, etc.
\_sdef{_item:q}{}%for chiasms with no leading alphabet letters
\_sdef{_item:Q}{\.ChiasmNumbering}
\_def\.begChiasm{\_begitems \.easylist \_style Q \.keepstyle}
\_def\.endChiasm{\_fornum 1..\_ilevel \_do{\_enditems}}

\_nspublic \begChiasm \endChiasm ;
```

15 Outline

opbible.opm

```
\_newdimen\.colsep
\.colsep=10pt

\_def\.Outline{
   \_medskip
%   \filbreak
   \.chaptit{\_mtext{outline}}%
   \_everylist={\_ifcase\_ilevel \_or \_style I \_or \_style A \_or \_style n \_fi}
   \_sdef{_item:A}{\_strut\_uppercase\_ea{\_athe\_itemnum}. }
   \_sdef{_item:I}{\_strut\_uppercase\_ea{\_romannumeral\_itemnum}. }
   \_hsize=.5\_hsize \_advance\_hsize by-\.colsep
   \_emergencystretch=40pt
   \_leftskip=0pt \_rightskip=0pt
}
\_def\.rightnote#1{\_par
   \_setbox0=\_hbox{\_kern\_hsize \_kern\.colsep
                    \_vtop{\_leftskip=0pt \_kern0pt\_noindent\_strut\_it#1}}
   \_ht0=0pt \_dp0=0pt \_box0 \_nointerlineskip
}
\_nspublic \Outline \rightnote ;
```

16 Timelines

- `\timeline`⟨*num*⟩ sets the total number of years (or other units) in time-line.
- `\timelinewidth`⟨*dimen*⟩ sets the width of time-line.
- `\l` is shortcut for `\baselineskip` (an be used in `\vskip` parameter).

opbible.opm

```
\_def\.l{\_baselineskip}
\_newcount\.timeline      \.timeline=100       % default
\_newdimen\.tlwidth       \.tlwidth=10cm % default
\_def\.timelinewidth{\_afterassignment\.timelinewidthA\.tlwidth}
\_def\.timelinewidthA{\_par\_hbox to\.tlwidth{}}

\_nspublic \l \timeline \timelinewidth ;
```

All objects used for creating time-line are defined by `\puttext`, i.e. they don't shift the current typesetting point.

`\arrowtext` ⟨*from*⟩`..`⟨*to*⟩ `(`⟨*settings*⟩`)` `{`⟨*text*⟩`}` creates a horizontal line with arrows. Its width and its position is given by ⟨*from*⟩`..`⟨*to*⟩ time units. The ⟨*settings*⟩ can include font selector, color settings of something similar for ⟨*text*⟩. The ⟨*text*⟩ is placed to the center of the line.

opbible.opm

```
\_def\.arrowtext #1..#2(#3)#4{%
   \_puttext \.pos{#1}0pt
      {\_lower.745ex\_hbox to\_dimexpr\.pos{#2}-\.pos{#1}{#3\.Larrow{ #4 }\.Rarrow}}
}
\_def\.Larrow{$\leftarrow$\_kern-.8em\_leaders\_vrule height.65ex depth-.42ex\_hfil}
\_def\.Rarrow{\_leaders\_vrule height.65ex depth-.42ex\_hfil\_kern-.8em$\rightarrow$}
\_def\.rule{\_leaders\_vrule height.12ex depth.12ex\_hfil}
\_def\.pos#1{\_expr{#1/\_the\.timeline}\.tlwidth}

\_nspublic \arrowtext ;
```

`\tlput` ⟨*above/below*⟩ ⟨*where*⟩ ⟨*llap or rlap or nothing*⟩ `(`⟨*format ot text*⟩`)` `{`⟨*text*⟩`}` puts the ⟨*text*⟩ to the timeline. The ⟨*text*⟩ can include more lines separated by `\cr`. The parameter ⟨*above/below*⟩ is `a` or `b` and means the ⟨*text*⟩ position: above the current point or below it. ⟨*where*⟩ is the position of the text in time units. ⟨*llap or rlap*⟩ is `\llap` or `\llap` and it menans that text is encapsulated to `\llap`, `\rlap`. If nothing is here the text is centered. The ⟨*format of text*⟩ can include the font setting, color setting etc.

opbible.opm

```
\_def\.tlput #1 #2 #3(#4)#5{%
   \_let\.Lhss=\_hss \_let\.Rhss=\_hss
   \_ifx#3\_rlap\_relax \_let\.Lhss=\_relax \_let\.Rhss=\_hss   \_fi
   \_ifx#3\_llap\_relax \_let\.Lhss=\_hss   \_let\.Rhss=\_relax \_fi
   \_puttext \.pos{#2}0pt {\_hbox to0pt{\.Lhss #4\.tltext#1{#5}\.Rhss}}
```

```
}
\_def\.tltext#1#2{\_ifx#1a\_vbox\_else
   \_vtop\_fi{\_kern0pt\_halign{\.Lhss##\.Rhss\_cr\_strut#2\_crcr}}%
}
\_nspublic \tlput ;
```

`\tline`⟨*from*⟩`..`⟨*to*⟩ prints the line. Its length and position is given by ⟨*from*⟩`..`⟨*to*⟩ time units. `\tlines{`⟨*data/separated/by/*⟩`}` creates a list of short vertical lines. Each line is represented by one `|`. The distance between lines (in time units) are given in the parameter.

opbible.opm

```
\_def\.tline #1..#2 {%
   \_puttext \.pos{#1}0pt {\_hbox to \_dimexpr\.pos{#2}-\.pos{#1}{\.rule}}
}
\_def\.tlines#1{\_puttext 0pt0pt{\_hbox{\_foreach #1|\_do##1|{\.vrul\_hskip\.pos{0##1}}}}}
\_def\.vrul{\_def\.vrul{\_kern-.12ex\_vrule height.7\.l depth.7\.l width.24ex \_kern-.12ex}}

\_nspublic \tline \tlines ;
```

17 Typesetting variants

By default, chapter numbers are in the outer margin and quotes characters too. The `\normalchapnumbers` macro moves chater numbers to the left side in the first paragraph, cquotes characters are removed and outer margins are reduced because there is no material in them.

opbible.opm

```
\_def\.normalchapnumbers{
   \_margins/2 a4 (25,25,20,20)mm
   \.lrmargin=0pt
   \_setbox0=\_box\.lqqbox \_setbox0=\_box\.rqqbox
   \_def\.printbeforefirst{%
      \_nobreak\_medskip
      \.trychapnote
      \_hangindent=\_parindent \_hangafter=-2
      \_noindent \_llap{\_vbox to0pt
         {\_kern-8pt\_hbox{\_setfontsize{at23pt}\_bf\Red\_the\.chapnum\_kern5pt}\_vss}}%
   }
}
\_nspublic \normalchapnumbers ;
```

18 Checking syntax

opbible.opm

```
\_def\.checksyntax#1 {%
   \_let\processbooks=\_relax
   \_ifx\_relax#1\_relax \_else
      \_begingroup
         \_the\.syntaxmacros
         \_wterm{^^J** checking file: #1 **^^J}
         \_input{#1}
         \_vfil\_break
      \_endgroup
   \_ea\.checksyntax \_fi
}

\_newtoks\.syntaxmacros
{\_catcode`<=13
\_global\.syntaxmacros={
\_def<#1>{\_bgroup
   \_message{checking \_unexpanded{<#1>}}%
   \_ifx\_relax#1\_relax \_errmessage{empty link}\.nobref\_else \_afterfi{\.checkbref#1>\.bref#1>}\_fi
   \_glet\.linkpre=\.linkpre \_glet\.linkfspec=\.linkfspec
   \_egroup
}
\_def\.checkbref#1#2>{%
   \_isinlist{.#1#2}{<}\_iftrue \_errmessage{duplicated \_string<}\.nobref\_else
   \_ifx"#1\.checkbrefQ #1#2>\_else \.checkbrefD #1#2>\_fi\_fi
}
```

```
\_def\.checkbrefQ "#1"#2#3>{\.checkbrefD #2#3>}
\_def\.checkbrefD #1>{%
      \_isinlist{.#1}{ }\_iftrue\.checkbrefS#1>\_else\.checkbrefN#1>\_fi
}
\_def\.checkbrefS #1 #2>{\.checkbrefN#2>}
\_def\.checkbrefN #1>{%
   \_def\.tmpb{#1}
   \_ifx\.tmpb\_empty \_errmessage{missing link data}\.nobref\_else
      \_replstring\.tmpb{:}{}\_replstring\.tmpb{-}{}\_replstring\.tmpb{_}{}%
      \_replstring\.tmpb{a}{}\_replstring\.tmpb{b}{}\_replstring\.tmpb{c}{}%
      \_setbox0=\_hbox{\_tmpnum=0\.tmpb\_relax}%
      \_ifdim\_wd0>0pt \_errmessage{nonnumeric link data}\.nobref\_fi
   \_fi
}
\_def\.nobref{\_def\.bref##1>{{\Red\_string<##1>}}}
\_def\.currbook{}
\_def\.prelinkB{BK}
\_def\.prelinkC{BK}
\_def\.prelinkV{0}
\_def\nochapbooks{BK}
\_let\<=<

\_def\x/#1/{\_def\.tmpb{#1}%
   \_isinlist\.tmpb\x\_iftrue \.badx
   \_else \_isinlist\.tmp<\_iftrue \.badx
   \_else \_isinlist\.tmp\enditems\_iftrue \.badx \_else \.x/#1/\_fi\_fi\_fi
}
\_def\.badx{\_errmessage{unclosed \_string\x/.../}}

\_def\Article[#1]{}
\_def\Cite #1 {\_par\_noindent{\_bf Cite: }}
\_def\insertCite #1#2{}

\_def\putArticle #1 #2[#3]#4(#5){}
\_def\putCite #1:#2 {\_par\_noindent{\_bf Cite: }}
\_def\putBot #1 #2[#3]#4(#5){\_vbox}

\_def\c[#1/#2]#3{#3}

\_long\_ea\_def\_csname Note\_endcsname #1 #2#3%

   {\_par \_let\.nextww\_undefined \_noindent{\_bf Note #1:} #3\_par}
}}
\_nspublic \checksyntax ;
```

19 Generating templates from templates

The \filegen{⟨*file-name-template*⟩}⟨*cr*⟩⟨*file-content-template*⟩⟨*cr*⟩\endfile saves ⟨*file-name-template*⟩ to \.filename and ⟨*file-content-template*⟩ to \.filecontent. Then it runs a loop over \genbooks. The \genbooks macro is defined by \BookTitle and user can re-define it.
The \.btitle{⟨*bmark or amark*⟩} expands to full title of the given book.

opbible.opm

```
\_newwrite\.outfile
\_def\.filegen #1 {\_par
   \_begingroup \_addto\genbooks{ }\_def\.filename{#1}%
   \_setverb \_endlinechar=`\^^J \.filegenA
}
\_ea\_def \_ea\.filegenA \_expanded{#1^^J\_csstring\\endfile#2^^J}{%
   \_def\.filecontent{#1}%
   \_ea\_foreach\genbooks \_do ##1 {%
      \_bgroup
      \_ifx^##1^\_else
      \_replstring\.filename{@@}{##1}%
      \_isfile{\.filename}\_iftrue \_opwarning{file "\.filename" exists already}%
      \_else
         \_wterm{creating file: \.filename}%
         \_immediate\_openout\.outfile={\.filename}%
         \_replstring\.filecontent{@@@}{\.btitle{##1}}%
```

```
          \_replstring\.filecontent{@@}{##1}%
          \_immediate\_write\.outfile{\.filecontent}\_immediate\_closeout\.outfile
       \_fi\_fi
       \_egroup
    }%
    \_endgroup
}
\_def\.btitle#1{\_ifcsname fb!#1\_endcsname \_trycs{btit!\_cs{fb!#1}}{#1}%
    \_else \_trycs{btit!#1}{#1}\fi
}
\_nspublic \filegen ;
```

20 TODO macros

The temporary macros are here. I plan to rewrite them.

opbible.opm

```

\_def\.quotationmarks#1#2{%
    \.cnvtext{"}{\.doquotmark}%
    \_def\.doquotmark {\_futurelet\.next\.doquotmarkA}%
    \_def\.doquotmarkA {%
       \_let\.doquotmarkB=#1\relax
       \_ea\_ifx\_space\.next \_let\.doquotmarkB=#2\_fi
       \_ifx\_space\.next \_let\.doquotmarkB=#2\_fi
       \_ifx\_endgraf\.next \_let\.doquotmarkB=#2\_fi
       \_ifx\_endcenter\.next \_let\.doquotmarkB=#2\_fi
       \_ifx.\.next \_let\.doquotmarkB=#2\_fi
       \_ifx,\.next \_let\.doquotmarkB=#2\_fi
       \.doquotmarkB}%
}
\_nspublic \quotationmarks ;

\_def\.chaptit#1{\_line{\_hss\.chapfont\Red#1\_hss}
    \_nobreak
}
\_def\.schaptit#1{\_bigskip\.chaptit{#1}\_nobreak\_medskip}

\_def\.subtit#1{\_par
    \_ifnum\.currversenum=1 \_else \_medskip\_fi
    \_line{\_indent\.subtitfont #1\_hss}\_nobreak
    \_ifnum\.currversenum=1 \_vskip-\_medskipamount\_fi
    \_smallskip
}
\_def\.subtitfont {\Red\_it}

\_nspublic \chaptit \schaptit \subtit ;

\_sdef{_mt:intro:en}{Introduction}    \_sdef{_mt:outline:en}{Outline}
\_sdef{_mt:intro:cs}{Úvod}            \_sdef{_mt:outline:cs}{Osnova}

\_def\dopsat{{\Red !!! DOPSAT !!! }}

\_def\.bibleinput#1 {\_bgroup
    \_catcode`##=13 \_bgroup\_lccode`~=`## \_lowercase{\_egroup\_let~}=\.processline
    \_input{#1}%
    \_egroup
}
\_let\FormatedBook=\_ignoreit  % for backward compatibility
\_let\CommentedBook=\_ignoreit % for backward compatibility
```

Active character < used for references.

opbible.opm

```
\_outer\_def\Note {\.Note}
\_outer\_def\ww   {\.ww}
\_outer\_def\ChapterPre  {\.ChapterPre}
\_outer\_def\ChapterPost {\.ChapterPost}
\_outer\_def\BookTilte   {\.BookTitle}

```

```
\_def\_afterload{\_adef<{\.bref}}
\_afterload

\_endnamespace
```

21 Index