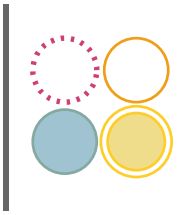
latexindent.pl

Version 3.13



Chris Hughes *

2021-10-30

`latexindent.pl` is a `Perl` script that indents `.tex` (and other) files according to an indentation scheme that the user can modify to suit their taste. Environments, including those with alignment delimiters (such as `tabular`), and commands, including those that can split braces and brackets across lines, are *usually* handled correctly by the script. Options for `verbatim`-like environments and commands, together with indentation after headings (such as `chapter`, `section`, etc) are also available. The script also has the ability to modify line breaks, and to add comment symbols and blank lines; furthermore, it permits string or regex-based substitutions. All user options are customisable via the switches and the YAML interface; you can find a quick start guide in **??** on page ??.

Contents

*and contributors! See **??** on page ??. For all communication, please visit [2].

SECTION 1

Fine tuning

N: 2019-07-13

`latexindent.pl` operates by looking for the code blocks detailed in **??** on page ??. The fine tuning of the details of such code blocks is controlled by the `fineTuning` field, detailed in Listing 1.

This field is for those that would like to peek under the bonnet/hood and make some fine tuning to `latexindent.pl`'s operating.

Warning!

Making changes to the fine tuning may have significant consequences for your indentation scheme, proceed with caution!

LISTING 1: `fineTuning`

```
fineTuning:
    environments:
      name: '[a-zA-Z@\*0-9_\\]+'
    ifElseFi:
      name: '(?!@?if[a-zA-Z@]*?\{)@?if[a-zA-Z@]*?'
    commands:
      name: '[+a-zA-Z@\*0-9_\:]+?'
    keyEqualsValuesBracesBrackets:
      name: '[a-zA-Z@\*0-9_\/.:\#-]+[a-zA-Z@\*0-9_\/.\h\{\}:\#-]*?'
      follow: '(?:(?<!\\)\{)|,|(?:(?<!\\)\[)'
    namedGroupingBracesBrackets:
      name: '[0-9\.a-zA-Z@\*><]+?'
      follow: '\h|\R|\{|\[|\$|\)|\('
    UnNamedGroupingBracesBrackets:
      follow: '\{|\[|,|&|\)|\(|\$'
    arguments:
      before: '(?:#\d\h*;?,?\/?)+|\<.*?\>'
      between: '_|\^|\*'
    trailingComments:
      notPreceededBy: '(?<!\\)'
    modifyLineBreaks:
      betterFullStop:
    '(?:\.\)(?!\h*[a-z]))|(?:(?<!(?:(?:e\.g)|(?:i\.e)|(?:etc))))\.(?!(?:[a-z]|[A-Z]|\-|~|\,|[0-9]))'
      doubleBackSlash: '\\\\(?:\h*\[\h*\d+\h*[a-zA-Z]+\h*\])?'
      comma: ','
```

The fields given in Listing 1 are all *regular expressions*. This manual is not intended to be a tutorial on regular expressions; you might like to read, for example, [1] for a detailed covering of the topic.

We make the following comments with reference to Listing 1:

1. the `environments:name` field details that the *name* of an environment can contain:
   (a) `a-z` lower case letters
   (b) `A-Z` upper case letters
   (c) `@` the `@` 'letter'
   (d) `\*` stars
   (e) `0-9` numbers

(f) _ underscores

(g) \ backslashes

The + at the end means *at least one* of the above characters.

2. the `ifElseFi:name` field:
   (a) `@?` means that it *can possibly* begin with `@`
   (b) followed by `if`
   (c) followed by 0 or more characters from `a-z`, `A-Z` and `@`
   (d) the `?` the end means *non-greedy*, which means 'stop the match as soon as possible'
3. the `keyEqualsValuesBracesBrackets` contains some interesting syntax:
   (a) `|` means 'or'
   (b) `(?:(?<!\\)\{)` the `(?:...)` uses a *non-capturing* group – you don't necessarily need to worry about what this means, but just know that for the `fineTuning` feature you should only ever use *non*-capturing groups, and *not* capturing groups, which are simply `(...)`
   (c) `(?<!\\)\{)` means a `{` but it can *not* be immediately preceded by a `\`
4. in the `arguments:before` field
   (a) `\d\h*` means a digit (i.e. a number), followed by 0 or more horizontal spaces
   (b) `;?,?` means *possibly* a semi-colon, and possibly a comma
   (c) `\<.*?\>` is designed for 'beamer'-type commands; the `.*?` means anything in between `<...>`
5. the `modifyLineBreaks` field refers to fine tuning settings detailed in **??** on page ??. In particular:
   (a) `betterFullStop` is in relation to the one sentence per line routine, detailed in **??** on page ??
   (b) `doubleBackSlash` is in relation to the `DBSStartsOnOwnLine` and `DBSFinishesWithLineBreak` polyswitches surrounding double back slashes, see **??** on page ??
   (c) `comma` is in relation to the `CommaStartsOnOwnLine` and `CommaFinishesWithLineBreak` polyswitches surrounding commas in optional and mandatory arguments; see **??** on page ??

It is not obvious from Listing 1, but each of the `follow`, `before` and `between` fields allow trailing comments, line breaks, and horizontal spaces between each character.

Warning!

For the `fineTuning` feature you should only ever use *non*-capturing groups, such as `(?:...)` and *not* capturing groups, which are `(...)`

Example 1 As a demonstration, consider the file given in Listing 2, together with its default output using the command

```
cmh:~$ latexindent.pl finetuning1.tex
```

is given in Listing 3.

LISTING 2: finetuning1.tex

```
\mycommand{
    \rule{G -> +H[-G]CL}
    \rule{H -> -G[+H]CL}
    \rule{g -> +h[-g]cL}
    \rule{h -> -g[+h]cL}
}
```

LISTING 3: finetuning1.tex default

```
\mycommand{
\rule{G -> +H[-G]CL}
\rule{H -> -G[+H]CL}
\rule{g -> +h[-g]cL}
\rule{h -> -g[+h]cL}
}
```

It's clear from Listing 3 that the indentation scheme has not worked as expected. We can *fine tune* the indentation scheme by employing the settings given in Listing 5 and running the command

```
cmh:~$ latexindent.pl finetuning1.tex -l=fine-tuning1.yaml
```

and the associated (desired) output is given in Listing 4.

LISTING 4: finetuning1.tex using Listing 5

```
\mycommand{
   \rule{G -> +H[-G]CL}
   \rule{H -> -G[+H]CL}
   \rule{g -> +h[-g]cL}
   \rule{h -> -g[+h]cL}
}
```

LISTING 5: finetuning1.yaml

```
fineTuning:
    arguments:
      between:
    '_|\^|\*|\->|\-|\+|h|H|g|G'
```

Example 2 Let's have another demonstration; consider the file given in Listing 6, together with its default output using the command

```
cmh:~$ latexindent.pl finetuning2.tex
```

is given in Listing 7.

LISTING 6: finetuning2.tex

```
@misc{ wikilatex,
author = "{Wikipedia contributors}",
title = "LaTeX --- {Wikipedia}{,}",
note = "[Online; accessed 3-March-2020]"
}
```

LISTING 7: finetuning2.tex default

```
@misc{ wikilatex,
author = "{Wikipedia contributors}",
title = "LaTeX --- {Wikipedia}{,}",
note = "[Online; accessed 3-March-2020]"
}
```

It's clear from Listing 7 that the indentation scheme has not worked as expected. We can *fine tune* the indentation scheme by employing the settings given in Listing 9 and running the command

```
cmh:~$ latexindent.pl finetuning2.tex -l=fine-tuning2.yaml
```

and the associated (desired) output is given in Listing 8.

LISTING 8: finetuning2.tex using Listing 9

```
@misc{ wikilatex,
   author = "{Wikipedia contributors}",
   title = "LaTeX --- {Wikipedia}{,}",
   note = "[Online; accessed 3-March-2020]"
}
```

LISTING 9: finetuning2.yaml

```
fineTuning:
    NamedGroupingBracesBrackets:
      follow: '\h|\R|\{|\[|\$|\)|\(|"'
    UnNamedGroupingBracesBrackets:
      follow: '\{|\[|,|&|\)|\(|\$|"'
    arguments:
      between: '_|\^|\*|---'
```

In particular, note that the settings in Listing 9 specify that NamedGroupingBracesBrackets and

UnNamedGroupingBracesBrackets can follow " and that we allow --- between arguments.

Example 3 You can tweak the fineTuning using the -y switch, but to be sure to use quotes appropriately. For example, starting with the code in Listing 10 and running the following command

```
cmh:~$ latexindent.pl -m
    -y='modifyLineBreaks:oneSentencePerLine:manipulateSentences:␣1,␣
    modifyLineBreaks:oneSentencePerLine:sentencesBeginWith:a-z:␣1,␣
    fineTuning:modifyLineBreaks:betterFullStop:␣
    "(?:\.|;|:(?![a-z]))|(?:(?<!(?:(?:e\.g)|(?:i\.e)|(?:etc))))\.(?!(?:[a-z]|[A-Z]|\-|
    issue-243.tex -o=+-mod1
```

gives the output shown in Listing 11.

LISTING 10: finetuning3.tex

```
We go; you see: this sentence \cite{tex:stackexchange} finishes here.
```

LISTING 11: finetuning3.tex using -y switch

```
We go;
you see:
this sentence \cite{tex:stackexchange} finishes here.
```

Example 4 We can tweak the fineTuning for how trailing comments are classified. For motivation, let's consider the code given in Listing 12

LISTING 12: finetuning4.tex

```
some before text
 \href{Handbook%20for%30Spoken%40document.pdf}{my document}
some after text
```

We will compare the settings given in Listings 13 and 14.

LISTING 13: href1.yaml -m

```
modifyLineBreaks:
    textWrapOptions:
        columns: 80
        all: 1
        perCodeBlockBasis: 1
    removeParagraphLineBreaks:
        all: 1
```

LISTING 14: href2.yaml -m

```
modifyLineBreaks:
    textWrapOptions:
        columns: 80
        all: 1
        perCodeBlockBasis: 1
    removeParagraphLineBreaks:
        all: 1

fineTuning:
    trailingComments:
      notPreceededBy:
    '(?:(?<!Handbook)(?<!for)(?<!Spoken))'
```

Upon running the following commands

```
cmh:~$ latexindent.pl -m finetuning4.tex -o=+-mod1 -l=href1
cmh:~$ latexindent.pl -m finetuning4.tex -o=+-mod2 -l=href2
```

we receive the respective output in Listings 15 and 16.

LISTING 15: finetuning4.tex using Listing 13

```
some before text \href{Handbook some after text%20for%30Spoken%40document.pdf}{my document}
```

LISTING 16: finetuning4.tex using Listing 14

```
some before text \href{Handbook%20for%30Spoken%40document.pdf}{my document} some after text
```

We note that in:

- Listing 15 the trailing comments are assumed to be everything following the first comment symbol, which has meant that everything following it has been moved to the end of the line; this is undesirable, clearly!
- Listing 16 has fine-tuned the trailing comment matching, and says that % cannot be immediately preceeded by the words 'Handbook', 'for' or 'Spoken', which means that none of the % symbols have been treated as trailing comments, and the output is desirable.

Another approach to this situation, which does not use `fineTuning`, is to use `noIndentBlock` which we discussed in **??** on page ??; using the settings in Listing 17 and running the command

```
cmh:~$ latexindent.pl -m finetuning4.tex -o=+-mod3 -l=href3
```

then we receive the same output given in Listing 16; see also `paragraphsStopAt` in **??** on page ??.

LISTING 17: href3.yaml -m

```
modifyLineBreaks:
    textWrapOptions:
        columns: 80
        all: 1
        perCodeBlockBasis: 1
    removeParagraphLineBreaks:
        all: 1
        paragraphsStopAt:
            verbatim: 0

noIndentBlock:
    href:
        begin: '\\href\{[^}]*?\}\{'
        body: '[^}]*?'
        end: '\}'
```

With reference to the `body` field in Listing 17, we note that the `body` field can be interpreted as: the fewest number of zero or more characters that are not right braces. This is an example of character class.

Example 5 We can use the `fineTuning` field to assist in the formatting of bibliography files.

Starting with the file in Listing 18 and running the command

```
cmh:~$ latexindent.pl bib1.tex -o=+-mod1
```

gives the output in Listing 19.

LISTING 18: bib1.bib

```
@online{paulo,
title="arararule,indent.yaml",
author="PauloCereda",
date={2013-05-23},
urldate={2021-03-19},
keywords={contributor},}
```

LISTING 19: bib1-mod1.bib

```
@online{paulo,
   title="arararule,indent.yaml",
   author="PauloCereda",
   date={2013-05-23},
   urldate={2021-03-19},
   keywords={contributor},}
```

Let's assume that we would like to format the output so as to align the = symbols. Using the settings in Listing 21 and running the command

```
cmh:~$ latexindent.pl bib1.bib -l bibsettings1.yaml -o=+-mod2
```

gives the output in Listing 20.

LISTING 20: bib1.bib using Listing 21

```
@online{paulo,
   title    = "arararule,indent.yaml",
   author   = "PauloCereda",
   date     = {2013-05-23},
   urldate  = {2021-03-19},
   keywords = {contributor},}
```

LISTING 21: bibsettings1.yaml

```
lookForAlignDelims:
   online:
      delimiterRegEx: '(=)'

fineTuning:
    keyEqualsValuesBracesBrackets:
      follow:
     '(?:(?<!\\)\{)|(?:(?<!\\)\[)'
     UnNamedGroupingBracesBrackets:
      follow: '\{|\[|,|&|\)|\(|\$|='
```

Some notes about Listing 21:

- we have populated the `lookForAlignDelims` field with the `online` command, and have used the `delimiterRegEx`, discussed in **??** on page ??;
- we have tweaked the `keyEqualsValuesBracesBrackets` code block so that it will *not* be found following a comma; this means that, in contrast to the default behaviour, the lines such as `date={2013-05-23}`, will *not* be treated as key-equals-value braces;
- the adjustment to `keyEqualsValuesBracesBrackets` necessitates the associated change to the `UnNamedGroupingBracesBrackets` field so that they will be searched for following = symbols.

Example 6 We can build upon Listing 21 for slightly more complicated bibliography files.

Starting with the file in Listing 22 and running the command

```
cmh:~$ latexindent.pl bib2.bib -l bibsettings1.yaml -o=+-mod1
```

gives the output in Listing 23.

LISTING 22: bib2.bib

```
@online{cmh:videodemo,
title="Videodemonstrationofpl.latexindentonyoutube",
url="https://www.youtube.com/watch?v=wo38aaH2F4E&spfreload=10",
urldate={2017-02-21},
}
```

LISTING 23: bib2-mod1.bib

```
@online{cmh:videodemo,
   title   = "Videodemonstrationofpl.latexindentonyoutube",
   url     = "https://www.youtube.com/watch?v               = wo38aaH2F4E&spfreload = 10",
   urldate = {2017-02-21},
}
```

The output in Listing 23 is not ideal, as the = symbol within the url field has been incorrectly used as an alignment delimiter.

We address this by tweaking the `delimiterRegEx` field in Listing 24.

LISTING 24: bibsettings2.yaml

```
lookForAlignDelims:
   online:
      delimiterRegEx: '(?<!v)(?<!spfreload)(=)'
```

Upon running the command

```
cmh:~$ latexindent.pl bib2.bib -l bibsettings1.yaml,bibsettings2.yaml -o=+-mod2
```

we receive the *desired* output in Listing 25.

LISTING 25: bib2-mod2.bib

```
@online{cmh:videodemo,
   title   = "Videodemonstrationofpl.latexindentonyoutube",
   url     = "https://www.youtube.com/watch?v=wo38aaH2F4E&spfreload=10",
   urldate = {2017-02-21},
}
```

With reference to Listing 24 we note that the `delimiterRegEx` has been adjusted so that = symbols are used as the delimiter, but only when they are *not preceeded* by either `v` or `spfreload`.