Manued Reference Manual
真鵺道

YAMAUCHI, Hitoshi
(yamauchi@mpi-sb.mpg.de)

2002-8-12(Mon)

Abstract

Manued is a method for proofreading of manuscripts, proposed by Ikuo TAKEUCHI in No.39 Programming symposium in Japan. Manued is a media independent proofreading method, however, it is especially effective for exchanging electronic text via E-mail. This document introduces the manued method and `manued.el` which is an emacs lisp implementation for supporting `manued` method. I wish this method and `manued.el` will be a help for the people who feels some difficulty to exchange a manuscript via Email with their editor or colleagues.

Contents

1 Introduction

1.1 Brief description of Manued

For the present, revising electric manuscripts is not easy as revising them on a piece of paper with red ink. Especially, in Email environment, it is hard to revise a manuscript since the kinds of usable characters are limited. However, many people want to exchange manuscripts by Email via the Internet. Conventionally, many people may use a method for revising an electric manuscript like below.

```
The Three Laws of Robotics by Azimuth.
                              ^^^^^^^

         This may be a mistake for Asimov. And the number of
         laws was changed from three to four, wasn't it?

1. A robot may not injure a robot, nor through inaction
                            ^^^^^ This must be "human being"

   allow a human being to come to harm.
```

This conventional method indicated correcting position with a symbol ‘ˆ’ and proofreading plans with a text. This conventional method always needs much input work and some kind of ambiguous is included in it, because proofreading plans are directed by natural language. The main problem of this conventional method comes from this ambiguous. Even if the manuscript is already inputed in a computer, this ambiguousness interferes automatic extraction of the revised document. Furthermore, information of correcting position with ‘ˆ’ sometimes may be no mean when the mail reading environment is changed. For example, someone reads a manuscript with proportional font and the other one does not. In other words, there is the case that it is difficult to specify a correcting point since each person has own environment to read an Email including a manuscript.

Proofreading of Manued method solves these problems. The input for designation of proofreading is relatively easy, and it can define exact designation of the correcting region and how to change the text. Therefore, there is no fail to miss the correcting position. And there is no mis-recognition of the proofreading text that is a comment or a substitution. In particular, on of the advantages of manued method is independence from environment. We should only assume the manuscript is written as a text file.

Manued has two main notation of proofreading rule. One is “delete and insert” rule. This is shown with notation of [original text (delete) /revised text (insert) ; comments]. Therefore, above text will be changed as follows.

> The Three Laws of Robotics by [Azimuth/Asimov; This may be a mistake with Asimov. And the number of laws was changed from three to four, isn't it?]. 1. A robot may not injure a [robot/human being], nor through inaction allow a human being to come to harm.

Of course, we can use this method with a piece of paper and a pen. However, we assume only an editor to edit a text file. Although, I implemented "manued-minor-mode" on the Emacs editor to make manued method more usefully. In manued-minor-mode, we can use following kinds of functions.

- Extracting the original document or the revised one from one manued document automatically
- Searching the proofreading position
- Assistance for inserting manued commands

Since manued method does not care environment, assisting tools for manued method can be implemented on other editors. Just only because I always use the emacs, I implemented the tool on the emacs. So, please inform me if you implemented an assistance tool for manued method on other environment. Examples of demanded environments are as follows.

- vi
- Microsoft Word
- Filter commands for extracting revised document, etc.

1.2 Information of this document

- Copyright (C) 1997-2002 YAMAUCHI, Hitoshi
- Mail Address

  yamauchi@mpi-sb.mpg.de

2 Quick Start of Manued

2.1 About Manued

Manued is a method for proofreading of manuscripts, proposed by Ikuo TAKEUCHI in No.39 Programming symposium of Japan. The aim of this method is to achieve easy proofreading when people exchange a manuscript via Email. I present `manued.el` as one of the implementations to assist the method method.

2.2 Installation

`manued.el` depends on the Emacs version 19 or later. please use `manued.el` on the Emacs version 19.34 or later, or mule based on them. Author uses it on the GNU Emacs 19.34.1, Mule Version 2.3 (SUETSUMUHANA). Installation of `manued.el` is as follows.

1. Put the `manued.el` file on your emacs 'load-path'.
2. Add the next line to your .emacs file.

   ```
   (autoload 'manued-minor-mode "manued" "manuscript editing mode" t)
   ```

2.3 Starting Manued

To try to `manued.el`, you can read the tutorial file(`manued.tut`) or directly enter to the `manued-minor-mode`.

- Read the tutorial
  You can find the `manued-e.tut` file in the archive of the manued package. Please read the `manued-e.tut` file in your emacs buffer (e.g. type M-x find-file) .
- Directly enter to the `manued-minor-mode`

  1. To enter `manued-minor-mode`, type "M-x manued-minor-mode".
  2. Type some texts in the buffer.
  3. To insert manued swap command, put a mark and type M-x manued-insert-swap-command (or type C-c C-m C-s).
  4. To extract revised document in another buffer, Type manued-show-newer-in-manued-buffer.
  5. Others: please select a menu item. Type M-n or M-p, this is the search command for manued command.

3 What is Manued?

3.1 Outline of Manued

Manued is a method for proofreading of manuscripts, proposed by TAKEUCHI, Ikuo in No.39 Programming symposium in Japan [3]. Manued stands for MANUscript EDiting Language (and there is another origin. Takeuchi is interested in 'Nue'. This is an imaginary creature of Japan. Ma-nue-dou has special meaning in Japanese). The aim of this method is to achieve easy proofreading with exchanging a manuscript via Email.(Here is original text in Japanese on the Web.)

When we revise a manuscript sent by Email, there are some problems as follows.

- We cannot use the special symbols like hand written characters for the proofreading a manuscript. The kind of Email usable-character-set is limited.

- We usually can not use colors.

- Some mail reader does not keep two dimensional information of document. When author's mail reader and editor's one are different, it is not guarantee that positioning information of the document is the same. One famous implementation of such mail reader is Eudora. This is not Eudora's defect. Originally Email protocol (RFC 822) does not have any rules about character width, fonts and how to display breaking lines. Therefore, these situations can usually occur.

- We do not want to depend on line feed position. To send exact correction information, we want to avoid the effect of editor software's line feed processing algorithm for a long line, which depends on each editor program.

To solve these problems, we used to exchanging printed paper including proofreading information with red ink via snail-mail. However, this method takes too long time and we have already high-speed Email. Using facsimile is another solution since it has acceptable transmission speed, but receiver must inputs corrections by reading pieces of paper with his/her eyes. So, some errors will be mixed to a manuscript by human error, and input costs for human being is very high. There is the OCR technology for recognizing hand written characters, but it seems that is not matured for general using. In the end, it seems not a good idea to change the media of the manuscript from electric stuffs to printed papers.

[3] says that proofreading for an electric text should have next features.

- Both revised document and original one are extracted automatically. (required)

- High readability for human being.

- Correction points should be easily found.

- We can put some comments for revised region.
- It should be easy to put revision information to a manuscript.
- Repeated correction directive should be abstracted.

Manued is proposed to satisfy these features and this stands for MANUscript EDiting language. The author presents an implementation to support proofreading with this language.

3.2 Examples of Manued document.

Let me show you some simple examples.

- He was **killed** at programming.
- When “Norwegian **woods**” is translated to Japanese, it is always **interpretented** as “a forest in Norway”. However, **1** of my friends says that the **light** meaning is a furniture made in Norway. Please tell me which is right?
- Why do people say “chapter **first**”?

I emphasized the point which I wanted to correct in this text. However, some browsers may not express them. I want to correct above text as follows.

- He was **skilled** at programming.
- When “Norwegian **wood**” is translated to Japanese, it is always **interpreted** as “a forest in Norway”. However, **one** of my friend says that the **right** meaning is a furniture made in Norway. Please tell me which is right?
- Why do people say “chapter **one**”?

To do this with conventional method, for example, we correct as follows.

```
He was killed at programming.
       ^ skilled. This small missing "s" is fatal!


When ‘‘Norwegian woods’’ is translated to Japanese, it is always
                     ^ delete s. I love the Beatles.

interpretented as ‘‘a forest in Norway’’. However, 1 of my friends
^ interpreted.                                     ^ Of course one is one,
                                                     but write ‘one’.

says that the light meaning is a furniture made in Norway.
              ^ right.
```

```
Please tell me which is right?


Why do people say ``chapter first''?
                            ^ one. (I have also same question but
                                     one is right.)
```

To display this conventional descriptions for browsers, I use special tags to preserve two dimensional information like verbatim (for LaTeX) or pre (for html). However, in Email environment (RFC 822), there is no such special directives and this conventional descriptions using ‘̂’ symbol may corrupted. For example, the same information will be shown in the next paragraph without the tags which preserves two dimensional information. The representation of above text will be varied with each environment, since usual browser ignores successive white spaces and line feeds. Some people can understand next documents, but the others see meaningless texts. It will be clear at a glance that this is not beneficial for the people who can not read next text.

He was killed at programming. ŝkilled. This small missing ”s” is fatal!

When “Norwegian woods” is translated to Japanese, it is always d̂elete s. I love the Beatles.

interpretented as “a forest in Norway”. However, 1 of my friends înterpreted. Ôf course one is one, but write ‘one’.

says that the light meaning is a furniture made in Norway. r̂ight.

Please tell me which is right?

Why do people say “chapter first”? ône. (I have also same question but one is right.)

In addition, even if an environment can preserve two dimensional information, it is hard to distinguish among correction information, only the comment and original text. Moreover, first you corrected the latter part of your manuscript and second you correct former part of the manuscript, latter corrected parts are sometimes shifted. In such situation, you must maintain the manuscript for keeping consistency of two dimensional information of revised position. This is not easy way. It is also stressed that cut and paste manipulation may be harmful for conventional method. Proofreading with Manued is independent from two dimensional information and can resolve conventional method using ”̂ symbol.

In Manued method, there are two replacement rules shown as follows.

- `[A/B]` : This means ‘delete A and insert B’. The variations are

  1. `[A/]` : Delete A (delete A and insert nothing).
  2. `[/A]` : Insert A (delete nothing and insert A).

- `[A|B|C]` : Swap A with C, so result will be ‘CBA’.
  The variation is

  1. `[A||C]` : Swap A with C.

According to above rules, the example texts will be presented as follows.

- He was **[/s; This small missing ”s” is fatal!]**killed at programming.
- When “Norwegian wood **[s/; I love the Beatles.]**” is translated to Japanese, it is always **[interpretented/interpreted]** as “a forest in Norway”. However, **[1/one; Of course one is one, but write ‘one’.]** of my friends says that the **[light/right]** meaning is a furniture made in Norway. Please tell me which is right?
- Why do people say “chapter **[first/one; I have also same question but one is right.]**”?

You can see this method is stronger than conventional method. Do you understand the overview of Manued? If there is a tool which proceeds these rules, manued method will be more powerful and useful. Therefore, I implemented `manued.el`, this program supports manued method. `manued.el` has next functions : highlighting correction regions, searching manued directives, inserting manued directives, extracting revised document and original one.

4 Syntax of Manued

4.1 Manued command and Manued definition commands of correction symbols

In Manued, there are "definition commands of correction symbols" and "manued commands". "Definition commands of correction symbols" is to avoid dependences of different environments and "manued commands" is to indicate proofreading points. We also call definition commands of correction symbols as def-command, because these commands are started with string 'def'. Def-commands define proofreading symbols in a manuscript. To correspond to next circumstances, def-commands is defined by Manued.

- Since usable characters are limited in Email environment, the characters indicating correction points with Manued will be varied according to a document.
- It is almost same with above, if fixed correction characters are used for indicating proofreading point by manued, some environment forces to use too many escape commands. This will degrade readability of manued document. We want to avoid this situation with using def-commands.

Def-command is always put in the beginning of document and must be put at the beginning of line. When it is not put at the beginning of line, Manued does not recognize they are the def-commands. The names of definition commands of correction symbols are 'def' plus each manued command name. For example, 'delete and insert' command is defined by 'defdelete' def-command.

When you want to set delete command symbol as '*', you should insert next line in your document.

```
defdelete *
```

Then correction region will be written as next.

```
[original text*revised text]
```

This means delete 'original text' and insert 'revised text'.

When your document has no def-command, default characters are used that suggested in [3] by Ikuo TAKEUCHI. I will also show the default def-commands in next the section.

4.2 Definition commands of correction symbols for `manued.el` ver.0.9

This section explains the def-commands.

Def-commands always appears in a proofreaded document as follows.

```
defparentheses  [ ]
defdelete       /
defswap         |
defcomment      ;
defescape       ~
deforder        newer-last
defversion      0.9.5
```

I will show the each line meaning and default values of each def-commands.

- defparentheses
  This def-command defines the start string and end string of manued command. (default value ‘`[ ]`’)

- defdelete
  This defines manued command of delete and insert. In default, delete the former part of this symbol and insert the latter part of this symbol. For example, `[A/B]` means delete A and insert B. You can exchange the delete part and insert part, see also deforder. (default value ‘`/`’)

- defswap
  This defines manued command of swap. For example, `[A|B|C]` means A and C will be swapped and results is CBA. (default value ‘`|`’)

- defescape
  This defines manued command of escape. For example, [This is ~[foo~]/These are ~[bars~]] will be delete “This is [foo]” and insert “These are [bars]”. However, too many escapes are very complicated to read. To avoid this, you can re-define manued command strings by using defparentheses. For example, write “defparentheses { }”, we can rewrite below to {This is [foo]/These are [bars]}. (default value ‘~’)
  Here, you must pay attention that escaped symbol is starting manued command strings only when it is not in manued command. This is avoid complicated situation like below.

  ```
  The ratio $a_n/b_n$ must be considered here.
  ~[\lim_{n \to \infty} \frac{a_n}{b_n} \]
  ```

  If any manued command symbol is treated in any place, all manued command in original text must be escaped. In this time, ’/’ of a_n/b_n and] is corresponding to manued command symbol. But these symbol is not in manued command. It is too complicated to escape them all. To avoid such situation, ‘starting manued region strings’ is the only escape target (in here, ‘[’) in plain text.

- defcomment
  This defines comment of manued command. Latter part of this symbol in manued command will be ignored. (default value ‘`;`’)

- deforder
  This defines newer part of delete command. In defdelete command of the manued method, the deleted part is called as the 'older' part, and the inserted part is called as the 'newer' part. The list in delete command is ordered as [first/last]. You can decide which is delete part. So, these (newer older) and (first last) combinations will be next 4 types.

  - newer-first
  - newer-last
  - older-first
  - older-last

  This deforder of definition commands of correction symbols is only define attribute of delete command, and does not define any manued command symbol. (default value '`newer-last`')

- defversion
  This def-command shows the version information for `manued.el`. This is useful information for the `manued.el` program. (default value '(manued-version-num)')

Definition commands of correction symbols should be located the top of a corrected manuscript and each def-command must be located at the beginning of line. If a manuscript does not include these def-commands, the default manued symbols will be used. `Manued.el` checks consistency of def-commands and warns to the user when consistency is broken, for example, if defparentheses has only one argument, manued.el warns it.

4.3 manued commands for `manued.el` version 0.9 or later

This section explains manued commands with default manued symbols.

- `[A/B]`
  Delete A, insert B The variations are

  1. `[A/]` : Delete A. (delete A, insert null string.)
  2. `[/A]` : Insert A. (delete null string, insert A.)

- `[A|B|C]`
  Swap A with C. This will be CBA.
  The variation is

  1. `[A||C]` : Swap A with C. This will be CA.

- `[; comment ]`
  Set a comment range from ';' to ']'.

- Supplement
  These manued symbols will be ignored at right after escape symbol. When you want to write this command strings, you can use escape symbol. For example, `~[` will be `[`.

Nest of these commands is possible, and process is done from the inside. However, non interpretable list of manued command is an error. For example, it is an error that there is no defRparenthesis corresponding to defLparenthesis, or it is also an error like `[A|B/C]` manued sequence.

4.4 Pretty print (after ver.0.9.5)

When you extract the new document with manued, do you want to modify the extracted part? For example, when you use a markup language and you want to emphasize the proofreading part with the language. If yes, there is a way to modify the extracted part.

First, you type

```
  M-x manued-set-pretty-print-on
```

and, you apply some Extracting document commands. I will describe this with next Example 1.

Example 1: Extract from `[A/B;C]` to `\colortext{blue}{A}\colortext{red}{B}\colortext{green}{C}`. Namely, when you want to add some colors at revised parts with TeX.

```
(setq manued-pretty-print-format-delete-list
  '("\\colortext{blue}{%s}" "\\colortext{red}{%s}" "\\colortext{green}{%s}"))
```

This should be included in your `.emacs` or `.manued.el`. The elements of this list correspond with each part of delete command like next:

```
(setq manued-pretty-print-format-delete-list
  '("First Element Format" "Second Element Format" "Comment Element Format"))
```

Each string element will be an argument of format function of the Emacs.

4.4.1 Examples of pretty print

Ex. 2: In `[A/B;C]`, when you want to output B part with `Large` strings in TeX way and want to delete part A and comment part.

```
 (setq manued-pretty-print-format-delete-list
   '("" "{\\Large %s}" ""))
```

Ex. 3: If you want to change the style of manued expression like from `[A/B;C]` to `{A*B@C}`.

```
(setq manued-pretty-print-format-delete-list
  '("{%s*" "%s" "@%s}"))
```

In manued, the comment part is optional and it is not needed. Therefore, when there is no comment in manued expression, the comment part format string is not outputed in default. However, this behavior is controlled by the variable `manued-pretty-print-null-comment-out-p`. Here, we describe only about delete command, but also the way to control swap command with pretty print mode. See 5.6.4 in details.

4.5 Example of manued documents

4.5.1 Example 1

Since defLparenthesis and defRparenthesis can indicate any strings without white space, next example is valid.

```
defparentheses ^_^[  ]^_^

Manued^_^[,/]^_^ stands for ‘‘^_^[Tao of manued/Manuscript Editing
Language; In Japanese, we call this Manuedou and dou means Tao.]^_^’’.
```

- Original version :
  Manued, stands for “Tao of manued”

- Revised version :
  Manued stands for “Manuscript Editing Language”.

4.5.2 Example 2

When you have an environment that can use multi-byte characters, you may use some def-commands with multi-byte characters. This document of Japanese version shows such example. But this document does not assume such environment, I omit the example multi-byte def-command.

4.5.3 Example 3

This looks like SGML, HTML, XML.

```
defparentheses      <manued>  </manued>
defdelete   /

‘‘Don’t worry, Miss Sakuara. The manuscript still is.
  <manued>/At here.</manued>’’

‘‘Give me the supreme <manued>sing/song.</manued>’’

‘‘ <manued>Reverse/Return</manued> to your original figure.’’
```

- Original version :
  “Don’t worry, Miss Sakuara. The manuscript still is.” “Give me the supreme sing” “Reverse to your original figure.”

- Revised version :
  “Don’t worry, Miss Sakuara. The manuscript still is. At here.” “Give me the supreme song. “Return to your original figure.”

4.5.4 Example 4

One of my friends, Mr. Tateoka, proposed below definition, this has some kind of interest since it looks like ‘sed’ commands.

```
defLparenthesis s/
defRparenthesis /g
defdelete //
defescape ~


Manued was proposed by Ikuo TAKEUCHI who grieved that editing
manuscripts by s/Electric mail//Email; Later, you use the word
Email, please standardize your terms./g was not smoothly done
because two dimensional information s/od//of/g document is not
kept in Email.
```

- Original version :
  Manued was proposed by Ikuo TAKEUCHI who grieved that editing manuscripts by Electric mail was not smoothly done because two dimensional information od document is not kept in Email.
- Revised version :
  Manued was proposed by Ikuo TAKEUCHI who grieved that editing manuscripts by Email was not smoothly done because two dimensional information of document is not kept in Email.

5 A Supporting tool `manued.el` for Manued.

5.1 Download

You can find the latest version and documents in following Web sites.

- http://www.mpi-sb.mpg.de/˜hitoshi/otherprojects/manued/index.shtml
- http://www.archi.is.tohoku.ac.jp/˜yamauchi/otherprojects/manued/index.shtml
- http://ne.cs.uec.ac.jp/ (Mirror site)

5.2 Short history of `manued.el`

`manued.el` is an Emacs lisp program for supporting the Manued proofreading method. Manued method itself is an easy proofreading method for correcting electric texts. Next features are considered to support for this method: emphasizing manued command (e.g. using colors), extracting revised text and original text. `manued.el` offers to add colors to manued commands, to search correction points, to extract revised text and original text and to support manued command inputs.

I (the author of `manued.el`) have been interested in Lisp language since I took the Masahiko SATO's class of functional language. At that time, I was an master candidate. However, I have not written any Lisp programs before writing `manued.el`. I sometimes tried to write lisp program, but I always failed because I also have no program which I really want to write and I had no friends who is a lisper. There was the HCL (Hokkaido Common Lisp) in my laboratory on Omron's Luna 68k, but I just touched it at a time and I forget it.

Several years are passed, I must exchange manuscripts to correct with research colleagues of Tohoku University via Email. I felt high unpleasant for that work. First, I printed out a manuscript and checked it by red ball-point pen. Second, I inputed correction points.

Moreover, I supervised a student. One day he send me his draft of his master thesis. The time of correcting work reaches several hours and the time of input work was taken more hours. I used conventional method for correcting manuscript. However, we can not avoid errors with human being inputing them and misunderstanding of comments will be occurred since correction symbol was vague in the conventional method.

In that days, I joined the programming symposium and heard the idea of Manued by Ikuo TAKEUCHI. Instantly I was fascinated this idea. I thought this is the program to write in Emacs lisp. I guessed many people in the hall of programming symposium agree that idea. Professor TAKEUCHI commented that “This program may be very easy to implement. I think the fourth grade student entered his/her graduation thesis can be accomplish to write this program.” I heard this comment and I thought this should be a good exercise to start writing lisp program for me. Fortunately, I have the best teacher to learn lisp and offer `manued.el` here. His name is MAEDA Atusi. I thank him.

5.3 Installation

`manued.el` is based on Emacs Version 19.34 or later. Please use Emacs Version 19.34 or later or Mule based on Emacs version 19.34 or later. The author's environment is GNU Emacs 19.34.1 (mule Version 2.3 (SUETSUMUHANA)).

1. Put `manued.el` the directory which is included load-path.
2. Add next line to '.emacs'.

```
(autoload 'manued-minor-mode "manued" "manuscript editing mode" t)
```

5.4 How to use `manued.el`

Read a manued document which is a target document of proofreading in a emacs buffer, then type 'M-x manued-minor-mode' to enter the manued-minor-mode. Below commands are available. Default key bindings is future work. Of course you can customize them.

- Starting mode and others
  - M-x manued-mode
    This is alias of next manued-minor-mode, see next item.
  - M-x manued-minor-mode
    Enter manued minor mode. `manued.el` after version 0.9.0 has only minor mode. The author usually uses the manued mode with TeX-mode, LaTeX-mode, indented-text-mode and mail draft mode only. Then, for easiness of maintenance for this code, the author abolished manued-major-mode.
  - M-x manued-show-version
    Show the version of `manued.el`.
  - M-x manued-insert-header
    Insert the def-command header for manued command.
- Hi-light
  - M-x manued-hilit-older
    Hi-light and the original part of document in delete command is emphasized.
  - M-x manued-hilit-newer
    Hi-light and the revised part of document in delete command is emphasized.
  - M-x manued-recenter-hilit
    Set current point to the center of the screen and re-hilit the buffer.
- Extracting document

- M-x manued-show-older-in-manued-buffer
  Extract the original document from current buffer to another buffer.
- M-x manued-show-newer-in-manued-buffer
  Extract the revised document from current buffer to another buffer.
- M-x manued-show-older-region
  Extract the original document from a designated region of current buffer. The designated region will be changed.
- M-x manued-show-newer-region
  Extract the revised document from a designated region of current buffer. The designated region will be changed.
- M-x manued-eval-last-manuexp
  Extract the revised document from the prior manued command.
- M-x manued-set-pretty-print-on
  Set pretty print mode on.
- M-x manued-set-pretty-print-off
  Set pretty print mode off.

- Insert the Manued commands

  - M-x manued-insert-swap-command
    Insert swap command. The 'A' part of swap command is filled with designated region. When *manued-is-swap-command-with-comment-on* is 't', comment symbol is inserted together.
  - M-x manued-insert-swap-command-toggle-comment
    Almost same as *manued-insert-swap-command*, but condition of comment insertion is reversed. Comment symbol is inserted when *manued-is-swap-command-with-comment-on* is nil.
  - M-x manued-insert-delete-command
    Insert delete command. The 'delete' part of delete command is filled with designated region. When *manued-is-delete-command-with-comment-on* is 't', comment symbol is inserted together.
  - M-x manued-insert-delete-command-toggle-comment
    Almost same as *manued-insert-delete-command*, but condition of comment insertion is reversed. Comment symbol is inserted when *manued-is-delete-command-with-comment-on* is nil.
  - M-x manued-insert-comment-region]
    Insert comment command. The designated region is surrounded with comment command.
  - M-x manued-insert-comment-at-point
    Insert comment command at the current point.

- Search

- M-x manued-next-l-parenthesis
  Search next manued command from current point.
- M-x manued-previous-l-parenthesis
  Search previous manued command from current point.

5.5 Customizing `manued-minor-mode`

Manued.el provides two main customizing methods. One of them is to change the emacs lisp variables instead of default values. I will show the list of customizable variables later. Another method is to use the hook function. The entry function 'manued-minor-mode' calls

```
(run-hooks 'manued-minor-mode-hook)
```

after all initialization is done. So, you can customize *manued-minor-mode* by using this hook function. (However, now I did not use this feature, this is only available.)

One of recommended methods is that you put the file `.manued.el` in your home directory and set variables in the file, then you read the file from `.emacs`. For this, add your `.emacs` file to next line.

```
(load-file "~/.manued.el")
```

5.6 Customizable variables

5.6.1 Manued definition commands of correction symbols (def-command)

When you set next variables in `.emacs` file or set by any other method, your definition is valid except when def-commands are appeared in manued document. The default values of def-command is also shown in '()'.

- manued-l-parenthesis-str
  The default value of start symbol of manued command. (default '`[`')
- manued-r-parenthesis-str
  The default value of end symbol of manued command. (default '`]`')
- manued-delete-str
  The default value of delete symbol of manued command. (default '`/`')
- manued-swap-str
  The default value of swap symbol of manued command. (default '`|`')
- manued-comment-str
  The default value of starting comment symbol of manued command. (default '`;`')
- manued-escape-str
  The default value of escape symbol of manued command. (default '˜')

- manued-order-str
  The default applying order of delete command. (default ‘older-first’) The candidate variables are “older-first”, “newer-last” or “older-last”, “newer-first”. The form of delete command of manued is [first/last], you must decide which is the revised part (= newer-part). See the item deforder in section 4.2.

`manued.el` checks that these def-commands have consistency or not. If there is no consistency (e.g. two def-commands has same symbol.), `manued.el` may warn it. Then you should change definition of def-commands.

5.6.2 Range specification of manued document

You can specify the range of manued document with next special strings. When these strings of range specifying are not found in manued document, start of manued document is set to top of the document and end of it is set to bottom of the document.

- manued-doc-begin-pat
  Strings which indicating starting point of manued document. (default ”-*-*-BEGINMANUED-*-*-”)

- manued-doc-end-pat
  Strings which indicating end point of manued document. (default ”-*-*-ENDMANUED -*-*-”)

5.6.3 Colors of Hi-Light

There are three choices for using color. Set following variable to `manued-use-color-hilit` according to your wish.

- Use color as possible as it can.(`t`)

- Never use color.(`nil`)

- According to font-lock-mode status.(`’follow-font-lock-mode`)

Example.

```
(setq manued-use-color-hilit ’follow-font-lock-mode)
```

A list which has 6 color names are used to specify the hi-lighting colors. Emacs teaches us what color names are available by typing ‘M-x list-colors-display’. Each elements of the list means as follows.

```
’(delete-first-color delete-last-color
  swap-alpha-color swap-beta-color swap-gamma-color
  comment-color command-color)
```

Where,

- delete-first-color, delete-last-color
  Specify colors of text in delete command as `[`delete-first-color`/`delete-last-color`]`.

- swap-alpha-color, swap-beta-color, swap-gamma-color
  Specify colors of text in swap command as `[`swap-alpha-color — swap-beta-color — swap-gamma-color`]`

- comment-color
  Specify the text color in comment region.

- command-color
  Specify the manued command symbol color. As default, manued command symbols are ‘`[`’,‘`/`’,‘`;`’,‘`]`’.

For example, according to default values, this color list is set with

```
'(red gray60 blue red green4 BlueViolet gray60).
```

The variables of keeping default color list is next.

- manued-first-hilit-color-list
  The color list of when first part is emphasized in delete command.

- manued-last-hilit-color-list
  The color list of when last part is emphasized in delete command.

5.6.4 Pretty print

- manued-pretty-print-format-delete-list
  Format list of pretty print for delete command.
  (default `'("\\textnormal{%s}" "\\textbf{%s}" "\\textit{%s}")`)

- manued-pretty-print-format-swap-list
  Format list of pretty print for swap command.
  (default `'("\\textbf{%s}---" "\\textbf{%s}" "---\\textbf{%s}" "(\\textit{%s})"`)

- manued-pretty-print-on-p
  If `t`, the document extraction command gives you the output with pretty print style. (default `nil`) This variable can be changed with commands (`M-x manued-set-pretty-print-on`, `M-x manued-set-pretty-print-off`). If you want to get the output as always pretty print style, set this variable to `t`.

- manued-pretty-print-null-comment-out-p
  When A manued expression has no comment, pretty print does not output the comment part format strings. However, if you want to output the comment part format whether a manued expression has or doesn't have a comment, you can set this variable `t`. Notice: this variable is not buffer local.

5.6.5 Other customization

- manued-mode-syntax-table
  Default value of syntax-table (default : text-mode-syntax-table)

- manued-mode-abbrev-table
  Default value of abbrev-table (default : text-mode-syntax-table)

- manued-mode-map, manued-minor-mode-map
  Default value of keymaps

- manued-is-delete-command-with-comment-on
  When this value is ‘t’, *manued-insert-delete-command* inserts delete command with comment symbol.

- manued-is-swap-command-with-comment-on
  When this value is ‘t’, *manued-insert-swap-command* inserts swap command with comment symbol.

- manued-is-auto-insert-header
  This value is a list of `'(auto-insert query-when-insert insert-point)`. This value controls inserting def-command and behavior of inserting when no def-command is found in manued document.

  - `auto-insert` : When this is tand when entering manued-minor-mode, `manued.el` tries to insert def-command header if def-command is not found in manued document.
  - `auto-insert` (This is effective iff `auto-insert` is t.) : When this is t, `manued.el` asks insert or not to user with interactively.
  - `insert-point` (This is effective iff `auto-insert` is t.) : There are three cases according to the value.
    - t : Insert header at current cursor position of the buffer.
    - nil : Insert header at the top of the buffer (point-min).
    - number : Insert header at this number position.

- manued-ask-if-formatted-buffer-is
  When this value is t, and when document extracted buffer is already exist, ask user to overwrite the buffer or not.

- manued-defcommand-head-str-list
  List of string which can add top of def-command. Default values are “%” and “%%”, since you can put def-command into the TeX comments.

5.7 Bugs

- The quality of this document is very poor. Suggestions and Comments are welcome.

- Current version (0.9.5) does not use `font-lock-mode`, so, some environment can not hi-light proofreading part. This is related with only coloring the text.

5.8 Copying

`manued.el` (C) Copyright 1998-2002 Yamauchi, Hitoshi

This software is free software. There is absolutely no warranty about this program. This software can be redistributed only under GNU copyleft (most recently version of GNU General Public License).

Suggestions, comments and bug reports are welcome to

> yamauchi@mpi-sb.mpg.de

5.9 Manued Mailing List

Manued mailing list has opened since 2000-3If you want to join that, please send an Email with no contents to

> manued-ml-subscribe@takopen.cs.uec.ac.jp

In this ML, Current public language is Japanese, however, you can also write a mail with English.

6 Manued Tips

- TeX and Manued (How to compile a TeX file which extracted by show newer command immidiately.)
  - When def-command harms TeX compilation.
  - When TeX option is deleted with manued processing.

6.1 TeX and Manued

How to compile TeX file which extracted by show-newer-command immediately.

6.1.1 When def-command harms TeX compilation.

I hope that I can compile the output of show-newer-command immediately. (First, you apply manued-show-newer-in-manued-buffer to the current buffer. Second, you write *manued* buffer which is result of the command to foo.tex file by using M-x write-file. Finally you compile 'foo.tex' file.) However the compilation may fail because of manued def-commands. One solution is not to use def-command in the manuscript and to set manued-def-* variables by elisp startup file(*.emacs*). But one of the meanings of def-command is to keep consistency between the def-command symbols and the manued commands within the document. Therefore, it is not a good idea to eliminate def-command header, especially a manuscript is exchanged via Email since this is just the case that such consistency should be kept.

`manued.el` can recognize extended def-command which is attached with some strings at top of def-command. You can set the head strings of def-command to *manued-defcommnd-head-str-list*. By default, this list includes '%' and '%%' which is the comment symbols of TeX. Next code is recognized by `manued.el` as def-command. However, a manued command must be put at the beginning of line and should not include blank between the head string with the manued command. You can also customize *manued-defcommnd-head-str-list* to change the header string.

```
%%defLparenthesis [
%%defRparenthesis ]
```

6.1.2 When TeX option is deleted with manued processing

Because the default manued command starts at '`[`' and ends with '`]`', this is same as the TeX option argument. I always use Japanese environment, I change the '`[`' and '`]`' to 2bytes parentheses character. However, people use ASCII environment may not use such method. So, one of this solution uses some special strings instead of '`[`' and '`]`'. Next setting is one example.

```
(setq manued-l-parenthesis-str ":[:")
(setq manued-r-parenthesis-str ":]:")
```

However this seems not so good solution. If you find some good strings for manued method, please inform me.

7 Manued Links

7.1 Other implementation of manued.el (By Prof. WADA)

Takeuchi and I introduce another implementation of manued on “bit” No. 12, 1999. Prof. Wada also implemented it on emacs. Why are two manued implementations represented on the same editor (emacs)? I don’t know the exact reason, however, I think emacs is one of the most easy way to implement an interactive text processing tools.

This section introduces the implementation of manued by Prof. Wada. This implementation focuses to “see” the manued document and much smaller than my implementation. This compactness allows to send a manuscript with the code to whom a person did not install `manued.el` and the person can instantly use it.

7.1.1 Implementation of manued by Prof. Wada.

```
;;
;; manued : editing manuscripts for Emacs
;; 1998 Eiiti Wada
;; manued.el, Copyright (C) 1999 Eiiti Wada
;;
;; (YAMAUCHI, Hitoshi translates the comments to English for explanation.)
;;
;;[A\B]   means ‘substitute A with B.’
;;[A\]    means ‘delete A.’
;;[\B]    means ‘insert B.’
;;[A|B|C] means ‘reorder ABC to CBA.’
;;[A||B]  means ‘reorder AB  to BA.’
;;[@C]    means comment.
;;
;; Someone said that it’s hard to read the manued document, so I quick
;; hacked a tool for easy browsing. Let’s try it. This tool works with
;; X Window version of Mule editor.
;;
;; First, put this emacs lisp program in the mule buffer.
;; Second, type ‘esc’ ‘esc’, then you will see the prompt ‘Eval:’
;; at the minibuffer, then type (eval-current-buffer) to evaluate whole
;; the program.
;;
;; Next, put the text of your Email in the mule buffer, type ‘esc’ ‘esc’
;; and (manued). Then, you can see the colored A, B, C potion in the
;; buffer.
;;
;; If you don’t like these colors, type ‘esc’ ‘esc’ again and execute
;; (list-faces-display), then you can get the name list of enable
;; colors.  According to the color list, change the color definition
;; part --- that is in the head part of this program --- to your
;; favorite colors.
;;

(make-face ’alpha)
(set-face-foreground ’alpha "blue")
(make-face ’beta)
(set-face-foreground ’beta "red")
(make-face ’gamma)
(set-face-foreground ’gamma "green4")
(setq facealpha ’alpha)
(setq facebeta  ’beta)
(setq facegamma ’gamma)

(defun proc-range (beg end)
 (let* (p0 p1 p2 ch)
       (goto-char beg)
       (setq beg (search-forward "[" end t))
       (while beg (proc-sec) (setq beg (search-forward "[" end 0)))))

(defun proc-sec nil
 (setq p0 (re-search-forward "[]|\[{]" end t))
 (setq ch (string-to-char (buffer-substring (1- p0) p0)))
 (cond ((= ch ?[) (skip) (proc-sec))
       ((= ch ?\\) (findrsb) (alpha beg p0) (beta p0 p2))
       ((= ch ?|) (findvl) (findrsb) (alpha beg p0) (gamma p0 p1) (beta p1 p2))
       ((= ch ?]) (gamma beg p0))
       ((= ch ?{) (skipcb) (proc-sec))))

(defun skip nil
 (let* (p ch)
       (setq p (re-search-forward "[][{]" end t))
       (while p
              (setq ch (string-to-char (buffer-substring (1- p) p)))
              (cond ((= ch ?]) (setq p nil))
                    ((= ch ?[)
                     (skip)
                     (setq p (re-search-forward "[][]" end t)))
                    ((= ch ?{) (skipcb))   ))))
(defun skipcb nil
 (let* (p ch)
       (setq p (re-search-forward "[}{]" end t))
       (while p
              (setq ch (string-to-char (buffer-substring (1- p) p)))
              (cond ((= ch ?}) (setq p nil))
                    ((= ch ?{)
                     (skipcb)
                     (setq p (re-search-forward "[}{]" end t)))))))

(defun findrsb nil
 (setq p2 (re-search-forward "[][{]" end t))
 (setq ch (string-to-char (buffer-substring (1- p2) p2)))
 (cond ((= ch ?[) (skip) (findrsb))
       ((= ch ?]))
       ((= ch ?{) (skipcb) (findrsb))))

(defun findvl nil
 (setq p1 (re-search-forward "[|[{]" end t))
 (setq ch (string-to-char (buffer-substring (1- p1) p1)))
 (cond ((= ch ?[) (skip) (findvl))
       ((= ch ?|))
       ((= ch ?{) (skipcb) (findvl))))

(defun alpha (l r)
 (put-text-property l (1- r) 'face facealpha)
 (proc-range l (1- r)))

(defun beta  (l r)
 (put-text-property l (1- r) 'face facebeta)
 (proc-range l (1- r)))

(defun gamma (l r)
 (put-text-property l (1- r) 'face facegamma)
 (proc-range l (1- r)))

(defun manued nil (proc-range 1 (point-max)))
```

I thank Prof. Wada Eiiti for permitting to public this code.

7.2 Other implementations

There is an implementation for a text editor named “Hidemaru” on Windows 95 and later.

7.3 Manueder (By Suzuki)

Manuediff is a diff program which outputs manued format. Example:

```
  manuediff file1 file2 > diff.with.manued.format
```

8 Acknowledgements

Thanks to Takeuchi Ikuo to offer "Manuscript Editing on E-mail" and its electric text. Without manued, I can not submit all my papers written in last two years.

I also thanks to Eiiti Wada who gave me an intellectual surprise by the concise implementation.

I also should say "thank you" to Maeda Atusi who teaches me the lisp language. I had no knowledge about lisp when I started to implement this. `manued.el` may not be written without his assistance.

Some people gave fruitful suggestions, comments with working codes to `manued.el` implementation. I will thank for all such people and especially next people. (Alphabetical order)

- KAWAMURA, Takao (bug fix etc. for version 0.9.3)
- NAKAJIMA, Mikio (bug fix etc. for version 0.9.3)
- Stefan Monnier (comments for version 0.9.1)
- Suzuki, Ken-ichi (manuediff)

References

[1] *Emacs and its Info (Emacs 20, XEmacs).*

[2] TAKEUCHI Ikuo. *Hajimete no hito no tame no LISP (LISP for beginners (in Japanese)).* Saiensu-sha, 1986.

[3] TAKEUCHI Ikuo. Manuscript editing on email. *39 th Programming Symposium in Japan*, pages 61–68, January 1998.

[4] AOYAGI Takuta. *UNIX Tanpen series Emacs Lisp (In Japanese).* Kougakutosho, 1997.

[5] HIROSE Yuuji. *Yasashii Emacs-Lisp kouza (In Japanese).* Katto sisutemu, 1999.