1 Intro

Loading packages.

op-bible.opm

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

Basic settings.

op-bible.opm

```
\typosize[11/13]       % typesetting size of Bible text
\hyperlinks\Blue\Blue  % hyperlinks activated

\parindent=20pt
\enablemte             % micro typographical extensions enabled
\showboxbreadth=0
```

Fonts.

op-bible.opm

```
\fontfam[Heros]        % fonts for notes
\isfile{f-biblon.opm}\iftrue
   \fontfam[biblon]    % fonts for Bible text
\else
   \fontfam[lmfonts]   % alternative font for Bible text
\fi

\fontdef\bookfont{\setfontsize{at19.pt}\bf}
\fontdef\chapfont{\setfontsize{at13.pt}\bf}
\fontdef\markfont{\setfontsize{at7pt}\rm}
```

Auxiliary macros. `\printwarn {`⟨*text*⟩`}` prints warning. `\sedef {`⟨*name*⟩`}{`⟨*body*⟩`}` is expanded `\sdef`.

op-bible.opm

```
\def\printwarn#1{\wterm{WARNING (l.\the\inputlineno) #1}}
\def \sedef #1{\_ea\_edef \_csname#1\_endcsname}
```

2 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.

op-bible.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 `\tmpb` macro. The `\tmpb` macro is processed after all actions of given verse are done.

`\replpre{`⟨*prefix*⟩`}{`⟨*text*⟩`}{`⟨*fail*⟩`}` replaces first occurrence of ⟨*text*⟩ by ⟨*prefix*⟩`{`⟨*text*⟩`}` in `\tmpb` macro. If the ⟨*text*⟩ is empty then ⟨*prefix*⟩`{}` is inserted at the beginning of the `\tmpb`.

If ⟨*text*⟩ does not exists then ⟨*fail*⟩ is processed. The ⟨*fail*⟩ macro can use `\text` where ⟨*text*⟩ is saved.

op-bible.opm

```
\def\replpre#1#2#3{%
   \ifx^#2^\def\tmp{#1{}}\ea\ea\ea\def\ea\ea\ea\tmpb\ea\ea\ea{\ea\tmp\tmpb}%
   \else
      \def\replpredo##1#2##2\end{%
         \ifx^##2^\def\text{#2}#3% <fail>
         \else \replsave ##1#1{#2}##2\end \fi
      }%
      \def\replsave##1#2\end{\def\tmpb{##1}}%
      \ea\replpredo\tmpb#2\end
   \fi
}
```

3 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 \CommentedBook token list) or ⟨*chapter-num*⟩:⟨*verse-from*⟩-⟨*verse-to*⟩ (only ⟨*verse-from*⟩ is used for generating ⟨*gen-vref*⟩.
\gentovref{⟨*gen-vref*⟩} expands to ⟨*full-vref*⟩.

op-bible.opm

```
\newtoks\CommentedBook
\def\gentovref#1{\the\CommentedBook/\gentovrefA#1-\end}
\def\gentovrefA#1-#2\end{#1}
```

\renumref ⟨*full-vref*⟩\relax does re-caclulating of ⟨*full-vref*⟩ using \renum data.

op-bible.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.

op-bible.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 is 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 \nexttww and ⟨*pword*⟩ is in \nexttwwA. Otherwise, the macros \nexttww and \nexttwwA are undefined.
\Note does exactly following:

- Allocates new ⟨*note-num*⟩,
- Transforms ⟨*gen-vref*⟩ to ⟨*full-vref*⟩ using \gentovref.
- Modifies ⟨*full-vref*⟩ if \renum was declared using \renumvref and saves the result to \fullvrefm.
- Use \nexttww and \nexttwwA as ⟨*tword*⟩ and ⟨*pword*⟩ if they are defined.
- Otherwise transforms ⟨*word*⟩ to ⟨*tword*⟩ by \transformword.
- Reads ⟨*pword*⟩ (word to be printed in the note) if the alternative syntax with ={⟨*pword*⟩} is used. Else ⟨*pword*⟩ is equal to ⟨*tword*⟩.
- Defines \notetext!⟨*note-num*⟩ as ⟨*text*⟩.
- Defines \noteref!⟨*note-num*⟩ as ⟨*full-vref*⟩.
- 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*⟩}}}.

op-bible.opm

```
\newcount\notenum
\outer\def\Note #1 #2{%
   \incr\notenum
   \edef\fullvref{\gentovref{#1}}%
   \edef\fullvrefm{\ea\renumvref\fullvref\relax}%
   \def\tmp{#1}\sedef{notepre!\the\notenum}{\ea\renumlabel\fullvrefm\relax}%
   \ifx\nexttww\unefined
      {\def\printwarn##1{}\xdef\tword{\transformword{#2}}}%
   \else \xdef\tword{\nexttww}\fi
   \isnextchar={\NoteA}{\NoteA={}}%
}
\ifx\_partokenset\undefined
   \def\defnoteA{\def\NoteA=##1##2\par}
\else
```

```
   \def\defnoteA{\def\NoteA=##1##2\_par}
\fi
\defnoteA{%
   \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
   \edef\tmp{%
      \noexpand\newaction{\fullvrefm}%
      {\noexpand\replpre{\noexpand\doNote{\the\notenum}}{\tword}{\noexpand\notefail{\the\notenum}}}}%
   \tmp
}
```

`\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 `\isdivis` and `\iscolon`.

op-bible.opm

```
\def\renumlabel#1/#2\relax{#2%
   \ea\isdivis\tmp-\iffalse\else --\ea\renumlabelA\tmp\relax#2\relax \fi
}
\def\renumlabelA#1:#2-#3\relax#4:#5\relax{%
   \iscolon#3:\iffalse \the\numexpr#5+#3-#2\relax \else #3\fi
}
\def\isdivis#1-#2\iffalse{\ifx^#2^}
\def\iscolon#1:#2\iffalse{\ifx^#2^}
```

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.

op-bible.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
}
```

And the `\doNote{`⟨*note-num*⟩`}{`⟨*tword*⟩`}` prints the real note text in the second step, when the verse text from `\tmpb` is processed.

op-bible.opm

```
\def\prevtmpb{}
\def\doNote#1#2{%
   \edef\tmpb{\cs{notepre!#1}}%
   \notelog{\space\space \csstring\\Note \tmpb\space {#2}={\cs{pword!#1}} (#1)}%
   \noteinsert{%
      {\bf \ifx\prevtmpb\tmpb \else \tmpb \enskip \global\let\prevtmpb=\tmpb \fi
       \trymakedest{n:\currverse}%
       \ea \ifx \csname pword!#1\endcsname \empty
                 \else \ea\ea\ea\upcasefirst \csname pword!#1\endcsname. \fi}%
      \cs{notetext!#1}}%
   {\Red#2}%
}
\def\_printfnotemark{}
\def\_textindent#1{\noindent}
```

The phrase `{`⟨*word*⟩`}` used in notes must be exactly the same as the word used in the `.txs` text. But we want to capitalize the first letter of the ⟨*word*⟩ when printing. You can say `\let\upcasefirts=\relax` if you don't want this feature.

op-bible.opm

```
\def\upcasefirst #1{\uppercase{#1}}
```

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`.

op-bible.opm

```
\let\notelog=\wlog
```

4 Inserting data from format files

`\fmtpre {`⟨*gen-vref*⟩`}{`⟨*what*⟩`}` adds ⟨*what*⟩ to `\tmpc`, i.e. at the beginning of the verse.
`\ftmadd {`⟨*gen-vref*⟩`}{`⟨*what*⟩`}` adds ⟨*what*⟩ to `\tmpb`, 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`.

op-bible.opm

```
\let\FormatedBook=\CommentedBook
\def\fmtpre#1#2{\newaction{\gentovref{#1}}{\addto\tmpc{#2}}}
\def\fmtadd#1#2{\newaction{\gentovref{#1}}{\addto\tmpb{#2}}}
\def\fmtins#1#2#3{\newaction{\gentovref{#1}}{\replpre{\fmtafter{#3}}{#2}{\fmtfail{#3}}}}
\def\fmtafter#1#2{#2#1}
\def\fmtfail#1{\fmtwarn\addto\tmpc{#1}}
\def\fmtwarn{\printwarn{\string\fmtins: \currverse: The text "\unexpanded\ea{\text}" not found}}
```

`\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 `\encenter` because curious error (about # character) should be occur without this checking.

op-bible.opm

```
\newdimen\centermargin \centermargin=4em
\def\begcenter{\par \medskip
   \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}
```

5 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 `\processverse` ⟨*full-vref*⟩⟨*space*⟩⟨*verse-text*⟩`\end` is repeatedly processed.

op-bible.opm

```
\eoldef\processline#1{\processverse \currbook/#1\end}
```

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

- defines `\currverse` as ⟨*full-vref*⟩,
- prepares `\currversenum`, `\curversetext`, `\currchapnum` from ⟨*full-vref*⟩,
- defines `\tmpb` as ⟨*verse-text*⟩,
- processes all actions from `\alist!`⟨*full-vref*⟩,
- if `\currchapnum` changed, prints new chapter by `\printchap`
- prints verse from `\tmpb` using `\printverse`

op-bible.opm

```
\newcount\chapnum
\def\processverse #1 #2\end{%
   \edef\currverse{#1}%
   \preparechapverse #1
   \let\prelinkV=\currversenum
   \def\tmpb{#2}\def\tmpc{}%
   \csname alist!#1\endcsname
   \ifnum\currchapnum=\chapnum \else
       \let\prelinkC=\currchapnum \chapnum=\currchapnum\relax \printchap \fi
   \printverse
}
\def\preparechapverse #1/#2:#3 {\def\currchapnum{#2}%
   \isdivisin #3-\iftrue \defversefromto #3\end
   \else \def\currversenum{#3}\let\currversetext=\currversenum
   \fi
}
\def\defversefromto #1-#2\end{\def\currversenum{#1}\def\currversetext{#1--#2}}
```

`\printverse` prints verse from `\currversenum` and (possibly changed) `\tmpb`. It prints the single raised verse number first.
`\printchap` prints beginning of the new chapter. `\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`).

op-bible.opm

```
\def\printverse{%
   \tmpc % material accumulated by \fmtpre
   \ifnum\currversenum=1 \printbeforefirst \fi
   \quitvmode \trymakedest{v:\currverse}%
   \raise5pt\hbox{\unless\ifnum\currversenum=1 \markfont\currversetext\fi}%
   \tmpb \space
}
\def\printchap{\bigskip}

\def\printbeforefirst{%
   \par\nobreak
   \vbox to0pt{\null\vskip-1ex
      \hbox to\parindent{\hss \chapfont\Red \the\chapnum\ \hss}\vss}\nobreak \vskip-2ex
   \noindent \hangindent=\parindent \hangafter=-2 \relax}
```

6 Book titles, prefaces etc.

The macro `\BookTile` ⟨*b-mark*⟩ ⟨*a-mark*⟩ `{`⟨*title*⟩`}` declares titles of each Bible books. The ⟨*b-mark*⟩ is a book mark used in file names and ⟨*a-mark*⟩ is an actual book mark used in printed text.

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*⟩.

op-bible.opm

```
\outer\def\BookTitle #1 #2 #3{\sxdef{btit!#2}{#3}\sxdef{f!#2}{#1}}
```

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` and `\BookPost` are defined similarly.

op-bible.opm

```
\long\def\myaddto#1#2{\ifcsname#1\endcsname
   \gobal\ea\addto\csname#1\endcsname{#2}\else \global\sdef{#1}{#2}\fi}
\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}}
```

7 Processing books of the Bible

The `\processbooks` macro does two loops over all `\printedbooks`. The `\printedbooks` list can or cannot be finalized by a space. The first loop body sets `\pbook!`⟨*a-mark*⟩ used for hyperlinks. The second loop body does:

- Defines `\bmark` as ⟨*b-mark*⟩ (a mark of the book used in file names)
- Defines `\amark` as ⟨*a-mark*⟩ (an actual mark of the book used in text)
- Defines `\btit` as the book title.
- Calls `\bex!`⟨*a-mark*⟩ in order to set something extra.
- Calls `\BibleBook{`⟨*title*⟩`}{`⟨*a-mark*⟩`}`
- Prints title of the book to the terminal and to the log.
- Inputs format definition file.
- Inputs notes file.
- Calls `\bpr!`⟨*a-mark*⟩ in order to print a preface of the book,
- Inputs txs file with original text of the Bible using `\bibleinput`, i.e. prints the text.
- Calls `\bpo!`⟨*a-mark*⟩ in order to print a closing text of the book.

op-bible.opm

```
\def\processbooks {\par
   \checknochapbooks
   \ea\processbooksA \printedbooks\ignoreit. {}
   \ea\processbooksB \printedbooks\ignoreit. {}
}
\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
         \ea\BibleBook\ea{\btit}{#1}
         \cs{bex!#1}
         \wterm{** \cs{btit!#1} {#1} **}
         \input{\fmtfile}
         \input{\notesfile}
         \cs{bpr!#1}
         \bibleinput{\txsfile}
         \cs{bpo!#1}
      \endgroup
      \ea \processbooksB
   \fi
}
```

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`.

op-bible.opm

```
\def\checknochapbooks {%
   \ifx\nochapbooks\undefined
      \printwarn{\noexpand\nochapbooks (boks without chapters) undefined.}%
      \def\nochapbooks{}%
   \else  \edef\nochapbooks{\space\nochapbooks\space}\fi
}
```

Note that 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.

8 Bible references

We prepare temporary macros first.
`\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.

op-bible.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}
```

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.

op-bible.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.

op-bible.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.

op-bible.opm

```
\def\brefD{%
   \ifnum 0\ltextV=0 \def\ltextV{}\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.

op-bible.opm

```
\def\<{\let\prelinkB=\currbook \let\prelinkC=\currchapnum \let\prelinkV=\currversenum \bref}
```

`\oncebref` includes an additional macros which have to be processed in the single link, for example `\reduceref`. The `\everybref` token list includes macros which have to be applied for all links.

op-bible.opm

```
\newtoks\everybref
\def\oncebref{}
```

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 destiantion is itroduction then the ⟨*full-vref*⟩ has more reduced format: ⟨*book*⟩`/`.
`\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.

op-bible.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
   \linklog{\sspace <\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\end{\def\linkfspec{#1}}
\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\sspace{\space\space\space\space}
\def\linkpost{\if v\linkpre \else \linkpre\fi \space}
```

`\tracinglinks` and `\notracinglinks` are defined here.

op-bible.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. The link is created always if a user declared `\tracingallrefs`.

op-bible.opm

```
\def\createlink{{%
   \ifx\brefH\empty \let\linktext=\ltextP\fi
   \ea\isprintedbook\linkfspec \iftrue
   \link[\linkpre:\linkfspec]{\Blue}{\linktext}%
   \else {\Blue\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. Macro `\ensuredest` creates the item `\Xcreatedest` to .ref file and it is read in the second TeX run. The `\trymakedest` macro is used ad the begining of each verse, note etc. Only referenced destinations are created.

op-bible.opm

```
\def\ensuredest{\openref \immediate\_wref\Xcreatedest{{\linkpre:\linkfspec}}}
\refdecl{
   \def\Xcreatedest#1{\sxdef{dest!#1}{}}
}
\def\trymakedest#1{\ifcsname dest!#1\endcsname \dest[#1]%
   \global \ea\let\csname dest!#1\endcsname \undefined \fi}
```

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{`⟨*tmarkA*⟩`}` `\def\var!2{`⟨*tmark-B*⟩`}` `\def\var!3{`⟨*tmark-C*⟩`}` etc.

op-bible.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
}
```

`\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⟩}
```

op-bible.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}
```

`\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*⟩.

op-bible.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
}
```

`\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.

op-bible.opm

```
\outer\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}}
```

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 `\smark` or `\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` are defined as `\outer` in order to better diagnose mistakes with number of parameters of `\ww` or missig empty line after `\Note` text. But we want to skip such objects in `\switch` parameters. This is the reason why we run `\unsetouter` before the `\switch` parameter is read and we run `\setouter` in order to return to the normal setting.

op-bible.opm

```
\newtoks\switchD
\def\switch {\let\switchN=\switchA \unsetouter \switchN}
\long\def\switchA #1#2{\switchD={\setouter #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 \unsetouter \ea\switchN \else \setouter \fi}
\long\def\switchI #1#2{\futurelet\next\switchB}
\def\switchC{\ifx\tmp\smark \the\switchD
             \else\ifx\tmp\tmark \the\switchD \fi\fi
}
\def\unsetouter{\slet{ww}{relax}\slet{Note}{relax}}
\def\setouter{\slet{ww}{iww}\slet{Note}{iNote}}
\let\iww=\ww     % backup of outer ww
\let\iNote=\Note % backup of outer Note
```

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

op-bible.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>}
```

op-bible.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}%
}
```

10 Inserting notes to the page

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

op-bible.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.

op-bible.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
   \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 toOpt` 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 `1fill1` 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).

op-bible.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
   \vskip 0pt plus1filll minus8pt
}
\_def \noterule {\_kern-3pt {\Black \_hrule}\_kern 2.6pt }
```

11 TODO macros

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

op-bible.opm

```
\def\chaptit#1{\ifhmode \setbox0=\lastbox \par \nobreak\vskip-\baselineskip \fi
   \medskip{\chapfont\Red#1}\endgraf\nobreak\medskip}

\newcount \chapnum
\def\source#1{}
\def\BibleBook#1#2{\def\currbook{#2}\let\prelinkB=\currbook
   \bigskip {\bookfont #1}\par\nobreak\medskip \chapnum=0 }

\def\dopsat{{\Red !!! DOPSAT !!! }}
\def\pg{??}

\def\setvariant#1{}
\def\bibleinput#1 {\bgroup
   \catcode`##=13 \bgroup\lccode`~=`## \lowercase{\egroup\let~}=\processline
   \input #1
   \egroup
}
```

Active character `<` used for references.

op-bible.opm

```
\def\_afterload{\adef<{\bref}}
\_afterload

\endinput
```