OpBible

Automatic Study Bible Typesetting Using TEX

Version 1.0

Petr Olšák, Šimon Konečný

Revised 9. 4. 2023

https://www.overleaf.com/project/62e3832c89a3bfe389d592ba

It is too small a thing for you to be my servant
to restore the tribes of Jacob
and bring back those of Israel I have kept
I will also make you a light for the Gentiles,
that my salvation may reach to the ends of the earth.
Isaiah 49:6

Preface

The verse from Isaiah 49 on the title page is inspiring: It would be a small thing to write and typeset using TEX just one Bible commentary as one's life's work. The OpBible offers the opportunity to write a commentary on any Bible version to anyone anywhere; once the annotation apparatus is complete, the study Bible is typeset.

Šimon Konečný's long-standing vision of using TEX to automatically typeset study Bibles came to fruition through a collaboration with Petr Olšák. Starting from the realistic assumption that those who will use OpBible the most will also be those who have the least experience with TEX, we tried to make the usage as simple as possible.

However, there is no point in pretending that typesetting a study Bible is something trivial. To claim that it is a little tricky is a bit understatement; it is rather quite a challenge for any programmer.

Our intention to allow the production of a single commentary apparatus to gop with multiple Bible translations simultaneously was further complicated by the fact that different versions of the Bible have different verse numbers in some places, differently broken paragraphs, different structures of poetic passages, different placement of headings added by the translators; that some translations have completely different book titles (and hence abbreviations), that file names cannot have diacritics, but in book references can, and countless other such oddities.

All the problems we encountered have been overcome and everything works just fine to our satisfaction. The resulting collection of tools is quite complicated, therefore the definitions usage is irritatingly sensitive to transgressions against TEX syntax.

That's why OpBible includes tools to make it easier to find where typos, forgotten parentheses, etc. have been left; and in this documentation, wherever we have seen fit, we add paragraphs entitled **What to look out for** and **Recommended**.

Petr Olšák
Šimon Konečný

Contents

1 What makes the OpBible specific?

The main advantages of the OpBible over traditional typesetting are:

- **Price:**
  - The software itself is free as public domain under an Open Source license (see Preface).
  - More importantly, however, is the price which you'll save on a typesetter you'd otherwise have to pay for several years to hand-typeset the Study Bible.
- **Time:** Typesetting an entire Study Bible takes minutes, not years. To provide the full text of a Study Bible that breaks up the pages so that the notes end up on the same pages with the verses they're commenting on, it takes a system with a quad-core Ryzen 3 processor ten minutes. If you're working on a particular book and only processing one, it's usually a matter of seconds.
- **Flexibility:** The result is not one single version of the Bible in which you cannot add any changes. If you decide to edit some notes, delete some or add more, etc., you have this new revised version immediately typeset and ready for print.
- **Variability:** The result may not be just one Bible. The notes can be written in a way that allows as many Bibles as there are different translations in a given language to be typeset with the same note apparatus. For example, consider these six English translations of the Bible: (1) Bible in Basic English (BBE); (2) Jubilee2000; (3) NETfree; (4) Updated King James Version (UKJV); (5) Restored Names King James Version (RNKJV); and (6) Webster. Now let's take a brief look at Daniel, chapter 2, where the animals, countries, and other stuff of Daniel's vision are rendered as follows:
  - *Bear* is the same everywhere but
  - *Leopard* of most tranlations is *Tiger* in Jubilee2000;
  - usual *Ram* becomes *Male sheep,* as well as *clay* changes *potter's earth* in BBE;
  - ordinary *brass* turns into *bronze* in NETfree;
  - *Greece* of BBE, Jubilee2000, and NETfree is *Grecia* in UKJV, RNKJV, and Webster.

  You can write your commentary in the way that the result would be all six translations at the same time, with no need to modify the notes file, while the notes (or charts, for that matter), will display the phrases as they appear in your current Bible. See sample book of Daniel, chapter two, the chart of Daniel's visions.

  Interactive output format. The result of the processing is a PDF file that is richly linked with active links. These are an invaluable aid for proof-reading before submission to the print (see below).
- **Precision:**
  - The average study Bible contains around 20,000 notes and within them around 80,000 references to Bible verses and other notes. The notes are written by humans; it is inevitable that they will contain errors from oversight or typos. It is practically beyond the human power to find and correct all of them. It is, however, within the power of A machine. OpBible prints warning messages, if, for example, a note refers to a non-existent note or verse.
  - It checks for the accurateness of references to verses and notes that do exist (thus they do not trigger an error message), but for some reason they are not the right ones. The control is made possible by the fact that all references are active links that display the immediate context when the cursor hovers over the linked location (if you are viewing the PDF with the correct version of Evince, for example). And when clicked on, it jumps straight there.
  - The phrase that the note comments on is highlighted with a different font (e.g. bold). However, it will also searched for in the Bible text inside the verse in question and the pages will be broken in the way that places the comment is on the same page as its phrase, not just by verse number. In case that there are that two notes commenting on two different words in the same verse, and that the page break will come between those words, the notes will follow the phrases on their pages.
  - The Bible texts are prepared (for example, by downloading them from the Internet) in separate files and there is no need for them to be modified in any way. All notes on them and other typesetting instructions are written in separate files. Then TeX will put everything together.

What to look out for: Although the intent of this program is to spread God's Word and the Good news contained therein, the OpBible itself *forgives nothing!* After all, what'd you expect, it's a software. Go to God for forgiveness.) As you'll read below, the OpBible loads one entire book of the Bible (the so-called core text) into memory at a time and only then begins to digest your notes and piece them together into pages of their respective verses. If you happen to make a mistake in TEXsyntax (e.g., forget a closing bracket, etc.), TEX will see the error somewhere in the middle of this matching procedure, and this will also result in a cluttered error message, in which it is highly likely that you will get lost and not find your way around.

The OpBible anticipates the occurrence of situations like this and offers useful tools to help you out of similar predicaments, but you'd better be careful and consistent. You won't go wrong with the first written note, to go through all the translations you plan to use to make sure everything works as it should. After that you only need to work with one of them, but it is advisable to run TEX every time after you finish each new note, so by its protests you can easily find the one you have a mistake in.

Recommended: Use the Linux operating system. OpBible, as a macro file for TEX, will run on any system with the same results, but the Linux installation has several several not-to-be-missed advantages. Among the most significant of these is the Evince PDF viewer, whose newer (as of 2023) version can display the immediate context of a hyperlinked reference by simply hovering the cursor over it without having to click on it. This is an invaluable aid for checking the accuracy of links to Bible passages or notes. Other systems do not (yet) offer this facility. While the hyperlinks in these systems do jump to the appropriate location when clicked, they no longer hit back to where the jump signal was sent from, which is tedious. In this documentation, we assume a Linux installation.

And of course, we recommend – or rather, consider it a prerequisite – at least basic literacy in TEX. If you don't have any experience with TEX, try starting with here. The time invested in this preliminary education will not be wasted; the more you understand TEX, the better (and more enjoyably) you will find writing your Bible notes.

2 What we need to run OpBible

It is necessary to have some kind of text editor that does not leave any hidden formatting information in the text (designed for editing programs, for example). It is up to the user what kind of editor suits them. The ideal editor is one that recognizes the programming language by its source file's extension and colors words according to its syntax. E.g. Vi, Emacs, ... choose, whichever would suit you.

You also need to have a TEX distribution with the LuaTEX program and OpTEX default macros, and finally you need some kind of output (i.e. PDF files) viwer (we recommend the last version of Evince).

It isnot importnt on which operating system you will run this, but we recommend Linux.

It is also possible to work in online mode without installing anything, see section 2.2.

2.1 TEXthe local machine distribution

We recommend the latest TEXlive. The distribution can be installed on any operating system directly from the web. It is also included in common Linux distributions.

TEXlive contains the program LuaTEX, which will process your input files and produce output PDFs. It also includes the default macro set OpTEX that the OpBible macros need, and extends it with the options described in this manual. The default set of macros (often called format) defines how documents are markup and controls formatting. The TEX distributions include other default macro sets. The best known is probably LATEX, but it does not need OpBible.

2.2 Variant: remote access on OverLeaf.com

Overleaf is a web-based environment for shared preparation of TEX documents, perhaps by multiple users. You do not need to have TEXlive isntalated and can run it online via Overleaf. TODOis also available: default project containing files for processing by OpBible. As an Overleaf user, you can copy (clone) it to your project and continue working there. However, working with the TEX distribution directly on your computer is much faster and more convenient.

In English, there are over 450 different translations of the Bible. Most of them are not being used anymore, but still, you can choose from more than sixty various English versions.

3 Running TEX

If you have a TEX distribution installed on your machine (for example, TEXlive 2022) and if you have a command line available on your system, then you can run TEX using:

```
optex main.tex
```

where main.tex is the name of the main file (it can have a different name). The optex command runs LuaTEX with the OpTEXmacros. The result of the processing is a file main.pdf and the processing message is in the log file main.log.

You can test if this works for you (without OpBible macros for now). Create a simple main.tex file in a text editor with this content:

```
\fontfam[lm]
Hello world!
\bye
```

and run optex main.tex. View the resulting PDF file main.pdf with a PDF viewer.

4 Structure of files processed by TEX

4.1 Main file with information about all other files

The main file is the file that is submitted to TEX first. For example, it is listed on the command line to start TEXu. It contains information about what other files to be read by TEX. Finally, TEX will create a PDF file of the same name as the main file name.

The main file for RUNNING OpBible (for example, main.tex) might look something like this:

```
\load[opbible] % of the OpBible macro
\enlang % initializing English hyphenation patterns

% Translation variants:
\def\tmark {BBE} %  Bible in Basic English
%\def\tmark {Jubilee2000} % Jubilee 2000
%\def\tmark {NETfree} % New English Tranlation
%\def\tmark {UKJV} % Updated King James Version
%\def\tmark {RNKJV} % Restored Names King James Version
%\def\tmark {Webster} % Webster Bible

\input {Cze-vars.tex} % Phrase declarations for different translation options
\input {Cze-books.tex} % Book titles and bookmarks \amark

\def\txsfile {sources/Eng\tmark-\amark.txs} % Location of txs files
\def\fmtfile {formats/fmt-Eng\tmark-\amark.tex} % Location of fmt files
\def\notesfile {notes/notes-\amark.tex} % Location of notes files
\def\introfile {others/intro-\amark.tex} % Location of book introduction files
\def\articlefile {others/articles-\amark.tex} % Location of article files

\def\printedbooks {%
 Gen Exod Lev Num Deut Josh Judg Ruth 1Sam 2Sam 1Kgs 2Kgs 1Chr 2Chr Ezra Neh Esth Job Ps
 Prov Eccl Song Isa Jer Lam Ezek Dan Hos Joel Amos Obad Jonah Mic Nah Hab Zeph Hag Zech Mal
 Matt Mark Luke John Acts Rom 1Cor 2Cor Gal Eph Phil Col 1Thess 2Thess 1Tim 2Tim Titus
 Phlm Heb Jas 1Pet 2Pet 1John 2John 3John Jude Rev
}

\processbooks % Generates document with books declared in \printedbooks
\bye
```

Now let's see what each of these lines does and which ones will require modification on your part for the specific needs of your project.

Using \load[opbible], the TEX loads macros of the OpBible package. This is the most important program that takes care of the typesetting.

The \enlang command sets the English word division patterns, so it assumes English text. The en is an ISO language abbreviation, you can use other languages: \cslang for Czech, \delang for German, \eslang for Spanish etc. All these language options are listed in the OpTEX documentation.

The command `\def\tmark {⟨mark⟩}` defines the macro `\tmark` as a mark of the translation used (you can remember Translation mark). The marks of all available translations are listed in the file `vars.tex`. One of them should be selected as the mark of the currently processed translation. For example, `BBE` is the mark for the Bible in Basic English. In the example, six of the usual options for defining a translation marker in the case of English Bibles are given. Only one option (the one actually selected) does not have the % comment in front of it.

If you are looking at a book in progress in the `BBE` translation, leave the `main.tex` file in the above form. When you want to switch to, say, the UKJV, you will use the percent sign to comment out (i.e. make invisible to TEX) the line with the BBE, but make visible (uncomment) the line with the UKJV. Then the `\tmark` definition section will look like this:

```
% Translation variants:
%\def\tmark {BBE} %  Bible in Basic English
%\def\tmark {Jubilee2000} % Jubilee 2000
%\def\tmark {NETfree} % New English Tranlation
\def\tmark {UKJV} % Updated King James Version
%\def\tmark {RNKJV} % Restored Names King James Version
%\def\tmark {Webster} % Webster Bible
```

One of the translations must always be active, in other words, the `\tmark` has to be defined. If you forget to put a percent sign before the line you want to comment out, the world won't fall apart; the very last definition that TEX loads, which will redefine any previous ones, will apply.

`\input {vars.tex}` reads the configuration about the translation variants from the `vars.tex` file. See section 4.2 for details. Do not touch this line, even though you will probably be editing the `vars.tex` file called by this line.

The `\input {books.tex}` reads the information about the marks (abbreviations) of the books of the Bible and they are book names are assigned. This information is discussed in more detail in section 4.3.

The macro `\txsfile` (defined by `\def`) specifies the location of `.txs` files in the directory structure. For each book of the Bible, there has to be one `.txs` file containing the core text for that book. File names vary by book mark, and if there are multiple translations, the file name also includes the translation mark. In the `\txsfile` macro, you can use `\tmark` as the translation mark and `\amark` or `\bmark` as the book mark. For book marks, see section 4.3, for the format of `.txs` files, see the discussion in the section 4.4. In the example, `.txs` files are located in the `sources/` directory and are named `Eng`⟨*translation-tag*⟩`-`⟨*translation-tag-book*⟩`.txs`, so for example `EngBBE-Gen.tex`.

The English translations mentioned above are ready to go, you don't have to create them for yourself. If you need some other existing translation, you need to get it into a format usable for OpBible, in the same form as the `*.txs` files in the `sources/` directory. The `maketxs` script (see 4.4) will help you prepare individual `.txs` books from an existing source.

If you happen to be creating a brand new translation and plan to use it with OpBible, it probably wouldn't hurt to compose files one by one for each book directly in the desired format, see also 4.4.

The `\fmtfile` macro defines the location of the files specifying the formatting of the core text. Each book of the Bible of each translation used has its own formatting file. This is something that (unlike the notes) cannot be common to all translations, because the paragraph breaks and added headings can (and do) differ for each translation. Our intent was so-called non-destructive editing, in other words, formatting the biblical text without interfering with it. These files are discussed in section 4.5.

The macro `\notesfile` defines the location of the notes files. That's where you will write your commenting notes.

Each book Bible has its own notes file. The notes refer to a place in the core text, and it is the job of TEX to create pages with both the core text and the commenting notes. For more details on how to write notes files, see section 5.1. Notice that the note files are common to all translations, i.e. there are no separate files distinguished by `\tmark`. The note writing rules allow for the possibility to have various translations over one common notation, as long as they are in one language (e.g. English; see section 6). If you want to write notes for a completely different language, it is the best to start a new project (preferably in another directory) with different `.txs` files, different formatting and note files.

The macro `\introfile` specifies the files where the introductions to each book. It is possible (and advisable) to create a separate introduction file for each book.

The macro \articlefile specifies the names of files which contain the theological articles. The articles and similar stuff can be placed practically anywhere Bible, on any page with the core text (i.e., NOT in the Introduction).

The macro \printedbooks contains the marks of the books you want TEX to process. The sample calls to process the entire Bible, i.e. all 66 books of the Protestant canon. If you're only doing test prints, for example, you can process only some of the books of the Bible and have an alternative definition in the main file, for example \def\printedbooks{Dan}. Just put it after the definition for the entire Bible, because later definitions of the same macro take precedence over any earlier ones.

The \processbooks command starts processing all the books specified in the macro \printedbooks. For each book, TEX will read the corresponding core text from .txs file, formats it using the data from the appropriate formatting file and appends the notes from the appropriate note file. You don't need to change anything here.

The \bye command will terminate TEX. Anything you type after this farewell to TEXwill be ignored.

You can also add your own macros and settings to the main file before \processbooks, which will affect the entire typesetting completely.

For example, you can put a \ChapterPre{⟨*code*⟩} or \ChapterPost{⟨*code*⟩} declaration. These codes are then executed before and after each chapter.

4.2 File declaring translation variants

If we are working with a single translation variant, there is no need to create this file and use it. Then just remove (comment out) the instruction to read it from the main file.

In the example in section 4.1, the file vars.tex is read, which should contain the declaration of translation variant tags using \variants:

```
\variants ⟨number-of-variants⟩ {⟨mark⟩} {⟨mark⟩} ... {⟨mark⟩}
```

where ⟨*number-of-variants*⟩ is the number of translation variants (into a common language, for example, English), and then all the tags of the variant translations are listed. For example,

```
\variants 6 {BBE} {Jubilee2000} {NETfree} {UKJV} {RNKJV} {Webster}
```

declares the abbreviations for the 6 English translation variants: BBE for Bible in Basic English, Jubilee2000 for Jubilee 2000, NETfree for New English Tranlation, UKJV for Updated King James Version, RNKJV for Restored Names King James Version, and Webster for the Webster Bible.

The translation variants thus defined must match the definitions of \tmark in the main main.tex file, including upper or lower case.

Consider in advance the number of translations you want to use (changing their number later will be very difficult, though not impossible) and especially the order of the translations: the same order in which they are declared in the definition of \variants will apply to the entire project. In all the notes commenting on a phrase that spells differently in different translations, you will list the different versions in that precise order.

If you know that a phrase or word will appear more often than just in a single comment, you can define it directly in this vars.tex file using the \vdef command. The number of phrases listed after \vdef must be exactly the same as the number of ⟨*number-variant*⟩, each of them enclosed in brackets, and they must correspond to the translation variants in the same order as the variants listed in the declaration of \variants. For example:

```
\vdef {Greece} {Greece} {Greece} {Grecia} {Grecia} {Grecia}
```

declares that the name of Greece is transcribed differently in different variants of translation: it is *Greece* in BBE, Jubilee2000, and NETfree, but changes to Grecia in UKJV, RNKJV, and Webster.

When we write notes concerning this country in the notes file, we will just write \x/Greece/ (the first translation variant in the definition of \variants) and this will be turned into the corresponding phrase used in the currently processed translation that we have declared in the main file using \def\tmark{...}.

So after changing \def\tmark{...} in the main file, all occurrences of of \x/Greece/ in the text of notes will automatically start behaving differently and adapt to the phraseology of that particular translation variant. Then such words can be inflected or added various endings, for example: The entry \x/Greece/'s will yield the form Grecia's in the note under the UKJV, RNKJV, and Webster translations but will remain `Greece's` with BBE, Jubilee2000, and NETfree. This is discussed in more detail in section 6.

The command `\variants` declaring the abbreviations of the translation variants is unique (the only one) for the variants file, whereas there can be more `\vdef` commands defining variant phrases in the file, because there are of course many phrases that are used in different translation variants, not just the country of Greece.

The whole passages of text can be handled differently depending on the translation variant set. The branching command `\switch`is used for this purpose. It is discussed in more detail in the section 6.3. For example, the names of individual translations (which are then used in the page header) can be declared differently for different translations using `\def\bibname`:

```
\switch {BBE}{\def\bibname{Bible in Basic English}}%
        {Jubilee2000}{\def\bibname{Jubilee 2000}}%
        {NETfree}{\def\bibname{New English Tranlation}}%
        {UKJV}{\def\bibname{Updated King James Version}}%
        {RNKJV}{\def\bibname{Restored Names King James Version}}%
        {Webster}{\def\bibname{Webster Bible}}%
```

This particular declaration is a part of the already prepared `vars.tex` file.

4.3 Books names file

In the main file there is an instruction to read the books names file, for example `\input {books.tex}`. This file must contain the commands `\BookTitle` in the format:

`\BookTitle` ⟨*a-mark*⟩ ⟨*b-mark*⟩ `{`⟨*non-abbreviated book title*⟩`}`

There must be at least one space between the marks and the book title. The beginning of a file read this way might look like this:

```
\BookTitle Gn Gen {The First Book of Moses (Genesis)}
\BookTitle Ex Exod {The Second Book of Moses (Exodus)}
\BookTitle Lev {The Third Book of Moses (Levicitus)}
\BookTitle Nu Num {The Fourth Book of Moses (Numeri)}
\BookTitle Dt Deut {The Fifth Book of Moses (Deuteronomy)}
\BookTitle Jos Josh {Joshua}
\BookTitle Sd Judg {Judges}
...
```

In the first column after `\BookTitle`, there are ⟨*a-marks*⟩, which are further used in the text of the notes and are used to create links to various places in the Bible.

In the second column, there are ⟨*b-marks*⟩, which can be the same as ⟨*a-marks*⟩, but may also be different. It is possible, for example, that the names of the `.txs` files were created by exporting from some software and the bookmarks are different than we need to use in the text of our notes. Then it is possible that in the main file declare the location of the `.txs` files using `\bmark` instead of `\amark`, i.e.

```
\def\txsfile {sources/Cze\tmark-\bmark.txs}
```

and have the files `CzeBKR-Gen.txs`, `CzeBKR-Exod.txs`, while in the text we use tags `Gn`, `Ex`, etc., not `Gen`, `Exod`.

The macro `\amark` contains the ⟨*a-mark*⟩ of the currently processed book and the macro macro `\bmark` includes the ⟨*b-mark*⟩ of the book currently being processed.

Note that the macro `\printedbooks` (in the file `main.tex`) with the marks of all the books we want to to process (see section 4.1) contains ⟨*a-marks*⟩, not ⟨*b-marks*⟩.

In the third parameter after `\BookTitle`, the book titles are in compound brackets.

File with 66 entries of `\BookTitle` is generated automatically after extracting the base texts from Sword using `mod2tex` and `maketxs` (see section 4.4). It is possible to use it, but ⟨*a-tag*⟩ and book titles is probably need to be manually modified according to the conventions of the translation, as demonstrated in the example above.

Additional information about individual books can be added to the book title file using the \BookException, \BookPre, \BookPost commands. They have the following syntax:

```
\BookException ⟨a-mark⟩ {⟨text-exceptions⟩}
\BookPre ⟨a-mark⟩ {⟨text-before-book⟩}
\BookPost ⟨a-mark⟩ {⟨text-after-book⟩}
```

While ⟨*text-exceptions*⟩ is inserted before reading all files of the book defined by the ⟨*a-mark*⟩ inside a loop to read all books with \processbooks, then ⟨*text-before-book*⟩ is inserted after the book files have been read but before before processing the first verse. Finally, ⟨*text-after-book*⟩ is inserted after the last verse of the book.

Example of using \BookException: if you want to have ⟨*a-marks*⟩ in the file names and they contain diacritical diacritical marks, you can modify the value of the macro \amark so that it does not contain diacritics as follows:

```
\BookException Ž {\def\amark{Z}}
\BookException Př {\def\amark{Pr}}
\BookException Pís {\def\amark{Pis}}
\BookException Ř {\def\amark{R}}
\BookException Žd {\def\amark{Zd}}
```

Then, for example, the notes-\amark.tex files are actually named notes-Z.tex, notes-Pr.tex etc. In the text of the notes you normally use the tags books Ž, Př, Pís, etc.

There are five books in the Bible that have only one chapter (Obadiah, Philemon, 2 and 3 John and Jude). Because the references to them are not written with the chapter number (Ph 1:4) but only with the verse number (Ph 4), we must teach TEX which ones they are so that the chapter number ...but know that it's the number 1, which is not written anywhere. When referring to such books, a different way of interpreting the reference is then used, see section 7.2. This is achieved by defining a macro \nochapbooks which must contain the ⟨*a-tags*⟩ of these books: \def\nochapbooks {Abd Fm 2Jn 3Jn Ju}, obviously identical to those already given in the \BookTitle definition (in the Cze-books.tex file).

4.4 Base text format, called .txs files

The base text of the Bible is assumed to be stored in .txs files. (text source). Each of the 66 books of the Bible is stored in its own .txs file. The names of the .txs files and their locations must match the \def\txsfile declaration. in the main.tex file (see section 4.1).

Each line of the .txs file contains one Bible verse quoted by #⟨*chapter-number*⟩:⟨*verse-number*⟩. The verses must be listed in the correct order. For example, the beginning of the CzeBKR-Da.txs file looks like this (parts of the text are omitted in the sample):

```
    #1:1 The years of the third reign of Jehoiakim king of Judah, ... and besieged him.
    #1:2 And the Lord delivered Jehoiakim into his hand ... into the house of the treasure of
his God.
    ...
```

The basic texts of the Bible can be obtained, for example, from the modules of the Word https://www.crosswire.org/sword/modules/ModDisp.jsp?modType=Bibles. Individual .txs files can then be generated using the following procedure (on Linux):

Unzip the ZIP downloaded from the above mentioned web page (the so-called module). You need have the libsword-dev package installed on your computer and the program mod2tex, which is part of OpBible. Use installmgr -l to find out list of downloaded modules. If you have set the current directory in the location where you where you unzipped the ZIPs and the modules directory was created, then the modules are are found. The modules contain text in binary format, we need them to convert them to text format. To do this, just type the following into the command line:

```
mod2tex module > file
```

where module is the name of the module. In the resulting file you have the complete base text of the translation (module). For example, after

```
mod2tex CzeBKR > CzeBKR.out
```

the complete translation of the King James Bible is in the file `CzeBKR.out`. This can now be split into `.txs` files with the command

```
maketxs CzeBKR.out
```

This command will create the `CzeBKR-books.tex` file in addition to the 66 `.txs` files, in which which contains the titles and abbreviations of each book, so there is:

```
\BookTitle Gen Gen {Genesis}
\BookTitle Exod Exod {Exodus}
\BookTitle Leo Leo {Leviticus}
\BookTitle Num Num Num {Numbers}
\BookTitle Deut Deut {Deuteronomy}
\BookTitle Josh Josh {Joshua}
\♪BookTitle Judg Judg {Judges}
...
```

These titles are not in English, because the Sword source does not contain them. It is therefore it is necessary to manually edit this file and insert English titles instead of English ones. The abbreviations of the books are listed twice as well. The first one should be changed according to convention of abbreviations in Czech Bibles and the second abbreviation can be left. Then the declaration

```
\def\txsfile {sources/Cze\tmark-\bmark.txs}
```

in the main file will cause the created `.txs` files to be searched for in the `sources/` directory and their names are assumed `CzeBKR-Gen.txs`, `CzeBKR-Exod.txs`, `CzeBKR-Lev.txs`, etc.

If the rare case should arise that you were to compile the Bible from several different sources, say the Old Testament you wanted in Dr. Jan Hejčl's translation and the New Testament in František Žilka's translation, you would have to juggle the file names a bit so that the resulting definition of `\tmark` would be the same for the whole Bible. The two `*.out` files (e.g. `HEJCL.out` and `ZILKA.out`)[1] would have to be merged into one and then named e.g. `CzeHecjlZilka.out` and then only executed `maketxs CzeHejclZilka.out`. Then you can have a `\def\tmark{HejclZilka}` definition in the main `main.tex` file and the resulting study Bible will have Hejcl's Old Testament and Zilko's New.

With `.txs` files you will no longer need to be manipulated, edited or added to. V the `sources/` directory, you can have a „data store“ of all the core texts Bible for all used translation variants at once. In the case of the six variants translation, you have $6times66 = 396$ files.

If you have something in the `.txs` files that you want to format differently in the end, it is possible to use `cnvtext{⟨what⟩}{⟨how⟩}` in the main file. TeX in each verse of `.txs` file will look up all occurrences of ⟨*co*⟩ and replace them with ⟨*how*⟩. For example, if you have sections of text in square brackets in the `.txs` files, i.e. `[something like this]` and you want to print them in italics, write in the main file:

```
\cnvtext{[}{\bgroup\it} \cnvtext{]}{\/\egroup}
```

It may happen that the `.txs` file does not use the correct typographic quotes (i.e. in English ”...”), but instead there are programming quotes `"..."`. Without interfering with the `.txs` file, this can be fixed by adding an instruction to the main file:

```
\quotationmarks{"}{"}
```

This will automatically replace the programming quotes in the `.txs` file with English quotes typographic characters. Similarly, you can declare the replacement of with English quotation marks or other quotation marks, for example by declaring `\quotationmarks{"}{"}`. The programming quotation mark is then implicitly replaced by the opening typographic quotation mark (the first parameter in the declaration), but if the is followed by a space, end-of-verse, end-of-paragraph, period, or comma, it is is replaced by a closing typographic quotation mark.

4.5 Entries specifying the formatting of the base text in `fmt-*.tex` files

The base text in `.txs` files does not contain no formatting or additional information, such as chapter titles or where to end a paragraph or switch from block formatting to formatting to the middle of lines and back.

[1] Their modules are not on Sword, the `*.out` files would have to be created by downloading them from e.g. `https://obohu.cz` and converting them to the desired form with some clever script (or manually?).

Since we don't want to interfere with the underlying text, [2] you need these additional data to the corresponding verses using the special commands `\fmtadd`, `\fmtpre` and `\fmtins`. These commands are typically in the `fmt-*-*.tex` files, for example, `fmt-CzeBKR-Da.tex`. It is advisable to maintain these formatting files dependent on the book (Daniel in the example above), but also on the translation (the King James Bible in the example). While it is possible to start with one file for each book and make the files for the other translations as copies of the default, but in the end, it may be necessary for the different variants of the base text formatting instructions slightly modified according to the translation used.

The syntax for using these commands is as follows:

```
\fmtpre{⟨chapter-number⟩:⟨verse-number⟩}{⟨fmt-quote⟩}
\fmtadd{⟨chapter-number⟩:⟨version-number⟩}{⟨fmt-quote⟩}
\fmtins{⟨chapter-number⟩:⟨version-number⟩}{⟨phrase⟩}{⟨fmt-quote⟩}
```

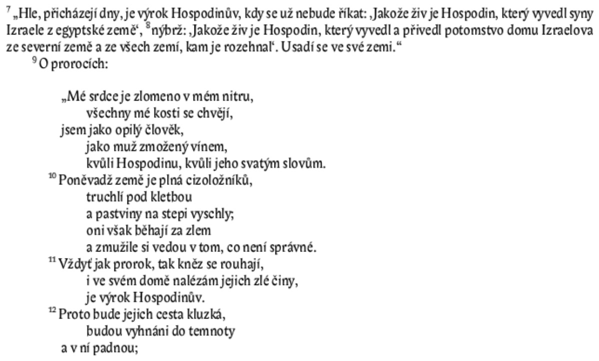
where ⟨*fmt-quote*⟩ is the „quote" that is passed to TEXu for formatting. For example, `\endgraf` marks the end of a paragraph. `\begcenter` opens a passage with centred text, and it must be closed (somewhere later) with `\endcenter`. Or `\chaptit{<text>}` inserts ⟨*text*⟩ as the chapter title, whereas `\schaptit{<text>}` inserts the pericope title somewhere other than before the first verse of the chapter, and makes space above and below the title.

The `\fmtpre` command inserts ⟨*fmt-quote*⟩ at the beginning of the specified verse (before the possibly printed verse number at the top index). The `\fmtadd` command inserts ⟨*fmt-quote*⟩ at the end of the specified verse. Finally, `\fmtins` inserts ⟨*fmt-loop*⟩ inside the verse after the first occurrence of the specified ⟨*phrase*⟩ that must literally exist in the verse. Otherwise, TEX prints a warning and does not insert ⟨*fmt-quote*⟩ at all.

An example of how the `\fmt*` commands can be used can be seen in the file `fmt-BKR-Da.tex`.

The command `\fmtfont`{⟨*chapter-number*⟩:⟨*chapter-number*⟩}{⟨*phrase*⟩}{⟨*font*⟩} is used to mark the selected phrase with the selected font. For example, the fmtfont1:26people in the book of Genesis will cause the word "people" will be printed in italics, because `\em` is an intelligent italic switch (it automatically adds the Italian correction `\/` after the word, which you don't have to worry about, as you would if you switched to italics with the regular `\it`). Any other font switch can be used instead of `\em`.

In addition to `\begcenter` and `\endcenter`, it is also possible to use controlled indentation with `\ind`⟨*number*⟩ (as indent). At the point of insertion, the line is terminated and the next line begins indented by ⟨*number*⟩ paragraph indents. Inserting such indent commands via `\fmtins` or `\fmtpre` can be quite laborious, however, and cluttered, yet the Bible is rife with poetic passages that require a lot of of differently indented lines. The command `\fmtpoetry` can be used for this purpose. we will first demonstrate its use in the example of the CEP translation of Jeremiah 23:9–15:

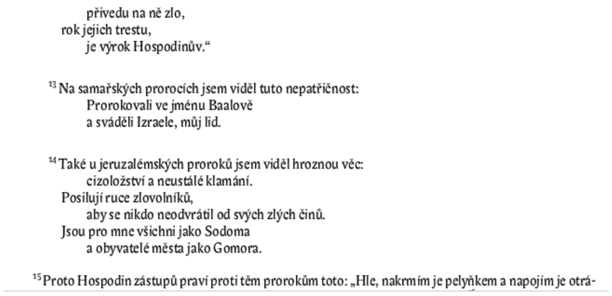


[7] „Hle, přicházejí dny, je výrok Hospodinův, kdy se už nebude říkat: ‚Jakože živ je Hospodin, který vyvedl syny Izraele z egyptské země', [8] nýbrž: ‚Jakože živ je Hospodin, který vyvedl a přivedl potomstvo domu Izraelova ze severní země a ze všech zemí, kam je rozehnal'. Usadí se ve své zemi."

[9] O prorocích:

„Mé srdce je zlomeno v mém nitru,
všechny mé kosti se chvějí,
jsem jako opilý člověk,
jako muž zmožený vínem,
kvůli Hospodinu, kvůli jeho svatým slovům.
[10] Poněvadž země je plná cizoložníků,
truchlí pod kletbou
a pastviny na stepi vyschly;
oni však běhají za zlem
a zmužile si vedou v tom, co není správné.
[11] Vždyť jak prorok, tak kněz se rouhají,
i ve svém domě nalézám jejich zlé činy,
je výrok Hospodinův.
[12] Proto bude jejich cesta kluzká,
budou vyhnáni do temnoty
a v ní padnou;

[2] Say that these are holy scriptures, hence „untouchable" text. The only thing that would entitle you to interfere with the basic text is the unlikely possibility that you might find an error in the source from the Word (or wherever you got it from), relative to the printed version. It is then really better to remove it on the spot.



přivedu na ně zlo,
rok jejich trestu,
je výrok Hospodinův.“

[13] Na samařských prorocích jsem viděl tuto nepatřičnost:
Prorokovali ve jménu Baalově
a sváděli Izraele, můj lid.

[14] Také u jeruzalémských proroků jsem viděl hroznou věc:
cizoložství a neustálé klamání.
Posilují ruce zlovolníků,
aby se nikdo neodvrátil od svých zlých činů.
Jsou pro mne všichni jako Sodoma
a obyvatelé města jako Gomora.

[15] Proto Hospodin zástupů praví proti těm prorokům toto: „Hle, nakrmím je pelyňkem a napojím je otrá-

The poetic part of this sample was planted like this:

```
\fmtpre{23:9}{\endgraf}
{\fmtpoetry{23:9}{prophecies:}
{23:10}{23:10}{23:10}{23:10}{23:10}
{\fmtpoetry{23:11}{\they blaspheme,}
{\fmtpoetry{23:12}{/ slippery,/// darkness// fall;/// evil,/// punishment,///}
{\fmtpoetry{23:13}{// impropriety:/// Baal's///}
{\fmtpoetry{23:14}{\the thing:// deception.// of the wicked,// of the deeds.// Sodom}
\fmtins{23:9}{prorocích:}{\bigskip}
\fmtpre{23:13}{\medskip}
\fmtpre{23:14}{\medskip}
\fmtpre{23:15}{\bigskip}
```

`\fmtpoetry{`⟨*chapter-number*⟩`:`⟨*version-number*⟩`}{`⟨*format-data*⟩`}` states format a particular of the referenced verse. In the ⟨*formatting-data*⟩, the words from line breaks followed by one or more slashes. Number of of slashes indicates how many paragraph indents the next line after the that word. ⟨*format-entries*⟩ must necessarily end with one or more slashes and may (but need not) begin with one or more slashes, if begin, the beginning of the line is indented by the appropriate number of paragraph indents, but the verse number is set off slightly to the left in the space of a paragraph indentation.

If we want to insert extra vertical spaces when using `\fmtpoetry`, it is is possible, but only after the `\fmtpoetry` command, as can be seen in the example where `\medskip` half line space and full line space `\bigskip`. The rule is that if you use `\fmtins` in the same place or `\fmtpre`, multiple formatting instructions will eventually be executed in in the reverse order of the `.fmt` file, so, for example, for verse 23:13 in the above example, first `\medskip` is executed at the beginning of the verse and then indent the next line by two paragraph indents according to `\fmtpoetry` (`\fmtpoetry{23:13}{// impropriety:...}`).

4.6 Notes in `notes-*.tex` files

The following files (located according to the `\notesfile` declaration in the main file) contain, among other things, notes on individual verses or parts of verses. To write notes is mainly used to write notes using the `\Note` command. Since this is the main reason why the the OpBible package was created, an entire section is devoted to it 5.1.

5 Notes and other objects linked to the base text

5.1 Notes linked to base text phrases: the `\Note` command

The main purpose of the OpBible tool is to create a PDF from the base text not only with the Bible text itself, but with linked notes. To do this, the for this purpose are, among others, the `notes-*.tex` files (for example, `notes-Gn.tex` for the book of Genesis), in which contain notes on individual phrases of the base text according to the following agreed format. Individual note is prefixed with the command `\Note` and has the form:

`\Note` ⟨*chapter-number*⟩`:`⟨*verse-number*⟩ `{`⟨*phrase*⟩`}` ⟨*text-notes*⟩
⟨*blank line*⟩

For example:

```
    \Note 1:2 {period} A reference to the vessels of the plundered temple, not to the deporta-
tion of the captives.
```

The example is from the `notes-Da.tex' file, i.e. the book notes file Daniel. Specifically, the note refe` *to-print*⟩`, i.e.  \begtt \Note` ⟨*chapter-number*⟩`:`⟨*verse-number*⟩ `{`⟨*phrase*⟩`}={`⟨*phrase-to-print*⟩`}` ⟨*text-notes*⟩ ⟨ Da.tex' file). It is therefore possible to have multiple notes for different phrases that are tied to the same Bible verse.

The individual notes in the source notes file are separated by blank lines. This is necessary, otherwise TEX would not know where the text ends when reading them of the note ends. It also increases the clarity of the source file. Additional lines of the note can (but need not) be indented.

If the note refers to the whole verse (i.e. without a specified phrase), write `{}`, i.e. an empty phrase to search for. For example:

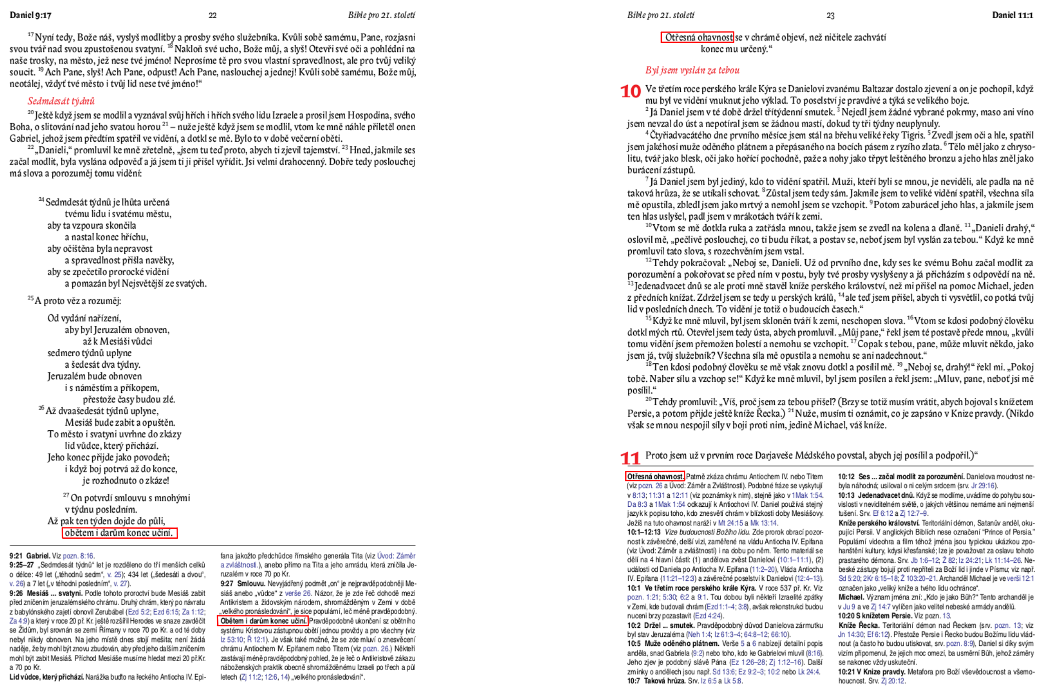
```
    \{Note 1:1-21 {}={Keeping ritual purity} The prophet quotes the context of his book
       by recounting a personal history (his and his friends') of captivity, education, loyalty
to God
       and service to King Nebuchadnezzar.
```

Moreover, this excerpt demonstrates that it is also possible to give the full range of verse numbers. If the ⟨*phrase*⟩ for the search is blank (as in this example), then the note will be placed on the same page as the beginning of the first verse in the specified range. If ⟨*phrase*⟩ is non-empty, it must occur in the first verse of the range of verses. The verse range will be printed in the beginning of the note anyway. The range symbol "-" is the only „minus" character normally available on keyboard. It is coded as 45. It must not be any special character that looks anything like a horizontal comma.

The order of the printed notes for a verse corresponds to the order of the phrases on which they refer to, in the main text. The order in which they are written in the source in the source file is therefore unaffected. Notes that bind using the empty parameter `{}` to the entire verse, are ordered first, and if there are more than one, then they are ordered in the source file.

5.2 Page break within a verse

If different annotations comment out different phrases within a single verse, and at the same time a page break after all formatting comes between those phrases in the base text, then the annotations follow their phrases to their respective pages. In other words, the page break is not determined by the verse number, but by the phrase, retrieved in the base text and synchronized with the note. See the example of Da 9:27, where such is the case, e.g., in B21; the search phrases are highlighted with red borders:



Daniel 9:17 22 *Bible pro 21. století*

[17] Nyní tedy, Bože náš, vyslyš modlitby a prosby svého služebníka. Kvůli sobě samému, Pane, rozjasni svou tvář nad svou zpustošenou svatyní. [18] Nakloň své ucho, Bože můj, a slyš! Otevři své oči a pohlédni na naše trosky, na město, jež nese tvé jméno! Neprosíme tě pro svou vlastní spravedlnost, ale pro tvůj veliký soucit. [19] Ach Pane, slyš! Ach Pane, odpusť! Ach Pane, naslouchej a jednej! Kvůli sobě samému, Bože můj, neotálej, vždyť tvé město i tvůj lid nese tvé jméno!“

Sedmdesát týdnů

[20] Ještě když jsem se modlil a vyznával svůj hřích i hřích svého lidu Izraele a prosil jsem Hospodina, svého Boha, o slitování nad jeho svatou horou [21] – nuže ještě když jsem se modlil, vtom ke mně náhle přiletěl onen Gabriel, jehož jsem předtím spatřil ve vidění, a dotkl se mě. Bylo to v době večerní oběti.

[22] „Danieli,“ promluvil ke mně zřetelně, „jsem tu teď proto, abych ti zjevil tajemství. [23] Hned, jakmile ses začal modlit, byla vyslána odpověď a já jsem ti ji přišel vyřídit. Jsi velmi drahocenný. Dobře tedy poslouchej má slova a porozuměj tomu vidění:

[24] Sedmdesát týdnů je lhůta určená
tvému lidu i svatému městu,
aby ta vzpoura skončila
a nastal konec hříchu,
aby očištěna byla nepravost
a spravedlnost přišla navěky,
aby se zpečetilo prorocké vidění
a pomazán byl Nejsvětější ze svatých.

[25] A proto věz a rozuměj:

Od vydání nařízení,
aby byl Jeruzalém obnoven,
až k Mesiáši vůdci
sedmero týdnů uplyne
a šedesát dva týdny.
Jeruzalém bude obnoven
i s náměstím a příkopem,
přestože časy budou zlé.
[26] Až dvaašedesát týdnů uplyne,
Mesiáš bude zabit a opuštěn.
To město i svatyni uvrhne do zkázy
lid vůdce, který přichází.
Jeho konec přijde jako povodeň;
i když boj potrvá až do konce,
je rozhodnuto o zkáze!

[27] On potvrdí smlouvu s mnohými
v týdnu posledním.
Až pak ten týden dojde do půli,
obětem i darům konec učiní.

9:21 Gabriel. Viz pozn. 8:16.
9:25–27 „Sedmdesát týdnů“ let je rozděleno do tří menších celků o délce: 49 let („téhodnů sedm“, v. 25); 434 let („šedesáti a dvou“, v. 26) a 7 let („v téhodni posledním“, v. 27).
9:26 Mesiáš ... svatyni. Podle tohoto proroctví bude Mesiáš zabit před zničením jeruzalémského chrámu. Druhý chrám, který po návratu z babylónského zajetí obnovil Zerubábel (Ezd 5:2; Ezd 6:15; Za 1:12; Za 4:9) a který v roce 20 př. Kr. ještě rozšířil Herodes ve snaze zavděčit se Židům, byl srovnán se zemí Římany v roce 70 po Kr. a od té doby nebyl nikdy obnoven. Na jeho místě dnes stojí mešita; není žádá naděje, že by mohl být znovu zbudován, aby před jeho dalším zničením mohl být zabit Mesiáš. Příchod Mesiáše musíme hledat mezi 20 př.Kr. a 70 po Kr.
Lid vůdce, který přichází. Narážka buďto na řeckého Antiocha IV. Epifana jakožto předchůdce římského generála Tita (viz Úvod: Záměr a zvláštnosti.), anebo přímo na Tita a jeho amrádu, která zničila Jeruzalém v roce 70 po Kr.
9:27 Smlouvu. Nevyjádřený podmět „on“ je nejpravděpodobněji Mesiáš anebo „vůdce“ z verše 26. Názor, že je zde řeč dohodě mezi Antikristem a židovským národem, shromážděným v Zemi v době „velkého pronásledování“, je sice populární, leč méně pravděpodobný.
Obětem i darům konec učiní. Pravděpodobně ukončení sz obětního systému Kristovou zástupnou obětí jednou provždy a pro všechny (viz Iz 53:10; Ř 12:1). Je však také možné, že se zde mluví o znesvěcení chrámu Antiochem IV. Epifanem nebo Titem (viz pozn. 26.) Někteří zastávají méně pravděpodobný pohled, že je řeč o Antikristově zákazu náboženských praktik obecně shromážděnému Izraeli po třech a půl letech (Zj 11:2; 12:6, 14) „velkého pronásledování“.

Bible pro 21. století 23 Daniel 11:1

Otřesná ohavnost se v chrámě objeví, než ničitele zachvátí
konec mu určený.“

Byl jsem vyslán za tebou

10 Ve třetím roce perského krále Kýra se Danielovi zvanému Baltazar dostalo zjevení a on je pochopil, když mu byl ve vidění vnuknut jeho výklad. To poselství je pravdivé a týká se velikého boje.

[2] Já Daniel jsem v té době držel třítýdenní smutek. [3] Nejedl jsem žádné vybrané pokrmy, maso ani víno jsem nevzal do úst a nepotíral jsem se žádnou mastí, dokud ty tři týdny neuplynuly.

[4] Čtyřiadvacátého dne prvního měsíce jsem stál na břehu veliké řeky Tigris. [5] Zvedl jsem oči a hle, spatřil jsem jakéhosi muže oděného plátnem a přepásaného na bocích pásem z ryzího zlata. [6] Tělo měl jako z chrysolitu, tvář jako blesk, oči jako hořící pochodně, paže a nohy jako třpyt leštěného bronzu a jeho hlas zněl jako burácení zástupů.

[7] Já Daniel jsem byl jediný, kdo to vidění spatřil. Muži, kteří byli se mnou, je neviděli, ale padla na ně taková hrůza, že se utíkali schovat. [8] Zůstal jsem tedy sám. Jakmile jsem to veliké vidění spatřil, všechna síla mě opustila, zbledl jsem jako mrtvý a nemohl jsem se vzchopit. [9] Potom zaburácel jeho hlas, a jakmile jsem ten hlas uslyšel, padl jsem v mrákotách tváří k zemi.

[10] Vtom se mě dotkla ruka a zatřásla mnou, takže jsem se zvedl na kolena a dlaně. [11] „Danieli drahý,“ oslovil mě, „pečlivě poslouchej, co ti budu říkat, a postav se, neboť jsem byl vyslán za tebou.“ Když ke mně promluvil tato slova, s rozechvěním jsem vstal.

[12] Tehdy pokračoval: „Neboj se, Danieli. Už od prvního dne, kdy ses ke svému Bohu začal modlit za porozumění a pokořovat se před ním v postu, byly tvé prosby vyslyšeny a já přicházím s odpovědí na ně. [13] Jedenadvacet dnů se ale proti mně stavěl kníže perského království, než mi přišel na pomoc Michael, jeden z předních knížat. Zdržel jsem se tedy u perských králů, [14] ale teď jsem přišel, abych ti vysvětlil, co potká tvůj lid v posledních dnech. To vidění je totiž o budoucích časech.“

[15] Když ke mně mluvil, byl jsem skloněn tváří k zemi, neschopen slova. [16] Vtom se kdosi podobný člověku dotkl mých rtů. Otevřel jsem tedy ústa, abych promluvil. „Můj pane,“ řekl jsem té postavě přede mnou, „kvůli tomu vidění jsem přemožen bolestí a nemohu se vzchopit. [17] Copak s tebou, pane, může mluvit někdo, jako jsem já, tvůj služebník? Všechna síla mě opustila a nemohu se ani nadechnout.“

[18] Ten kdosi podobný člověku se mě však znovu dotkl a posílil mě. [19] „Neboj se, drahý!“ řekl mi. „Pokoj tobě. Naber sílu a vzchop se!“ Když ke mně mluvil, byl jsem posílen a řekl jsem: „Mluv, pane, neboť jsi mě posílil.“

[20] Tehdy promluvil: „Víš, proč jsem za tebou přišel? (Brzy se totiž musím vrátit, abych bojoval s knížetem Persie, a potom přijde ještě kníže Řecka.) [21] Nuže, musím ti oznámit, co je zapsáno v Knize pravdy. (Nikdo však se mnou nespojil síly v boji proti nim, jedině Michael, váš kníže.

11 Proto jsem už v prvním roce Darjaveše Médského povstal, abych jej posílil a podpořil.)“

Otřesná ohavnost. Patmě zkáza chrámu Antiochem IV. nebo Titem (viz pozn. 26 a Úvod: Záměr a Zvláštnosti). Podobné fráze se vyskytují v 8:13; 11:31 a 12:11 (viz poznámky k nim), stejně jako v 1Mak 1:54. Da 8:3 a 1Mak 1:54 odkazují k Antiochovi IV. Daniel používá stejný jazyk k popisu toho, kdo znesvětí chrám v blízkosti doby Mesiášovy. Ježíš na tuto ohavnost naráží v Mt 24:15 a Mk 13:14.
10:1–12:13 *Vize budoucnosti Božího lidu.* Zde prorok obrací pozornost k závěrečné, delší vizi, zaměřené na vládu Antiocha IV. Epifana (viz Úvod: Záměr a zvláštnosti) i na dobu po něm. Tento materiál se dělí na 4 hlavní části: (1) andělova zvěst Danielovi (10:1–11:1), (2) události od Daniela po Antiocha IV. Epifana (11:2–20), Vláda Antiocha IV. Epifana (11:21–12:3) a závěrečné poselství k Danielovi (12:4–13).
10:1 Ve třetím roce perského krále Kýra. V roce 537 př. Kr. Viz pozn. 1:21; 5:30; 6:2 a 9:1. Tou dobou byli někteří Izraelité zpátky v Zemi, kde budovali chrám (Ezd 1:1–4; 3:8), avšak rekonstrukci budou nuceni brzy pozastavit (Ezd 4:24).
10:2 Držel ... smutek. Pravděpodobný důvod Danielova zármutku byl stav Jeruzaléma (Neh 1:4; Iz 61:3–4; 64:8–12; 66:10).
10:5 Muže oděného plátnem. Verše 5 a 6 nabízejí detailní popis anděla, snad Gabriela (9:2) nebo toho, kdo ke Gabrielovi mluvil (8:16). Jeho zjev je podobný slávě Pána (Ez 1:26–28; Zj 1:12–16). Další zmínky o andělech jsou např. Sd 13:6; Ez 9:2–3; 10:2 nebo Lk 24:4.
10:7 Taková hrůza. Srv. Iz 6:5 a Lk 5:8.
10:12 Ses ... začal modlit za porozumění. Danielova moudrost nebyla náhodná; usiloval o ni celým srdcem (srv. Jr 29:16).
10:13 Jedenadvacet dnů. Když se modlíme, uvádíme do pohybu souvislosti v neviditelném světě, o jakých většinou nemáme ani nejmenší tušení. Srv. Ef 6:12 a Zj 12:7–9.
Kníže perského království. Teritoriální démon, Satanův anděl, okupující Persii. V anglických Biblích nese označení “Prince of Persia.” Populární videohra a film téhož jména jsou typickou ukázkou zpohanštění kultury, kdysi křesťanské; lze je považovat za oslavu tohoto prastarého démona. Srv. Jb 1:6–12; Ž 82; Iz 24:21; Lk 11:14–26. Nebeské zástupy bojují proti nepříteli za Boží lid i jinde v Písmu; viz např. Sd 5:20; 2Kr 6:15–18; Ž 103:20–21. Archanděl Michael je ve verši 12:1 označen jako „veliký kníže a tvého lidu ochránce“.
Michael. Význam jména zní: „Kdo je jako Bůh?“ Tento archanděl je v Ju 9 a ve Zj 14:7 vylíčen jako velitel nebeské armády andělů.
10:20 S knížetem Persie. Viz pozn. 13.
Kníže Řecka. Teritoriální démon nad Řeckem (srv. pozn. 13; viz Jn 14:30; Ef 6:12). Přestože Persie i Řecko budou Božímu lidu vládnout (a často ho budou utiskovat, srv. pozn. 8:9), Daniel si díky svým vizím připomenul, že jejich moc omezí, ba usměrní Bůh, jehož záměry se nakonec vždy uskuteční.
10:21 V Knize pravdy. Metafora pro Boží vševědoucnost a všemohoucnost. Srv. Zj 20:12.

5.3 Commands to insert other objects

In addition to the \Note commands, the notes-*.tex file can be used to write instructions for inserting additional objects related to a particular verse in the base text. Such an object is placed in the [3] on the bottom of the page under the two columns of notes or quotes to be placed on top of the page. We define the link to the location of the text using the ⟨*chapter-number*⟩:⟨*verse-number*⟩, similar to Note. The object is placed on the same page where the corresponding verse begins. If the object does not fit on the page below the verse, it is placed at the bottom of the of the following page.

Insert images using \putImage (section 5.4), articles using \putArticle (section 5.5), quotes using \putCute (section 5.6), images across two pages in an open book using \putSpanImage or \putStanText (section 9.1).

The order in which you write the commands to drive these objects in the notes-*.tex file has no effect on the final appearance after printing. You can have all objects of one type concentrated in one place in the source file, but you can also have them placed between \Notes notes, typically by the verse number to which the objects are bound.

5.4 Insert images

To place images with a link to a specific verse, use the \putImage command:

```
\putImage ⟨chapter-number⟩:⟨verse-number⟩ {⟨caption⟩} [⟨image⟩] (⟨parameters⟩) {⟨file⟩}
```

For example:

```
\putImage 2:1 {Daniel's vision} [daniel'svision] () {Nabuco.pdf}
```

- ⟨*chapter-number*⟩:⟨*version-number*⟩ indicates the location to which the image is placed.
- ⟨*caption*⟩ is the caption to the image. In the default typographic settings, it is printed in a strip that separates the top of the page from the the image itself.
- ⟨*label*⟩ specifies a label for the image for reference. You can then click on the reference it using ...see image \ref[⟨*image*⟩] on the \pg page, which prints in the case of our example ...see image Daniel’s Vision on page 123. In doing so, the text in this link will be hyperlinked to the place where the where the image occurs. If you don’t want to use ⟨*link*⟩, type [] (i.e., as if it were an empty funnel).

[3] According to the implicit definition of the appearance of each page, this can be redefined.

- ⟨*parameters*⟩ specifies any additional instructions for formatting the image. They can be missing, as the example suggests. By default, the image is stretched to the full width of the page. If you want it smaller, for example, type ⟨*parameters*⟩ in the space for `\picw=9cm` which will cause the image to be 9 cm wide and centered.
- ⟨*file*⟩ is the full name of the image file. It can have the extension `pdf` (for vector images) or `png` or `jpg` for bitmap images. Image files must be stored in the `images/` directory. If are located elsewhere, the parameter `\picdir` must be set for that location, for example `\picdir={bibleImages/}`. This is sensible to set in the main file, see section 4.1.

5.5 Insert articles

An article with text (typically longer than a note) can be placed similarly to images, i.e. with the default typographic setting at the bottom of the of the page where the verse specified by ⟨*chapter-number*⟩:⟨*verse-number*⟩. If it does not fit there, it is inserted at the bottom of the the bottom of the next page. If it does not fit on a single page, its next page is inserted section at the bottom of the next page (and so on) until the whole article is inserted into the text).

Write the text of the articles for each book of the Bible in a file called a saved using the macro `\articlefile`, for example the `articles-Gn.tex` file contains all the articles for the book of Genesis. As the contents of what this file looks like will be described in a moment.

In the `notes-*.tex` file, you need to specify the article location request using `\putArticle`:

```
\putArticle ⟨article-number⟩:⟨article-number⟩ {⟨title⟩} [⟨article-number⟩] (⟨parameters⟩)
```

For example:

```
\putArticle 6:1 {Who was Darius the Mede?} [6] ()
```

- ⟨*chapter-number*⟩:⟨*verse-number*⟩ indicates the location to which the image is placed.
- ⟨*title*⟩ is the title of the article. It is printed similarly to the image.
- ⟨*article-number*⟩ is some numeric designation of the article that must be within the within a single book. Typically articles are linked to chapters, then the ⟨*article-number*⟩ should be the number of the corresponding chapter. According to this number, the article is not placed in the rate (this is what the ⟨*chapter-number*⟩:⟨*chapter-number*⟩), however, this number is used in references. For example, you can type `...see <"article" Da 6>a` and it will print ...see article Da 6. The format and properties of such links are described in section 7. In addition, the ⟨*article-number*⟩ is used to find the actual text of the article in the corresponding `atricles-*.tex` file. The
- ⟨*parameters*⟩ are parameters that specify the formatting of the article. TODO...

The text of the articles to be inserted must be in the `articles-*.tex` file. This file must contain a line quoted by `\Article` as follows:

```
\Article [⟨article-number⟩]
```

This is followed by the text of the article. Then it can continue with the next `Article [`⟨*article-number*⟩`]` followed by more text, etc. In one file all the articles for a given book are thus collected.

If you specify the request `\putArticle` and the corresponding file `article-*.tex` or the corresponding line in `\Article [`⟨*article-number*⟩`]` does not exist, the TEXem will end with an error.

5.6 Insert citations at the top of the page

The command `\putCite` ⟨*chapter-number*⟩`:`⟨*chapter-number*⟩ `{`⟨*text*⟩`}` inserts when default typographic setting of ⟨*text*⟩ as a quote at the beginning of page containing ⟨*chapter-number*⟩:⟨*verse-number*⟩.

The ⟨*text*⟩ itself may contain a `quotedby {<author>}` at the end. In such a case in this case, ⟨*author*⟩ is printed on a new line (in the case of a quotation placed on the left page) or, assuming there is enough space, it is indented more on the last line to the right (for a quote placed on the left-hand page).

For a controlled transition to a new line, you can use the `\nl` (new line) command in ⟨*text*⟩.

5.7 Insert citations in the margin of the article text

Article text is formatted into two columns by default. It is possible to break the outer column and insert a quote or other idea. This text will stick out into the outer margin.

Inside the article (i.e. just after \Article [⟨*article-number*⟩]) is you need to insert a declaration of the quoted text using: \Cite ⟨*letter*⟩ {⟨*text*⟩}. Here ⟨*letter*⟩ is typically A. But if you want to insert more than one quotes, you need to distinguish them with additional letters, i.e. B, C, etc., and all quotes at the beginning of the article.

The \Cite command only declares the quote. Its actual insertion into the text \insertCite ⟨*letter*⟩\left and at the same time elsewhere in the text of the article use \insertCite ⟨*letter*⟩\right. If the article is on the left-hand side of a double page, the quote is placed only by \insertCite⟨*letter*⟩\left, in other words \insertCite⟨*letter*⟩\right is ignored. If the article is on the right-hand side of an open double page, the placement is is governed only by \insertCite ⟨*letter*⟩\right. Just below the line in which the the appropriate \insertCite appears, the quote will actually appear. The line itself is is not split because of this (in other words, the paragraph is not terminated because of the insertion of the quote).

The reason why it is necessary to give two locations for \insertCite is as follows. Forward we don't know on which page (odd or even) the article with the inserted quote will appear. Since the quote should be in the outer margin, it should be placed on the left page in the first column, and on the right page in the second column, somewhere slightly different. The location of \insertCite ⟨*letter*⟩\left should therefore correspond to a row in the first column and \insertCite ⟨*letter*⟩\right to another row in the second column of the article.

It is wise to debug (i.e., test in advance) what the placement of the quote looks like for both options that may occur (left/right). If you want to to see how it works for the variant that doesn't match the correct position page, you can use the command at the beginning of the article text % \swapCites But this should not be left on for the final print, so activated \swapCites will cause a warning to the terminal and to the log file.

6 Different (but similar) versions of the base text

See section 4.2 for an introduction to this issue. There is detailed documentation on each option.

6.1 Declaring translation variants and using the \x command

If the variants are not declared with the \vdef command, then \x/⟨*phrase*⟩/ command used in the text will output ⟨*phrase*⟩. However, it is possible declare translation variants. The number of variants must be specified with the \variants command. (see section 4.2). This is done once in the whole document. Then the \vdef commands can follow, always with as many parameters as the number of variants specified by the \variants command. For example,

```
\variants 6 {BKR} {PSP} {CSP} {CEP} {B21} {SNC}
\{Belteshazzar} {Belteshazzar} {Belteshazzar} {Belteshazzar} {Belteshazzar}
\{Darius} {Dareios} {Dareios} {Darjaves} {Darjaves} {Darjaves} {Darjaves}
\{Darius} {Dariavesh} {Dariavesh} {Dariavesh} {Dariavesh} {Dariavesh}
\{Daniel} {Daniel} {Daniel} {Daniel} {Daniel} {Daniel} {Daniel}
\{Seven years} {Seven periods} {Seven times} {Seven years} {Seven periods} {Seven years}
```

If \def\tmark {⟨*variant*⟩} is now declared in the main file, then the \x/⟨*phrase*⟩/ will turn into the ⟨*phrase*⟩ of the specified ⟨*variant*⟩. In doing so, the ⟨*phrase*⟩ parameter of the \x command must be identical to the first phrase specified in the \vdef command. If, in our case, the main file is \def\tmark{BKR}, then

```
\x/Baltazar/ will output Baltazar
\x/Darius/ prints Darius
\x/seven years/ lists seven years
\x/whatever/ lists whatever
```

But if there is a \def\tmark{CEP} in the main file, then

```
\x/Balthasar/ will print Belthasar
\x/Darius/ lists Darius
\x/seven years/ lists seven years
\x/whatever/ will print anything and the terminal will warn,
                                          that the phrase /whatever/ has no declared translation.
```

For example, a note in the notes-Da.tex file (for the book of Daniel) might read:

```
\Note 5:31 {Darius}={\x/Darius/Median} Some schools claim that this and other
   (<6:1>, <6:6>, <6:9>, <6:25>, <6:28>; <9:1>; <11:1>)
   The references to \x/Darius/ Medes in the book of \x/Daniel/ are historical errors.
```

and this remark is printed at `\def\tmark{CEP}` as:

> **5:31 Darius the Mede** Some schools claim that this and other (6:1, 6:6, 6:9, 6:25, 6:28; 9:1; 11:1) references to Darius the Mede in the book of Daniel are historical errors.

Selected parameters of `\vdef` may be empty (written as `{}`), indicating an undefined phrase for that language. If such a phrase needs to be used with `\x/.../`, a warning will be printed. In addition, the parameter may contain a single `"` character, indicating that the the same phrase is used as in the previous parameter. So our example above might also look like this:

```
\vdef {Baltazar} {Belteshaccar} {Beltšasar} {"}        {Baltazar} {Belteshazzar}
\{Darius} {Dariavesh} {Dareios} {Dariavesh}          {"}
\{Darius} {Dariavesh} {Dareios} {Darjaves} {"}          {"}
\{Daniel} {Daniel} {Daniel} {"}        {"}           {"}
\vdef {seven years} {seven periods} {seven times} {seven years} {seven periods} {seven years}
```

Caution: Quotation marks in place of a phrase cannot be used in definitions of `\ww` (see chapter 6.2) that precede `\Note`. They only work in the `\vdef` definition in the variant file, valid for the entire Bible.

6.2 Variant phrase declarations for pairing notes with text

The `\Note` command may be immediately preceded by a declaration of the search word by variant translations using `\ww` (this is short for watchword). The `\ww` command has as many parameters as there are translation variants declared `\variants` command, and these parameters can be simple (in the format `{`⟨*wanted-phrase*⟩`}`) or compound (in the format `{`⟨*search-phrase*⟩`}={`⟨*what-to-print*⟩`}`). The immediately following `\Note` will then ignore its parameter for the search phrase and use the parameter from `\ww` corresponding to the language variant being processed. For example:

```
    \ww {literary arts and language} %BKR
        {%PSP
        {%CSP
        {%CEP
        {Babylonian writing and language} %B21
        {Chaldean language}={Chaldean language ... with national writing} %SNC
    \1:4 {} Babylonian literature was ...
```

The `Note 1:4` given here locates the text in verse 1:4 for `literary art and language`, when a variant translation of the KJV is being processed, and looks up the text “to the Chaldean literary art and language” when the CEP translation variant is being processed. The example assumes that six `\variants` have been declared using the `\variants` command translation variants in the specified order. You can also specify a different phrase for searching and for printing in a note, as shown in the following example:

```
    \ww {Baltazar} %BKR
        {Belteshaccar}={Belteshaccar ... Shadrach ... Méšach ...i Avéd-negó} %PSP
        {Belteshsar}={Belteshsar ... Shadrach ... Méšak ...i Abed-nego} %CSP
        {{Belteshazzar}={Belteshazzar, Hananiah Shadrach, Mishael Meshach, and Azariah Abed-
nego}
        {Baltasar}={Baltasar, Hananiah Shadrach, Mishael Meshach, and Azariah Abednego}
        {Baltshazzar}={Baltshazzar, Hananiah Shadrach, Mishael Meshach, and Azariah Abed-nego}
    \Note 1:7 {} The exact meaning of these names is a matter of debate...
```

The search and replace phrases are used exactly as they are written in parameters of the `\www` command. This does not apply to `\Note` notes, which do not have `\ww` in front of it. Then when using

`\Note` ⟨*string*⟩ `{`⟨*search-phrase*⟩`}` ⟨*text*⟩ ⟨*blank line*⟩

or

`\Note` ⟨*string*⟩ `{`⟨*search-phrase*⟩`}={`⟨*what-to-print*⟩`}` ⟨*text*⟩ ⟨*blank line*⟩

the ⟨*search-phrase*⟩ is first transformed by the data from `\vdef`. Only if this data does not exist for the search phrase, ⟨*search-phrase*⟩ is used so, as is.

recommend: For clarity, it is worthwhile to have each translation on a new line, and to label it after the commenting percentage, so that we know where what belongs without groping. The last line of the annotation file for a particular book that TeX loads should contain a single `\endinput` command. Whatever follows below this instruction on subsequent lines will not be seen by TeX. (But don't confuse it with `\end` or `\bye` so that it doesn't skip its run at this point, but continues reading other files.)

So under `\endinput` we can have a few lines ready, e.g. in this form:

```
\ww {}={} % BKR
    {}={} % PSP
    {}={} % CSP
    {}={} % CEP
    {}={} % B21
    {}={} % SNC
\1:1 {}
```

and then just copy these lines in place of the new note, edit the chapter number and verses after `\Note`, or delete `={}` where they are not needed. We don't lose track of where which phrases belong where, whether we write them out by hand or copy them from a Bible program or online resource.

6.3 Branching of text processing by translation variants

Use the `\switch` command to branch the input text processing in depending on the set value of the `\tmark` parameter, i.e. depending on the language variant currently being processed. The command has the following syntax:

```
\switch {⟨list of variants⟩} {⟨what to do⟩}%
        {⟨list of variants⟩} {⟨what to do⟩}% ... etc.
        {⟨list of variants⟩} {⟨what to do⟩}
```

The pairs `{`⟨*list of variants*⟩`}{`⟨*what to do*⟩`}` can be given as many times as you like. After each pair `{{`⟨*list of variants*⟩`}{`⟨*what to do*⟩`}` (except the last one pair) must be followed immediately and without spaces by another such pair, Therefore, when moving to the next line, write a percentage after the closing parenthesis to cover the gap from the end of the line. Gaps at the beginning of the next line do not matter. You can read the percentage after the pair as „next pair continues“.

The ⟨*list of variants*⟩ is a single translation variant or a list of translation variants separated by with a comma and no spaces. The TeX then works as follows. a variant defined by the `\mark` parameter occurs in the ⟨*variant list*⟩, the the following ⟨*what to do*⟩ is executed. If there is no such variant, the following ⟨*what to do*⟩ is skipped. Example:

```
\switch {BKR} {Balsazar}%
        {PSP} {Belshazzar}%
        {CSP,CEP,B21,SNC} {Belshazzar}
```

The example shows how to print the word Belshazzar depending on the language variant being processed.

Once TeX finds a match and does ⟨*what to do*⟩, then the possible following entries within the same `\switch` command are skipped. Furthermore, the rule is that if ⟨*variant list*⟩ is empty, ⟨*what to do*⟩ is always executed if not skipped according to the previous rule. So an empty ⟨*variant list*⟩ on at the end of the `\switch` parameter pairs is evaluated as „other cases“. The example above can also be written as follows:

```
\switch {BKR} {Balsazar}%
        {PSP} {Belshazzar}%
        {} {Belshazzar}
```

The `\switch` command can be used not only for single phrases within notes `\Note`, but also to entire sections of input text containing, for example, `\Note`. several `\Note` notes, several definitions, etc.

The `\switch` command cannot be used in the parameters of other macros. There it only works command `\x/`⟨*phrase*⟩`/`.

6.4 Renumbering verses according to translation variants

Some translation variants have different verse numbering. In this case, the the `\renum` command can be used as follows:

```
\renum ⟨short-book⟩ ⟨default-ch-ch-chapter⟩:⟨default-ch-verse⟩ = ⟨translation⟩ ⟨ch-chapter⟩:⟨ch-from⟩-⟨ch-
to⟩
```

where ⟨*translation*⟩ is the tag for a particular translation. Instead of ⟨*default-ch-h-handle*⟩:⟨*default-ch-verse*⟩, it is used for `\def\tmark{`⟨*translation*⟩`}` is actually ⟨*ch-kap*⟩:⟨*ch-from*⟩. Such renumbering does not only apply to this verse, but the entire stretch of verses defined by the range ⟨*ch-from*⟩-⟨*ch-do*⟩.

For example:

```
\renum Da 6:1 = CSP 6:2-29
\renum Da 6:1 = CEP 6:2-29
\renum Da 6:1 = B21 6:2-29
\renum Da 6:1 = SNC 6:2-29
```

The Pauline Study Translation has the same numbering as the King James Bible at the turn of chapters 5 and 6, so there is no need to renumber it. The other translations give the content of the verse in BDB 6:1 (`It pleased` Darius'...) under 6:2, and the whole of chapter 6 is offset by one number from the KJV. KJV and PSP 6:28 is in the CSP, CEP, B21, and SNC 6:29. We number the notes according to the translation given as the first parameter of the definition of `\variants` in the `Cze-vars.tex` file, i.e. in our example `BKR`, including references, e.g. see note 5:31. Renumbered translations change the note number according to the actual verse number that the note comments on, including the reference, which is printed as See note 6:1.

If the renumbering is to refer to a single verse, the identical ⟨*from*⟩ and ⟨*from*⟩ should be given, as in

```
\renum Da 5:31 = CSP 6:1-1
\renum Da 5:31 = CEP 6:1-1
\renum Da 5:31 = B21 6:1-1
\renum Da 5:31 = SNC 6:1-1
```

After such a declaration, the final remark

```
\Note 5:31 {Darius}={\x/Darius/Median} Some schools claim that this and other
   (<6:1>, <6:6>, <6:9>, <6:25>, <6:28>; <9:1>; <11:1>)
   The references to \x/Darius/ Medes in the book of \x/Daniel/ are historical errors.
```

prints as follows:

> **6:1 Darius the Mede** Some schools claim that this and other (6:2, 6:7, 6:10, 6:26, 6:29; 9:1; 11:1) references to Darius the Mede in the book of Daniel are historical errors.

Even such tidbits as shifting a number by a mere fraction of a verse can be dealt with. BDB Da 2:28 "there is a God in heaven who reveals secret things" is in the SNC "God who is in heaven reveals secrets." 2:27; So before the note on this phrase, we will put `Renum Da 2:28 = SNC 2:27-27`. But then we also need a note on the phrase "in the latter days" of the same verse, but – world wonder – the phrase "in the latter days" is already is in the SNC in verse 2:28! We don't need to speculate why the SNC translators try such tricks on us by sending half a verse under a different number than everyone else; it is enough that we know how to handle it: We write renum Da 2:28 = SNC 2:28-28" before this new note, and everything works as it should: Where the numbering diverges, it renumbers; where it it matches, it stays the same.

7 Methods of creating hyperlinks

A hyperlink is a section of text by which the reader, even after printing it, can tell to what other place in the text (or in the internet) he can look at. Thus, it typically contains a numerical indication of page or the number of a chapter, section, etc. In addition, if the reader is working with a PDF viewer, then this section of text is also active, i.e., when the mouse hovers over mouse cursor over this text can be clicked and the PDF viewer will go to the specified point in the document (or to the Internet).

The Bible is invariably structured text. It contains (in the Protestant canon) 66 books with established markers for those books, each book has its chapters numbered from one and each chapter has verses numbered from one. So there is no need to have TeX generate these numbers automatically (as it does when typesetting, say, a technical text that is broken down into chapters and sections), and thus there is no need to use reams in the source in the source document (which TeX assigns to the generated numbers during processing) and to reference the these funnels, as described in section 1.4.3 of the OpTeX documentation. It is much more efficient to refer directly to a specific place in the Bible that already has for many centuries now, with a fixed book, chapter number and verse number. (The Archbishop of Canterbury, Stephen Langton, in the early 13th century, when he was teaching at the University of Paris [and not yet an archbishop] divided the Bible into chapters. Then, in the mid-16th century, the French printer Robert Estienne divided the New Testament into verses and added the Old Testament, which Jewish scribes had centuries earlier divided into verses, but cunningly not into chapters. Since 1553, when Estienne published the first French Bible so numbered, we have used this system to this day.)

We write references to a specific place in the Bible between < and >. these characters is printed as written (with the exceptions noted below). But TeX must be able to interpret the reference correctly in order to make it active with a possible click-through to the correct place in the Bible. This is done by for this purpose, which is the subject of this section.

Basic rule with full details

The reference between < and > is of the form `"`⟨*text*⟩`"` ⟨*data*⟩ or just ⟨*data*⟩. However, the complete ⟨*data*⟩ is of the form ⟨*book*⟩ ⟨*chapter*⟩`:`⟨*verse*⟩. Here ⟨*book*⟩ is the abbreviation of the book (it must be followed by space), ⟨*chapter*⟩ is the chapter number, and ⟨*verse*⟩ is the verse number. Example:

```
... see also verse <Jr 8:13>
... see also <"verse" Jr 8:13>
```

In the first case it prints ... see also verse Jr 8:13 and in the second case ... see also verse Jr 8:13. The active (allowing click-through) will be in both cases the area marked in blue here and the click-through leads to Jr 8:13.

7.1 Link specifier

The link ending character > may be followed closely by a link specifier, which is one of the letters:

- `n` ... refers to a note,
- `g` ... refers to a gloss (which will come into play if another version of OpBible with double-column typesetting of the base text is ever created),
- `a` ... refers to an article,
- `i` ... refers to the introduction.

The link specifier is not printed, it is just internal information where it should the active link (i.e. mouse click). If it is not followed by a terminating character > is followed by any of these specifiers, it is a link to the verse (this is probably the most common case). Example of a note reference:

```
... see <"note on" Jr 7:4>n for more information.
```

prints ... see note to Jr 7:4 for more information. The click-through leads to the first note to Jr 7:4, not to the verse itself.

In the case of a reference to an article (specifier `a`), the full entry has the format ⟨*book*⟩ ⟨*chapter*⟩, i.e., the verse information is missing because the articles can be understood as introductions to chapters. In the case of a reference to the introduction to a book (specifier `i`), the full entry is in the format ⟨*book*⟩ (lacking both chapter and verse information) because it are book introductions. In other cases, the full entry has the format as it was discussed in section ?? except as described in section 7.2

7.2 Exception for the format of the full entry for some books

Books Abd, Fm, 2Jn, 3Jn, and Ju are not divided into chapters. In a reference to a verse, note to a verse or a gloss to one of these books, the chapter information is missing and the format of a complete reference looks like this: ⟨*book*⟩ ⟨*verse*⟩. To let TeX know that to apply this exception, the list of abbreviations for these books needs to be defined in macro `\nochapbooks`. For example, the `Cze-books.tex` file says

```
\def\nochapbooks{Abd Fm 2Jn 3Jn Ju}
```

Since there are different book abbreviations for different languages, this macro is needed to define this macro depending on the language used.

7.3 Incomplete

Sometimes the reader can determine the location of a verse from the context, so ⟨*data*⟩ may not be complete. In incomplete data, there may be a missing ⟨*book*⟩ or ⟨*book*⟩ ⟨*chapter*⟩: or all of it. For example:

```
... we also see an analogy in <"verses" Jr 8:13>, <9:7> and <11:3>
... see verses <Jr 8:13>, <15>, <17>
... see all <"verses" Jr 8:13>--<22>,
... (cf. <Jr 8:13> and <"his note">n).
```

Printed: ... We also see an analogy in verses Jr 8:13, 9:7 and 11:3 ... see verses Jr 8:13, 15, 17 ... see all of verse Jr 8:13–22. ... (cf. Jr 8:13 and note).

The entries `9:7`, `11:3`, `15`, `17`, `22` and the last blank in these examples are incomplete. The reader knows that they refer to the book of Jeremiah and that, where no chapter is given, they refer to chapter 8 of Jeremiah. In the last example with a blank, the reader knows that it is a note k Jr 8:13. links (to be clicked on) because the incomplete entry takes unspecified information from the previous entry. This rule applies locally to a single text object: note, article, introduction, etc. If the very first entry in the very first entry in a text object is incomplete, the unspecified information is replaced with by the abbreviation of the book currently being processed, or by the number of the current chapter, or verse.

If an incomplete entry is prefixed with \, the unspecified information is taken from the book or chapter or verse currently being processed, regardless of which entry precedes. For example:

```
\Note 4.5 {} The idea is repeated in <Jr 4:5>. But you will also find it mentioned in \<8:3>.
```

Here the reference 8:3 leads to verse 3 of chapter 8 of the actual book. If there the sign was not there, then this reference would lead to 8:3 of Jeremiah.

An incomplete entry is printed as is, as incomplete. The above rules on its are only done internally so that the active link will work properly after mouse click.

There is another different format for incomplete data: the verse number is missing. This is indicated by the presence of the book abbreviation and the absence of a colon and it is not a book from the `\nochapbooks` exception list. Thus, the entry is in the form ⟨*book*⟩ ⟨*chapter*⟩ (separated by a space). In this case, TEX will create internal reference to the first verse of the chapter. Examples:

```
... see <"chapter" Da 7>
... Joseph's story (<Gn 39-41>)
```

In the second example, the range of chapters is given and TEX creates an internal link to the first verse of chapter 39. Compare also with section 7.4.

7.4 Format for the verse range and for the section in the verse

In each entry, it is possible to have ⟨*verse*⟩ instead of ⟨*chapter*⟩:⟨*verse*⟩ to write a range of verses in the format ⟨*from*⟩-⟨*to*⟩. The TEX will create an internal link to the first verse of the range only and turns the hyphen (character `-`, ASCII 45) in the range into a dash. Examples:

```
<Jr 8:3-7>,
<Jr 8:3-9:5>,
<3-7>,
<8:3-7>.
```

For example, the first link in this example prints as Jr 8:3–7 and only offers a link to Jr 8:3.

Sometimes you need to link to a section of a verse, not the whole verse. This is done by appending a letter immediately after the verse number. For example,

```
... see <Da 9:11b>
```

You can append such a letter to both complete and incomplete references. For the purpose of hyperlink, these letters are ignored but printed. Thus, the example given prints ... see Da 9:11b, but the link goes to Da 9:11.

7.5 Concealment of data

When none of the rules are sufficient to create an internal link from a link listed above, you can make everything in the link to print, enclose it in "..." and the subsequent entry to create the internal link, to hide. To do this, just append the character " just after the closing quotation mark underscore _ followed by the data. This entry is not printed. For example,

```
<"First Book of Samuel"_1Sa 1:1>
```

prints only the text First Book of Samuel, which internally refers to the first verse of this book.

7.6 Renumbering the reference

If a link points to a verse that has a different translation in a particular language numbering specified by the \renum macro, then enter the reference according to the default numbering and TEX will recalculate it itself according to the data specified in \renum. It prints the recalculated data and uses it for the internal reference. Cf. 6.4.

7.7 Data reduction

You may want to write the complete data in parentheses for the reference and require it to be automatically reduction to incomplete if they refer to the current book. If they do not refer to the current book, the entry will remain complete. You can do this by by adding a \re before the opening parenthesis of the entry (the eventual reduction rule applies only to this single entry) or by using the \reduceref command. If you you use it in a note (or in a TEXgroup), the reduction rule is applied for all subsequent complete entries of that note (of that TEXgroup). When used in a rate declaration, the eventual reduction rule is applied to all complete data in the document. Example:

```
\re<"verse" Da 7:3>
```

is printed as verse Da 7:3 if this reference is given outside the book Daniel. However, when this reference is given in the book of Daniel, it is printed only verse 7:3, which internally refers to Da 7:3.

The link reduction rule set by \reduceref can be turned off with the \noreduceref command. From there to the end of the note (TEX groups) links behave as if \reduceref were not turned on.

7.8 The marker for a book can be printed differently

If the bookmark is declared differently in different translations using \vdef, then use only the ⟨*book*⟩ entry in the references according to the first variant. However, if any alternative translation is set using \tmark variant, the link will eventually print according to the \vdef entry of that translation variant. Internally, however, links are linked according to the base variant. This property This feature is useful if you are declaring \vdef for tags 1Pa and 2Pa (Paralipomenon) alternative text for 1Let and 2Let (Chronicles) in variant B21 (Bible for the 21st century). When then you type the reference <"see" 1Pa 2:3>, you get see 1Pa 2:3 in the usual translation variants, but it will print see 1Let 2:3 if you use translation variant B21.

7.9 Failed links, i.e. links to a non-existent place

If a reference is made to a non-existent verse or a non-existent note, then there are two possibilities. If it is a reference to a book that is deliberately not printed (because, for example, we are working with a test copy of only selected books of the Bible, see also \printedbooks in section 4.1), then the link is indeed colored, as if it were active, but it is not, and the TEXem processing log no warning appears. However, if the link points to a non-existent verse or note of the printed Bible, then the link is active, the click goes to the last page of the PDF file, and a warning appears in the log that the link is incorrect. However, the first time TEX is processed, all links and there is a large number of warnings about incorrect links in the log. Only when TEXem is next processed are links pointing to existing site are correctly interlinked.

7.10 Link Tracing

By default, detailed link tracing is enabled in the log file. Turn this off with \notracinglinks and back on with \tracinglinks. In addition, use \tracingouterlinks to disable the suppression of link warnings on non-existent books, allowing the log to find any non-existent links from due to a typo in the bookmark.

7.11 Chapter and book links

To link to a book with a link to the beginning of that book, write `\cref[⟨book⟩]`, for example, `\cref[Gn]` will print Gn with a link to the beginning of the book of Genesis. Similarly, `\cref[Gn 2]` will print Gn 2 with a click through to the beginning of the Day of Rest chapter. If you want to print something else, then for `[...]` must be closely followed by `{⟨text⟩}`, where *⟨text⟩* is the text that to be printed and become the active link. So, for example `\cref[Gn 2]{Rest Day}` will print Rest Day with a link to the corresponding chapter.

7.12 Links to pages

You can place an unprintable page link target in your text using `\pglabel[⟨glue⟩]` and then you can link to the page using `\pgref[⟨glue⟩]`. In place of `\pgref[⟨glyph⟩]` will print the number of the clickable page. Similar to `\cref`, you can use `\pgref[⟨picker⟩]{⟨text⟩}` to print a *⟨text⟩* other than the page number that clicks through to the place where where the `\pglabel[⟨glue⟩]` is located.

8 Maps, images and their legends

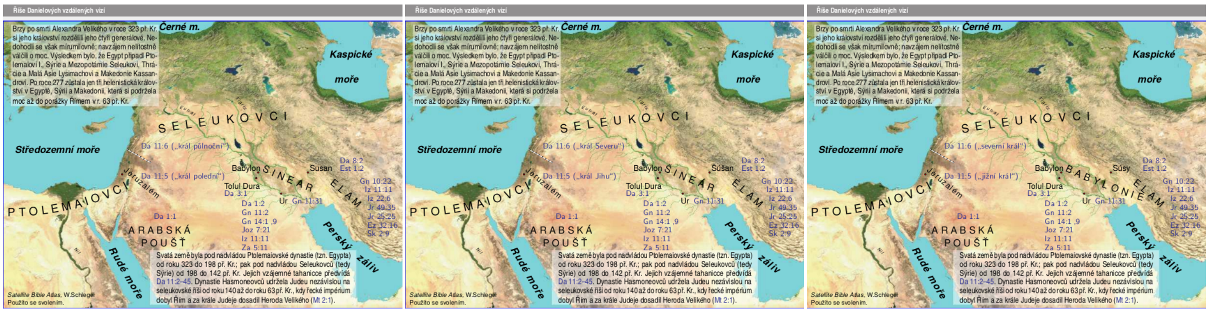
8.1 Translation variants

Just as you can change the wording of a search phrase in a note to match the actual base text, you can prepare the description of the map (and similar graphical objects showing expressions that vary in different translations) to always match the current version of the Bible.

Assuming that we have a blank map that we want to describe with its legend, any text is placed over an image as follows:

```
\puttext 5mm 62mm{{\bi Mediterranean}}}
```

The first dimension (5mm) determines the horizontal offset on the x-axis, the second the vertical offset on the y-axis, with the coordinate `0mm 0mm` being the bottom left corner of the image.



Říše Danielových vzdálených vizí

Brzy po smrti Alexandra Velikého v roce 323 př. Kr. si jeho království rozdělili jeho čtyři generálové. Nedohodli se však mírumilovně; navzájem nelítostně válčili o moc. Výsledkem bylo, že Egypt připadl Ptolemaiovi I., Sýrie a Mezopotámie Seleukovi, Thrácie a Malá Asie Lysimachovi a Makedonie Kassandrovi. Po roce 277 zůstala jen tři helenistická království v Egyptě, Sýrii a Makedonii, která si podržela moc až do porážky Římem v r. 63 př. Kr.

Svatá země byla pod nadvládou Ptolemaiovské dynastie (tzn. Egypta) od roku 323 do 198 př. Kr.; pak pod nadvládou Seleukovců (tedy Sýrie) od 198 do 142 př. Kr. Jejich vzájemné tahanicce předvídá Da 11:2–45. Dynastie Hasmoneovců udržela Judeu nezávislou na seleukovské říši od roku 140 až do roku 63 př. Kr., kdy řecké impérium dobyl Řím a za krále Judeje dosadil Heroda Velikého (Mt 2:1).

Satellite Bible Atlas, W.Schlegel
Použito se svolením.

Map from the Introduction to Daniel: BKR CEP B21

For example, you can notice how in the sample above not only the phrases „midnight king" and „noon king" (BKR) change to „north king" and „south king" (CEP) and „south king" and „King of the North" (B21), which is a common active link as in the notes, but also the map description itself , e.g. The name of the city of Susan (just above the Persian Gulf) from the King's name to Shushan in CEP and to Susa in B21. metamorphoses to Sinear and to Babylonia. The definitions of `\vdef` in the `Cze-vars.tex` file take care of this.

The image must be loaded before the description can start, and everything in it must be inside the definition of `\insertBot`. The above sample has been written as

```
\insertBot {Daniel's Remote Viewing Realm}[map](){
 \inspic{fertile-crescent-crop.pdf}% blind map
  \Heros \cond \setfontsize{at 9pt}\rm %font
  \vskip-1mm
\putstext 2mm 108mm {\top{\hsize6.5cm %box width\top
                    \baselineskip10pt %line spacing inside the box
                    \noindent %space saving, no need to indent
\leftskip=3pt \rightskip=3pt %how much the semi-transparent shadow will overlap the text
  Soon after the death of Alexander the Great ... until the defeat by Rome in 63 B.C.}%end\vtop
  }%end \putstext
  \LMfonts\sans \setfontsize{at9pt}\rm
  \puttext 145mm 29mm {⟨”Acts 2:9”_Acts 2:9⟩}
  \puttext 145mm 32.5mm {⟨”Ez 32:16”_Ez 32:16⟩}
.
.
.
  \puttext 2mm 5mm{{{{Heros \setfontsize{at 7pt}\it Satellite Bible Atlas,\rm W.Schlegel}}
  \puttext 2mm 2mm{\Heros \setfontsize{at 7pt}\rm Used with permission.}
  }%end \insertBot
```

What to watch out for: There must not be a blank line inside `\insertBot`.

Command `\putBot` ⟨*verse number*⟩ `{`⟨*title*⟩`}` `[`⟨*slash*⟩`]` `(`⟨*commands*⟩`)` `{`⟨*code*⟩`}` behaves similarly to `\insertBot`, but inserts the image or diagram not on the current page, but to the page where the verse specified by its ⟨*verse-string*⟩ is located. Such commands can be used in an annotation file, for example at the end of the file.

8.2 Macro `\town` for the town symbol on the map

The towns of Jerusalem, Babylon, Tolul Dura, Susan and Ur are visible on the map as tiny circles with a red center and black perimeter. The properties of this ring can be set with the macro `\townparams`, whose default values are as follows:

```
\def\townparams{
   \hhkern=.8pt % radius of the sphere
   \lwidth=.5pt % contour line thickness
   \fcolor=\Red % ring color
   \lcolor=\Black % contour line colour
}
```

The macro `\town` itself places this marker with coordinates, similar to the macro `\puttext`, but without additional text, e.g.

```
  \town 101.5mm 53mm %town Babylon
```

8.3 Tilted text

In the example we have the name of the city of Jerusalem printed at an oblique angle so that it does not clash with „king of the South“ and can be seen clearly. This can be achieved by adding additional parameters to the `\puttext` definition:

```
\puttext 48mm 55mm {\c[-40/\kern1pt\pdfrotate{0}]{Jerusalem}}
```

The number `-40` was used to tilt the inscription. For further details see 8.4.

8.4 Inscriptions along the curve

Some of the inscriptions on the map require „stretch“ according to the terrain, especially the names of large areas, in this case the Ptolemaic and Seleucids, or the Persian Gulf, or perhaps the more minor names of rivers (Euphrates, Tigris, Nile). We will change the additional parameters to `\puttext` as we have already seen when placing Jerusalem on the map in the v 8.3.

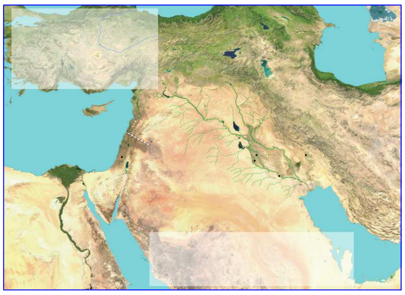
The `\kern` command determines the spacing between letters; the digits in the `\pdfrotate` definition determine the strength of the curvature. A negative value bends the lettering concavely (like a rainbow), a positive value convexly (like a bowl). The dynasties of Alexander's descendants are written on our map as follows:

```
\puttext 62mm 70mm {\c[10/\kern7pt\pdfrotate{-1}]{SELEUKOVCI}}
\puttext 2mm 37mm {\c[0/\kern4pt\pdfrotate{2.5}]{PTOLEMAI}}
```

If we needed a sign that waved in the shape of the letter S (it would be concave and convex at the same time), we would have to assemble it from two or more `\puttext` statements, glued together to look like one continuous text.

8.5 Partially transparent background of continuous text

In the 8.1 sample, the double continuous text in the map was originally placed in lighter rectangles prepared by the graphical editor (left image). The text then had to be inserted into these hit the text; if it didn't work, one had to redo the image and try to hit it again.



It worked, but it was tediously laborious and time-consuming. OpBible offers a more imaginative solution. no pre-made areas. Before using `\putstext` for the first time, the level of transparency of the white shadow can be set by adjusting the default value of `\def\shadowparameter{.1}` (right image). `\def\shadowparameter{1}` means a solid opaque white background; a smaller number means more transparency. However, this value is then stored in the page-resources of the output PDF and is used the same on all subsequent pages, so it cannot be changed and have it in different places in the same document different on different pages. If perhaps there should be an unexpectedly excessive demand for the ability to change the transparency level on the fly, this may be an incentive for implementation in a possible future version of OpBible. For the moment, we did not find it necessary to complicate the macros by creating more and more page-resources, so the user should be satisfied with the option to set the transparency of the shadow under the text on maps uniformly for the entire Bible.

9 Timeline inclusion tools

9.1 Image or text over two pages

If we want to insert an image or text across two pages in an open double page, we can use

- `\insertSpanImage`: inserts a prepared PDF image, can be used in the introduction of a book,
- `\insertSpanText`: insert text (for example a timeline), can be used in the book introduction,
- `\putSpanImage`: insert a prepared PDF image, anchored relative to the number chapter and verse, can be used in the notes file,
- `\putSpanText`: like `\putSpanImage`, but inserts text instead of an image.

The `\insertSpanImage` and `\insertSpanText` commands place the image or text at the bottom of two pages according to the following rule. Suppose that the command itself is executed when TeX creates the current page with number c. Then

- if c is even and the image or text fits the current page in portrait, it will be placed on pages c and $c + 1$,
- if c is even and the image or text does not fit on the current page, it will be placed on pages $c + 2$ and $c + 3$,
- if c is odd, the image or text will be placed on pages $c + 1$, $c + 2$.

This ensures that the image or text will always be visible on the double page of an open book.

The `\putSpanImage` or `\putSpanText` commands work according to the same rule, only the number c in this case corresponds to the page number on which the the beginning of the verse specified in the command parameter.

These commands have the following parameters:

`\insertSpanImage {Title} [⟨column⟩] (⟨parameters⟩) {⟨filename⟩}` Here, ⟨*Title*⟩ is used in the header of the image and ⟨*snapshot*⟩ can be set, to refer to the image (and thus create an active link) using `\ref`. If you don't want to use the funnel, leave the parameter empty: `[]`. Furthermore, ⟨*parameters*⟩ can specify how the image is placed, typically this parameter is empty: `()`. Finally, ⟨*filename*⟩ is the name of the image file including the extension. Typically this is a PDF file, i.e. it has the extension `.pdf`.

`\insertSpanText` `{Title}` `[`⟨*column*⟩`]` `(`⟨*parameters*⟩`)` `{`⟨*text*⟩`}` The parameters are the same as `\insertSpanImage`, only the last parameter is different. It contains the text to be printed across two pages. Typically there might be a set of `\timeline`, `\timelinewidth`, `\arrowtext`, `\tlput` commands can be used, `\tline`, `\tlines` to create a timeline, see section 9.2.

`\putSpanImage` ⟨*chapter*⟩`:`⟨*string*⟩ `{Title}` `[`⟨*column*⟩`]` `(`⟨*parameters*⟩`)` `{`⟨*filename*⟩`}` In addition, there is a ⟨*chapter*⟩:⟨*verse*⟩ specifying the verse whose beginning is on page *c* and the image is positioned according to the rule above. The other parameters are the same as for `\insertSpanImage`.

`\putSpanText` ⟨*chapter*⟩`:`⟨*verse*⟩ `{Title}` `[`⟨*image*⟩`]` `(`⟨*parameters*⟩`)` `{`⟨*text*⟩`}` behaves the same as `\insertSpanText`, but with the addition of the ⟨*chapter*⟩:⟨*string*⟩ parameter has the same meaning as in the `\putSpanImage` command.

9.2 Commands to create a timeline

These commands can be used in the ⟨*text*⟩ parameter of the `\insertSpanText` or `\putSpanText` commands.

First, you need to specify the number of years (or other units) for the full width of the timeline. All other data will be entered in these units. In the following text we will refer to these units as years. Set the timeline parameters using `\timeline` and `\timelinewidth`:

```
\timeline ⟨number of years⟩
\timelinewidth ⟨width of the axis⟩
```

For example, after

```
\timeline 500
\timelinewidth 25cm
```

the axis will be 25cm wide and 500 years will be included in that width, so one year will represent a width of 0.5 millimeters. However, it is more usual to specify the width of the axis as some fraction of the total width of the page (or double-sided image, if using axis in `\insertSpanText` or `\putSpanText`). For example: `\timelinewidth=.95\hsize`.

The timeline is built line by line. Commands for text or line segments that are on the same line are written below each other in any order. To move to the next line is the `\vskip` ⟨*dimension*⟩ command, while the dimension can be specified in in multiples of the line height using the unit `\l`. i.e. `\vskip 1.5\l` means a shift of one and a half lines down.

To insert text, use `\tlput` ⟨*flag*⟩ ⟨*position*⟩ ⟨*space*⟩ `(`⟨*setting*⟩`)` `{`⟨*text*⟩`}`. The ⟨*flag*⟩ parameter can be `a` if we want the text to be above the current rate position, or `b` if we want the text to be below the current rate position. When the ⟨*flag*⟩ is `a`, multiline text extends upwards and when `b` extends downwards. The ⟨*position*⟩ parameter is the location on the timeline (in years) to which the text should be attached. From this point, the text will flow to the right if ⟨*space*⟩ is specified by the `\rlap` instruction, it will flow to the left if the ⟨*string*⟩ is specified by the `\llap`. Finally, if ⟨*query*⟩ is empty, the text will have centered lines and ⟨*position*⟩ then corresponds to this line center. The ⟨*setting*⟩ parameter can be be empty, i.e. `()`, or it may contain font settings, font color, etc. for following ⟨*text*⟩. A ⟨*text*⟩ contains the text to be printed. In the case of multi-line text, separate the lines with `\cr`. Example:

```
\tlput b 25 (\it) {Abraham\cr is 100 years old}
```

The text has two lines, their common center is below point 25 of the timeline.

`\tline` ⟨*from*⟩`..`⟨*to*⟩ creates a horizontal line starting at ⟨*from*⟩ and ending at ⟨*to*⟩. The data are in years.

`\tlines` ⟨*w1*⟩`|`⟨*w2*⟩`|`⟨*w3*⟩`|` (and possibly others) inserts short vertical lines on the axis. The number of ⟨*w*⟩`|` parameters can be arbitrary, each representing one vertical line, and the numbers between them indicate the distance between adjacent lines in years. For example

```
\tline 0..100
\tlines 20|20|20|20|20|
```

creates a 100th horizontal line and vertical lines on top of it to denote 20, 40, 60 and 80 years.

`\arrowtext` ⟨*from*⟩`..`⟨*to*⟩ `(`⟨*setting*⟩`)` `{`⟨*text*⟩`}` prints a horizontal line from ⟨*from*⟩ to ⟨*to*⟩ (data in years) and the middle of the line is broken and ⟨*text*⟩ is written at that point. At the edges of the line are arrows pointing outwards from the line.

10 Page formatting variants

By default, a single-column typeface is set for the main text of the Bible and for book introductions and annotations. Two-column type is set for verse notes.

By default, chapter numbers are capitalized and in the outer margin. In the outer margin are also enlarged quotation marks attached to quotations. The command `\normalchapnumbers` command changes this setting: chapter numbers are then inserted in the left upper corner of the first paragraph and the enlarged quotation marks are removed. The outer margin is then shrunk because there is no more print material in it.

By default, each `\Note` occupies a new paragraph in a two-column typeface. By using `\mergednotes` it is possible to ensure that all notes referenced to a single verse are always combined into a common paragraph. But it comes at a cost of not ensuring that the occurrence of phrases in the main text is on the same page as the phrases in the note. The entire paragraph of the linked note will only link to the beginning of the verse, i.e. the beginning of the verse and the beginning of the note to the verse are on the same page, the phrases themselves are not searched.

Other page formatting options are still in the planning stages and are not in this version of OpBible implemented.

11 Error search options

It may happen that you make a typo in the `notes-*`, `intro-*` files, `articles-*`. If you had included the file directly in the whole Bible processing, TeX would report an error at a completely different point in time than when it read the file and tracing back such an error is very difficult, and typically the error occurs at a different stage of processing and thus is often reported at a very incomprehensible and misleading.

However, it is possible to process newly written files directly first, but without but not linked to the main text, and in this mode errors are reported more directly. To search for errors directly, use `\checksyntax` ⟨*file list*⟩ `{}` Here the file list are the names of the files checked without the `.tex` extension, so you can have, for example, at the end of the main file:

```
\checksyntax intro-Dan articles-Dan notes-Dan {}
```

performs a direct check of the enumerated files. The output is the text of these files without any formatting emphasis. Syntax errors in the files are but will show up in a straightforward manner.

Note: using `\checksyntax` disables the `\processbooks` command, which in does nothing, so the base text of the Bible is not loaded at all.

11.1 Generating default files `notes`, `fmt`, `intro`

If the files `notes`, `fmt`, `intro` do not exist, OpBible does not get angry and behaves as if they were empty. But you might want to have these files for all the books of the Bible already prepared in the appropriate directories and with a short introduction. Templates for such files can be write them in a separate file (e.g. `templates.tex`) and then file using the `optex templates` command. This will generate the default files according to the specified templates for all the books of the Bible. If a file with the specified name already exists, it is retained (i.e., it is not overwritten by the default file) and a warning will appear on the term and in the `log` file.

First you need a `books.tex` file with the names and abbreviations of all the books of the Bible, as mentioned in the 4.3 section. This can be generated, for example from Sword using the `mod2tex` script, see section 4.4.

In the `templates.tex` file, the macro must first be loaded using `\input. opbible.opm`, then the `books.tex` file, and then you can specify file templates with the `\filegen` command as follows:

```
\filegen {⟨file-name⟩}
⟨file-content⟩
\endfile
```

while ⟨*filename*⟩ must contain and ⟨*file-content*⟩ can contain a double character `@@`, which is automatically replaced by the book abbreviation ⟨*bmark*⟩ specified in `bible.tex` and generated as many default files as there are book titles declared in `books.tex`. For example, `templates.tex` might look like this:

```
\input opbible.opm
\input books.tex

\filegen {intro-@@.tex}
% Book Introduction @@
\endfile

\filegen {notes-@@.tex}
% Book notes @@

%\ww {} {} {} {} {} {} {}
% BKR PSP CSP CEP B21 SNC
%\Note 1:1 {} text terminated with a blank line
\endfile
```

The `optex templates` command will generate 66 files in this case `intro-1Jn.tex`, `intro-1kor.tex`, etc. with the specified single line and 66 `notes` files with specified five-line content.

You can also use the `@@@` triple character in the ⟨*file content*⟩, which is interchanged from the full title of the book. So after specifying `% Book Introduction @@@` you will get files with the text `% Introduction` →the book of Genesis, then `% Introduction to the Book of Exodus` etc.

In the example, the file names are chosen so that they are generated in the current directory. If you have prepared subdirectories in this directory corresponding to names, it is possible to generate files directly into them, for example, write `\filegen {intros/intro-@@.tex}`.

To generate a different list of files than the ⟨*bmark*⟩ list listed in the `books.tex` file, define `books.tex` after reading macro `\genbooks` and your chosen shortcuts. For example, after

```
\def\genbooks {Gn Ex Lv Nu Dt Joz}
```

the `\filegen` command will only generate templates for the five books with the specified abbreviations. You can also define `\genbooks` before each `\filegen`, then you'll have different book abbreviations for different subfile types.

on a second chance after death. no one would have done it. ...that is, at the point of the most extreme reality. nine times out of ten you'll become original without even noticing it. and then, with the money in your suitcase, you arrive at the agreed-upon spot in the middle of the bridge. the designated end of the bridge as you were instructed. A van emerges from the darkness, stops in the middle of the bridge, your daughter gets out, you're relieved she's alive and hopefully okay, she picks up the suitcase and throws it into the car, then she turns towards you, slaps her left hand on her right bicep while holding up her right fist with the back of her hand towards you while screaming: „Thank you, dumbass!"

12 Summary of basic commands and definitions

I would put everything the user might need to set up in

12.1 Typically in the main file

- `\tmark`: `\def\tmark{`⟨*t-short*⟩`}`: declaration of the t-short of a translation, e.g. BKR, PSP.
- `\txsfile`: `\def\txsfile {`⟨*filename mask*⟩`}`: declaration of the location of `.txs`. You can use `\amark` or `\bmark` in the filename mask.
- `\notesfile`: `\def\notesfile {`⟨*filename mask*⟩`}`: location declaration annotation files.
- `\introfile`: `\def\introfile {`⟨*filename mask*⟩`}`: location declaration of files with book introductions.
- `\articlefile`: `\def\articlefile {`⟨*filename mask*⟩`}`: declaration of the location of the of article files.
- `\fmtfile`: `\def\fmtfile {`⟨*filename mask*⟩`}`: declaration of the location of file formatting data.
- `\input {`⟨*filename*⟩`}`: read the next file in the given location.
- `\printedbooks`: `\def\printedbooks {`⟨*list of a-bookmarks*⟩`}`: list of books to be processed by `\processbooks`.
- `\processbooks`: instruction to process the Bible.

The above commands need to be defined in the main file, read `books.tex`, `vars.tex` and finally start processing with the command `\processboks`.

12.2 Typically in the file `books.tex`

- `\BookTitle` ⟨*a-tag*⟩ ⟨*b-tag*⟩ ⟨*title*⟩: declares for each book a-tag, b-tag and title. As each individual book is processed, the `\def\amark{`⟨*a-mark*⟩`}` and `\def\bmark{b-mark}`. The a-brand is used in references in the text of the Bible, and the b-mark may be used in file names.
- `\BookException` ⟨*a-mark*⟩ `{`⟨*code*⟩`}`: before processing a book, the ⟨*code*⟩ is executed.
- `\BookPre` ⟨*a-tag*⟩ `{`⟨*code*⟩`}`: after printing the introduction before the main text the declared book, ⟨*code*⟩ is executed.
- `\BookPost` ⟨*a-tag*⟩ `{`⟨*code*⟩`}`: after printing the main text of the declared book, ⟨*code*⟩ is executed.
- `\nochapbooks`: `\def\nochapbooks{`⟨*list*⟩`}`: list the a-brands of the books that are not divided into chapters.

12.3 Typically in the file `vars.tex`

This file declares variant phrases for different versions of the Bible translation.

- `\variants` ⟨*number*⟩ ⟨*list of t-short translation variants*⟩: declares the number of translation variants and a list of t-strings. The number must match the number of t-shortcuts as well as the number of parameters of the `\vdef` and `\ww` commands.
- `\vdef` ⟨*phrase variants*⟩; declares the variants of the phrase, the first of which is reference. The phrase variants are given by the literal `{`⟨*this*⟩`}`.

12.4 Typically in the notes file

- `\Note` ⟨*verse number*⟩ `{`⟨*commented phrase*⟩`}` ⟨*note text terminated by a blank line*⟩: declares a note.
- `\ww` ⟨*variants of phrase*⟩: must precede `\Note`. V each translation variant, the following note will be bound to the phrase of a particular phrase according to the currently defined t-tag.
- `\x/`⟨*phrase*⟩`/`: prints the phrase according to the current t-tag and according to `\vdef` declaration. The ⟨*phrase*⟩ parameter is the reference phrase and is printed this parameter if no t-tag is defined or if the first t-character from the list of t-characters defined by `\variants`.
- `\putBot` ⟨*verse number*⟩ `{`⟨*title*⟩`}` `[`⟨*slash*⟩`]` `(`⟨*commands*⟩`)` `{`⟨*code*⟩`}`: sets up an image/diagram at the bottom of the page containing the ⟨*verse number*⟩.

12.5 Typically in a book introduction file

... TODO, add

12.6 Typically in an article file

... TODO, complete

12.7 Typically in a file with formatting data

... TODO, complete

12.8 When creating maps

... TODO, complete

12.9 When creating timelines

... TODO, complete

13 Index