

Package ‘sinew’

July 23, 2025

Type Package

Title Package Development Documentation and Namespace Management

Version 0.4.0

Date 2022-03-27

Description Manage package documentation and namespaces from the command line.
Programmatically attach namespaces in R and Rmd script, populates
'Roxygen2' skeletons with information scraped from within functions and
populate the Imports field of the DESCRIPTION file.

Depends R (>= 3.2.0)

Imports rstudioapi, utils, tools, sos, stringi, yaml, crayon, cli,
rematch2

License MIT + file LICENSE

Suggests rcmdcheck, git2r, shiny, miniUI, withr, usethis, fs, details,
roxygen2, testthat, knitr, rmarkdown

URL <https://github.com/yoniced/sinew>

BugReports <https://github.com/yoniced/sinew/issues>

NeedsCompilation no

RoxygenNote 7.1.2

VignetteBuilder knitr

Author Jonathan Sidi [aut, cre],
Anton Grishin [ctb],
Lorenzo Busetto [ctb],
Alexey Shiklomanov [ctb],
Stephen Holsenbeck [ctb]

Maintainer Jonathan Sidi <yoniced@gmail.com>

Repository CRAN

Date/Publication 2022-03-31 10:30:02 UTC

Contents

create_yaml	2
interOxyAddIn	3
ls_param	4
makeDictionary	4
makeOxyFile	5
makeOxygen	6
make_force_packages	8
make_import	8
moga	9
parse_check	10
pretty_addin	11
pretty_merge	11
pretty_namespace	12
pretty_rmd	13
pretty_sinew	14
rmOxygen	15
sinew_opts	16
tabular	17
untangle	17
untangle_examples	19
update_desc	20
Index	21

create_yaml	Create _sinewconfig.yml
-------------	-------------------------

Description

Create _sinewconfig.yml file in project root directory

Usage

create_yaml()

Value

nothing

Author(s)

Jonathan Sidi

Examples

```
## Not run:  
create_yml()  
  
## End(Not run)
```

interOxyAddIn	<i>Interactive add-in</i>
---------------	---------------------------

Description

Launches an interactive addin for insertion of roxygen2 comments in files. Allows selection of extra parameters for [makeOxygen](#)

Usage

```
interOxyAddIn()
```

Details

Open an .R file in Rstudio's source editor.

This addin requires shiny and miniUI to be installed (listed as Suggests in Description)

- Launch the add-in via Addins -> interactiveOxygen or interOxyAddIn() in the console.
 - Add-in opens in the viewer panel.
- Select function's/dataset's name in the source editor.
 - If objects cannot be found, the addin prompts to source the file.
 - Choose parameters for [makeOxygen](#)
 - * Click Insert
- Select next object's name
- Rinse/Repeat
- Click Quit when done with the file.

Value

Nothing. Inserts roxygen2 comments in a file opened in the source editor.

Author(s)

Anton Grishin, Jonathan Sidi

Examples

```
if(interactive()) interOxyAddIn()
```

ls_param	<i>Return roxygen2 parameter calls from parameter dictionary</i>
----------	--

Description

Return roxygen2 parameter calls from the intersection of the parameters listed in the package dictionary and the formals of a function

Usage

```
ls_param(obj, dictionary = "man-roxygen/Dictionary-1.R", print = TRUE)
```

Arguments

obj	function or name of function
dictionary	character, path_to_dictionary, Default: 'roxygen-man/Dictionary-1.R'
print	boolean print output to console, Default: TRUE

Value

character vector

Examples

```
repo='https://raw.githubusercontent.com/yoniced/sinew/master/'
dict_loc=file.path(repo,'man-roxygen/Dictionary-1.R')
ls_param(sinew::makeOxygen,dictionary=dict_loc)
```

makeDictionary	<i>Parse package R files to create dictionary of unique parameter definitions</i>
----------------	---

Description

Given list of R files function returns roxygen2 template consisting of intersecting parameter definitions

Usage

```
makeDictionary(path, save_path = FALSE)
```

Arguments

path	character or character vector of paths to files to parse
save_path	boolean that allows for function to write template to man-roxygen subdirectory, Default: FALSE

Value

character/character vector of intersecting parameters

Examples

```
makeDictionary('R')
```

makeOxyFile

Inserts roxygen2 skeletons in file(s).

Description

Applies makeOxygen function to all functions/dataframes in supplied file(s)

Usage

```
makeOxyFile(input = NULL, overwrite = FALSE, verbose = interactive(), ...)
```

Arguments

input	character, vector of path(s) to one or more .R files, a path to directory containing .R files, Default: NULL
overwrite	logical, If TRUE overwrites file(s), FALSE writes "Oxy"- prefixed files in the same directory, Default: FALSE
verbose	logical, If TRUE will print output to console and open edited files in the editor viewer, Default: interactive()
...	additional parameters passed to makeOxygen

Details

If an object cannot be found it will be sourced into a temporary environment. If the file already contains roxygen2 comments they will be deleted to avoid duplication. Some functions may require attaching additional packages. For instance, if functions were defined with purrr's compose or partial functions, omission of purrr:: in definitions will require library(purrr) before proceeding with makeOxyFile.

Value

Nothing. Writes files with roxygen2 comments as a side effect

Author(s)

Anton Grishin

See Also

[makeOxygen](#)

Examples

```
# copy dummy package to tempdir
file.copy(system.file('pkg',package = 'sinew'),tempdir(),recursive = TRUE)
pkg_dir <- file.path(tempdir(),'pkg')
pkg_dir_R <- file.path(pkg_dir,'R')

# update namespaces in package functions
pretty_namespace(pkg_dir_R, overwrite = TRUE)

# test on one R file
# this will create a new R file called 'oxy-yy.R' in the same directory
makeOxyFile(file.path(pkg_dir_R,'yy.R'))

# Remove the file
unlink(file.path(pkg_dir_R,'oxy-yy.R'))

# Test on all R files in directory and overwrite the contents
makeOxyFile(pkg_dir_R, overwrite = TRUE)

# Remove Skeleton
rmOxygen(file.path(pkg_dir_R,'yy.R'))
rmOxygen(file.path(pkg_dir_R,'zz.R'))

# adds more fields to defaults, passes "cut" to make_import
sinew_opts$append(list(add_fields=c("concept", "describeIn")))
makeOxyFile(file.path(pkg_dir_R,'yy.R'), cut = 5)

# cleanup
unlink(pkg_dir, recursive = TRUE, force = TRUE)
sinew_opts$restore()
```

makeOxygen

Populate Roxygen2 Skeleton

Description

Creates roxygen2 skeleton including title, description, import and other fields for an object in the global environment or a function of an attached namespace.

Usage

```
makeOxygen(
  obj,
  add_default = TRUE,
  add_fields = sinew_opts$get("add_fields"),
  use_dictionary = NULL,
  print = TRUE,
  ...
)
```

Arguments

<code>obj</code>	function or name of function
<code>add_default</code>	boolean to add defaults values to the end of the PARAM fields, Default: TRUE
<code>add_fields</code>	character vector to add additional roxygen2 fields, Default: c("details", "examples", "seealso", "rdname", "export")
<code>use_dictionary</code>	character, path_to_dictionary, Default: NULL
<code>print</code>	boolean print output to console, Default: TRUE
<code>...</code>	arguments to be passed to <code>make_import</code>

Details

`add_fields` can include any slot except for the defaults (title, description, param, return). The order in `add_fields` determines the order of printout. The roxygen2 fields to add are list below, for more information go to [Generating Rd files](#). If `obj` is 'data.frame' or 'tibble' then the fields c('export', 'examples', 'seealso', 'rdname') will be ignored.

Field	Skeleton
<code>author</code>	AUTHOR [AUTHOR_2]
<code>backref</code>	src/filename.cpp
<code>concept</code>	CONCEPT_TERM_1 [CONCEPT_TERM_2]
<code>describeIn</code>	FUNCTION_NAME DESCRIPTION
<code>details</code>	DETAILS
<code>example</code>	path/relative/to/package/root
<code>export</code>	
<code>family</code>	FAMILY_TITLE
<code>field</code>	FIELD_IN_S4_RefClass DESCRIPTION
<code>format</code>	DATA_STRUCTURE
<code>importClassesFrom</code>	PKG CLASS_a [CLASS_b]
<code>importMethodsFrom</code>	PKG METHOD_a [METHOD_b]
<code>include</code>	FILENAME.R [FILENAME_b.R]
<code>inherit</code>	[PKG::]SOURCE_FUNCTION [FIELD_a FIELD_b]
<code>inheritDotParams</code>	[PKG::]SOURCE_FUNCTION
<code>inheritSection</code>	[PKG::]SOURCE_FUNCTION [SECTION_a SECTION_b]
<code>keywords</code>	KEYWORD_TERM
<code>name</code>	NAME
<code>rdname</code>	FUNCTION_NAME
<code>references</code>	BIB_CITATION
<code>section</code>	SECTION_NAME
<code>source</code>	\url{ http://somewhere.important.com/ }
<code>slot</code>	SLOTNAME DESCRIPTION
<code>template</code>	FILENAME
<code>templateVar</code>	NAME VALUE
<code>useDynLib</code>	PKG [routine_a routine_b]

Examples

```
makeOxygen(stats::lm)
```

make_force_packages	<i>Create lists of package exports</i>
---------------------	--

Description

Useful for supplying packages to the force argument to pretty_namespace.

Usage

```
make_force_packages(packages)
```

Arguments

packages	(character) packages to include in list. When duplicate function names exist the order of packages determines which function will be selected first - IE the first package with the function name will include that function, the second package with the function name will not have it listed.
----------	--

Value

(named list) with package names as names as all exports as a character vector

Examples

```
make_force_packages(c("utils"))
```

make_import	<i>Populate import fields for documentation</i>
-------------	---

Description

Scrape R script to create import and importFrom calls for roxygen2, namespace or description files

Usage

```
make_import(
  script,
  cut = NULL,
  print = TRUE,
  format = "oxygen",
  desc_loc = NULL
)
```


Arguments

<code>script</code>	character, connection to pass to <code>readLines</code> , can be file path, directory path, url path
<code>cut</code>	integer, number of functions to write as <code>importFrom</code> until switches to <code>import</code> , Default: <code>NULL</code>
<code>print</code>	boolean, print output to console, Default: <code>TRUE</code>
<code>format</code>	character, the output format must be in <code>c('oxygen','description','import')</code> , Default: <code>'oxygen'</code>
<code>desc_loc</code>	character, path to <code>DESCRIPTION</code> file, if not <code>NULL</code> then the Imports fields in the <code>DESCRIPTION</code> file, Default: <code>NULL</code>

Examples

```
# copy dummy package to tempdir
file.copy(system.file('pkg',package = 'sinew'),tempdir(),recursive = TRUE)

pkg_dir <- file.path(tempdir(),'pkg')
pkg_dir_R <- file.path(pkg_dir,'R')
pkg_dir_DESC <- file.path(pkg_dir,'DESCRIPTION')

# update namespaces in package functions
pretty_namespace(pkg_dir_R, overwrite = TRUE)

# update imports/importsFrom for roxygen2 tags
make_import(pkg_dir_R,format = 'oxygen')

# update Imports for DESCRIPTION file output to console
make_import(pkg_dir_R,format = 'description')

# update Imports for DESCRIPTION file overwrite file
make_import(pkg_dir_R,format = 'description', desc_loc = pkg_dir)

cat(readLines(pkg_dir_DESC),sep = '\n')

# cleanup tempdir
unlink(pkg_dir, force = TRUE, recursive = TRUE)
```

moga

Make Oxygen Great Again

Description

Update/append an R file that has roxygen2 headers already with updated information

Usage

```
moga(path, ..., force.fields = NULL, dry.run = TRUE, overwrite = FALSE)
```

Arguments

path	character path to R file
...	arguments to be passed to new makeOxygen
force.fields	character, vector a field names that are in current header that are to be updated Default: NULL
dry.run	boolean, write lines to console the output, Default: TRUE
overwrite	boolean, overwrite contents of input file, Default: FALSE

Details

Cross references fields already in the roxygen2 header and adds any new ones from the updated call. To force a change to a field add field name to force.fields.

Value

character

Examples

```
# We want to update the contents of the Roxygen2 with the new parameter "b"
# without touching the other fields

# Before
cat(readLines(system.file('example_moga.R',package = 'sinew')),sep = '\n')

# After
moga(system.file('example_moga.R',package = 'sinew'))
```

parse_check	<i>parse_check</i>
-------------	--------------------

Description

check for fail of pretty_parse > parse, and offers to open file to offending line

Usage

```
parse_check(p, txt, ask)
```

Arguments

p	result of pretty_parse > parse
txt	input text to pretty_parse
ask	boolean, If TRUE then a menu will be created for the use to choose between competing namespaces for a function, Default: TRUE

`pretty_addin`*Interactively run pretty functions in R and Rmd files*

Description

Addin that scans the file source contents and attaches namespace information.

Usage

```
pretty_addin()
```

Details

Either saved or untitled R or Rmd files in the source editor may be used.

In R files Highlight specific text, or not highlight at all and the whole document will be used.

In Rmd files highlight subsets of chunks to add namespaces directly in the chunks, or not highlight at all and the whole document will be used to create a new chunk at the top of the document with relevant namespaces needed to render the Rmd.

`pretty_merge`*pretty_merge*

Description

handles force and ignore arguments

Usage

```
pretty_merge(e1, e2)
```

Arguments

<code>e1</code>	(data.frame) typically sym. funs with list of all parsed functions in txt
<code>e2</code>	(list) typically force or ignore with list of namespaces and the respective functions to force or ignore

```
pretty_namespace      Append namespace to functions in script
```

Description

Autoappend namespace to functions in script by searchpath order

Usage

```
pretty_namespace(
  con = NULL,
  text = NULL,
  ask = TRUE,
  askenv = new.env(),
  force = NULL,
  ignore = NULL,
  overwrite = FALSE,
  sos = FALSE
)
```

Arguments

con	character, path to file or directory that contains script, Default: NULL
text	character, vector that contains script, Default: NULL
ask	boolean, If TRUE then a menu will be created for the use to choose between competing namespaces for a function, Default: TRUE
askenv	environment, environment that stores names of functions to force in ask, Default: new.env()
force	list, named list of functions to force over the internal search (see details), Default: NULL
ignore	list, named list of functions to ignore (see details), Default: NULL
overwrite	boolean, overwrite original file, Default: FALSE
sos	boolean, apply sos search for uninstalled libraries, Default: FALSE

Details

Searches for functions in the loadedNamespace, help.search and then [findFn](#). If force is not NULL but a named list eg `list(stats=c('rnorm', 'runif'), utils = 'head')`, then the value pairs will be used in place of what was found using the search path. If ignore is not NULL but a named list eg `list(stats=c('rnorm', 'runif'), utils = 'head')`, then if the functions are found they will not have a namespace attached to them.

If you want to toggle off the summary console printing you can set it globally via `sineu_opts$set(pretty_print=FALSE)`.

Value

character

Author(s)

Jonathan Sidi

See Also[findFn](#), [help.search](#)**Examples**

```
txt <- '#some comment'
yy <- function(a=4){
  head(runif(10),a)
  # a comment
}

zz <- function(v=10,a=8){
  head(runif(v),a)
}'

pretty_namespace(text=txt)
```

pretty_rmd

*Attach namespacing to Rmarkdown chunks***Description**

Apply pretty_namespace to Rmarkdown document

Usage

```
pretty_rmd(
  input,
  output = tempfile(fileext = ".Rmd"),
  open_output = TRUE,
  create_library = TRUE,
  chunks = NULL,
  ...
)
```

Arguments

input	character, path to input Rmd file
output	character, path to output Rmd file, Default: NULL
open_output	boolean, open the output on.exit, Default: TRUE
create_library	boolean, create library chunk, Default: TRUE
chunks	numeric, indices of chunks to run on, Default: NULL
...	arguments to pass to pretty_namespace

Details

If output is NULL then the returned lines are printed to console. If chunks is NULL then all the chunks are used.

Value

character

Author(s)

Jonathan Sidi

See Also

[pretty_namespace](#)

Examples

```
## Not run:
  if(interactive()){
    pretty_rmd(input = system.file('example.Rmd',package = 'sinew'))
  }

## End(Not run)
```

```
pretty_sinew
```

```
Convert File to R directory with pretty and oxygen
```

Description

One function to run [pretty_namespace](#), [untangle](#) and [makeOxyFile](#)

Usage

```
pretty_sinew(con = NULL, text = NULL, dir.out = NULL, keep.body = TRUE)
```

Arguments

con	character, path to file or directory that contains script, Default: NULL
text	character, vector that contains script, Default: NULL
dir.out	character, path to save new R files, Default: NULL
keep.body	boolean, if TRUE all non-functions will be saved to body.R in the parent directory of dir.out, Default: TRUE

Details

If dir.out is set to NULL all outputs are redirected into file.path(tempdir(), 'sinew')

Value

Nothing, side effects is to create files

Author(s)

Jonathan Sidi

rmOxygen

Remove roxygen2 Comments From an .R File

Description

Strips .R files of roxygen2 style comments (#')

Usage

```
rmOxygen(.file, showonexit = TRUE)
```

Arguments

<code>.file</code>	character, path to an .R file, character vector of length 1
<code>showonexit</code>	logical, show file on exit. Default: TRUE

Value

Nothing. Overwrites files as a side effect

Author(s)

Anton Grishin

Examples

```
## Not run:  
rmOxygen("./myRfunctions/function1.R")  
  
## End(Not run)
```

sinew_opts	<i>Default and current sinew options</i>
------------	--

Description

Options for functions in the sinew package. When running R code, the object `sinew_opts` (default options) is not modified by chunk headers (local chunk options are merged with default options), whereas `sinew_opts_current` (current options) changes with different chunk headers and it always reflects the options for the current chunk.

Usage

`sinew_opts`

`sinew_opts_current`

Format

An object of class `list` of length 5.

An object of class `list` of length 5.

Details

Normally we set up the global options once in the first code chunk in a document using `sinew_opts$set()`, so that all *latter* chunks will use these options. Note the global options set in one chunk will not affect the options in this chunk itself, and that is why we often need to set global options in a separate chunk.

A list of default chunk options, can be retrieved via `sinew_opts$get()`

Note

`sinew_opts_current` is read-only in the sense that it does nothing if you call `sinew_opts_current$set()`; you can only query the options via `sinew_opts_current$get()`.

Examples

```
sinew_opts$get()
```

tabular	<i>Tabular for roxygen2</i>
---------	-----------------------------

Description

Convert data.frame to roxygen2 tabular format

Usage

```
tabular(df, header = TRUE, ...)
```

Arguments

df	data.frame to convert to table
header	boolean to control if header is created from names(df), Default: TRUE
...	arguments to pass to format

Value

character

Source

[roxygen2 formatting](#)

See Also

[format](#)

Examples

```
tabular(mtcars[1:5, 1:5])
tabular(mtcars[1:5, 1:5], header=FALSE)
```

untangle	<i>Split an R script by functions</i>
----------	---------------------------------------

Description

Split a R script with multiple functions into multiple single function R files.

Usage

```
untangle(  
  file = "",  
  text = NULL,  
  dir.out = "",  
  keep.body = TRUE,  
  dir.body = dirname(dir.out)  
)
```

Arguments

file	character, path to R file, Default: ""
text	character, vector of R commands, Default: NULL
dir.out	character, path to save new R files, Default: NULL
keep.body	boolean, if TRUE all non-functions will be saved to body.R in the parent directory of dir.out, Default: TRUE
dir.body	character, path to save body.R files, Default: dirname(dir.out)

Details

Functions that are objects in lists are treated as objects and will stay in body.R .

Value

list of separate functions

Author(s)

Jonathan Sidi

Examples

```
test_dir <- file.path(tempdir(), 'sinew_test')  
  
dir.create(test_dir)  
  
txt <- "#some comment  
yy <- function(a=4){  
  head(runif(10),a)  
  # a comment  
}  
  
v <- 20  
  
#another comment  
zz <- function(v=10,a=3){  
  head(runif(v),pmin(a,v))  
}
```

```
zz(v)

"

untangle(text = txt,dir.out = test_dir)

list.files(tempdir(), recursive = TRUE, pattern = '.R$')

cat( readLines(file.path(test_dir,'yy.R')), sep = '\n')

cat( readLines(file.path(test_dir,'zz.R')), sep = '\n')

cat( readLines(file.path(tempdir(),'body.R')), sep = '\n')

unlink(test_dir, force = TRUE, recursive = TRUE)
```

untangle_examples	<i>Convert examples blocks in roxygen2 header to script</i>
-------------------	---

Description

Converts and aggregates roxygen2 examples into a single output file.

Usage

```
untangle_examples(input, output = "./roxy_ex_to_file.R")
```

Arguments

input	character, file or directory
output	character, file path of output, Default: './roxy_ex_to_file.R'

Details

If output is set to NULL then output returned as invisible character object.

Value

writes R file to disk

Author(s)

Jonathan Sidi

`update_desc`*Update Package Description File*

Description

Update package DESCRIPTION file Imports field

Usage

```
update_desc(path, overwrite = TRUE)
```

Arguments

<code>path</code>	character, path to R folder containing package functions
<code>overwrite</code>	logical, overwrite the file, Default: TRUE

Details

If overwrite is FALSE then the output will be returned to the console.

Author(s)

Jonathan Sidi

Examples

```
# copy dummy package to tempdir
file.copy(system.file('pkg', package = 'sinew'), tempdir(), recursive = TRUE)

pkg_dir <- file.path(tempdir(), 'pkg')
pkg_dir_R <- file.path(pkg_dir, 'R')
pkg_dir_DESC <- file.path(pkg_dir, 'DESCRIPTION')

# update namespaces in package functions
pretty_namespace(pkg_dir_R, overwrite = TRUE)

# send result to the console
update_desc(pkg_dir_R, overwrite = FALSE)

# overwrite the Imports field
update_desc(pkg_dir_R, overwrite = TRUE)

# view DESCRIPTION file
cat(readLines(pkg_dir_DESC), sep = '\n')

# cleanup tempdir
unlink(pkg_dir, recursive = TRUE, force = TRUE)
```

Index

- * **Internal**
 - parse_check, [10](#)
- * **datasets**
 - sinew_opts, [16](#)
- * **interactive**
 - interOxyAddIn, [3](#)
- * **namespace**
 - pretty_addin, [11](#)
 - pretty_namespace, [12](#)
 - pretty_rmd, [13](#)
 - pretty_sinew, [14](#)
- * **opts**
 - sinew_opts, [16](#)
- * **populate**
 - make_import, [8](#)
 - makeOxyFile, [5](#)
 - makeOxygen, [6](#)
 - moga, [9](#)
 - update_desc, [20](#)
- * **untangle**
 - untangle, [17](#)
 - untangle_examples, [19](#)
- * **utility**
 - create_yaml, [2](#)
 - ls_param, [4](#)
 - makeDictionary, [4](#)
 - rmOxygen, [15](#)
 - tabular, [17](#)

create_yaml, [2](#)

findFn, [12](#), [13](#)

format, [17](#)

help.search, [13](#)

interOxyAddIn, [3](#)

ls_param, [4](#)

make_force_packages, [8](#)

make_import, [8](#)

makeDictionary, [4](#)

makeOxyFile, [5](#), [14](#)

makeOxygen, [3](#), [5](#), [6](#)

menu, [10](#), [12](#)

moga, [9](#)

parse_check, [10](#)

pretty_addin, [11](#)

pretty_merge, [11](#)

pretty_namespace, [12](#), [14](#)

pretty_rmd, [13](#)

pretty_sinew, [14](#)

rmOxygen, [15](#)

sinew_opts, [16](#)

sinew_opts_current (sinew_opts), [16](#)

tabular, [17](#)

untangle, [14](#), [17](#)

untangle_examples, [19](#)

update_desc, [20](#)