The **hanging** package*

Author: Peter Wilson, Herries Press
Maintainer: Will Robertson
`will dot robertson at latex-project dot org`

2009/09/02

Abstract

The **hanging** package provides facilities for defining hanging paragraphs and hanging punctuation.

Contents

1 Introduction

Some authors may wish to use hanging paragraphs in their documents. Normally only the first line of a paragraph is indented. A hanging paragraph is a paragraph like this one where lines other than the first have indentation. Other authors might wish to use hanging punctuation. In this style of typesetting punctuation marks that come at either the start or end of a line are typeset outside the normal text block.

The **hanging** package provides facilities for both hanging paragraphs and hanging punctuation.

This manual is typeset according to the conventions of the LaTeX DOCSTRIP utility which enables the automatic extraction of the LaTeX macro source files [GMS94].

---

*This file (`hanging.dtx`) has version number v1.2b, last revised 2009/09/02.

Section 2 describes the usage of the package. Commented source code for the package is in Section 3.

2 The **hanging** package

2.1 Hanging paragraphs

The `hanging` package provides a command for producing a single hanging paragraph and an environment for typesetting a series of hanging paragraphs.

`\hangpara`

The command `\hangpara{`⟨*indent*⟩`}{`⟨*afternum*⟩`}` placed at the start of a paragraph will cause it to be typeset as a hanging paragraph.

The ⟨*indent*⟩ argument specifies the amount of indentation. The value must be a length. A positive value will indent the left hand side of the paragraph and a negative value will cause the right hand side of the text to be indented.

The ⟨*afternum*⟩ argument controls the number of lines that will be indented. The value can be a positive or negative integer, say N. If $N > 0$ then lines *after* the N^{th} line (i.e., lines $N + 1$ and onwards) will be indented. If $N < 0$ then the first N lines will be indented. This paragraph has been typeset according to the specification `\hangpara{3em}{-2}` whereas the first paragraph in the Introduction was typeset according to `\hangpara{3em}{2}`.

A word of caution: There must be no linebreak between the `\hangpara` command and the start of the paragraph text. Either put the command at the start of the paragraph text, or on the previous line with a following %.

`hangparas`

The `hangparas` environment can be used for typeseting a sequence of hanging paragraphs. The environment takes the same two arguments as the `\hangpara` command. That is,
`\begin{hangparas}{`⟨*indent*⟩`}{`⟨*afternum*⟩`}`.

Another word of caution: The `hangparas` environment changes the definition of `\everypar` and may have unexpected interactions with any enclosed commands or environments that also change `\everypar`. If this does happen, then use the `\hangpara` command for the problematic paragraph(s) instead of the `hangparas` environment.

2.2 Hanging punctuation

`hangpunct`

Text within the `hangpunct` environment will be typeset with hanging punctuation. Personally, I think that this is more of a curiosity than anything else. Others disagree and think it is an essential part of fine typography.

Use of the environment may have unpleasant side effects as the punctuation characters are made active (i.e., they act as commands rather than as glyphs). The hanging punctuation characters are left and right single and double quotes (`‘’`), period (or full stop) (`.`), exclamation mark (`!`), question mark (`?`), colon and semicolon (`:;`), and the comma (`,`).

If you really care about hanging punctuation, Hàn Thế Thành's `pdf(La)TeX` has built in support via `\pdfprotrudechars`. This is implemented in the `pdfTeX` program itself [Tha00], so is robust. The `pdf(La)TeX` system is available from CTAN but the latest version, if not on CTAN, can be obtained from `ftp://ftp.cstug.cz/pub/tex/lo9cal/cstug/thanh/pdftex`.

The package takes several options which stop punctuation characters from being hung. The available options include:

notcomma	don't hang the comma (`,`)
notperiod	don't hang the period (full stop) (`.`)
notquote	don't hang any quote marks (‘ ‘‘ ’’ ’)
notquery	don't hang the question mark (`?`)
notexcl	don't hang the exclamation mark (`!`)
notcolon	don't hang the colon (`:`)
notscolon	don't hang the semicolon (`;`)

If none of these options are given then all the available punctuation marks will be hung.

There is one other option:

fine The default settings for the amount of punctuation hang correspond to those given by Donald Knuth in the *TeXbook* [Knu84], and are independent of the local font in use (use `pdfLaTeX` for font-dependent settings). Stefan Ulrich[1] suggested a less aggresive hang, which is provided by the fine option. This option also stops hanging question and exclamation marks.

`\nhpt`
`\nhlq`
`\nhrq`

If a non-hanging period, or left or right quote character is required in it's character form within the `hangpunct` environment, then respectively the `\nhpt`, `\nhlq` and `\nhrq` commands can be used. For example, if a length is to be changed you cannot write:
`\setlength{\mylength}{2.76cm}`. Instead you must code this as:
`\setlength{\mylength}{2\nhpt76cm}`.

3 The package code

Announce the name and version of the package, which requires LaTeX 2$_\varepsilon$.

```
⟨*usc⟩
\NeedsTeXFormat{LaTeX2e}
\ProvidesPackage{hanging}[2009/09/02 v1.2b hanging paragraphs and punctuation]
```

---

[1] Private email on 2001/03/14 (`ulrich@cis.uni-muenchen.de`).

3.1 Hanging paragraphs

`\hangpara` The command for a single hanging paragraph. Just package up the TeX `\hang...` commands, together with `\noindent` to switch off any existing `\parindent` indentation.

```
\newcommand{\hangpara}[2]{\hangindent#1\hangafter#2\noindent}
```

`hangparas` The environment for typesetting multiple hanging paragraphs. It takes the same arguments as the `\hangpara` command. At the start, set the paragraph indent to zero, then change `\everypar` to `\hangpara`. At the end of the environment make sure that the last paragraph was finished via `\par`, otherwise the user would have to leave a blank line or use `\par` explicitly.

```
\newenvironment{hangparas}[2]{\setlength{\parindent}{\z@}
  \everypar={\hangpara{#1}{#2}}}{\par}
```

3.2 Hanging punctuation

The code for hanging punctuation is based on the Dirty Tricks example on page 395 of the TeXbook [Knu84].

I made an experimental attempt to include hanging hyphens, but this broke too many other things all over the place to be of practical use. I believe Knuth when he says that hanging hyphens require a font with a zero-width hyphen. I think that the basic problem is that the '-' character, as input, represents both a hyphen and a minus sign; diddling with a hyphen means that the minus sign is similarly diddled. This package is meant to be independent of the font, and following Knuth, providing a hanging hyphen is font-dependent. Fortunately, most of the time TeX manages to avoid hyphens at line ends.

Stefan Ulrich (`ulrich@cis.uni-muenchen.de`) pointed out that the EC fonts include a hyphen with a smaller, but non-zero, width. Using this then the last non-hyphen character at the end of a hyphenated line will be a little closer to the margin. To use the smaller EC hyphen, Stefan suggested doing:

```
\usepackage[T1]{fontenc}
\hyphenchar\font=127
```

`\ifh@ngcomma` `\ifh@ngfstop` `\ifh@ngquote` `\ifh@ngquery` `\ifh@ngexcl` `\ifh@ngcolon` `\ifh@ngscolon` `\ifh@ngfine`

We need some booleans for the options.

```
\newif\ifh@ngcomma  \h@ngcommatrue
\newif\ifh@ngfstop  \h@ngfstoptrue
\newif\ifh@ngquote  \h@ngquotetrue
\newif\ifh@ngquery  \h@ngquerytrue
\newif\ifh@ngexcl   \h@ngexcltrue
\newif\ifh@ngcolon  \h@ngcolontrue
\newif\ifh@ngscolon \h@ngscolontrue
\newif\ifh@ngfine   \h@ngfinefalse
```

Now do the options.

```
\DeclareOption{notcomma}{\h@ngcommafalse}
```

```
\DeclareOption{notperiod}{\h@ngfstopfalse}
\DeclareOption{notquote}{\h@ngquotefalse}
\DeclareOption{notquery}{\h@ngqueryfalse}
\DeclareOption{notexcl}{\h@ngexclfalse}
\DeclareOption{notcolon}{\h@ngcolonfalse}
\DeclareOption{notscolon}{\h@ngscolonfalse}
\DeclareOption{fine}{\h@ngfinetrue}
\ProcessOptions\relax
\ifh@ngfine
  \h@ngqueryfalse \h@ngexclfalse
\fi
```

`\h@ngcommawd` `\h@ngfstopwd` `\h@ngquotewd` `\h@ngdquotewd` `\h@ngquerywd` `\h@ngexclwd` `\h@ngcolonwd` `\h@ngscolonwd`

We need some lengths for kerning for each punctuation mark that might be hung. These are set to the width of the character.

```
\newlength{\h@ngcommawd}  \settowidth{\h@ngcommawd}{,}
\newlength{\h@ngfstopwd}  \settowidth{\h@ngfstopwd}{.}
\newlength{\h@ngquotewd}  \settowidth{\h@ngquotewd}{‘}
\newlength{\h@ngdquotewd} \settowidth{\h@ngdquotewd}{‘‘}
\newlength{\h@ngquerywd}  \settowidth{\h@ngquerywd}{?}
\newlength{\h@ngexclwd}   \settowidth{\h@ngexclwd}{!}
\newlength{\h@ngcolonwd}  \settowidth{\h@ngcolonwd}{:}
\newlength{\h@ngscolonwd} \settowidth{\h@ngscolonwd}{;}
```

These values are changed if the fine option is given. The following is by courtesy of Stefan Ulrich.

```
\ifh@ngfine
  \setlength{\h@ngcommawd}{0.35\h@ngcommawd}
  \setlength{\h@ngfstopwd}{0.30\h@ngfstopwd}
  \setlength{\h@ngquotewd}{0.30\h@ngquotewd}
  \setlength{\h@ngdquotewd}{0.35\h@ngdquotewd}
  \setlength{\h@ngcolonwd}{0.20\h@ngcolonwd}
  \setlength{\h@ngscolonwd}{0.20\h@ngscolonwd}
\fi

```

`\h@ngallowhyphens`

An incantation to be used to allow hyphenation in a ‘word’ beginning with a quote mark.[2]

```
\newcommand{\h@ngallowhyphens}{\nobreak\hskip\z@skip}
```

`\nhpt` `\nhlq` `\nhrq`

As the punctuation is going to be made active, we need commands for some of the characters when they will be used in constants.

```
\newcommand{\nhpt}{.}
\newcommand{\nhlq}{‘}
\newcommand{\nhrq}{’}
```

---

[2] Hubert Gaesslein (`hubertjg@virtualphotonics.cm`) pointed out that the definition of `h@ngallowhyphens` should end with `z@skip` rather than my original `z@`, in case it was used in situations I hadn't thought of.

`\h@nglqq` `\h@ngrqq` `\h@ngrquote` `\h@nglquote` `\h@ngcomma` `\h@ngfstop` `\h@ngquery` `\h@ngexcl` `\h@ngcolon` `\h@ngscolon`

Commands for punctuation with kerning.

```
\newcommand{\h@nglqq}{``}
\newcommand{\h@ngrqq}{''}
\newcommand{\h@ngrquote}{'\kern-\h@ngquotewd\kern\h@ngquotewd}
\newcommand{\h@nglquote}{\ifhmode\kern\h@ngquotewd\vadjust{}\else\leavevmode\fi
                         \kern-\h@ngquotewd`\h@ngallowhyphens}
\newcommand{\h@ngcomma}{,\kern-\h@ngcommawd\kern\h@ngcommawd}
\newcommand{\h@ngfstop}{.\kern-\h@ngfstopwd\kern\h@ngfstopwd}
\newcommand{\h@ngquery}{?\kern-\h@ngquerywd\kern\h@ngquerywd}
\newcommand{\h@ngexcl}{!\kern-\h@ngexclwd\kern\h@ngexclwd}
\newcommand{\h@ngcolon}{:\kern-\h@ngcolonwd\kern\h@ngcolonwd}
\newcommand{\h@ngscolon}{;\kern-\h@ngscolonwd\kern\h@ngscolonwd}
```

`\activatepunct`

At this point we make the punctuation characters active.[3]

Activate the punctuation characters, leaving the left quote till last.

```
\def\activatepunct{%
\ifh@ngcomma  \catcode`\,=\active \fi
\ifh@ngfstop  \catcode`\.=\active \fi
\ifh@ngquote  \catcode`\'=\active \fi
\ifh@ngquery  \catcode`\?=\active \fi
\ifh@ngexcl   \catcode`\!=\active \fi
\ifh@ngcolon  \catcode`\:=\active \fi
\ifh@ngscolon \catcode`\;=\active \fi
\ifh@ngquote  \catcode`\`=\active \fi
}
```

Define the simple punctuation macros in terms of the respective `\h@ng...` commands, using `\gdef` as they are instantiated in a group.

```
\begingroup
\activatepunct
\ifh@ngcomma  \gdef,{\h@ngcomma}  \fi
\ifh@ngfstop  \gdef.{\h@ngfstop}  \fi
\ifh@ngquery  \gdef?{\h@ngquery}  \fi
\ifh@ngexcl   \gdef!{\h@ngexcl}  \fi
\ifh@ngcolon  \gdef:{\h@ngcolon}  \fi
\ifh@ngscolon \gdef;{\h@ngscolon}  \fi
```

Life is more complicated for quote marks as they have to check for ligatures.

```
\ifh@ngquote
  \gdef'{\futurelet\next\h@ngrqtest}
  \gdef`{\futurelet\next\h@nglqtest}
```

`\h@ngrqtest` `\h@nglqtest` `\h@ngrquotes` `\h@nglquotes`

Commands for left and right quotes ligature tests.

```
  \gdef\h@ngrqtest{\ifx\next'\let\next=\h@ngrquotes\else\let\next=\h@ngrquote\fi\next}
  \gdef\h@nglqtest{\ifx\next`\let\next=\h@nglquotes\else\let\next=\h@nglquote\fi\next}
  \gdef\h@ngrquotes'{\h@ngrqq\kern-\h@ngdquotewd\kern\h@ngdquotewd}
```

---

[3] My thanks to Harald Hanche-Olsen `hanche@math.ntnu.no` for showing me how to do this instead of my initial hack of putting all the following code into a file to be input by the environment.

```
83  \gdef\h@nglquotes`{\ifhmode\kern\h@ngdquotewd\vadjust{}\else\leavevmode\fi
84    \kern-\h@ngdquotewd\h@nglqq\h@ngallowhyphens}
85 \fi
```

Finally, end the group started earlier.

```
86 \endgroup
```

`hangpunct` The definition of the hanging punctuation environment is now very simple. It just calls `\activatepunct`.

```
87 \newenvironment{hangpunct}{\activatepunct}{}
```

The end of this package.

88 ⟨/usc⟩

References

[GMS94] Michel Goossens, Frank Mittelbach, and Alexander Samarin. *The LaTeX Companion.* Addison-Wesley Publishing Company, 1994.

[Knu84] Donald E. Knuth. *The TeXbook.* Addison-Wesley Publishing Company, 1984.

[Tha00] Hàn Thế Thành. *Micro-typographic extensions to the TeX typesetting system.* Dissertation. Masaryk University Brno, Faculty of Informatics, October 2000. (Available as `http://www.fi-muni.cz/~thanh/download/thesis.pdf`)

Index

Numbers written in italic refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in roman refer to the code lines where the entry is used.