

# Package ‘Rd2roxygen’

January 17, 2026

**Type** Package

**Title** Convert Rd to 'Roxygen' Documentation

**Version** 1.18

**Imports** roxygen2 (>= 4.0.0), xfun (>= 0.55), formatR (>= 1.0)

**Suggests** litedown (>= 0.9)

**Description** Functions to convert Rd to 'roxygen' documentation. It can parse an Rd file to a list, create the 'roxygen' documentation and update the original R script (e.g. the one containing the definition of the function) accordingly. This package also provides utilities that can help developers build packages using 'roxygen' more easily. The 'formatR' package can be used to reformat the R code in the examples sections so that the code will be more readable.

**License** GPL

**URL** <https://github.com/yihui/Rd2roxygen>

**BugReports** <https://github.com/yihui/Rd2roxygen/issues>

**VignetteBuilder** litedown

**RoxygenNote** 7.3.3

**Encoding** UTF-8

**NeedsCompilation** no

**Author** Hadley Wickham [aut],  
Yihui Xie [aut, cre] (ORCID: <<https://orcid.org/0000-0003-0645-5666>>)

**Maintainer** Yihui Xie <[xie@yihui.name](mailto:xie@yihui.name)>

**Repository** CRAN

**Date/Publication** 2026-01-17 05:00:02 UTC

## Contents

Rd2roxygen-package . . . . .	2
create_roxygen . . . . .	3
parse_and_save . . . . .	3

parse_file . . . . .	4
Rd2roxygen . . . . .	5
reformat_code . . . . .	6
roxygen_and_build . . . . .	7

<b>Index</b>	<b>9</b>
--------------	----------

---

Rd2roxygen-package	<i>Convert Rd to roxygen documentation and utilities to enhance documentation and building packages</i>
--------------------	---

---

## Description

This package contains functions to convert Rd to roxygen documentation. It can parse an Rd file to a list ([parse\\_file](#)), create the roxygen documentation ([create\\_roxygen](#)) and update the original R script (e.g. the one containing the definition of the function) accordingly ([Rd2roxygen](#)). This package also provides utilities which can help developers build packages using roxygen more easily ([rab](#)).

## Note

There is no guarantee to generate perfect roxygen comments that can be converted back to the original Rd files. Usually manual manipulations on the roxygen comments are required. For example, currently ‘@S3method’ is not included in the comments, and ‘@rdname’ is not supported either (users have to move the roxygen comments around and add the appropriate tags by themselves). Patches (as pull requests) are welcome through GitHub: <https://github.com/yihui/Rd2roxygen/>.

This package is not thoroughly tested, so it is likely that it fails to convert certain parts of Rd files to roxygen comments. As mentioned before, you have to manually deal with these problems. You are welcome to report other serious issues via <https://github.com/yihui/Rd2roxygen/issues>.

## Author(s)

Hadley Wickham and Yihui Xie

## See Also

Useful links:

- <https://github.com/yihui/Rd2roxygen>
- Report bugs at <https://github.com/yihui/Rd2roxygen/issues>

## Examples

```
## see the package vignette: vignette('Rd2roxygen')
```

---

create_roxygen	<i>Create the roxygen documentation</i>
----------------	---

---

**Description**

The parsed information is converted to a vector of roxygen tags.

**Usage**

```
create_roxygen(info, usage = FALSE)
```

**Arguments**

info	the named list of the parsed documentation
usage	logical: whether to include the usage section in the output (this can be useful when there are multiple functions in a single usage section, but generally it is not necessary because roxygen can generate the usage section automatically)

**Value**

a character vector

**Author(s)**

Hadley Wickham; modified by Yihui Xie <<https://yihui.org>>

**Examples**

```
rd.file = system.file("examples", "parse_and_save.Rd", package = "Rd2roxygen")
options(roxygen.comment = "##' ")
create_roxygen(parse_file(rd.file))
```

---

parse_and_save	<i>Parse the input Rd file and save the roxygen documentation into a file</i>
----------------	---

---

**Description**

Parse the input Rd file and save the roxygen documentation into a file

**Usage**

```
parse_and_save(path, file, usage = FALSE)
```

**Arguments**

path	the path of the Rd file
file	the path to save the roxygen documentation
usage	logical: whether to include the usage section in the output

**Value**

a character vector if file is not specified, or write the vector into a file

**Author(s)**

Hadley Wickham; modified by Yihui Xie <<https://yihui.org>>

---

parse_file	<i>Parse the input Rd file to a list</i>
------------	--

---

**Description**

This function uses the function parse\_Rd in the **tools** package to parse the Rd file.

**Usage**

```
parse_file(path)
```

**Arguments**

path	the path of the Rd file
------	-------------------------

**Value**

a named list containing the documentation sections as strings

**Author(s)**

Hadley Wickham; modified by Yihui Xie <<https://yihui.org>>

**Examples**

```
rd.file = system.file("examples", "parse_and_save.Rd", package = "Rd2roxygen")
parse_file(rd.file)
```

---

Rd2roxygen*Convert all the Rd files of a package to roxygen comments*

---

**Description**

This function takes a package root directory, parses all its Rd files under the man directory and update the corresponding R source code by inserting roxygen documentation in to the R scripts.

**Usage**

```
Rd2roxygen(pkg, nomatch, usage = FALSE)
```

**Arguments**

pkg	the root directory of the package
nomatch	the file name (base name only) to use when an object in the Rd file is not found in any R source files (typically this happens to the data documentation); if not specified, the default will be 'pkg'-package.R
usage	logical: whether to include the usage section in the output

**Value**

NULL (but the process of conversion will be printed on screen)

**Note**

ESS users may use `options(roxygen.comment = "##' ")` to ensure the generated roxygen comments begin with "##' ", which is the default setting in Emacs/ESS.

Re-run this function on a package will remove the previous roxygen comments before functions in R scripts.

**Author(s)**

Yihui Xie <<https://yihui.org>>

**Examples**

```
## a demo package
pkg = system.file("examples", "pkgDemo", package = "Rd2roxygen")
file.copy(pkg, tempdir(), recursive = TRUE) # copy to temp dir first
od = setwd(tempdir())

## take a look at original R scripts
file.show("pkgDemo/R/foo.R")

options(roxygen.comment = "##' ")

## convert Rd's under man to roxygen comments
```

```
Rd2roxygen(file.path(tempdir(), "pkgDemo"))

file.show("pkgDemo/R/foo.R") # what happened to foo.R and bar.R?

setwd(od) # restore working directory
```

---

reformat\_code

---

*Format the code in the usage and examples sections*


---

## Description

The function `tidy_source` in the **formatR** package is used to polish the Rd files generated by **roxygen2** in the usage and examples sections.

## Usage

```
reformat_code(path, ...)
```

## Arguments

<code>path</code>	the path of the Rd file
<code>...</code>	other arguments passed to <code>tidy_source</code>

## Value

NULL; as a side effect, the original Rd file will be updated

## Note

If the usage or examples code is not syntactically correct, it will not be reformatted and a message will be printed on screen. One possible situation is the percent symbol %, which should be escaped even in the examples code (cf Writing R Extensions), and this can make the code syntactically incorrect, e.g. `a %in% b` should be `a \%in% b` but the latter is not valid R code. Anyway, this function will try to unescape the percent symbols before reformatting the code, then escape them.

## Author(s)

Yihui Xie <<https://yihui.org>>

## See Also

`tidy_source`

## Examples

```
rd.file = system.file("examples", "reformat_code_demo.Rd", package = "Rd2roxygen")
file.copy(rd.file, tempdir())
fmt.file = file.path(tempdir(), "reformat_code_demo.Rd")

file.show(fmt.file) ## show the raw Rd

reformat_code(fmt.file)
file.show(fmt.file) ## the formatted Rd
```

---

roxygen_and_build	<i>Roxygenize a package, clean up and build/check the package</i>
-------------------	---

---

## Description

After the source package is roxygenized, this function will build the package. Optionally it also installs or checks the package, reformats the code in the example sections. Note [rab](#) is an alias of [roxygen\\_and\\_build](#).

## Usage

```
roxygen_and_build(
  pkg,
  build = TRUE,
  build.opts = "--no-manual",
  install = FALSE,
  install.opts = if (build) "" else "--with-keep.source",
  check = FALSE,
  check.opts = "--as-cran --no-manual",
  remove.check = TRUE,
  reformat = TRUE,
  before = NULL,
  ...
)

rab(
  pkg,
  build = TRUE,
  build.opts = "--no-manual",
  install = FALSE,
  install.opts = if (build) "" else "--with-keep.source",
  check = FALSE,
  check.opts = "--as-cran --no-manual",
  remove.check = TRUE,
  reformat = TRUE,
  before = NULL,
  ...
)
```

**Arguments**

pkg	the root directory of the source package
build	whether to build the package
build.opts	options to be passed to R CMD build
install	whether to install the package
install.opts	options to be passed to R CMD INSTALL
check	whether to check the package
check.opts	options to check the package (e.g. "--no-examples")
remove.check	whether to remove the directory generated by R CMD check
reformat	whether to reformat the example code; see <a href="#">reformat_code</a>
before	an R expression to be evaluated under the package root directory before the package is roxygenized and built
...	other arguments passed to <a href="#">roxygenize</a>

**Value**

NULL

**Author(s)**

Yihui Xie <<https://yihui.org>>

**Examples**

```
## Not run:
roxygen_and_build("Rd2roxygen", install = TRUE)
## or simply
rab("Rd2roxygen", install = TRUE)

## End(Not run)
```



# Index

`create_roxygen`, [2](#), [3](#)

`parse_and_save`, [3](#)

`parse_file`, [2](#), [4](#)

`rab`, [2](#), [7](#)

`rab (roxygen_and_build)`, [7](#)

`Rd2roxygen`, [2](#), [5](#)

`Rd2roxygen-package`, [2](#)

`reformat_code`, [6](#), [8](#)

`roxygen_and_build`, [7](#), [7](#)

`roxygenize`, [8](#)

`tidy_source`, [6](#)