mp

Chris Dardis

Monday 6th April, 2015

Contents

1 Package Attributes 2

2 Groups 3

2.1 mp ................................................................. 3
2.2 mp-files ......................................................... 3

3 Functions 3

3.1 mp-mp (&optional FILENAME) ................................. 3
3.2 mp-get-file (&optional NAMESTEM EXTENSION) .......... 5
3.3 mp-R-nw-or-org (&optional FILENAME) ..................... 7
3.4 mp-skeleton (FILENAME listOfChunks) ...................... 9
3.5 mp-insert-chunk (CHUNK) ...................................... 14
3.6 mp-update (&optional FILENAME listofChunks) ............ 16
3.7 mp-ox-settings nil .............................................. 18
3.8 mp-org-tex (&optional FILENAME) .......................... 19
3.9 mp-nw-tex (&optional FILENAME) .......................... 20
3.10 mp-latexmk (&optional FILENAME) ....................... 22
3.11 mp-highlight (BUFFER) ...................................... 24
3.12 mp-view-pdf (&optional FILENAME) ....................... 25
3.13 mp-el-tex (&optional INCLUDESOURCE FILENAME) .... 25

4 Variables (customizable) 35

4.1 mp-entwiner .................................................... 35
4.2 mp-latex ....................................................... 35
4.3 mp-args-latex ................................................ 36
4.4 mp-pdf-viewer ................................................ 36
1 Package Attributes

- Commentary:

  Makes a pdf from the materials in the ‘default-directory’ (or will search up the directory tree for the appropriate file type).
  These may include .R, .tex, .Rnw, .org and .bib files. Indexes, glossaries and table of contents are supported.
  Place the following in your init.el file:
  (add-to-list 'load-path "~/path/to/directory") (require ’mp)
  and, optionally, For HTML export of .org files:
  (require ’htmlize)
  Define f8 as prefix for code chunk in ESS, as there are 8 characters in the prefix:
  (fset ’chunk1 "## —- ") (add-hook ’ess-mode-hook (lambda () (local-set-key (kbd "<f8>") ’chunk1)))
  There is also a simple function for producing .pdf’s from an .el package file.
Function-local/temporary variables are named using the camelCase convention or, for single words, as e.g. variable1.

2 Groups

2.1 mp
This group consists of the elements of ‘make-pdf’. This is a series of variables and functions to simplify the process of pdf creation using R, \LaTeX X and the intermediaries (entwiners) knitr, Sweave and Org (‘org-mode’).

2.2 mp-files
This group is part of ‘mp’. These custom variables are whole files, which are stored as strings in elisp.

3 Functions

3.1 mp-mp (&optional FILENAME)

Documentation: ‘Entwine’ elements in a directory to produce and view a .pdf.

If no FILENAME is supplied, it will try to find the most recently modified of the following file types (in the order below) and pass this to the appropriate method.

Also if ‘p’ is entered as a single character in place of FILENAME, it will try to open the appropriate .pdf file.

- .R -> ‘mp-R-nw-or-org’
- .Rnw -> ‘mp-nw-tex’
- .tex -> ‘mp-latexmk’
- .org -> ‘mp-org-tex’
- .el -> ‘mp-el-tex’
- .pdf -> ‘mp-view-pdf’
(interactive "F FILENAME:"

(let
  (df1)
  (when FILENAME
    (setq FILENAME
      (file-name-nondirectory FILENAME)))
  (when
    (equal FILENAME "")
    (setq FILENAME nil))
  (when
    (and FILENAME
      (equal "p"
        (substring FILENAME
          (-
            (length FILENAME)
            1)
            (length FILENAME))))
    (buffer-file-name)
    (setq FILENAME
      (car
       (file-expand-wildcards
        (concat
         (file-name-sans-extension
           (file-name-nondirectory
            (buffer-file-name)))
           ".pdf")))))
  (unless
    (buffer-file-name)
    (set 'df1
      (directory-files default-directory nil ".*.pdf" t))
    (set 'df1
      (sort df1 'file-newer-than-file-p))
    (setq FILENAME
      (car df1))))
  (unless FILENAME
    (setq df1
      (directory-files default-directory nil "\R$\|\RNw$\|\org$\|\el$\|\pdf$")
    (set 'df1
      (sort df1 'file-newer-than-file-p))
3.2  mp-get-file (&optional NAMESTEM EXTENSION)

Documentation: Find the appropriate file based on the namestem and extension provided (as strings).

The function searches in the current ‘default-directory’. If no matching file is found, it will search up the directory tree.
(string= NAMESTEM "")
(not
(equal nil NAMESTEM))
(not
(string= NAMESTEM
(file-name-sans-extension fileName))))
(setq fileName
(member
(concat NAMESTEM "." EXTENSION)
(file-expand-wildcards
(concat "." EXTENSION))))
(unless fileName
(set 'df1
(file-expand-wildcards
(concat "\\" EXTENSION)))
(setq fileName
(car
(sort df1 'file-newer-than-file-p)))
(unless fileName
(set 'df1
(locate-dominating-file default-directory
(lambda
(x)
(directory-files x nil NAMESTEM))
))
(setq default-directory df1)
(setq fileName
(car
(directory-files default-directory nil NAMESTEM))))
(unless fileName
(error
(concat "No ." EXTENSION " file found in " default-directory)))
(when
(string-match-p
(concat "\\" EXTENSION "$")
(buffer-name))
(save-buffer))
(message "mp-get-file...done")
(message fileName)
(fileName)
3.3 **mp-R-nw-or-org** *(optional FILENAME)*

**Documentation:** Generate an .Rnw or .org file from an .R file with code chunks.

If no FILENAME is supplied, it will try to find the appropriate .R file in the current directory with `mp-get-file`.

If there is no .Rnw or .org file in the corresponding directory, it will generate one with `mp-skeleton`.

If such a file already exists, it will update it with `mp-update`.

It is called by `mp-entwine`.

```lisp
(interactive "F FILENAME:" )

(if FILENAME
  (setq FILENAME
    (file-name-nondirectory FILENAME))
  (setq FILENAME
    (mp-get-file "" "R" )))

(message "mp-R-nw-or-org...")

(let
  (beg1 end1 elem1 elem2
    (listOfChunks 'nil))
  (save-excursion
    (goto-char
      (point-min))
    (while
      (re-search-forward "##------"
        (point-max) t)
    (save-excursion
      (set 'beg1
        (point))
      (move-end-of-line 1)
      (set 'end1
        (point))
      (set 'elem1
        (buffer-substring-no-properties beg1 end1)))
  (if
    (string= mp-entwiner "knitr")
    (set 'elem2 nil)
    (progn
```
(set 'beg1 (point))
(set 'end1
(save-excursion
(search-forward "## ----" (point-max)
  t)))
(if end1
  (set 'end1 (- end1 (length "## ----" )))
  (set 'end1 (point-max)))
(set 'elem2
  (buffer-substring-no-properties beg1 end1))]
(add-to-list 'listOfChunks
  (list elem1 elem2) t)))
(let
  ((fileStem
    (file-name-sans-extension
      (file-name-nondirectory FILENAME)))
   ext1)
  (set 'ext1
    (if
      (string= "Org" mp-entwiner) "org" "Rnw"))
  (set 'elem1
    (member
      (concat fileStem ext1) (file-expand-wildcards (concat "*" ext1))))
  (if elem1
    (mp-update
      (concat fileStem ext1) listOfChunks)
    (mp-skeleton
      (concat fileStem ext1) listOfChunks))
  (message "mp-R-nw-or-org...done")
  (if
    (string= "Org" mp-entwiner)
3.4 mp-skeleton *(FILENAME listOfChunks)*

**Documentation:** Generate an .Rnw or a .org file from a ‘list’ of chunks of ‘R’ code.

If no FILENAME is supplied, it will try to find the appropriate .R file in the current directory with ‘mp-get-file’.

This will read all ‘chunks’ (specified by ‘## --- chunkName’) from the current .R file.

It makes a basic .Rnw or .org file from the chunks. The preamble for .Rnw files is ‘mp-preamble’.

The chunkName is inserted above each chunk, with an optional prefix and suffix. This is subsection(chunkName) by default; see ‘mp-chunk-brackets’.

If ‘mp-entwiner’ is set to ‘Sweave’, packages ‘Sweave’ and ‘lmodern’ are also added.

If ‘mp-latex’ is set to ‘xelatex’, package ‘fontspec’ with font settings is added (see ‘mp-xetex-font’).

The preamble for .org files in ‘mp-org-latex-header’. The author is given by ‘user-full-name’, if available, otherwise by ‘user-login-name’.

The file will be saved with the same name as the associated .R file.

Buffer options for Org export are set with ‘mp-ox-settings’.

This function may be called by ‘mp-R-nw-or-org’.

(interactive "F FILENAME: X listOfChunks: ")

(if FILENAME
  (setq FILENAME
    (file-name-nondirectory FILENAME))
  (setq FILENAME
    (mp-get-file "" "Rnw")))

(find-file FILENAME)

(setq mp-preamble
  (assq-delete-all nil mp-preamble))
(when
  (string= mp-latex "pdflatex")
  (setq mp-latex-font
    (rassq-delete-all nil mp-latex-font)))

(unless
  (string= mp-latex "pdflatex")
  (setq mp-xetex-font
    (rassq-delete-all nil mp-xetex-font)))

(when
  (string= mp-entwiner "Org")
  (defun fun1
    (STRING)
    (let
      (list1)
      (set 'list1
        (split-string STRING "\n"))
      (mapc
        (lambda
          (x)
          (insert
            (format "#+LATEX_HEADER: %s \n" x)))
        list1)))
    (let
      (head1)
      (set 'head1
        (mapconcat
          (lambda
            (x)
            (when
              (equal 'org
                (cadr x)))
              (caddr x))))
        mp-preamble "\n")
      (fun1 head1)
      (set 'head1
        (mapconcat
          (lambda
            (x)
            (when
              (equal 'all
(cadr x))
   (cddr x)))
mp-preamble "\n")
(fun1 head1)
(if
   (string= mp-latex "pdflatex")
   (mapc
      (lambda
         (x)
         (fun1
            (car x)))
      mp-latex-font)
   (mapc
      (lambda
         (x)
         (fun1
            (car x)))
      mp-xetex-font)))

(when
   (not
      (string= mp-entwiner "Org")))
(defun f1
   (KEY)
   (mapc
      (lambda
         (x)
         (when
            (equal KEY
               (cadr x))
            (insert
               (concat
                  (cddr x)
                  "\n")
               mp-preamble)))
   (f1 'class)
   (when
      (string= mp-entwiner "knitr")
      (f1 'knitr))
   (when
      (string= mp-entwiner "Sweave")
      (f1 'sweave))
   (if
(string= mp-latex "pdflatex")
(mapc
 (lambda
  (x)
  (insert
   (car x)))
 mp-latex-font)
(mapc
 (lambda
  (x)
  (insert
   (car x)))
 mp-xetex-font))
(f1 'all)
(insert "\n%%%
−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−
%%%
\begin{document}
%%%
")
(when
 (string= mp-entwiner "knitr")
 (insert mp-knitr-opts)
 (insert "\n%%% knitr read chunks\n<<readChunks, include=FALSE>>=\n")
 (insert
 (concat "read_chunk('" fileStem ".R')")
 (insert "\n@")
(insert "\n")
(let
 (author1)
 (set 'author1
 (if
   (equal user-full-name "")
   user-login-name user-full-name))
 (insert
 (if
   (string= mp-entwiner "Org")
   (concat "#+TITLE:" fileStem)
   (concat "\title{" fileStem "}")))
(insert "\n")
(when
 (string= mp-entwiner "knitr")
 (insert mp-knitr-opts)
 (insert "\n%%% knitr read chunks\n<<readChunks, include=FALSE>>=\n")
 (insert
 (concat "read_chunk('" fileStem ".R')")
 (insert "\n@")
(insert "\n")
(let
 (author1)
 (set 'author1
 (if
   (equal user-full-name "")
   user-login-name user-full-name))
 (insert
 (if
   (string= mp-entwiner "Org")
   (concat "#+TITLE:" fileStem)
   (concat "\title{" fileStem "}")))
(insert "\n")
\maketitle
%%% page numbers appear top−right
\pagestyle{headings}
\tableofcontents


(let ((i 0)
  (elem1 nil)
  (beg1 end1)
  (while (< i
    (length listOfChunks))
  (set 'elem1
    (nth i listOfChunks))
  (mp-insert-chunk elem1)
  (incf i)))

(when (not
  (string= mp-entwiner "Org"))
  (insert "\n\maketitle\n\%\%
page numbers appear top−right\n\pagestyle{headings}\n\tableofcontents\n\n")

(when (not
  (string= mp-entwiner "Org"))
  (insert "\n%%%
\end{document}"

(write-region
  (point-min)
  (point-max)
  (buffer-name))

(set-visited-file-name
  (buffer-name)
  t)

(save-buffer)

(when
  (string= mp-entwiner "Org")
  (mp-ox-settings))
3.5 mp-insert-chunk (CHUNK)

Documentation: Insert a CHUNK into an existing .Rnw or .org file.

The CHUNK is given as a list, where the ‘car’ is the name of the chunk
and the ‘cdr’ is the chunk contents.

The CHUNK is enclosed in by ‘mp-chunk-brackets’. ‘mp-Sweave-opts’
or ‘mp-org-header-args’ will also be added as needed.

(set 'beg1
  (if
    (string= mp-entwiner "Org")
    (nth 2 mp-chunk-brackets)
    (nth 0 mp-chunk-brackets)))

(set 'end1
  (if
    (string= mp-entwiner "Org")
    (nth 3 mp-chunk-brackets)
    (nth 1 mp-chunk-brackets)))

(insert
  (concat beg1
    (prin1-to-string
      (first CHUNK)
      t)
    end1 "\n\n"))

(set 'beg1
  (if
    (string= mp-entwiner "Org")
    "#+NAME: " <<")

(set 'end1
  (if
    (string= mp-entwiner "Org")
    "" ">>="))

(insert
  (concat beg1
    (prin1-to-string
      (first CHUNK)
      t)))
(when
  (string= mp-entwiner "Sweave")
  (setq mp-Sweave-opts
    (rassq-delete-all nil mp-Sweave-opts))
  (insert
    (concat ","
      (mapconcat 'car mp-Sweave-opts ","))))

(insert
  (concat end1 "\n"))

(when
  (string= mp-entwiner "Org")
  (setq mp-org-header-args
    (assq-delete-all nil mp-org-header-args))
  (set 'end1
    (mapconcat
      (lambda
        (x)
        (concat
          (cadr x)
          " \n"
          (cddr x)))
      mp-org-header-args " ")
    mp-org-header-args "")
  (insert
    (concat "#+begin_src R " end1 "\n")))

(when
  (not
    (string= mp-entwiner "knitr"))
  (insert
    (prinl-to-string
      (second CHUNK)
      t)
    "\n")
  (set 'end1
    (if
      (string= mp-entwiner "Org"
        "#+end_src " @")
    (insert
      (concat end1 "\n\n"))))
3.6 mp-update (&optional FILENAME listOfChunks)

Documentation: Update an .Rnw or .org file with a ‘list’ of chunks. If no FILENAME is supplied, it will try to find the appropriate .R file in the current directory with ‘mp-get-file’.

The ‘list’ should be in the form of a value-pair, indicating the name and contents of each chunk e.g. (‘foo’ ‘barbarbar’).

If ‘mp-entwiner’ is set to ‘Sweave’ or ‘Org’, this function is called by ‘mp-R-nw-or-org’.

(interactive "F FILENAME: X listOfChunks: ")

(if FILENAME
  (setq FILENAME
    (file-name-nondirectory FILENAME))
  (if
    (string= mp-entwiner "Org")
    (setq FILENAME
      (mp-get-file ""
        "org"))
    (setq FILENAME
      (mp-get-file ""
        "Rnw")))))

(setq org-startup-folded nil)

(find-file FILENAME)

(message "mp-update...")

(let
  ((namesCurrChunks nil)
   prefix1 suffix1 beg1 end1 old1 chunk1 chunkName1 chunkValue1)
  (set 'prefix1
    (if
      (string= "Org" mp-entwiner)
        "#[+]NAME: " "<<")
    (set 'suffix1
      (if
        (string= "Org" mp-entwiner)
        "$" ".,")))
(save-excursion
  (goto-char (point-min))
  (while
    (re-search-forward prefix1 (point-max) t)
    (save-excursion
      (set 'beg1 (point))
      (search-forward-regexp suffix1 nil t)
      (set 'end1 (point))
      (set 'chunk1
        (buffer-substring-no-properties beg1 end1))
      (add-to-list 'namesCurrChunks chunk1 t)))
  (search-forward-regexp prefix1)
  (move-beginning-of-line -2)
  (let
    ((i 0) (elem1 nil))
    (while
      (< i (length listOfChunks))
      (set 'elem1 (nth i listOfChunks))
      (unless
        (set 'old1
          (member (car elem1) namesCurrChunks)
          (mp-insert-chunk elem1))
        (when old1
          (set 'chunkName1 (first elem1))
          (set 'chunkValue1 (second elem1))
          (goto-char (point-min))
          (search-forward-regexp (concat prefix1 chunkName1) (point-max) t))
    )
  )
)
(move-beginning-of-line
 (if
   (string= "Org" mp-entwiner)
   3 2))
(set 'suffix1
 (if
   (string= "Org" mp-entwiner)
   "#+end_src" "@"))
(insert
 (prin1-to-string chunkValue1 t))
(set 'beg1
 (point))
(search-forward suffix1 nil t)
(end-of-line 0)
(set 'end1
 (point))
(delete-region beg1 end1)
(move-beginning-of-line 4))
(incf i))))

(save-buffer)

(when
  (string= mp-entwiner "Org")
  (mp-ox-settings))

(message "mp-update...done")

3.7 mp-ox-settings nil

Documentation: Set ‘org-mode’ export settings. All variables are set as
buffer-only (see ‘make-local-variable’).

  Adds support for ‘R’, ‘latex’ and ‘emacs-lisp’ to ‘org-babel-load-languages’.
  Sets the following to ‘nil’: ‘org-confirm-babel-evaluate’ and ‘org-latex-with-
  hyperref’ and ‘org-latex-table-caption-above’. Sets ‘org-latex-listings’ to ‘t’.
  Adds ‘listings’ and ‘color’ to ‘org-latex-packages-alist’.

(interactive)

(require 'ox-latex)

(require 'ob-R)
3.8  mp-org-tex (&optional FILENAME)

Documentation: Use an .org file to make a .tex (\TeX) file. If no FILENAME is supplied, it will try to find the appropriate .R file in the current directory with 'mp-get-file'.

(if FILENAME
  (setq FILENAME
3.9 \texttt{mp-nw-tex (optional FILENAME)}

\textbf{Documentation:} Generate a \texttt{.tex} (\LaTeX) file from an \texttt{.Rnw} file. Add \texttt{mp-sweaveSty} and \texttt{mp-upquoteSty} to the current directory if required.

If no \texttt{FILENAME} is supplied, it will try to find the appropriate \texttt{.R} file in the current directory with \texttt{mp-get-file}'.

Once complete, \texttt{mp-latex-pdf} will be run on the output.
(when
  (assoc t mp-upquoteSty)
  (with-temp-file "upquote.sty"
    (insert
dr
    (assoc t mp-upquoteSty))))
)

(let
  ((fileStem
    (file-name-sans-extension
      (file-name-nondirectory FILENAME)))
    procRes defDir)
  (set 'defDir default-directory)
  (unless
    (directory-files default-directory nil
      (concat fileStem ".Rnw")
    (mp-R-nw-or-org
      (concat fileStem ".R"))
  (when
    (string= mp-entwiner "knitr")
    (with-temp-file "knit.R"
      (insert
        (concat "knitr::knit(" fileStem ".Rnw")")))
    (pop-to-buffer "*make−pdf*"
      (setq default-directory defDir)
    (goto-char
      (point-max))
    (set 'procRes
      (call-process "Rscript" nil t t "knit.R"))
  (unless
    (= procRes 0)
  (error
    (concat "Error with knitr"))))
)

(when
  (string= mp-entwiner "Sweave")
  (when
    (assoc t mp-sweaveSty)
    (unless
      (directory-files default-directory nil "Sweave.sty")
    (with-temp-file "Sweave.sty"
      (insert
        (cdr
        (assoc t mp-sweaveSty)))))
  (pop-to-buffer "*make−pdf*"
    (setq default-directory defDir)
  (goto-char
    (point-max))
  (set 'procRes
    (call-process "\$Rscript" nil t t "Sweave.sty"))
  (unless
    (= procRes 0)
  (error
    (concat "Error with Sweave"))))
)
(pop-to-buffer "*make−pdf*"
(setq default-directory defDir)
(setq 'procRes
  (call-process "R" nil t t "CMD" "Sweave"
    (concat fileStem ".Rnw"))

(unless
  (= procRes 0)
  (error
    (concat "Error with Sweave")))))
(mp-latexmk
  (concat fileStem ".tex")))
(message "mp−nw−tex...done")

3.10 mp-latexmk (&optional FILENAME)

Documentation: Use a .tex (LaTeX, XeTeX) file to make a .pdf file, using the
method given by ‘mp-latex’ and ‘latexmk’. Add a ‘latexmkrc’ file to the
‘default-directory’ to help with this. If another ‘latexmkrc’ is on your path,
the local copy will override this.

If no FILENAME is supplied, it will try to find the appropriate .tex file
in the curren directory with ‘mp-get-file’.

Once complete it will open the file with ‘mp-view-pdf’. It is called by
the functions ‘mp-nw-tex’ and ‘mp-org-tex’.

See the manual for details: URL ‘http://ctan.mackichan.com/
support/latexmk/latexmk.pdf’.

(interactive "F FILENAME:"

(unless FILENAME
  (setq FILENAME
    (mp-get-file "")"tex")))

(message "mp−latexmk...")

(let
  (bibBuf extraArgs defDir
    (fileStem
      (file-name-sans-extension
        (file-name-nondirectory FILENAME))

  (when
(assoc t mp-latexmkrc)
(unless
  (directory-files default-directory nil "^\.latexmkrc$")
(with-temp-file ".latexmkrc"
  (insert
    (cdr
     (assoc t mp-latexmkrc)))
  (insert "$pdf_mode = 1; \n$postscript_mode = \n$dvips_mode = 0; \n"
  (insert
    (concat "$pdflatex = " mp-latex " %O %S \n")
  )))
  (when
    (set 'bibBuf
      (find-buffer-visiting
        (concat fileStem ".bib"))))
  (with-current-buffer bibBuf
    (save-buffer))
  (set 'extraArgs
    (mapconcat 'car mp-args-latex " ")
  (set 'defDir default-directory)
  (pop-to-buffer "*make-\npdf*
  (setq default-directory defDir)
  (goto-char
    (point-max))
  (insert "\nRUNNING LATEXMK\n\n")
  (lexical-let
    ((fileStem fileStem))
  (set-process-sentinel
    (start-process-shell-command "async-\npdf" "*make-\npdf*"
      (concat "latexmk " fileStem " "
        extraArgs))
  (lambda
    (process event)
  (message "mp-late\nxmk...done")
  (mp-highlight "*make-\npdf")
  (mp-view-pdf
    (concat fileStem ".pdf")
  (when
    (not
      (string-match-p "finished" event))
  (error "Error in latexmk")))))))
3.11  **mp-highlight (BUFFER)**

**Documentation:** Highlight important words in output when generating .pdf files.

Runs in the current buffer.

```lisp
(interactive "B Buffer: ")

(unless BUFFER
  (setq BUFFER
         (current-buffer)))

(pop-to-buffer BUFFER)

(defun fun1
  (REGEXP FACE)
  (save-excursion
    (while
      (re-search-forward REGEXP nil t)
      (set-text-properties
       (match-beginning 0)
       (match-end 0)
       FACE))))

(save-excursion
  (goto-char (point-min))
  (fun1 "^Run.*$" '(face highlight))
  (fun1 ".arning.*$" '(face holiday))
  (fun1 ".itation.*$" '(face holiday))
  (fun1 ".ference.*$" '(face holiday))
  (fun1 "Error.*$" '(face holiday))
  (fun1 "Fatal.*$" '(face holiday))
  (fun1 "ignored" '(face warning))
  (fun1 "\+[+\?erfull.*$" '(face error))

```
3.12 mp-view-pdf (optional FILENAME)

**Documentation:** View FILENAME with `mp-pdf-viewer`.

```lisp
(defun "You can't.-$"
     '(face error)))
```

3.13 mp-el-tex (optional INCLUDESOURCE FILENAME)

**Documentation:** Generate a .tex file from a .el file containing a package.
If INCLUDESOURCE is non-nil, the source code for the functions and the default values of the variables in the package are included also.
The keywords to identify in the package preamble are given in `mp-el-package-attributes`.
It is designed for packages which are contained completely in one file.

```lisp
(defun mp-el-tex (INCLUDESOURCE FILENAME)
  (interactive "F FILENAME:")
  (if FILENAME
      (setq FILENAME
            (file-name-nondirectory FILENAME))
    (setq FILENAME
            (mp-get-file "" "pdf")))
  (start-process-shell-command "mp-view" nil
   (cons nil (concat mp-pdf-viewer "" FILENAME)))
```

```lisp
(defun mp-el-tex (INCLUDESOURCE FILENAME)
  (interactive (list
                (y-or-n-p "Include source? ")
                (read-file-name "File name?")))
  (unless FILENAME
    (setq FILENAME
          (mp-get-file "" "el")))
  (unless
    (or INCLUDESOURCE
     (equal INCLUDESOURCE ""))
    (setq INCLUDESOURCE t))
```
(setq mp-preamble
  (assq-delete-all nil mp-preamble))

(when
  (string= mp-latex "pdflatex")
  (setq mp-latex-font
    (rassq-delete-all nil mp-latex-font)))

(unless
  (string= mp-latex "pdflatex")
  (setq mp-xetex-font
    (rassq-delete-all nil mp-xetex-font)))

(message "mp-el-tex...")

(let*
  (beg1 end1 r1 elem1
   (l0 'nil)
   (l1 'nil)
   (fileStem
     (file-name-sans-extension
      (file-name-nondirectory FILENAME))))
  (defun fun1
    (KEYWORD)
    (set 'r1
      (concat "^."  
        (substring  
          (symbol-name KEYWORD)  
          1)  
        ":?"))
    (save-excursion
      (set 'elem1
        (search-forward-regexp r1 nil t)))
    (when elem1
      (set 'beg1
        (point))
      (end-of-line)
      (set 'end1
        (point))
      (set 'elem1
        (buffer-substring-no-properties beg1 end1)))
    (add-to-list '10
      (cons KEYWORD elem1))
(goto-char (point-min))
(save-excursion
  (goto-char (point-min))
  (set 'beg1 (point))
  (forward-comment (buffer-size))
  (set 'end1 (point))
  (set 'elem1 (buffer-substring-no-properties beg1 end1)))

(when elem1
  (with-temp-buffer
    (insert elem1)
    (goto-char (point-min))
    (save-excursion
      (while (re-search-forward ";;" nil t)
        (replace-match "" nil nil)))
    (set 'case-fold-search nil)
    (mapc 'fun1 mp-package-attributes)
    (save-excursion
      (when (search-forward-regexp ";;;###autoload" nil t)
        (add-to-list 'l0 (cons 'autoload "TRUE") t)))
    (search-forward-regexp "\[cC]ommentary:?" nil t)
    (set 'beg1 (point))
    (search-forward "Code" nil t)
    (beginning-of-line -2)
    (set 'end1 (point))
    (set 'elem1 (buffer-substring-no-properties beg1 end1))
    (set 'elem1 (replace-regexp-in-string " ;;\s+" "newline" elem1))
    (add-to-list 'l0 27)
(cons 'Commentary elem1)

(set '10
    (rassq-delete-all nil 10)))
10)
(while
 (not
  (=
   (point)
   (point-max)))
(forward-sexp)
(save-excursion
 (set 'beg1
      (point))
(set 'elem1
    (sexp-at-point))
(add-to-list 'll
            (cons
             (car elem1)
             elem1)
            t)))

(find-file
 (concat fileStem ".org"))
(erase-buffer)
(defun fun2
 (STRING)
 (let
   (list1)
       (set 'list1
            (split-string STRING "\n")
        (mapc
          (lambda
            (x)
            (insert
             (format "\#+LATEX_HEADER: %s \n" x)))
          list1)))
     (let
       (head1)
       (set 'head1
            (mapconcat
             (lambda
              (x)
              (when

28
(equal 'org
  (cadr x))
  (cddr x))
  mp-preamble "\n")
(fun2 head1)
(set 'head1
  (mapconcat
    (lambda
      (x)
      (when
        (equal 'all
          (cadr x))
          (cddr x))
  mp-preamble "\n")
(fun2 head1))
(if
  (string= mp-latex "pdflatex")
  (mapc
    (lambda
      (x)
      (fun2
        (car x)))
    mp-latex-font)
  (mapc
    (lambda
      (x)
      (fun2
        (car x)))
    mp-xetex-font))
(insert "\n")
(when 10
  (insert "\n\n")
  (insert "\n\n")
  (mapc
    (lambda
      (x)
      (insert
        (concat "—•"
          (symbol-name
            (car x))
          "•:")
      (insert
        (concat

(cdr x)
 "\n")
 (insert "\n")
 10)
 (insert "\n")
(setq ll
 (sort ll 'equal))
(when
 (assoc 'defgroup ll)
 (insert "\n* Groups\n
")
(mapc
 (lambda
 (x)
  (when
  (eq 'defgroup
   (car x))
  (insert
   (concat "** "
    (symbol-name
     (nth 1
      (cdr x))
    "\n\n"))
  (insert
   (documentation-property
    (eval
     (cdr x))
    'group-documentation))
  (insert "\n\n")))
ll)
(set 'll
 (assq-delete-all 'defgroup ll)))
(when
 (assoc 'defun ll)
 (insert "\n* Functions\n
")
(mapc
 (lambda
 (x)
  (when
   (eq 'defun
    (car x))
   (insert
    (concat "** "
     (symbol-name
      (nth 1
       (cdr x))))
    "\n\n")
  (insert
   (documentation-property
    (eval
     (cdr x))
    'function-documentation)))
(l1)
(set 'l1
 (assocdefgroup ll))
(set 'l1
 (assq-delete-all 'defgroup l1)))
(nth 1
  (cdr x)))))
(insert
  (concat "/"
    (prinl-to-string
      (help-function-arglist
        (eval
          (cdr x))))
    "\n\n")
(insert
  (concat "*Documentation*:"
    (documentation
      (eval
        (cdr x))
      t)
    "\n\n")
(when INCLUDESOURCE
  (insert "\n#+begin_src lisp \n"
    (set 'elem1
      (indirect-function
        (cdr x)))
    (pop elem1)
    (pop elem1)
    (pop elem1)
    (while elem1
      (set 'r1
        (pop elem1))
      (when
        (eq 'cons
          (type-of r1))
        (insert
          (concat
            (pp r1)
            "\n")
          (insert "\n#+end_src \n"))))
  (insert "\n#+begin_src lisp \n"
    (set 'l1
      (assq-delete-all 'defun l1)))
  (when
    (assoc 'defcustom l1)
    (insert "\n* Variables (customizable)\n"
      (mapc
        (lambda
          (nth 1
            (cdr x))))))
(x)

(when
  (eq 'defcustom
    (car x))
(insert
  (concat "** "
    (symbol-name
      (nth 1
        (cdr x)))
    "\n\n"))

(when
  (set 'elem1
    (documentation-property
      (eval
        (cdr x))
    'variable-documentation t))
(insert "\Documentation:\n")
(insert
  (concat elem1 "\n\n"))

(when INCLUDESOURCE
  (when
    (set 'elem1
      (get
        (nth 1
          (cdr x))
      'standard-value))
(insert "\Standard value:\n")
(insert "\#+begin_src lisp \n"
(insert (format "%s" elem1))
(insert "\n\#+end_src \n")

(when
  (set 'elem1
    (get
      (nth 1
        (cdr x))
    'custom-type))
(insert "\Type:\n")
(insert "\#+begin_src TeX")
(mapcar
  (lambda
    (x)
    (insert
(format "\n%s" x)))
elem1)
(insert "\n#+end_src \n")
(when
  (set 'elem1
    (get
      (nth 1
        (cdr x))
      'custom-options))
  (insert "*Options: \n")
  (insert "#+begin_src lisp"

    (mapc
      (lambda
        (x)
        (insert
          (format "\n%s" x)))
elem1)
    (insert "\n#+end_src \n"))
  (when
    (set 'elem1
      (get
        (nth 1
          (cdr x))
        'custom-links))
    (insert "*Links: \n")
    (mapc
      (lambda
        (x1)
        (if
          (set 'r1
            (member :tag x1))
          (insert
            (concat "[[" (car
              (last r1))
            "\n"]")
            (cadr r1)
            "\n"]\n")))
          (insert
            (concat "[[" (last x1)
              (last x1)
              "\n"]\n"))))
elem1))))))}
(set 'l1
  (assq-delete-all 'defcustom l1)))

(when
  (assoc 'defvar l1)
  (insert "\n  Variables\n")
  (mapc
    (lambda
      (x)
      (when
        (eq 'defvar
          (car x))
        (insert "\n** \n"
          (symbol-name
            (nth 1
              (cdr x)))
          "\n")
        (insert "\n")
        (when
          (set 'elem1
            (documentation-property
              (eval
                (cdr x))
              'variable-documentation t))
          (insert
            (concat "\n\n*Documentation*:\n  elem1 "\n\n"))))
  l1)
  (setq l1
    (assq-delete-all 'defvar l1))
  (insert "\n")
  (when l1
    (setq l1
      (sort l1 'equal))
    (insert "\n  Additional code\n")
    (mapc
      (lambda
        (x)
        (unless
          (equal nil
            (cdr x))
          (insert
            (concat "\n** \n"))))
  l1)
4 Variables (customizable)

4.1 mp-entwiner

Documentation:
The method to 'entwine' files in a directory.
One of: 'knitr', 'Sweave' or 'Org' (see 'org-mode').

Standard value:

(knitr)

Type:
radio
(const :doc Default :value knitr)
(const :doc Sweave with .nw :value Sweave)
(const :doc Weave with .org :value Org)

4.2 mp-latex

Documentation:
This is the command for generating a .pdf from a .tex (TeX) file. Other options are also possible.

Standard value:

(pdflatex)

Type:
4.3 mp-args-latex

**Documentation:**
Alist of command-line arguments to be added to ‘mp-latex’. If non-nil, the argument will be added.

**Standard value:**

```
((quote ((-interaction=nonstopmode . t))))
```

**Type:**
- **alist**: key-type
  - (choice :tag other (string :tag other))
  - :value-type
    - (boolean :tag Activate :value nil)

**Options:**
- `-interaction=nonstopmode`
- `-shell-escape`
- `-8bit`
- `-interaction=errorstopmode`
- `-enc`
- `-etex`
- `-mltex`
- `-output-format=pdf`

4.4 mp-pdf-viewer

**Documentation:**
This is the command line/ shell command to view a .pdf file. It is used by the functions ‘mp-latex-pdf’ and ‘mp-latexmk’. The executable needs to be in your ‘exec-path’. Some useful URLs for downloads are given in the ‘custmomize’ help.

**Standard value:**
**Type:**

<table>
<thead>
<tr>
<th>radio</th>
</tr>
</thead>
<tbody>
<tr>
<td>(const :doc Default. Cross platform. evince)</td>
</tr>
<tr>
<td>(const :doc Alternative for Windows. sumatrapdf)</td>
</tr>
<tr>
<td>(const :doc Good for Linux/Ubuntu. xpdf)</td>
</tr>
<tr>
<td>(const :doc Adobe Acrobat Reader. acrord32)</td>
</tr>
<tr>
<td>(string :tag Enter an alternative program here. )</td>
</tr>
</tbody>
</table>

**Links:** Adobe Acrobat Reader

- xpdf
- sumatra
- evince

### 4.5 mp-preamble

**Documentation:**

This is the preamble for documents created with `mp-skeleton-nw`. It is an alist in the form (KEY1. (KEY2 . VALUE)). If KEY1 is non-nil, the list is included. KEY2 is a symbol indicating when to include. VALUE is a string.

The preamble inserted depends on the value of `mp-entwiner`. For example, when `mp-entwiner` is set to `knitr` all elements of the list with KEY1 non-nil and KEY2=‘knitr’ will be added to the preamble.

When editing this with `customize`, Use ‘C-j’ for carriage return

See `\TeX-doc` i.e. `(\TeX-doc packageName)` for details on particular packages.

**Standard value:**

```latex
((quote ((t class . \%%
  \documentclass{article}) (t knitr . \%%
  \%% modified from default setup for knitr
  \%%
  \usepackage[graphicx]
  \usepackage[color]
  \usepackage{framed}
  \%% recommended with 'knitr'
  \usepackage{alltt}
  \usepackage{mathtools}
```

37
Type:

alist
:tag

:key-type
(boolean :tag Activate :value nil)
:value-type
(cons :tag Cons-cell (choice :tag KEY2. Include in... (sexp
  :tag knitr :value knitr) (sexp :tag sweave :value sweave
) (sexp :tag org :value org) (sexp :tag all :value all)

39
4.6 mp-knitr-opts

Documentation:
Default setup options for 'knitr'.
This is passed to 'chunks' in 'knitr' by 'mp-R-nw-or-org'. This list is not
eexhaustive.
Common options are given as vectors with a choice indicated by the
index, in square brackets.

Standard value:

```r
library(knitr)
### Set global chunk options
opts_chunk$set(
  eval=TRUE,
  echo=TRUE,
  results=c('markup', 'asis', 'hold', 'hide')[1],
  collapse=FALSE,
  warning=TRUE, message=TRUE, error=TRUE,
  split=FALSE, include=TRUE, strip.white=TRUE,
  ## code decoration
  tidy=FALSE, prompt=FALSE, comment='##',
  highlight=TRUE, size='normalsize',
  background=c('#F7F7F7', colors()[479], c(0.1, 0.2, 0.3))[1],
  ## cache
  cache=FALSE,
  ## plots
  fig.path=c('figure', 'figure/minimal-'),
  fig.keep=c('high', 'none', 'all', 'first', 'last')[1],
  fig.align=c('center', 'left', 'right', 'default')[1],
  fig.show=c('hold', 'asis', 'animate', 'hide')[1],
  dev=c('pdf', 'png', 'tikz')[1],
  fig.width=7, fig.height=7, #inches
  fig.env=c('figure', 'marginfigure')[1],
  fig.pos=c('t', 'b', 'p', 'H')[1])
### Set R options
```
Type:

choice

(string :format %v :value )

Links: Code chunks and package options

Hooks - knitr documentation

4.7 mp-Sweave-opts

Documentation:
These options will be added to all code ‘chunks’ generated with ‘Sweave’. The value is used by ‘mp-insert-chunk’.
Options are given as a list of (KEY . VALUE) pairs. Non-nil for the VALUE means the argument will be added.
Only one plot per chunk is supported by ‘Sweave’. Manually changing to fig=TRUE in the .Rnw file generated is one simple approach to including plots.
Other possible options include:

• results= tex
  \hspace{1cm} – if set to ‘tex’, results will be read as \text{\LaTeX}

• echo=TRUE
  \hspace{1cm} – if =FALSE, no R code is included in output

• fig=FALSE
  \hspace{1cm} – if TRUE, a figure for graphics is included

• png=TRUE
- if =FALSE, no .png graphics are generated

• strip.white=false
  - If =all, all blank lines are removed;
  - If =true, blank lines removed from top and bottom

• width=6
• height=6
  - inches, for figures

• print=FALSE
  - If =TRUE, wrap all expressions in print()

• EPS=TRUE
  - If =FALSE, no EPS figures are produced. EPS figures are re-quired for \LaTeX but not PDF\LaTeX

• keep.source=FALSE
  - If =TRUE, do not deparse source before ‘echo’ing i.e. include original source ‘as-is’

• quiet=FALSE
  - If =TRUE, all progress messages are suppressed

• split=FALSE
  - If =TRUE, split over multiple files

• term=TRUE
  - If =FALSE, only output from print() and cat() is ‘echo’ed

**Standard value:**

```r
((quote ((results=verbatim . t) (results=tex) (echo=TRUE . t) (fig=FALSE . t) (png=TRUE . t) (strip.white=false . t) )))
```

**Type:**

42
alist
:keyword
(choice :tag other (string :tag other))
:valuetype
(boolean :tag Activate :value nil)

Options:
results=verbatim
results=tex
echo=TRUE
fig=FALSE
png=TRUE
strip.white=false
strip.white=all
strip.white=true
width=6
height=6
print=FALSE
EPS=TRUE
keep.source=FALSE
quiet=TRUE

4.8 mp-org-header-args

Documentation:
These options will be added to all code ‘chunks’ when ‘mp-entwiner’ is set to ‘Org’. The value is used by ‘mp-insert-chunk’.
Options are given as a list of (KEY1 . (KEY2 . VALUE)) pairs. Non-nil for KEY1 means the argument will be added.
For multiple plots per chunk, export the results to ‘x.pdf’ then in the .tex document change ‘includegraphics’ to ‘includepdf’ (from the \LaTeX package ‘pdfpages’).
There is a link to help on the customize screen.

Standard value:
((quote ((t :session . *session*) (t :exports . both) (t :results . output) (t :results . verbatim) (t :results . code)))))

Type:
alist
:key-type
(boolean :tag Activate :value nil)
:value-type

Links: orgmode manual - header arguments
4.9 mp-latex-font

Documentation:
This variable is used by ‘mp-skeleton-nw’.
It specifies the default font(s) to use when ‘mp-latex’ is set to ‘pdflatex’
(XeTeX) or ‘lualatex’.
Some common options are given in the form:

• serif

• sans-serif

• monospace font

Standard value:

```
((quote ((%%%
  %% Palatino family
  \usepackage[T1]{fontenc}
  \usepackage[mathpazo]
  \linespread{1.05}
  \usepackage[scaled]{helvet}
  \usepackage[courier] . t) (%%%
  %% Times family
  \usepackage[T1]{fontenc}
  \usepackage[mathptmx]
  \usepackage[scaled=.90]{helvet}
  \usepackage[courier]) (%%%
  %% Garamond family
  \usepackage[T1]{fontenc}
  \usepackage[urw-garamond]{mathdesign}
  \usepackage[lmodern]
  \usepackage[courier]
  \linespread{1.0609}) (%%%
  %% KP (Kepler) family; displays math
  \usepackage[T1]{fontenc}
  \usepackage[kpfonts]) (%%%
  %% Nimbus family
  \usepackage[T1]{fontenc}
  \usepackage[tgtermes]
  \usepackage[scale=.85]{tgtermes}
  \usepackage[tgcursor])))
```

Type: 45
4.10 mp-xetex-font

Documentation:

This variable is used by ‘mp-skeleton-nw’.
It specifies the default font(s) to use when ‘mp-latex’ is set to ‘xelatex’ (XeTeX) or ‘lualatex’.

• Some common commands include:
  – setmainfont
  – setsansfont
  – setmonofont

• Some common fonts and their \TeX Gyre equivalents include:

<table>
<thead>
<tr>
<th>Origin</th>
<th>\TeX Gyre</th>
</tr>
</thead>
<tbody>
<tr>
<td>Palatino</td>
<td>Pagella</td>
</tr>
<tr>
<td>Times</td>
<td>Termes</td>
</tr>
<tr>
<td>Helvetica</td>
<td>Heros</td>
</tr>
<tr>
<td>ITC Avant Garde Gothic</td>
<td>Adventor</td>
</tr>
<tr>
<td>New Century Schoolbook</td>
<td>Schola</td>
</tr>
</tbody>
</table>

• Common font options include:
  – Ligatures
    * Required/NoRequired,
    * Common/NoCommon,
    * Rare/Discretionary,
    * Historic, \TeX
  – Letters
Uppercase   SmallCaps   UppercaseSmallCaps
PetiteCaps  UppercasePetiteCaps   Unicase

- Numbers
  * Uppercase, Lowercase
  * Proportional, Monospaced
  * SlashedZero
  * Arabic

- Fractions
  * On, Alternate

- Style
  Alternate   Italic   Swash
  Historic   TitlingCaps

**Standard value:**

```latex
\usepackage[fontspec]{fontspec}
\defaultfontfeatures{Mapping=tex-text}
\setmainfont[ Ligatures={Rare, TeX, NoCommon},
             Numbers={Lowercase}]{Linux Libertine O}
\fontsize{12 pt}{16 pt}
\selectfont.
t))
```

**Type:**

alist
: key-type
(choice :tag other (string :tag other :format
Takes the form \setfont[options]{font} %v))
:value-type
(boolean :tag Activate :value nil)

**Links:** Free fonts at fontsquirrel
  Fonts spec documentation
4.11 mp-chunk-brackets

Documentation:

Brackets placed before and after each chunkName. Specify these in the form of a list of strings as follows: (openingForLaTeX closingForLaTeX openingForOrg closingForOrg)

This variable is used by ‘mp-insert-chunk’.

The names are specified in the corresponding .R file by ‘## —- chunkName’.

This may be used to enclose chunkName in a \TeX command e.g. ‘subsection{chunkName}’. The corresponding Org headline or level prefix, ‘* ’, is used when generating a skeleton .org file. Org uses up to 8 headline levels.

See also ‘org-level-faces’ and ‘org-heading-components’.

Standard value:

{{quote (\subsection{ } ** ))

Type:

choice :tag

(list :tag section (string :tag LaTeX :value \section{} ) (string :tag org :value *) (string :tag org :value ))
(list :tag subsection (string :tag LaTeX \subsection{} ) (string :tag org :value ** ) (string :tag org :value ))
(list :tag subsubsection (string :tag LaTeX \subsubsection{} ) (string :tag org :value *** ) (string :tag org :value ))
(list :tag paragraph/list item (string :tag LaTeX :value \paragraph{} ) (string :tag org :value **** ) (string :tag org :value ))
(list :tag no section (string :tag LaTeX :value ) (string :tag LaTeX :value ) (string :tag org :value ) (string :tag org :value ))
(list :tag other (string :tag LaTeX :value prefix ) (string :tag LaTeX :value suffix ) (string :tag org :value prefix) (string :tag org :value suffix))
4.12 mp-sweaveSty

Documentation:

The file ‘Sweave.sty’.

This is given in the form (KEY . VALUE) where VALUE is a string. If any KEY is t and if no such file is found in the ‘default-directory’, the first value which is t will be included in the working directory as ‘Sweave.sty’.

It is essential to have ‘Sweave.sty’ on your path when ‘mp-entwiner’ is set to ‘Sweave’. This variable is used by ‘mp-nw-tex’.

When editing this with ‘customize’, Use ‘C-j’ for carriage return.

Standard value:

```latex
\NeedsTeXFormat{LaTeX2e}
\ProvidesPackage{Sweave}{}
%%
\RequirePackage{ifthen}
\newboolean{Sweave@gin}
\setboolean{Sweave@gin}{true}
\newboolean{Sweave@ae}
\setboolean{Sweave@ae}{true}
%%
\DeclareOption{nogin}{\setboolean{Sweave@gin}{false}}
\DeclareOption{noae}{\setboolean{Sweave@ae}{false}}
\ProcessOptions
%%
\RequirePackage{graphicx,fancyvrb}
\IfFileExists{upquote.sty}{\RequirePackage{upquote}}{}
%%
\ifthenelse{\boolean{Sweave@gin}}{\setkeys{Gin}{width=0.8\textwidth}}{{%}
\ifthenelse{\boolean{Sweave@ae}}{{\RequirePackage[T1]{fontenc}}{{%}
\RequirePackage[ae]{T1}\RequirePackage{ae}
}}{{%}
}}

\DeclareVerbatimEnvironment{Sinput}{Verbatim}{fontshape=sl}
\DeclareVerbatimEnvironment{Soutput}{Verbatim}{}
\DeclareVerbatimEnvironment{Scode}{Verbatim}{fontshape=sl}
%%
\ifdefdefined\message{{\string Environment Schunk is already defined,}}
```
Type:

alist
:key-type
  (boolean :tag Activate :value nil)
:value-type
  (string :format %v :value )

4.13 mp-latexmkrc

Documentation:
The file ’.latexmkrc’.

This is given in the form (KEY . VALUE) where VALUE is a string. If any KEY is t and if no such file is found in the ‘default-directory’, the first value which is t will be included in the working directory as ’.latexmkrc’.

It is an initialization file or ’runcom’(commands) or for latexmk.

This variable is used by ‘mp-latexmk’. It is recommended to have this on your path. It supports the use of glossaries, acronyms and indices amongst others. No support is provided for the (now deprecated) ‘glossary’ package.

A link to the source is available as a link in the customize help.
When editing this with ‘customize’, Use ’C-j’ for carriage return.
Standard value:

```latex
\[ \text{(quote ((t .}
# Custom dependency for 'glossaries' package
add_cus_dep('glo', 'gls', 0, 'makeglo2gls');
sub makeglo2gls{
  system("makeindex -s
     \$_[0].ist" -t
     \$_[0].glg" -o
     \$_[0].gls"\n     \$_[0].glo" ");
}
# The 'glossaries' package, with the [acronym] option,
# produces a .acn file when processed with (xe/pdf)latex and
# then makeindex to process the .acn into .acr and
# finally runs of (xe/pdf)latex to read in the .acr file.
add_cus_dep('acn', 'acr', 0, 'makeacn2acr');
sub makeacn2acr{
  system("makeindex -s
     \$_[0].ist" -t
     \$_[0].alg" -o
     \$_[0].acr"\n     \$_[0].acn" ");
}
# Example of an added custom glossary type that is used
# in some of the 'glossaries' example files
# This is for the 'new glossary type' command
# \newglossary[ng][nlg]{notation}{not}{ntn}{Notation}
add_cus_dep('ntn', 'not', 0, 'makentn2not');
sub makentn2not{
  system("makeindex -s
     \$_[0].ist" -t
     \$_[0].nlg" -o
     \$_[0].not"\n     \$_[0].ntn" ");
}
# Dependencies for custom indexes using the 'index' package
add_cus_dep('adx', 'and', 0, 'makeadx2and');
sub makeadx2and{
  system("makeindex -o
     \$_[0].and"\n     \$_[0].adx" ");
}
```

51
add_cus_dep('ndx', 'nnd', 0, 'makendx2nnd');
sub makendx2nnd {
  system("makeindex -o \\
    \$_[0].nnd" \\
    \$_[0].ndx");
}
add_cus_dep('ldx', 'lnd', 0, 'makeldx2lnd');
sub makeldx2lnd{
  system("makeindex -o \\
    \$_[0].lnd" \\
    \$_[0].ldx");
}
# Custom dependency and function for 'nomencl' package
add_cus_dep('nlo', 'nls', 0, 'makenlo2nls');
sub makenlo2nls{
  system("makeindex -s nomencl.ist -o \\
    \$_[0].nls" \\
    \$_[0].nlo");
}
# Custom dependency and function(s) for 'epstopdf' package
# deletes an outdated pdf-image, and triggers a pdflatex-run:
add_cus_dep( 'eps', 'pdf', 0, 'cus_dep_delete_dest' );
# FOR USERS OF epstopdf v1.5 and later ONLY:
# load it as \usepackage[update,prepend]{epstopdf}
# detects an outdated pdf-image, and triggers a pdflatex-run
# Custom dependency to convert tif to png
add_cus_dep('eps', 'pdf', 0, 'cus_dep_require_primary_run');
add_cus_dep('tif', 'png', 0, 'maketif2png');
sub maketif2png{
  system("convert \\
    \$_[0].tif" \\
    \$_[0].png");
}
))))

Type:
alist :key-type 
  (boolean :tag Activate :value nil) :value-type 
  (string :format %v :value )
4.14  mp-upquoteSty

Documentation:

The file `upquote.sty'.

This is given in the form (KEY . VALUE) where VALUE is a string. If
any KEY is t and if no such file is found in the `default-directory', the first
value which is t will be included in the working directory as `upquote.sty'.

This variable is used by `mp-nw-tex'. This is favored by `knitr' and
`Sweave' to improving code display.

When editing this with `customize', Use 'C-j' for (carraige) return/ new-
line.

Standard value:

{(quote ((t .
%% This is file `upquote.sty',
%% generated with the docstrip utility.
%%
%% The original source files were:
%% upquote.dtx (with options: `package')
%%
%% Copyright (C) 2000 by Michael A. Covington
%% Copyright (C) 2003 by Frank Mittelbach
%% Copyright (C) 2012 by Markus Kuhn (current maintainer)
%%
%% Released under the LaTeX Project Public License v1.3c or
%% later
%% See http://www.latex-project.org/lppl.txt
%%
\NeedsTeXFormat{LaTeX2e}
\ProvidesPackage{upquote}
 [2012/04/19 v1.3 upright-quote and grave-accent glyphs in
 verbatim]
\newcommand\upquote@cmtt{cmtt}
\newcommand\upquote@OTone{OT1}
\ifx\encodingdefault\upquote@OTone\else\RequirePackage{textcomp}\fi
\else
4.15 mp-package-attributes

Documentation:

Keywords to search for in the initial comments of a package in an .el file.

These are used by ‘mp-el-tex’.

These are all read to the ‘end-of-line’.

Standard value:

```
((quote (Filename Copyright Author Maintainer Created
   Version Keywords Homepage Package-Version
   Package-Requires License URL Doc Keywords Compatibility)
))
```

Type:

sexp
5 Variables

5.1 mp-minor-mode-map

Documentation:
Defines the keymap for this minor mode.
The only keymap used by default is 'C-M-\l' for 'mp-mp'.

6 Additional code

6.1 require

(require (quote org))

6.2 define-minor-mode

(define-minor-mode mp-mode "
Define mp mode to make .pdfs.
Mp mode is a global minor mode.
It's LIGHTER (displayed on the mode line) is ' mp '." t "mp"
    mp-minor-mode-map :group (quote mp) :global t :version
    0.1 (message "mp mode toggled"))

6.3 provide

(provide (quote mp))