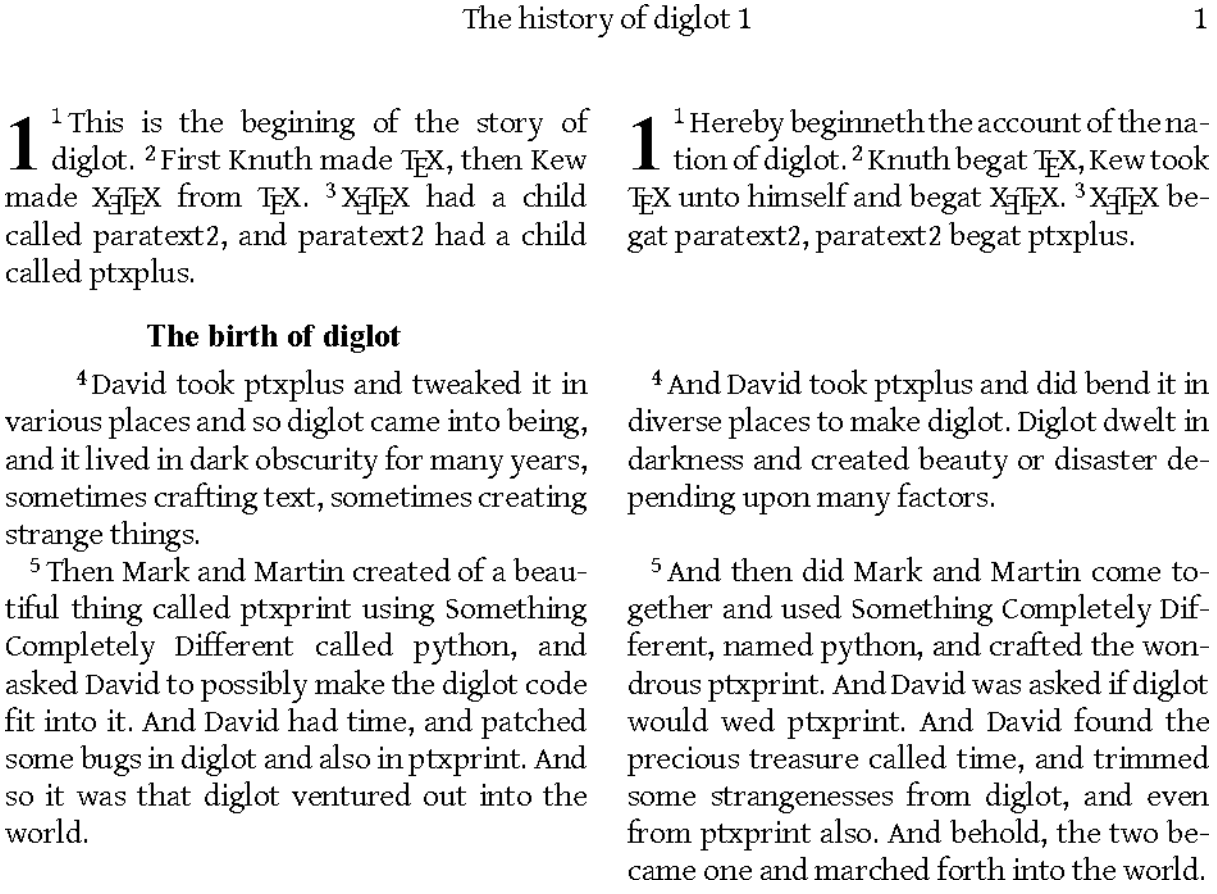
Diglot options and settings (XeTeX macros)

General overview

The diglot options allow for typesetting two (or more) versions in parallel, aligned by verse or paragraph. If that makes no sense, have a look at the samaple below, where the right hand column's paragraphs are one line shorter, but still the paragraphs in the two columns line up. To do this the two peices of text require an extra pre-processing step and then a number of extra configuration controls to make things really beautiful. The preprocessing step picks 'pairs' (or groups) of things that ought to be aligned, and puts these into what we can call a *chunk*. These pairs/tripples/quadruples might be verses, paragraphs at the same place, and so on. At the moment this step is done by a python program, one of 2 perl programs, or by hand for small sections of text. The python program integrates better with ptxprint, one of the perl programs is considerably older and both have more powerful/complex options.



The history of diglot 1 1

1 [1] This is the begining of the story of diglot. [2] First Knuth made TEX, then Kew made XETEX from TEX. [3] XETEX had a child called paratext2, and paratext2 had a child called ptxplus.

The birth of diglot

[4] David took ptxplus and tweaked it in various places and so diglot came into being, and it lived in dark obscurity for many years, sometimes crafting text, sometimes creating strange things.

[5] Then Mark and Martin created of a beautiful thing called ptxprint using Something Completely Different called python, and asked David to possibly make the diglot code fit into it. And David had time, and patched some bugs in diglot and also in ptxprint. And so it was that diglot ventured out into the world.

1 [1] Hereby beginneth the account of the nation of diglot. [2] Knuth begat TEX, Kew took TEX unto himself and begat XETEX. [3] XETEX begat paratext2, paratext2 begat ptxplus.

[4] And David took ptxplus and did bend it in diverse places to make diglot. Diglot dwelt in darkness and created beauty or disaster depending upon many factors.

[5] And then did Mark and Martin come together and used Something Completely Different, named python, and crafted the wondrous ptxprint. And David was asked if diglot would wed ptxprint. And David found the precious treasure called time, and trimmed some strangenesses from diglot, and even from ptxprint also. And behold, the two became one and marched forth into the world.

The structure of a diglot-friendly usfm file.

The preprocessing step "shuffles" the two files together, interspersing them with instructions to switch sides. (For as-yet-uncertain reasons, these instructions seem to work much better if preceeded by \p). There are four instructions: `\lefttext \righttext \nolefttext \norighttext`. The `\nolefttext` specifies that the text that follows ought to begin below any text remaining in the the left-hand column, and start a chunk that will not have any corresponding left-hand text. A sample of the file that produced the above image is given below.

```
\lefttext
\s The birth of diglot
\p
\norighttext
\p
\v 4 David took ptxplus and tweaked it in various places and so diglot came
   into being, and it lived in dark obscurity for many years, sometimes
   crafting text, sometimes creating strange things.
\p
```

```
\righttext
\p
\v 4 And David took ptxplus and did bend it in diverse places to make diglot.
   Diglot dwelt in darkness and created beauty or disaster depending upon many
   factors.
```

The structure of a polyglot-friendly usfm file

The polyglot version of the above would look like this:

```
\polyglotcolumn L
\s The birth of diglot
\p
\polyglotendcols
\polyglotcolumn L
\p
\v 4 David took ptxplus and tweaked it in various places and so diglot came
   into being, and it lived in dark obscurity for many years, sometimes
   crafting text, sometimes creating strange things.
\p
\polyglotcolumn R
\p
\v 4 And David took ptxplus and did bend it in diverse places to make diglot.
   Diglot dwelt in darkness and created beauty or disaster depending upon many
   factors.
\polyglotendcols
```

Note that now there is simply a column designator which follows `\polyglotcolumn` and the chunk-ending `\polyglotendcols`. The equivalent of `\norighttext` is to simply not specify anything for that column.

Additional columns should be defined in the controlling `.tex` file:

```
\newPolyglotCol A
```

Note that this must occur before any USFM files or stylesheets are loaded.

Diglot-specific configuration items to go in custom stylesheets:

Any `\Marker` can (theortically) be generic (e.g. 'p') or apply to a single column only ('pL' or 'pR' for left and right versions of 'p').

Background: when the code looks up a parameter for style 'p' it first looks up the parameter for 'pL' if in the left column or 'pR' if its in the right column, and then if it can't find one it looks up the unadorned 'p' properties.

Example: A user who wants to confuse us has:

```
\Marker p
\Fontsize 11
\FontName Gentium

\Marker pR
\Fontsize 12

\Marker pL
\FontName DoulosSIL
```

The left column will use: DoulosSIL (column specific) at 11pt (default), and the right column will use: Gentium (default) at 12pt (column specific).

Configuration items to go in foo-setup.tex (or main .tex file)

Extra columns

`\newPolyglotCol A` This specifies that just `L` and `R` are a bit boring, and you wish to use another column (`A`) as well. The perl program to merge files uses L,R,A,B,C.... as column identifiers. So far there is no python code to cope with polyglots.

Footnote control

- `\diglotSepNotestrue` If the footnotes from the 2 languages should be split (true) or merged together (false) (default: `\diglotSepNotestrue`). Merging footnotes is almost certainly not a wise choice if both texts have footnotes, but if only one side has notes then it probably makes a lot of sense. The exact order of the footnotes is probably complicated and may even be unpredictable.
- `\diglotBalNotesfalse` If a left column footnote steals space from the right column also, and vise-versa (default: `\diglotBalNotesfalse`). If this is a good idea or not probably depends on a lot of factors.
- `\DistinctNoteNumbering{f}` (default)
- `\ParallelNoteNumbering{f}` If the command `\ParallelNoteNumbering{f}` is given, then `\f` footnotes will have two parallel counts, so that left and right texts will be numbered (or picked from the callers list) separately. The default keeps a single count, so each footnote/cross-reference is numbered distinctly. This setting is ignored if merged footnotes are active, to avoid confusion.

The writers of hooks or control files that reset note numbering at chapters, sections etc, might want to use one of these options as appropriate:

- `\resetautonum{f}` Reset numbering for note `f`, in the present column
- `\resetSpecAutonum{fL}` Reset numbering for note `f`, column `L`
- `\resetAllAutonum{f}` Reset numbering for note `f` in all columns

###True/false options

- `\diglottrue` If there is diglot material this **must be set true** (i.e. `\diglottrue`), **before** the style sheet is loaded (default: `\diglotfalse`).
- `\OmitChapterNumberLtrue` , `\OmitChapterNumberLfalse` and `\OmitChapterNumberLdefault`
- `\OmitChapterNumberRtrue` , `\OmitChapterNumberRfalse` and `\OmitChapterNumberRdefault`
- `\OmitVerseNumberOneLtrue` , `\OmitVerseNumberOneLfalse` and `\OmitVerseNumberOneLdefault`
- `\OmitVerseNumberOneRtrue` , `\OmitVerseNumberOneRfalse` and `\OmitVerseNumberOneRdefault` Column-specific control over chapter and verse numbers. The 'third state' of this boolean (which is the default) permits the 'global' boolean (without the `L` or 'R') to have control.
- `\VisTracetrue`
- `\VisTraceExtratrue` Debugging options for really sticky problems; see end of this document.

Header macros

- `\rangerefL`, `\rangerefR`, `\rangerefA` (and their companions `\firstrefX` and `\lastrefX`) have now been defined, which display the book/chapter/verse ranges on a given column only. The appropriate font will be selected from the stylesheets.
- Also available: `\usdateX`, `\ukdateX`, `\isodateX`, `\hrsminsX`, `\timestampX` which include font selection.
- `\headfootX{...}` which selects the relevant font
- `\bookX `` and\bookaltX“‘ exist but as these are normally used in the rangeref (etc) expressions, they include no font switching, so they would need wrapping in `\headfootX{ }`

Page layout options

- `\def\ColumnGutterFactor{15}` Gutter between the 2 cols, (measured in `\FontSizeUnit`s), just like in two column mode.
- `\ColumnGutterRuletrue` There should be a vertical rule(line) between columns of text.

- `\FigGutterRuletrue` There should be a vertical rule between column-figures if there is one between the columns of text
- `\NoteGutterRuletrue` There should be a vertical rule between footnotes if there is one between the columns of text
- `\JoinGutterRuletrue` There should be no gap in the vertical rule between the one for the text body and the one for the notes. If false, there is no vertical rule in the gap controlled by `\AboveNoteSpace`.
- `\def\DiglotLFraction{0.55}`
  Fraction of the space that is used by column L. Similarly `\DiglotRFraction`, `\DiglotAFraction` etc. Unless multiple page layout (experimental) is used, the sum of all the fractions should be 1.0. If mulpiple pages layout is used, the sum of all fractions on their respective pages should be 1.0. No automatic verification of this is currently done, you'll just get ugly results.
- `\def\DiglotLeftFraction{0.5}` `\def\DiglotRightFraction{0.5}` Deprecated synonym for `\def\DiglotLFraction` and `..glotRFraction...`

Hopefully, the above fractional controls (and the font-sizes from they style sheet) should enable even the most widely different translation styles and languages to balance in an overall pleasing way, without huge gaps under every chunk on one column.

Column-specific configuration parameters

The following may be defined with a column-specific suffix (`L`, `R`, `A`, ...). `AdornVerseNumber`, `VerticalSpaceFactor`, `LineSpacingFactor`, `regular`, `bold`, `italic`, `bolditalic`, `SpaceStretchFactor`, `SpaceShrinkFactor`, `MakeChapterLabel`

These dimensions can similarly have column-specific values: `FontSizeUnit`, `IndentUnit`

If the column-specific value is not defined, then the 'global' (not-specific) value will apply.

###Deprecated true/false options

- `\useLeftMarkstrue`
- `\useRightMarkstrue` When you've got just one page and two texts, and one text goes until verse 15 and the other manages to fit verse 16 and 17 on as well, what do you put in the header 15 or 17? Intuitively, we'd probably expect the first 'mark' [i.e. chapter:verse] on the page to be from the left-hand column, and the last from the right, but this possibly becomes confused with `\nolefttext`, and even with short sections where the first verse set is actually on the right. These two options control whether marks from the left-hand column and right-hand column are used in the standard heading macros (`\rangeref`, `\firstref` and `\lastref`). Setting both to true might *normally* work, but sometimes it won't; it is best to pick one side to populate the headers, or use the side-specific variants below. The use of side-specific variants is critical for font-switching to work correctly.
- `\LeftMarkstrue` This used to be the only control that affected what went into the header (defaulting to true). It is now a short-hand for \useLeftMarkstrue\useRightMarksfalse.

Hooks

Like the markers, hooks can be made to apply to left or right columns. e.g.:

```
\sethook{start}{q1L}{\hangversenumber}
```

will apply the `\hangversenumber` only for the left column.

Setting hyphenation languages

- `\def\languageL{english}` Left column is in english. (Requires that the language's hypenation patterns have been loaded).
- `\def\languageR{nohyphen}` Right column should not be hypenated

Mixing diglot and monoglot text

The font-switching code requires that `\diglottrue` must be specified before any style sheets are loaded. However, it is now possible to have language-switching without the diglot layout:

```
\zglot|L\*
\monoglotcolumn L
```

These (equivalent) commands performs all the font switching etc. that might be expected for a diglot text, but *without* any of the column switching code being activated. The shorter milestone-like format is preferred as it is USFM-compliant, but the longer form works. The expectation is that this will be used in front-matter and back-matter books, etc. where a single or dual-column layout is best but font (and stylesheet) switching is still desired.

Although this switches on `\diglottrue`, so that font switching functions correctly, it should be used in a USFM file that was started in single or dual column mode (`\singlecolumn \diglotfalse`). It will probably cause unpredictable results if used in a normal diglot (or polyglot) file.

Hybrid files, which contain a mixture of diglot and monoglot (including serial-monoglot text as with `\monoglotcolumn`) material are possible, but they are basically untested.

`\diglotfalse\singlecolumn` and `\diglotfalse\doublecolumns` will switch to monoglot text, and `\diglotcolumns` will switch to diglot text.

Other settings

- `\Alternative` The PDF bookmarks are produced (by default) with a `/` separating the chapter name. The slash is actually produced by `\Alternative`, in case slash not the correct symbol to use.
- `\def\KeepMyBrokenAdjList{}` If this is defined, then old-stlye (broken) paragraph numbering for adjust lists and triggers.

Easy solutions to common problems

Avoiding mismatched titles

Sometimes the title or section headers can be misaligned. Seen by, for instance a book title being out of place by quarter of a line. This is because the main program adjusts the title spacing in ways that the diglot code cannot discover (yet?). It is most noticable in book titles, but can also occur in multi-line section headings. The cause is normally that one side contains a taller letter than the other side, or a letter that descends below the line further.

If that's the case, and say one side has no descenders and the other contains a 'p', then the nasty work-around is to add `\dstrut p` to the side which has no 'p'. `\dstrut` swallows the letter that comes after it and replaces it with a non-visible object of zero width, exactly as high and deep as the letter it destroyed.

My cross-references look ugly

With very small columns, and long booknames, you can end up with things looking like this:

[c] **3:10** Deuteronomy 27:26 [d] **3:11** Habakkuk 2:4 [e] **3:12** Leviticus 18:5 [f] **3:13** Deuteronomy 21:23

Or with left justification:

We could tell XeTeX that *anything* is better than line breaks between the origin reference and the text. This nasty bit of code gets rid of the gap and then puts it back, without allowing line breaking:

```
\sethook{end}{xo}{\dimen0=\lastskip\unskip\penalty 10000\hskip \dimen0  minus
   0.5\dimen0\penalty 10000}
```

[c] **3:10** Deuteronomy 27:26
[d] **3:11** Habakkuk 2:4
[e] **3:12** Leviticus 18:5
[f] **3:13** Deuteronomy 21:23

With full justification, you get this:

[c] **3:10** Deuteronomy 27:26
[d] **3:11** Habakkuk 2:4
[e] **3:12** Leviticus 18:5
[f] **3:13** Deuteronomy 21:23

which looks *far worse*, so you really want left justification: Note that this *looks* like one-reference per paragraph, but it's not. If the there were shorter words, a long list of references from one book or the booknames are shortened, then the text will suddenly form paragraphs. This may be just what you want, or it may not be.

XeTeX crashes saying:

```
! Missing number, treated as zero.
<to be read again>
                   \note-fR
\s@tn@tep@r@ms ...count \csname note-#1\endcsname
                                                  =\ifp@ranotes 500 \else 10...

\\...Notes \s@tn@tep@r@ms {#1}\s@tn@tep@r@ms {#1R}
                                                  \else \s@tn@tep@r@ms {#1}\...
<inserted text> \\{f}
                      \\{fe}\\{x}
\initn@testyles ...@tn@tep@rams \the \n@tecl@sses

\ptxfile ...lR {} \initp@rastyles \initn@testyles
                                                  \openadjlist "\the \AdjLis...
```

This is a very characteristic error message, and can be recognised from the first three lines (or the last two). A diglot `.usfm` file is being processed, and XeTeX is trying to set up a diglot version of a footnote, (in this case `\note-fR`, see the third line, but it might be some other "\note-, like xR). However because the\diglottrue"' command wasn't given when the footnote "'\f"' was defined, there is no "'\note-fR"'. Solution: "'\diglottrue"' **must** be in force when style sheets are loaded if diglot typesetting is to be used.

Debugging commands (here be dragons)

These diglot-specific commands are only necessary when something's going horribly wrong. They produce a lot of additional output (even more in the log file) and might help to solve mysterious problems.

- `\tracing{d}` Generate 15-20 lines of debugging information per chunk. Most will be numbered.
- `\tracing{D}` Generate large quantities of additional debugging information. Most are not numbered.
- `\VisTracetrue` A debugging option to help match the numbers from the log file with position in the output. Any time a chunk is added to a page, also put the current debugging number in there. If all is working correctly, there should be no change to the document's pagination from turning this on or off.
- `\VisTraceExtratrue` This command adds even more markers, but the markers may alter what appears on which page.
- `\diglotDbgJoinboxes=132` At various points in the process, boxes (see later) get joined together, by a macro called `\joinboxes`. This is a debugging option to help check that what's happening there is what ought to be happening. XeTeX has a debugging command `\showbox`, which stops processing and writes information about a given box (in the case here, a box is the stack of lines separated by spacing). This command fires the `\showbox` command if the number given is the current debug message number when joinboxes is called (and various other places). Note that `\showboxbreadth=99` and`\showboxdepth=99"' control how much detail is shown before truncation.
- `\def\diglotDbgeachcol{134}`

- `\def\diglotDbgdiglotDbgupdtPtl{213}` Trigger detailed debugging code for a particular occurance of the (frequently met) `\each@col` and `\upd@tep@rtial` macros.
- `\diglotDebugFollowContentstrue` This is the extreme version of `\diglotDbgJoinboxes` above. Rather than just showing boxes at a single point in the code, this will show the box contents at most points of potential interest. Combined with `\tracing{d}\tracing{D}`, log files on the order of 10Mbytes per page are to be expected.