Special Comment Lines; Hidden Preferences

1   Magic Comment Lines

A "Magic Comment Line" is a command in the source which is interpreted by TeXShop. Such lines must occur in the first twenty lines of a source file.

- % !TEX program =
- % !TEX encoding =
- % !TEX root =
- % !TEX numberingCorrection =
- % !TEX useTabs
- % !TEX useTabsWithFiles
- % !TEX tabbedFile{ }(optional short name)
- % !TEX pdfSinglePage
- % !TEX spellcheck =
- % !BIB program =
- % !TEX parameter =
- % !TEX useOldSyncParser
- % !TEX useConTeXtSyncParser
- % !TEX useAlternatePath
- % !TEX bookDisplay
- % !TEX standardDisplay
- % !TEX PageDirectionL2R
- % !TEX PageDirectionR2L

2 Program Examples

TeX source files can be typeset by many programs: TeX, LaTeX, pdfLaTeX, luaLaTeX, etc. TeXShop has an "Engine mechanism" to extend this list to user defined shell scripts, which can run a sequence of programs one after another. The Special Comment "program" line determines which program typesets a particular source file. But using this line is never required because TeXShop has other ways to determine the typesetting program.

If the special comment line is present, it overrides all other methods of determining the typesetting program. Thus if a particular source uses Unicode and requires XeLaTeX, adding a Special Comment line will guarantee that it is always typeset by the correct program.

The syntax of Special Comment lines is very picky about required spaces, cases, etc. Thus I recommend Ramon Figueros-Centeno's "Program" macro, because it lists all active engines and the user can simply click the required engine; Ramon's macro will automatically insert a Special Comment with exactly the right syntax.

Below are some examples:

```
% !TEX program = tex
% !TEX program = latex
% !TEX program = pdftex
% !TEX program = pdflatex
% !TEX program = xelatex
% !TEX program = pdflatexmk
```

Note that "tex" and "latex" produce TeX and DVI typesetting, in which TeX outputs a dvi file, which is then processed to produce a final pdf file. "pdftex" and "pdflatex" use pdftex to directly output pdf files. Note also that "xelatex" and "pdflatexmk" are engines defined by scripts in ∼Library/TeXShop/Engines.

3 Encoding Examples

These Special Comment lines fix the encoding used to read and write a source file, overriding all other methods of setting this encoding. It is important to use the exact name for an encoding used by the Macintosh, so I again recommend Ramon Figueros-Centeno's "Encoding" macro, which lists all possibilities and requires only a click on a choice to produce the correct syntax line.

```
% !TEX encoding = UTF-8 Unicode
% !TEX encoding = Iso Latin 9
% !TEX encoding = Mac Chinese Traditional
```

4 Tabbed Window Examples

The examples in this section still work, but they are mostly replaced by the tabbing features introduced in TeXShop 4.72. Read the changes document about that version for details.

The special comment "root" is explained elsewhere in the TeXShop Help Panel, and used when a project is divided into a commanding root file and included chapter files.

Since macOS Sierra, the Macintosh has supported tabbed windows. See the documentation of Sierra for instruction on creating these tabs using only the Finder.

TeXShop has three special comment commands to extend this build-in system support for tabs.

```
% !TEX useTabs

% !TEX useTabsWithFiles
% !TEX tabbedFile{Galois.aux}(Aux)
% !TEX tabbedFile{Galois.log}
% !TEX tabbedFile{~/Graphics/faireyes.eps}
```

The first line above assumes a root file containing "include" lines for the chapter sources. This special comment creates a window containing the root and various chapter sources as tabs in a single window. In this case, TeXShop searches the root file for the include-lines which name these chapter files.

The last four lines describe a more general method of creating tabbed views. This method requires more information from the user, but produces more flexible tabs. The "useTabsWithFiles" line introduces the method, which then produces a tabbed window for each "tabbedFile" line. This line contains a partial or full path to the file in curly brackets. The tabs will be named using these paths, but this can create long tab names, so it is possible to provide shorter names for the tabs inside round brackets. These shorter names are optional. Optional names are only available in High Sierra and above.

5 More Examples

```
% !TEX numberingCorrection = 0 + current - desired
```

This command lists the number of preliminary pages before the main material is given on pages starting with page 1. TeXShop uses this information to synchronize the PageNumber box on the toolbar with actual page numbers in the document.

```
% !TEX pdfSinglePage
```

This command was proposed by a user for Beamer. The user preferred Multiple Page mode for the display of articles and books, but wanted Beamer documents to appear as single pages in preview, ready to be projected onto a screen. Place this special comment in the source of any document whose preview window should appear in single page mode.

```
% !TEX spellcheck = German
```

This command causes the Spelling and Grammer panel to use the German dictionary when the document is active, even if by default another dictionary is generally used.

```
% !BIB program = biber
```

This causes the BibTeX command to run an alternate program, in this case biber.

```
% !TEX parameter =
```

When engine scripts run, they are passed a parameter giving the full path to the source file. This command passes a second parameter to such engine scripts. The engine script can ignore this parameter, but in some cases it is useful to write a script which is able to perform several tasks, depending on the new parameter.

```
% !TEX useOldSyncParser
% !TEX useConTeXtSyncParser
```

Synctex was written by Jerome Laurens. It causes a TeX engine to output a .synctex file with information needed to jump from a spot in the source to a corresponding spot in the preview, and vice-versa. Laurens also provides parser code for front ends, making it possible for front end authors to provide syncing ability without much extra work. In TeX Live 2017, Laurens fixed bugs and extended both pieces of code.

The ConTeXt system for luaTeX is written by Hans Hagen. Hagen often replaces pieces of TeX Live with his own code in ConTeXt. In 2017, Hagen wrote his own version of the sync code, but unfortunately he based it on Lauren's earlier version 2016 of the code, The 2017 parser cannot handle this code.

After a good deal of work, TeXShop contains both the 2016 and 2017 versions of the parser. By default it uses the 2017 version, but ConTeXt users can switch to the older parser using the special comment above. I don't yet know what will happen in 2018, but shoehorning in both parser libraries was an unpleasant task I am not likely to attempt again.

(Added in 2021): Hans modified the sync code in ConTeXt, Then he wrote routines which front ends can call to obtain sync information from the ConTeXt synctex file. In short, Laurens' parser code is completely replaced by code in ConTeXt. The special comment line "useConTeXtSyncParser" causes TeXShop to call this ConTeXt code. From 2021 on, this is the preferred sync method for ConTeXt users.

One consequence of this development is that Hans can modify sync code if he wishes without breaking sync in TeXShop. This is a welcome development. Thanks to Nicola Vitacolonna for calling this to my attention and urging TeXShop to adopt the new calls.

```
% !TEX useAlternatePath
```

Most TeXShop users pair it with the TeX Live distribution of TeX. This distribution is updated once a year, when new versions of the various executable files are released. The style files, class files, fonts, etc. in TeX Live are updated daily.

This organization works for most TeX binaries. ConTeXt is an exception because Hans Hagen updates it regularly and often. Luckily, there is an easy way to install and update ConTeXt, supported by a beautiful series of web pages called the "ConTeXt Garden." See https://wiki.contextgarden.net/Main_Page. Notice the link on this page titled "Install ConTeXt and start typesetting." This link downloads and installs everything needed to typeset using ConTeXt, placing it in a location of the user's choice. One typical place is ∼/bin/context.

TeXShop needs to be reconfigured to use this distribution rather than TeX Live. This is easy for someone who always typesets with ConTeXt, but troublesome for users who write some documents in LaTeX and some in ConTeXt. TeXShop now has a Preference setting in the Engine tab called "Alternate Path" where users can insert the full path to their ConTeXt distribution. The special command line "useAlternatePath" can then be added to the top of a ConTeXt document to use that path to ConTeXt rather than the standard path to TeX Live.

```
% !TEX bookDisplay
% !TEX standardDisplay

% !TEX PageDirectionL2R
% !TEX PageDirectionR2L
```

In "double page" and "double multi-page" modes, the Preview window shows the first page by itself and then shows a series of double pages. By convention, there are an odd number of pages in a book before the actual content begins, so this causes the left and right pages to appear as they would if a user opened the actual book. These magic lines allow users to change this behavior on a document-by-document basis. The first magic line causes the first page to appear by itself, and the second causes it to appear as a pair with the second page.

In Japan, text can be either vertical or horizontal, and vertical text pages should appear from right to left. The last two lines set this preference on a document by document basis.

6   Hidden Preferences

TeXShop has a large number of Preference Settings which make it possible for users to customize the behavior of the program to their liking. The most important settings are made visible by TeXShop Preferences, but more obscure settings are hidden and only available through the Terminal in /Applications/Utilities. To apply such a preference, it is important to quit TeXShop, then enter the setting in Terminal and push RETURN, and then restart TeXShop.

This document lists all hidden preferences mentioned in the Changes document covering TeXShop 3.07 and later. version 3.07. A complete list of hidden preferences is given in the TeXShop Manual.

TeXShop Versions Before 4.08

- defaults write MakeatletterEnabled YES

- defaults write TeXShop NSFontDefaultScreenFontSubstitutionEnabled -bool YES

- defaults write TeXShop SyncTeXOnly YES

- defaults write TeXShop ScreenFontForLogAndConsole -bool YES

- defaults write TeXShop WatchServer NO

- defaults write TeXShop AutoSaveEnabled NO

- defaults write TeXShop SourceInterlineSpace 1.0

- defaults write TeXShop ResetSourceTextColorEachTime YES

- defaults write TeXShop SwitchSides YES

- defaults write TeXShop InterpolationValue 3

- defaults write TeXShop FixPreviewBlur YES

- defaults write TeXShop FixLineNumberScroll NO

- defaults write TeXShop SourceScrollElasticity NO

- defaults write TeXShop YosemiteScrollBug NO

- defaults write TeXShop ReverseSyncRed 1.00

- defaults write TeXShop ReverseSyncGreen 1.00

- defaults write TeXShop ReverseSyncBlue 0.00

- defaults write TeXShop FixSplitBlankPages NO

- defaults write TeXShop IndexColorStart YES

- defaults write TeXShop OriginalSpelling YES

- defaults write TeXShop ContinuousHighSierraFix NO

- defaults write TeXShop TabsAlsoForInputFiles YES

- defaults write TeXShop FlashFix NO

- defaults write TeXShop FlashDelay 0.25

Version 4.08

- defaults write ColorImmediately YES

- defaults write OpenWithSourceInFront NO

Remark: The first item is obsolete. If the second item is YES, the source window opens in front of the preview window when a document is first opened.

Version 4.76 and Later

- defaults write TeXShop FixVoiceOver NO

- defaults write TeXShop DisplayAsBook YES