R Packages: Document R Code

?mean

The name of the function, and the library it is in.

mean {base} R Documentation

Arithmetic Mean

Description

What it does.

Generic function for the (trimmed) arithmetic mean.

Usage

```
mean(x, ...)

## Default S3 method:
mean(x, trim = 0, na.rm = FALSE, ...)
```

The function's name, and in the parentheses the named arguments it expects, in the order it expects them. If an argument has a default value, it is shown. Arguments without default values (e.g. x) must be provided by you.

More details on each named argument. This will tell you what class of thing each argument has to be—an object, a number, a data frame, a logical value, etc.

Arguments

`x`	An R object. Currently there are methods for numeric/logical vectors and date, date-time and time interval objects. Complex vectors are allowed for `trim = 0`, only.
`trim`	the fraction (0 to 0.5) of observations to be trimmed from each end of `x` before the mean is computed. Values of trim outside that range are taken as the nearest endpoint.
`na.rm`	a logical value indicating whether `NA` values should be stripped before the computation proceeds.
`...`	further arguments passed to or from other methods.

The ellipsis allows other arguments to be passed to and from the function.

What the function

Image by Kieran Healy

roxygen2: In-Line Documentation for R

roxygen2: In-Line Documentation for R

Write documentation *with* your functions

roxygen2: In-Line Documentation for R

Write documentation with your functions

render with `document()`

Insert roxygen skeleton

`Code > Insert Roxygen Skeleton`

Insert roxygen skeleton

`Code > Insert Roxygen Skeleton`

Ctrl/Cmd + Shift + Alt/Opt + R

```
add_one <- function(x) {
  x <- x + 1
  x
}
```

Insert Roxygen Skeleton

```
#' Title
#'
#' @param x
#'
#' @return
#' @export
#'
#' @examples
add_one <- function(x) {
  x <- x + 1
  x
}
```

Roxygen comments

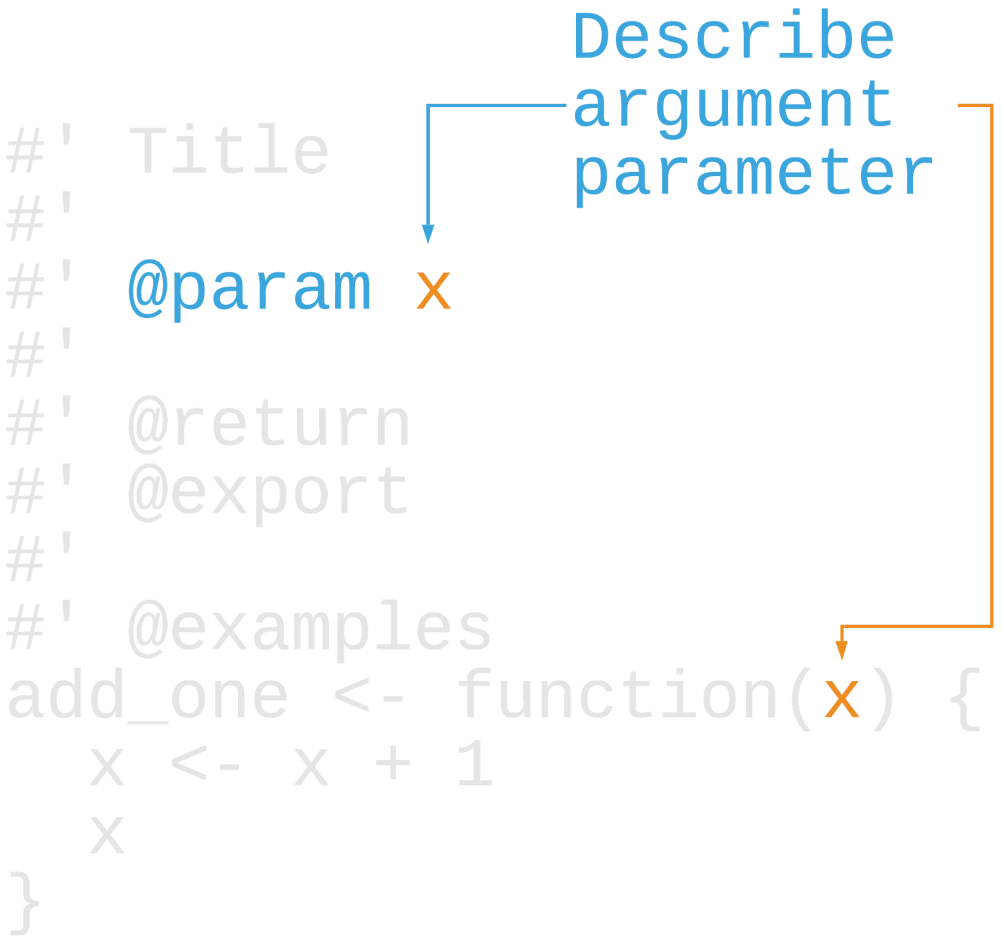
```
#' Title
#'
#' @param x
#'
#' @return
#' @export
#'
#' @examples
add_one <- function(x) {
  x <- x + 1
  x
}
```

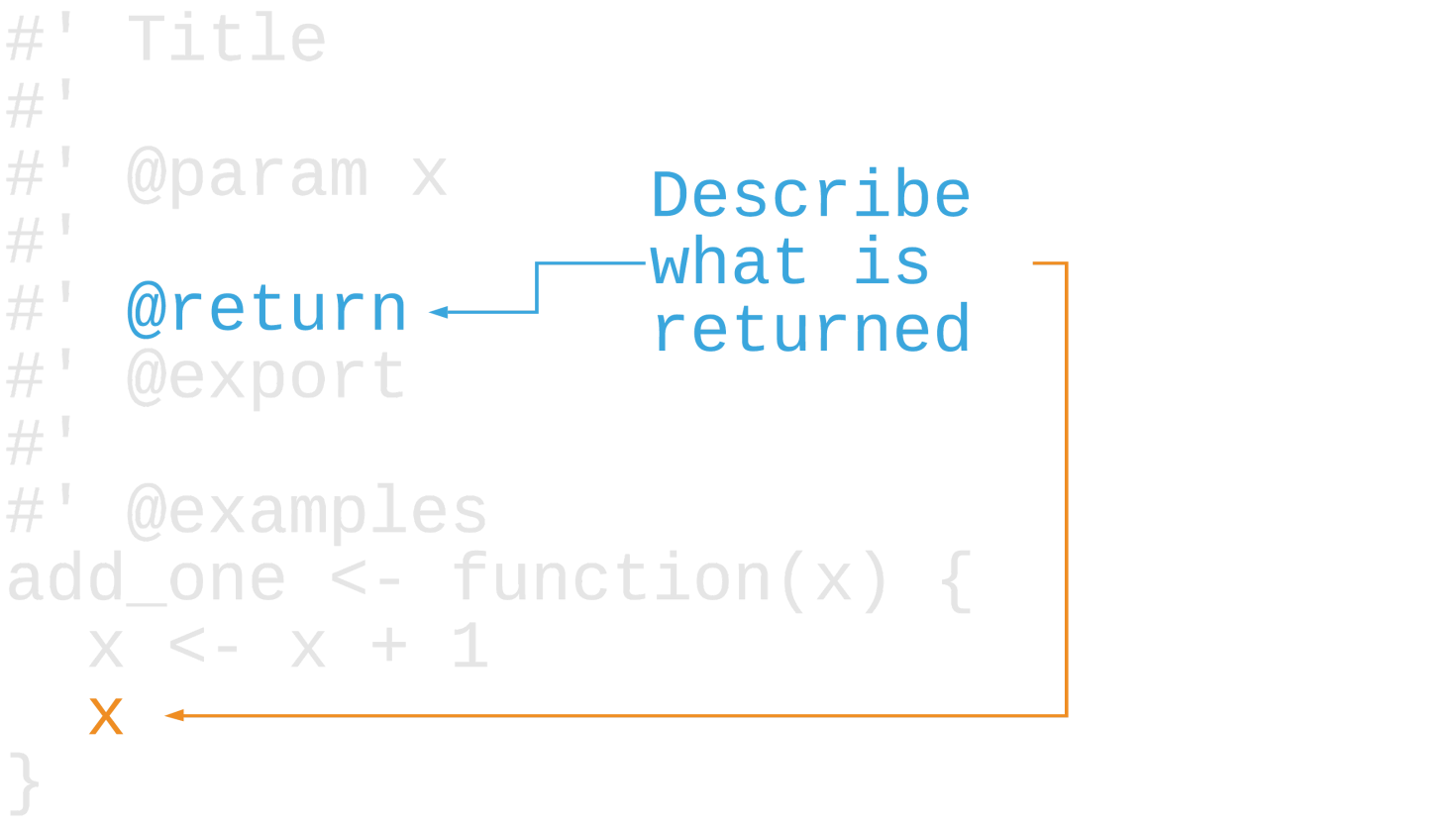
```
#' Title
#'
#' @param x
#'
#' @return
#' @export
#'
#' @examples
add_one <- function(x) {
  x <- x + 1
  x
}
```

Roxygen tags

```
#' Title
#'
#' @param x
#'
#' @return
#' @export
#'
#' @examples
add_one <- function(x) {
  x <- x + 1
  x
}
```

Description and Details


Describe
argument
parameter
#' Title
#'
#' @param x
#'
#' @return
#' @export
#'
#' @examples
add_one <- function(x) {
x <- x + 1
x
}

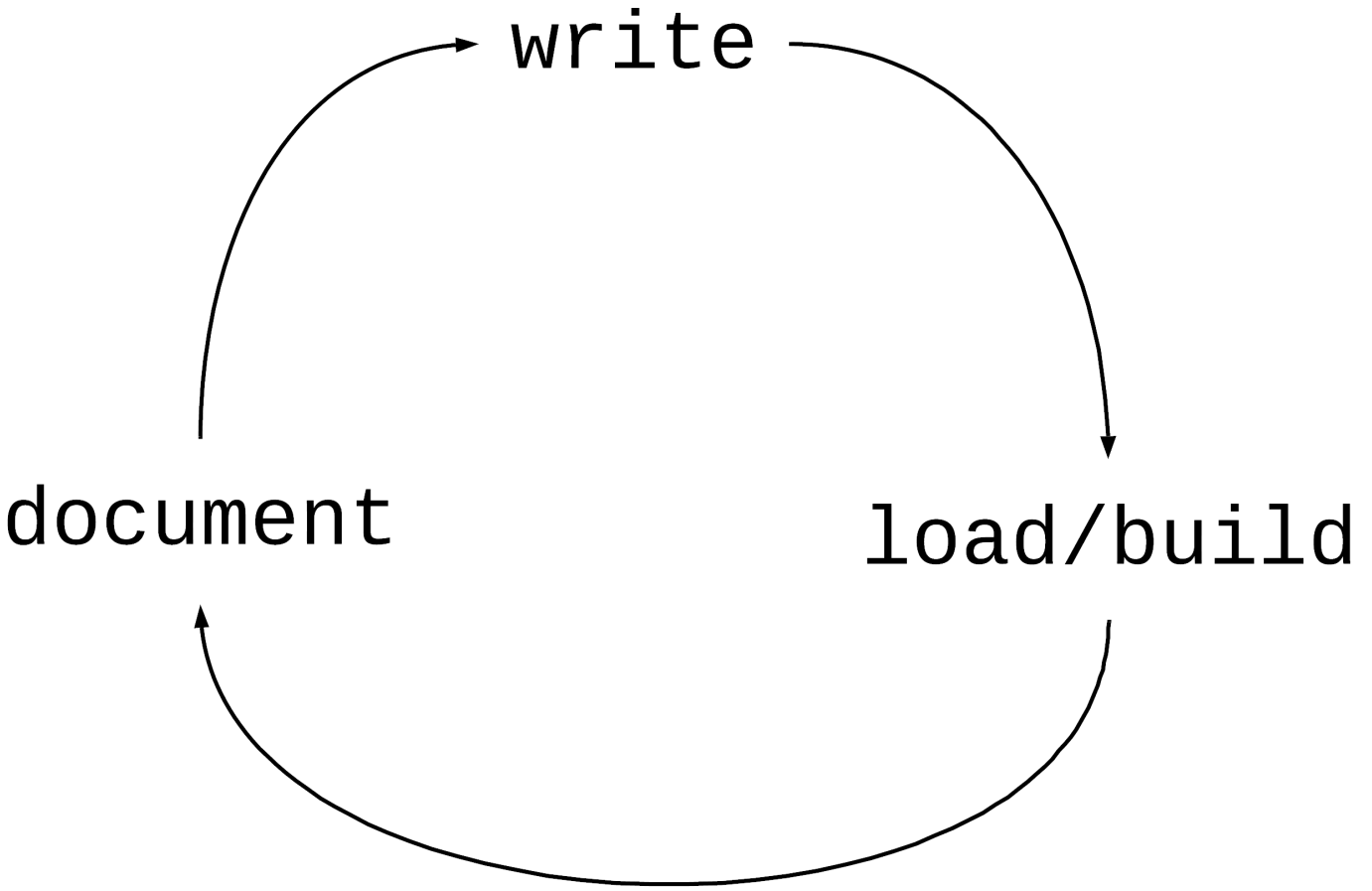

#' Title
#'
#' @param x
#'
#' @return
#' @export
#'
#' @examples
add_one <- function(x) {
x <- x + 1
x
}
Describe what is returned

```
#' Title
#'
#' @param x
#'
#' @return
#' @export
#'
#' @examples
add_one <- function(x) {
  x <- x + 1
  x
}
```

Export the function pkg::add_one()

```
#' Title
#'
#' @param x
#'
#' @return
#' @export
#'
#' @examples
add_one <- function(x) {
  x <- x + 1
  x
}
```

write examples


write
load/build
document

Write roxygen, run

`document()`

Write roxygen, run `document()`

```
theme_mako <- function(base_size = 14) {
  ggplot2::theme_dark(base_size = base_size) +
    ggplot2::theme(
      panel.background = ggplot2::element_rect(fill = "#0D98BA")
    )
}
```

Write roxygen, run `document()`

```r
#' A dark theme with a mako-like background
#'
#' @param base_size base font size
#'
#' @return a ggplot2 theme
#' @export
#'
#' @examples
#'
#' ggplot2::quickplot(iris$Sepal.Length) + theme_mako()
#'
theme_mako <- function(base_size = 14) {
  ggplot2::theme_dark(base_size = base_size) +
    ggplot2::theme(
      panel.background = ggplot2::element_rect(fill = "#0D98BA")
    )
}
```

```
document()
```

```
shinRa
├── .Rbuildignore
├── .gitignore
├── DESCRIPTION
├── NAMESPACE
├── R/
│   ├── themes.R
├── man
│   ├── theme_mako.Rd
└── shinRa.Rproj
```

man/theme_mako.Rd

```
## % Generated by roxygen2: do not edit by hand
## % Please edit documentation in R/themes.R
## \name{theme_mako}
## \alias{theme_mako}
## \title{A dark theme with a mako-like background}
## \usage{
## theme_mako(base_size = 14)
## }
## \arguments{
## \item{base_size}{base font size}
## }
## \value{
## a ggplot2 theme
## }
## \description{
## A dark theme with a mako-like background
## }
## \examples{
##
## ggplot2::quickplot(iris$Sepal.Length) + theme_mako()
##
## }
```

?theme_mako

theme_mako {shinRa}

A dark theme with a mako-like background

Description

A dark theme with a mako-like background

Usage

```
theme_mako(base_size = 14)
```

Arguments

base_size	base font size

Value

a ggplot2 theme

Examples

```
ggplot2::quickplot(iris$Sepal.Length) + theme_mako()
```

Syntax

LaTeX like. See more at https://r-pkgs.org/man.html

`use_roxygen_md()` lets you write in Markdown. See more at https://roxygen2.r-lib.org/articles/rd-formatting.html

Your Turn 1

Open the NAMESPACE file. What do you see?

Let's add documentation. Run `use_roxygen_md()`

Open `r/themes.R`. Insert a roxygen skeleton for `theme_avalanche()`.

Change the title to "AVALANCHE ggplot2 themes"

Hit Enter/Return twice after the title. Make sure the new lines start with `#'`. Add this text: "Minimalistic ggplot themes for use on AVALANCHE reports."

Run `document()` or press `Ctrl/Cmd + Shift + D`. Read the help page for your function with `?theme_avalanche`.

Finally, look at the NAMESPACE file again. What changed?

Your Turn 1

```
exportPattern("^[^\\.]")
```

Your Turn 1

```r
#' AVALANCHE ggplot2 themes
#'
#' Minimalistic ggplot themes for use on AVALANCHE reports.
#'
#' @param base_size
#' @param ...
#'
#' @return
#' @export
#'
#' @examples
theme_avalanche <- function(base_size = 14, ...) {
  ggplot2::theme_minimal(base_size = base_size, ...) +
    ggplot2::theme(panel.grid.minor = ggplot2::element_blank())
}
```

Your Turn 1

```
# Generated by roxygen2: do not edit by hand

export("%>%")
export(db_con)
export(get_resident_data)
export(theme_avalanche)
export(theme_avalanche_h)
export(theme_avalanche_v)
import(data.table)
importFrom(magrittr,"%>%")
```

Argument descriptions

```
#' [other roxygen code]
#' @param x The name of a database to retrieve
get_data <- function(x) {
  # code to get data
}
```

Argument descriptions: @inheritParams

```
#' [other roxygen code]
#' @param x The name of a database to retrieve
get_data <- function(x) {
  # code to get data
}

#' [other roxygen code]
filter_table <- function(x) {
  tbl <- get_data(x)
  # code to filter data
}
```

Argument descriptions: @inheritParams

```
#' [other roxygen code]
#' @param x The name of a database to retrieve
get_data <- function(x) {
  # code to get data
}

#' [other roxygen code]
#' @inheritParams get_data
filter_table <- function(x) {
  tbl <- get_data(x)
  # code to filter data
}
```

Examples

Examples can be any kind of R code

```
#' [other roxygen code]
#' @examples
#'
#' library(dplyr)
#' get_data("daily_actice_users") %>%
#'   filter(date == lubridate::today())
get_data <- function(x) {
  # code to get data
}
```

Examples

Examples can be any kind of R code

```
#' [other roxygen code]
#' @examples
#'
#' library(dplyr)
#' get_data("daily_actice_users") %>%
#'   filter(date == lubridate::today())
get_data <- function(x) {
  # code to get data
}
```

Examples

If you don't want to run examples, wrap them in `\dontrun{}` or `\donttest{}`

```
#' [other roxygen code]
#' @examples
#'
#' \dontrun{
#'   get_data("daily_active_users")
#' }
get_data <- function(x) {
  # code to get data
}
```

Your Turn 2

Let's keep working on the documentation for `theme_avalanche()`:

Remove `@param base_size` and replace it with: @inheritParams ggplot2::theme_minimal

For `@param ...`, add: Additional arguments passed to [ggplot2::theme_minimal()]

For `@return`, add: a ggplot theme.

For `@examples`, add two line breaks (make sure the new lines have roxygen comments!). Add this code: `ggplot2::qplot(iris$Sepal.Length) + theme_avalanche()`

Rebuild the documentation and check the help page.

Your Turn 2

```r
#' AVALANCHE ggplot2 themes
#'
#' Minimalistic ggplot themes for use on AVALANCHE reports.
#'
#' @inheritParams ggplot2::theme_minimal
#' @param ... Additional arguments passed to [ggplot2::theme_minimal
#'
#' @return a ggplot theme.
#' @export
#'
#' @examples
#'
#' ggplot2::qplot(iris$Sepal.Length) + theme_avalanche()
#'
theme_avalanche <- function(base_size = 14, ...) {
  ggplot2::theme_minimal(base_size = base_size, ...) +
    ggplot2::theme(panel.grid.minor = ggplot2::element_blank())
}
```

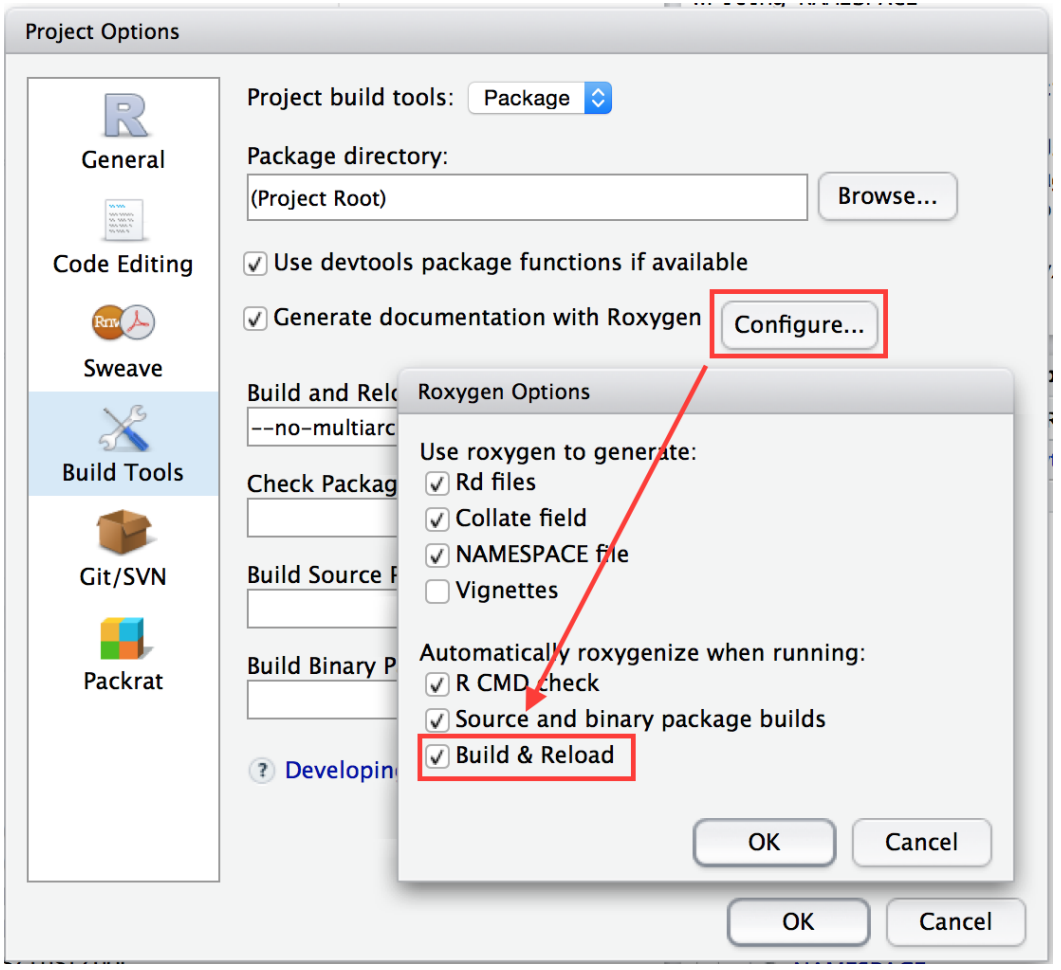


Image from [R Packages, ed. 2]

Quoth Jenny Bryan:

1. Use functions.
2. A few little functions >> a monster function
3. Small well-named helper >> commented code

Helper functions

```
plot_daus <- function(daily_users) {
  daily_users <- daily_users %>%
    dplyr::mutate(date = as.Date(time)) %>%
    dplyr::group_by(date)
    dplyr::select(user_id) %>%
    dplyr::distinct() %>%
    dplyr::summarize(n = dplyr::n())

  ggplot2::ggplot(ggplot2::aes(daily_users, x, n)) +
    ggplot2::geom_col()
}
```

Helper functions

```
plot_daus <- function(daily_users) {
  daily_users <- count_daus(daily_users)

  ggplot2::ggplot(ggplot2::aes(daily_users, x, n)) +
    ggplot2::geom_col()
}

count_daus <- function(daily_users) {
  daily_users %>%
    dplyr::mutate(date = as.Date(time)) %>%
    dplyr::group_by(date)
    dplyr::select(user_id) %>%
    dplyr::distinct() %>%
    dplyr::summarize(n = dplyr::n())
}
```

Show of Hands

Which of these functions will be added to NAMESPACE?

```r
#' Plot daily active users
#'
#' @param ...
#' @export
plot_daus <- function(...) {
  # ... code to plot daily active users
}

#' Count daily active users
#'
#' @param ...
count_daus <- function(...) {
  # ... code to count daily active users
}
```

Show of Hands

Which of these functions will be added to NAMESPACE?

```r
#' Plot daily active users
#'
#' @param ...
#' @export
plot_daus <- function(...) {
  # ... code to plot daily active users
}

#' Count daily active users
#'
#' @param ...
count_daus <- function(...) {
  # ... code to count daily active users
}
```

Exported functions vs internal functions

`@export =`
`shinRa::plot_daus()`

```
library(shinRa)

plot_daus()
```

Exported functions vs internal functions

NO `@export` =

`shinRa:::count_daus()`

```
library(shinRa)

count_daus()
```

🤔🤔🤔🤔🤔🤔

Strategies for documenting helper functions:

1. Don't document them 🤷
2. @keywords internal
3. @noMd

Joining documentation

```
#' [other roxygen code]
#' @param x The name of a database to retrieve
get_data <- function(x) {
  # code to get data
}

#' [other roxygen code]
#' @param x @inheritParam get_data
filter_table <- function(x) {
  tbl <- get_data(x)
  # code to filter data
}
```

Joining documentation

```
#' [other roxygen code]
#' @param x The name of a database to retrieve
get_data <- function(x) {
  # code to get data
}

#' @rdname get_data
#' @export
filter_table <- function(x) {
  tbl <- get_data(x)
  # code to filter data
}
```

Joining documentation

```
#' [other roxygen code]
#' @param x The name of a database to retrieve
#' @name data_helpers
NULL

#' @rdname data_helpers
#' @export
get_data <- function(x) {
  # code to get data
}

#' @rdname data_helpers
#' @export
filter_table <- function(x) {
  tbl <- get_data(x)
  # code to filter data
}
```

Joining documentation

```r
#' [other roxygen code]
#' @param x The name of a database to retrieve
#' @name data_helpers
NULL

#' @rdname data_helpers
#' @export
get_data <- function(x) {
  # code to get data
}

#' @rdname data_helpers
#' @export
filter_table <- function(x) {
  tbl <- get_data(x)
  # code to filter data
}
```

Your Turn 3

In `R/themes.R`, join the documentation of `theme_avalanche_h()` and `theme_avalanche_v()` to `theme_avalanche()` by replacing the roxygen code for the first two functions with "#' @rdname theme_avalanche".

Make sure both functions still have an export tag, as well!

Re-render the documentation and read the help page for `?theme_avalanche_h()`

Your Turn 3

```r
#' @rdname theme_avalanche
#' @export
theme_avalanche_h <- function(base_size = 14, ...) {
  ggplot2::theme_minimal(base_size = base_size, ...) +
    ggplot2::theme(
      panel.grid.minor = ggplot2::element_blank(),
      panel.grid.major.x = ggplot2::element_blank()
    )
}

#' @rdname theme_avalanche
#' @export
theme_avalanche_v <- function(base_size = 14, ...) {
  ggplot2::theme_minimal(base_size = base_size, ...) +
    ggplot2::theme(
      panel.grid.minor = ggplot2::element_blank(),
      panel.grid.major.y = ggplot2::element_blank()
    )
}
```

Package documentation

`use_package_doc()`

--

usethis also likes this for `use_import_from()` and other metadata

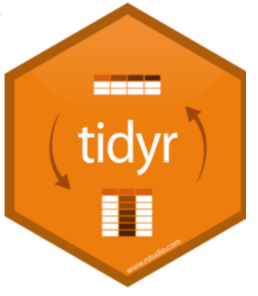
help("tidyr")

tidyr-package {tidyr} R Documentation

tidyr: Tidy Messy Data

Description

Tools to help to create tidy data, where each column is a variable, each row is an observation, and each cell contains a single value. 'tidyr' contains tools for changing the shape (pivoting) and hierarchy (nesting and 'unnesting') of a dataset, turning deeply nested lists into rectangular data frames ('rectangling'), and extracting values out of string columns. It also includes tools for working with missing values (both implicit and explicit).



Author(s)

Maintainer: Hadley Wickham hadley@rstudio.com

Authors:

- Lionel Henry lionel@rstudio.com

Other contributors:

- RStudio [copyright holder]

See Also

Useful links:

- https://tidyr.tidyverse.org
- https://github.com/tidyverse/tidyr
- Report bugs at https://github.com/tidyverse/tidyr/issues

[Package *tidyr* version 1.0.0 Index]

pkgdown

pkgdown

`use_pkgdown()`,
`use_pkgdown_github_pages()`

Renders documentation, README, vignettes, and more as a website.

Here's a good example

https://roxygen2.r-lib.org/

There are a lot more documentation tricks.

Read the vignettes!