The rcs-multi Package

Martin Scharrer
martin@scharrer-online.de

http://www.ctan.org/pkg/rcs-multi

Version v0.1a – 2011/04/02

1 Introduction

This package allows to typeset version control (VC) information provided by RCS[1] keywords (e.g. `$Id: ... $`) in LaTeX documents which can contain of multiple `.tex` files included using `\include` or `\input`. It is based on the `svn-multi` which macros have been renamed and modified to support RCS.

This package reads the keywords of all files and provides the VC information of of the most recent changed file of the document to the user through a set of macros. This information is written to the auxiliary `.aux` file during the first LaTeX run and read back at the next which introduces the same delay known from the table of contents. The standard LaTeX switch `\nofiles` can be used to suppress the file generation.

1.1 Scope of Keywords

This package provides the RCS keyword data in two different scopes: document-global and file-local.

Document Global

The document global macros, like `\rcsrev` , return the latest version control information (keyword data) for the whole multi-file document, i.e. the information of the latest changed file of the document. To collect, sort and provide this information is the main functionality of this package.

Local to Current File

There are also file-local macros, e.g. `\rcsfilerev` , which return the version control information of the current file, i.e. the file they are used in. It is assumed here that every file using this macros calls first the macro `\rcsid` . See section 2.1 for more details about this macro. Please note that the file-local macros technically actually return the *last registered* information from the last `\rcsid` .

---

[1] RCS homepage: http://www.gnu.org/software/rcs/

2 Usage

The version control information are provided by RCS keywords which first need to be read in by dedicated macros and can then be typeset using different macros.

2.1 Including RCS Keywords

RCS keywords are included using `\rcsid` . This macro should be written very early in each file, i.e. in the preamble of the main document soon after `\documentclass` and `\usepackage{rcs-multi}` and as first in *every* subfile before an `\chapter` or similar macro. It does not create any output. See section 2.2 to learn how to typeset the keyword values.

```
\rcsid{$Id$}
\rcsid{$Header$}
```

The macro is for the `Id` keyword and must be written like shown. A trailing colon with or without spaces after the keyword name (‘`Id`’) is also valid but **everything else** except a valid RCS string will cause a TeX parse error. The difference between `Id` and `Header` is that the latter includes the full URL/path and not only the filename.

```
\rcs{$⟨keyword⟩$}
```

This macro let you typeset rcs keywords directly. The dollars will be stripped and the rest is typeset as normal text.

```
\rcskwsave{$⟨keyword⟩$}
```

This macro lets you include and save any keyword you like. The keyword can be already expanded or not (no value and only “`:`” or nothing after the key name). This macro is also used internally and does not create any output. Please note that the argument is read verbatim and that there should be no space between the macro and the argument’s left brace.

2.2 Typesetting the Keyword Values

The following macros can be used to typeset the keyword values anywhere in the document. Please note that not all LaTeX fonts have all special characters, e.g. ‘`_`’ is not provided in the standard roman font. To proper typeset file names and URLs containing these letters you can use either teletype font (`\texttt`) or use `{\urlstyle{rm}\rcsnolinkurl{...}}` which requires the `hyperref` package.

```
\rcsrev
\rcsdate
\rcsauthor
```

These macros hold the keyword values of the whole document, i.e. of the most recent revision. They can be used everywhere in every file of the LaTeX document, after `\usepackage{rcs-multi}` of course. Please see section 2.3 how to typeset parts of the date.

`\rcsfilerev`
`\rcsfiledate`
`\rcsfileauthor`

These macros hold the keyword values of the current LaTeX file, but only if it contains a `\rcsid` or `\rcsidlong` macro. Otherwise the macros hold either zero values or the values of the last file dependent on whether an option is enabled which enabled the `fink` package. Please see section 2.3 how to typeset parts of the date. See `\rcskw` below for all other keywords.

`\rcsmainfilename`

The macro `\rcsmainfilename` hold the filename of the main LaTeXfile. It can be used to typeset this information anywhere in the document which might be more descriptive as the name of the current file (which can be typeset with `\rcskw {HeadURL}` or `\rcskw {Filename}` after `\rcsid` or `\rcsidlong`, respectively).

`\rcssetmainfile`

This will declare the current file as the main LaTeX file by defining the above macros. It will automatically be called at the end of the preamble so the user normally doesn't have to use it by him- or herself as long it isn't needed in the preamble.

Please note that this macro changes the definition of `\rcsmainfilename` directly without going over the auxiliary file. Calling it in several files will make this two macros inconsistent.

`\rcskw{`⟨*keyword name*⟩`}`

All keywords saved with `\rcsid` or `\rcskwsave` can be typeset by this macro which is a holdover from a very early version of this package when multiple files where not supported. It takes one argument which must be a RCS keyword name. It then returns the current value of this keyword or nothing (`\relax`) when the keyword was not set yet. Examples:

```
\textsl{Revision: \rcskw{Revision}}
URL: \url{\rcskw{HeadURL}}
```

In the second example `\url` (`hyperref` package) is used to add a hyperlink and to avoid problems with underscores (_) inside the URL. `rcsmulti` is also providing a macro `\rcsnolinkurl` which works like `\url` but doesn't adds an hyperlink. See the description of this macro for more details.

If the given keyword doesn't exists a package warning is given to allow spelling errors to be tracked down. This doesn't work well when `\rcskw` is used inside `\url`. In this case the warning code will be typeset(!) verbatim into the document by `\url`.

`\rcskwdef{`⟨*keyword name*⟩`}{`⟨*value*⟩`}`

This macro is used to define the keyword values. This is normally only called internally but could be used by the user to override single keywords. The values can then be typeset by `\rcskw`. Note that this macro has no influence on the calculation of the latest revision.

2.3 Accessing Date Values

```
\rcsyear     \rcsfileyear
\rcsmonth    \rcsfilemonth
\rcsday      \rcsfileday
\rcshour     \rcsfilehour
\rcsminute   \rcsfileminute
\rcssecond   \rcsfilesecond
```

Whenever the date information is read, i.e. by `\rcskwsave` `{Date}` or `\rcsid` , the following macros are set to the appropriate date parts for the current file (the `\rcsfile...` versions) and for the whole document.

```
\rcstime
\rcsfiletime
```

This macros return the time part of the date only and simply return the corresponding hour, minute and second macros with a colon as separator.

```
\rcspdfdate
```

Returns the last changed date of the whole document in a format needed for `\pdfinfo`. Can be used like this:

```
\pdfinfo{ /CreationDate (D:\rcspdfdate) }
```

to set the PDF creation date to the last changed date if you use `pdflatex` to compile your LaTeX document.

```
\rcstoday
\rcsfiletoday
```

These macros typeset the document-global or current-file, respectively, using the format of `\today` which depends on the used language. To adjust the language of your document use the `babel` package.

2.4 Using Full Author Names

If you like to have the full author[2] names, not only the usernames, in your document you can use the following macros. First you have to register all authors of the document with `\rcsRegisterAuthor` and then you can write e.g. `\rcsFullAuthor{\rcsauthor}` or `\rcsFullAuthor{\rcsfileauthor}`.

```
\rcsRegisterAuthor{⟨author⟩}{⟨full name⟩}
```

This macro registers *⟨full name⟩* as full name for *⟨author⟩* (a RCS username) for later use with `\rcsFullAuthor` .

---

[2] This means RCS authors, e.g. the persons who commit changes into the rcs repository.

```
\rcsFullAuthor{⟨author name or macror⟩}
\rcsFullAuthor*{⟨author name or macro⟩}
```

Takes the username as argument and returns the full name if it was registered first with `\rcsRegisterAuthor`, otherwise it returns the given username. The star version returns the username in parentheses after the full name. This is normally used in one of the following forms:

```
\rcsFullAuthor{\rcsauthor}
\rcsFullAuthor{\rcsfileauthor}
```

2.5 Using Full Revision Names

Like the author's also revision names/tags can be registered and used later. These macros were implemented on user request and have the drawback that you have to guess the next revision number of your document in order to get correct results when you like to tag the to-be-checked-in revision. Please note that this has nothing to do with the normal tagging.

```
\rcsRegisterRevision{⟨revision number⟩}{⟨tag name⟩}
```

This registers *⟨tag name⟩* as tag name for *⟨revision number⟩* for later use with `\rcsFullRevision` .

```
\rcsFullRevision{⟨revision number or macro⟩}
\rcsFullRevision*{⟨revision number or macro⟩}
```

Takes a revision number coming from a macro like `\rcsrev` , `\rcsfilerev` or a number as argument and returns the full name if it was registered first with `\rcsRegisterRevision` , otherwise it returns “Revision *⟨revision number⟩*”. The star version returns also the revision number leaded by ‘r’ in parentheses after the tag name, e.g. `Name (1.2)`.

2.6 Verbatim URLs with and without Hyperlinks

```
\rcsnolinkurl{⟨macro with returns special text⟩}
```

This macro allows you to write `\rcsnolinkurl{\rcskw{HeadURL}}` and get the Head URL typeset verbatim. However `\url{\rcskw {HeadURL}}` (`hyperref` package) gives you the same result with a hyperlink. Both macros require the `hyperref` package which is not automatically loaded by `rcsmulti`. Please load it manually when you like to use `\rcsnolinkurl` .

Please note that you can't use `hyperref`'s `\nolinkurl` because it won't expand `\rcskw` .

3 Implementation

3.1 Package Header

```
\NeedsTeXFormat{LaTeX2e}[1999/12/01]
\ProvidesPackage{rcs-multi}
 [\filedate\space\fileversion\space RCS Keywords for multi-file LaTeX documents]
```

3.2 General Internal Macros

Some internal used macro which don't fit in any other section.

`\rcs@ifempty`

#1: string

Tests if the given argument is empty. If so the first of the next two token will be expanded, the second one otherwise.

```
\def\rcs@ifempty#1{%
  \begingroup
  \edef\rcs@temp{#1}%
  \ifx\rcs@temp\empty
    \endgroup
    \expandafter
    \@firstoftwo
  \else
    \endgroup
    \expandafter
    \@secondoftwo
  \fi
}
```

`\rcs@ifequal`

#1: string a
#2: string b

Tests if the given arguments are identical, e.g. same strings. If so the first of the next two token will be expanded, the second one otherwise.

```
\def\rcs@ifequal#1#2{%
  \begingroup
  \edef\rcs@stringa{#1}%
  \edef\rcs@stringb{#2}%
  \ifx\rcs@stringa\rcs@stringb
    \endgroup
    \expandafter
    \@firstoftwo
  \else
```

```
    \endgroup
    \expandafter
    \@secondoftwo
  \fi
}
```

\rcs@ifvalidrev

#1: macro name

Checks if the given macro (by name) is a valid revision, i.e. defined and greater than zero.

```
\def\rcs@ifvalidrev#1{%
  \begingroup
  \@ifundefined{#1}%
    {\def\rcs@temp{-1}}%
    {\expandafter\edef
     \expandafter\rcs@temp\expandafter{\csname #1\endcsname}}%
  \ifnum\rcs@temp>-1\relax
    \endgroup
    \expandafter
    \@firstoftwo
  \else
    \endgroup
    \expandafter
    \@secondoftwo
  \fi
}
```

3.3 Definition of init values

```
% Init values
\def\rcsrev{0.0}            \def\@rcs@rev{0.0}
\def\rcsdate{}              \def\@rcs@date{}
\def\rcsauthor{}            \def\@rcs@author{}
\def\rcsyear{0000}          \def\@rcs@year{0000}
\def\rcsmonth{00}           \def\@rcs@month{00}
\def\rcsday{00}             \def\@rcs@day{00}
\def\rcshour{00}            \def\@rcs@hour{00}
\def\rcsminute{00}          \def\@rcs@minute{00}
\def\rcssecond{00}          \def\@rcs@second{00}
\def\rcsname{}              \def\@rcs@name{}
\def\rcsurl{}               \def\@rcs@url{}
\def\rcsmainfilename{}
\def\rcsmainurl{\rcsmainfilename}
\def\rcs@temp{}
\def\rcs@lastkw{}
```

```

\def\rcsfilerev{0.0}
\def\rcsfiledate{}
\def\rcsfileauthor{}
\def\rcsfileyear{0000}
\def\rcsfilemonth{00}
\def\rcsfileday{00}
\def\rcsfilehour{00}
\def\rcsfileminute{00}
\def\rcsfilesecond{00}
\def\rcsfileurl{}
\def\rcsfilename{}
```

3.4 Time and *Today* macros

`\rcstime`

`\rcsfiletime`

This macros simple use the hour, minute and second macros.

```
\def\rcstime{\rcshour:\rcsminute:\rcssecond}
\def\rcsfiletime{\rcsfilehour:\rcsfileminute:\rcsfilesecond}
```

These macros use the `\today` macro to typeset the current date using the local language settings. Thanks and credit goes to Manuel Pégourié-Gonnard for suggesting this feature and for providing some code.

`\rcstoday`

```
\newcommand*{\rcstoday}{%
  \begingroup
    \year\rcsyear \month\rcsmonth \day\rcsday
    \relax \today
  \endgroup
}
```

`\rcsfiletoday`

```
\newcommand*{\rcsfiletoday}{%
  \begingroup
    \year\rcsfileyear \month\rcsfilemonth \day\rcsfileday
    \relax \today
  \endgroup
}
```

3.5 Id macro

`\rcsid`

Calls `\rcskwsave` with `\@rcsidswtrue` so that the Id keyword will be parsed at the end of `\rcskwsave` .

```
\newcommand*{\rcsid}{%
  \@rcsidswtrue
  \rcskwsave
}
\newif\if@rcsidsw
\@rcsidswfalse
```

`\rcs@scanId`

#1: file name
#2: revision
#3: date (YYYY-MM-DD)
#4: time (HH:MM:SS)
#5: author (username)
#6: rest

Scans rcs Id (after it got parsed by `\rcskwsave`). Awaits only Id value without leading '`Id:`' and a trailing `\relax` as end marker. It calls `\@rcs@scandate` to extract the date information and `\@rcs@updateid` to update global Id values and also sets the appropriate keywords.

```
\def\rcs@scanId#1,v #2 #3 #4 #5 #6\relax{%
  \@rcs@scandate{#3 #4}%
  \@rcs@updateid{#2}{#3 #4}{#5}{#1}%
  \rcskwdef{Filename}{#1}%
  \rcskwdef{Date}{#3 #4}%
  \rcskwdef{Revision}{#2}%
  \rcskwdef{Author}{#5}%
}
```

`\@rcs@updateid`

#1: rev
#2: date
#3: author (username)
#4: url

We first define the expanded arguments to variables for the user. The expansion is needed because the arguments content is mostly generic like `\rcs@value` which can change very soon after this macro.

```
\def\@rcs@updateid#1#2#3#4{%
  \xdef\rcsfilerev{#1}%
  \xdef\rcsfiledate{#2}%
  \xdef\rcsfileauthor{#3}%
  \xdef\rcsfileurl{#4}%
  \rcs@getfilename\rcsfileurl
```

Then we check if the revision is non-empty (not yet expanded by RCS?) and larger then the current maximum value `\@rcs@rev`. If yes we save all value to save them in the .aux-file later.

```
  \ifx\rcsfiledate\empty\else
    \begingroup
      \edef\@tempa{\@rcs@year\@rcs@month\@rcs@day}
      \edef\@tempb{\rcsfileyear\rcsfilemonth\↙
         rcsfileday}
      \ifnum\@tempa<\@tempb
        \rcs@update
      \else\ifnum\@tempa=\@tempb
        \edef\@tempa{\@rcs@hour\@rcs@minute\↙
           @rcs@second}
        \edef\@tempb{\rcsfilehour\rcsfileminute\↙
           rcsfilesecond}
        \ifnum\@tempa<\@tempb
          \rcs@update
        \fi
      \fi\fi
    \endgroup
  \fi
}
```

`\rcs@updateid`

Updates the max-hold macros with the values of the current file VC information.

```
\def\rcs@update{%
  \xdef\@rcs@rev{\rcsfilerev}%
  \xdef\@rcs@date{\rcsfiledate}%
  \xdef\@rcs@author{\rcsfileauthor}%
  \xdef\@rcs@year{\rcsfileyear}%
  \xdef\@rcs@month{\rcsfilemonth}%
  \xdef\@rcs@day{\rcsfileday}%
  \xdef\@rcs@hour{\rcsfilehour}%
  \xdef\@rcs@minute{\rcsfileminute}%
  \xdef\@rcs@second{\rcsfilesecond}%
  \xdef\@rcs@name{\rcsfilename}%
  \xdef\@rcs@url{\rcsfileurl}%
}
```

`\rcs@catcodes`

Changes all TEX-special character to category “other”. The newline aka return is changed to category “ignore” so line breaks are not taken as part of the verbatim arguments.

```
\def\rcs@catcodes{%
  \let\do\@makeother
  \dospecials
  \catcode‘\^^M9
  \catcode‘\ 10
  \catcode‘\{1
  \catcode‘\}2
}
```

`\rcs@gdefverb`

#1: macro

```
\def\rcs@gdefverb#1{%
  \begingroup
    \def\rcs@temp{#1}%
    \begingroup
      \rcs@catcodes
      \rcs@gdefverb@
}
```

`\rcs@defverb@`

#1: verbatim stuff

```
\def\rcs@gdefverb@#1{%
    \endgroup
    \expandafter\gdef\rcs@temp{#1}%
  \endgroup
}
```

3.6 Keyword Macros

`\rcskwsave`

Enabled verbatim mode and uses a sub macro to read the arguments afterwards.

```
\def\rcskwsave{%
  \begingroup
    \rcs@catcodes
    \rcskwsave@readargs
}
```

`\rcskwsave@readargs`

#1: $kw: value$

Reads full argument, calls parse submacro and ends catcode changes. If `\rcskwsave` was called by `\rcsid` scans the id keyword by calling the scan macro.

```
\gdef\rcskwsave@readargs#1{%
    \rcskwsave@read#1\relax
  \endgroup
  \if@rcsidsw
    \rcs@ifequal{\rcs@lastkw}{Id}%
      {\ifx\rcskwId\empty\else
        \expandafter
        \rcs@scanId\rcskwId\relax
        \@rcsidswfalse
      \fi}{%
    \rcs@ifequal{\rcs@lastkw}{Header}%
      {\ifx\rcskwHeader\empty\else
        \expandafter
        \rcs@scanId\rcskwHeader\relax
        \@rcsidswfalse
        \fi}{}%
      }%
  \fi
  \ignorespaces
}
```

`\rcskwsave@read`

#1: keyword line without surrounding $ $

Reads the full keyword and strips the dollars.

```
\begingroup
\catcode‘\$=12
\gdef\rcskwsave@read $#1$\relax{%
  \rcs@checkcolon#1:\relax
}
\endgroup
```

`\rcskwsave@parse`

#1: key

#2: value

Parse the keyword and save it away.

```
\begingroup
\catcode‘\$=11
\gdef\rcskwsave@parse$#1:#2${%
  \expandafter\xdef\csname rcskw#1\endcsname{#2}%
}%
\endgroup
```

`\rcskwdef`

#1: key
#2: value

```
\newcommand{\rcskwdef}[2]{%
  \gdef\rcs@lastkw{#1}%
  \expandafter\xdef\csname rcskw#1\endcsname{#2}%
}
```

Example: `\rcskwdef{Rev}{2.3}` will define `\rcskwRev` as '`2.3`'.

We define default values for normal keywords. Keyword `Filename` is the name given by `Id` and not a real keyword.

```
\rcskwdef{Rev}{0.0}
\rcskwdef{Date}{}
\rcskwdef{Author}{}
\rcskwdef{Filename}{}
\rcskwdef{HeadURL}{}
```

`\rcskw`

#1: keyword name

Macro to get keyword value. Just calls **`\rcskw`** ⟨*ARGUMENT*⟩ where the argument interpreted as text. So e.g. `\rcskw{Date}` is the same as `rcskwDate` but this could be changed later so always use this interface to get the keyword values.

```
\newcommand{\rcskw}[1]{%
  \@ifundefined{rcskw#1}%
    {\PackageWarning{rcs-multi}{RCS keyword '#1' not defined (typo?)}}%
    {\csname rcskw#1\endcsname}%
}%
```

3.7 Keyword check and strip macros

The following macros are used to test whether the given keywords are fully expanded or not. RCS supports unexpanded keywords as input with or without colon and with or without trailing space(s), i.e. a: `$KW$`, b: `$KW:$` or c: `$KW: $`. To avoid LaTeX syntax errors in this pre-commit state the keyword is checked by the following macros. Unexpanded keywords result in an empty value. Also leading and trailing spaces are removed.

`\rcs@checkcolon`

#1: key
#2: potential value, might be empty

Checks if the keyword contains a colon. It is called by `\rcskwsave@read` with a trailing `:\relax` so that #2 will be empty if there is no earlier colon or will hold the value with this trailing colon otherwise. The first case means that the keyword is unexpanded without colon (case a) which leads to an empty value. In the second case `\rcs@stripcolon` is called to strip the colon and surrounding spaces. The final value is returned by `\rcs@value`.

```
\def\rcs@checkcolon#1:#2\relax{%
  \rcs@ifempty{#2}%
    {\rcskwdef{#1}{}}%
    {\rcs@stripcolon#2\relax\rcskwdef{#1}{\rcs@value}}%
}
```

`\rcs@stripcolon`

#1: potential value

Strips the previous added colon (for `\rcs@checkcolon`). The remaining argument is checked if it's empty (case b) or only a space (case c). Otherwise the keyword is expanded and `\rcs@stripspace` is called to strip the spaces.

```
\def\rcs@stripcolon#1:\relax{%
  \rcs@ifempty{#1}%
    {\gdef\rcs@value{}}%
    {\rcs@ifequal{#1}{ }%
      {\gdef\rcs@value{}}%
      {\rcs@stripspace#1\relax\relax}%
    }%
}
```

`\rcs@stripspace`

#1: first character
#2: rest of string

Strips leading space if present and calls `\rcs@striptrailingspace` to strip the trailing space.

```
\def\rcs@stripspace#1#2\relax{%
  \rcs@ifequal{#1}{ }%
    {\gdef\rcs@value{#2}}%
    {\rcs@striptrailingspace#1#2\relax}%
}
```

`\rcs@striptrailingspace`

#1: string

Strips trailing space using the macros parameter text. Must be called with `\relax` as end marker.

```
\def\rcs@striptrailingspace#1 \relax{%
  \gdef\rcs@value{#1}%
}
```

3.8 Date Macros

`\@rcs@scandate`

#1: date

Scans data information in Id keyword and saves them in macros.

```
\def\@rcs@scandate#1{\@rcs@scandate@#1\empty\relax}

\def\@rcs@scandate@#1/#2/#3 #4:#5:#6#7#8\relax{%
  \gdef\rcsfileyear{#1}%
  \gdef\rcsfilemonth{#2}%
  \gdef\rcsfileday{#3}%
  \gdef\rcsfilehour{#4}%
  \gdef\rcsfileminute{#5}%
  \gdef\rcsfilesecond{#6#7}%
}
```

`\rcspdfdate`

Returns date in a format needed for `\pdfinfo`.

```
\def\rcspdfdate{%
  \rcsyear\rcsmonth\rcsday
  \rcshour\rcsminute\rcssecond00'00'%
}
```

3.9 Mainfile Makros

`\rcssetmainfile`

Saves the current filename and URL to macros. Will be called automatically in the preamble.

```
\newcommand{\rcssetmainfile}{%
  \xdef\rcsmainfilename{\rcsfilename}%
  \xdef\rcsmainfileurl{\rcsfileurl}%
}
\AtBeginDocument{\rcssetmainfile}
```

3.10 Register and FullName Macros

`\rcsRegisterAuthor`

#1: author username
#2: Full Name
Saves the author's name by defining `rcs@author@`⟨*username*⟩ to it.

```
\newcommand{\rcsRegisterAuthor}[2]{%
  \expandafter\def\csname rcs@author@#1\endcsname{#2}%
}
```

`\rcsFullAuthor`

`\rcsFullAuthor*`

We test if the starred or the normal version is used and call the appropriate submacro `rcsFullAuthor@star` or `rcsFullAuthor@normal`.

```
\newcommand{\rcsFullAuthor}{%
  \@ifnextchar{*}%
    {\rcsFullAuthor@star}%
    {\rcsFullAuthor@normal}%
}%
```

`\rcsFullAuthor@star`

#1: username
Both submacros are calling `rcsFullAuthor@` but with different arguments. The star macro also removes the star of course.

```
\def\rcsFullAuthor@star*#1{%
  \edef\rcs@temp{#1}%
  \rcsFullAuthor@{\rcs@temp}{~(\rcs@temp)}%
}%
```

`\rcsFullAuthor@normal`

#1: username

```
\def\rcsFullAuthor@normal#1{%
  \edef\rcs@temp{#1}%
  \rcsFullAuthor@{\rcs@temp}{}%
}%
```

`\rcsFullAuthor@`

#1: username
#2: previous defined trailing string

`rcsFullAuthor@` now sets the author's full name. Note that #2 is empty when the normal version is called.

```
\def\rcsFullAuthor@#1#2{%
  \@ifundefined{rcs@author@#1}%
    {#1}%
    {\csname rcs@author@#1\endcsname #2}%
}
```

`\rcsRegisterRevision`

#1: revision number
#2: tag name

Saves the revision's name or tag by defining `rcs@revision@`⟨*revisionnumber*⟩ to it.

```
\newcommand{\rcsRegisterRevision}[2]{%
  \expandafter\def\csname rcs@revision@#1\endcsname{#2}%
}
```

`\rcsFullRevision`

`\rcsFullRevision*`

We test if the starred or the normal version is used and call the appropriate submacro `rcsFullRevision@star` or `rcsFullRevision@normal`.

```
\newcommand{\rcsFullRevision}{%
  \@ifnextchar{*}%
    {\rcsFullRevision@star}%
    {\rcsFullRevision@normal}%
}
```

`\rcsFullRevision@star`

#1: revision number

Both submacros are calling `rcsFullRevision@` but with different arguments. The star macro also removes the star of course.

```
\def\rcsFullRevision@star*#1{%
  \edef\rcs@temp{#1}%
  \rcsFullRevision@{\rcs@temp}{~(r\rcs@temp)}%
}
```

`\rcsFullRevision@normal`

#1: revision number

```
\def\rcsFullRevision@normal#1{%
  \edef\rcs@temp{#1}%
  \rcsFullRevision@{\rcs@temp}{}%
}
```

`\rcsFullRevision@`

#1: revision number
#2: previous defined trailing string

`rcsFullRevision@` now sets the revision name. Note that #2 is empty when the normal version is called.

```
\def\rcsFullRevision@#1#2{%
  \@ifundefined{rcs@revision@#1}%
    {Revision #1}%
    {\csname rcs@revision@#1\endcsname #2}%
}
```

3.11 Other macros

This section contains macros which don't fit in any other section.

`\rcs`

Strips the `$ $` around the keyword. A space must be before the final dollar.

```
\providecommand{\rcs}[1]{\@revs#1}
\def\@rcs$#1 ${#1}
```

`\rcsnolinkurl`

#1: URL

This code is taken from the `hyperref` package and is the definition of `\url` just without the part which creates the actual hyperlink. This needs of course the `hyperref` package. A warning is given if it isn't loaded.

```
%% Adapted from the \url macro of the `hyperref` package.
\DeclareRobustCommand*{\rcsnolinkurl}{%
  \@ifundefined{hyper@normalise}%
    {\PackageWarning{rcs-multi}{Package hyperref is needed for \noexpand
     \rcsnolinkurl.}}%
    {\hyper@normalise\rcsnolinkurl@}%
}%
\def\rcsnolinkurl@#1{\Hurl{#1}}%
```

`\rcs@getfilename`

#1: URL
This macro expands the content using the temporary macro and sets it in front of the `\rcs@getfilename` sub-macro together with `/{}` to make sure the macro does not break at values without directories. A `\relax` is used as end marker.

```
\def\rcs@getfilename#1{%
  \begingroup
    \edef\rcs@temp{#1}%
    \expandafter\@rcs@getfilename\rcs@temp/{}\relax
}%
```

`\@rcs@getfilename`

#1: URL part before first slash
#2: URL part after first slash
Splits the content at the first slash (/) and checks if the remainder is empty. If so the end marker got reached and the part before the slash is the filename which is returned. Otherwise the macro recursively calls itself to split the remainder.

```
\def\@rcs@getfilename#1/#2\relax{%
    \rcs@ifempty{#2}%
      {\endgroup\gdef\rcsfilename{#1}}%
      {\@rcs@getfilename#2\relax}%
}%
```

3.12 Write to Auxiliary file

`\rcs@writeaux`

This macro writes the `.aux` auxiliary file and is called from a `\AtEndDocument` macro later on.

```
\def\rcs@writeaux{%
    \immediate\write\@mainaux{^^J%
      \noexpand\gdef\noexpand\rcsrev{\@rcs@rev}^^J%
      \noexpand\gdef\noexpand\rcsdate{\@rcs@date}^^J%
      \noexpand\gdef\noexpand\rcsauthor{\@rcs@author↙
         }^^J%
      \noexpand\gdef\noexpand\rcsyear{\@rcs@year}^^J%
      \noexpand\gdef\noexpand\rcsmonth{\@rcs@month}^^↙
         J%
      \noexpand\gdef\noexpand\rcsday{\@rcs@day}^^J%
      \noexpand\gdef\noexpand\rcshour{\@rcs@hour}^^J%
      \noexpand\gdef\noexpand\rcsminute{\@rcs@minute↙
         }^^J%
      \noexpand\gdef\noexpand\rcssecond{\@rcs@second}^^J%
      \noexpand\rcs@gdefverb\noexpand\rcsname{\@rcs@name}^^J%
      \noexpand\rcs@gdefverb\noexpand\rcsurl{\@rcs@url}^^J%
    }%
}
```

At the end of document the values are written to the auxiliary file.

```
\AtEndDocument{%
  \if@filesw
    \ifx\@rcs@date\empty\else
      \rcs@writeaux
    \fi
  \fi
}
```