

Develop Packages in R

Part 2: Document Package and Functions

April 2019

Isabella Gollini

Isabella.Gollini@ucd.ie

[@IsabellaGollini](#)

Bruna Wundervald [@bwundervald](#)

Chiara Cotroneo [@selenocysteina](#)

Jo Nieć [@joannaniec](#)

[@RLadiesDublin](#)



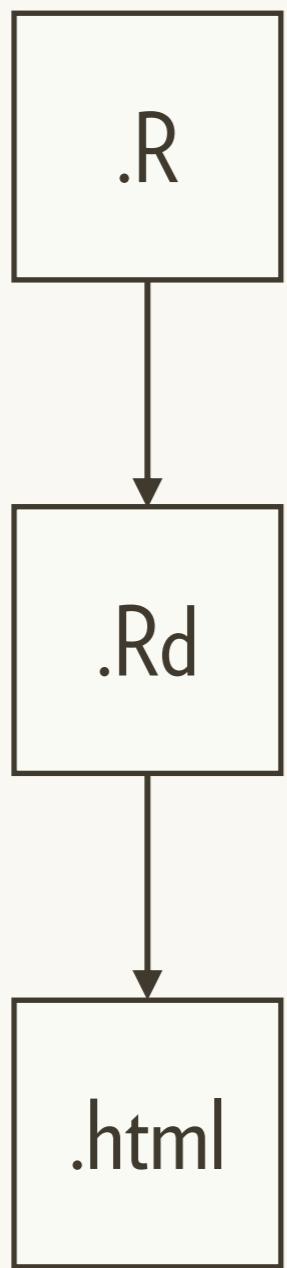
<https://rladies.org/>

<https://forwards.github.io>

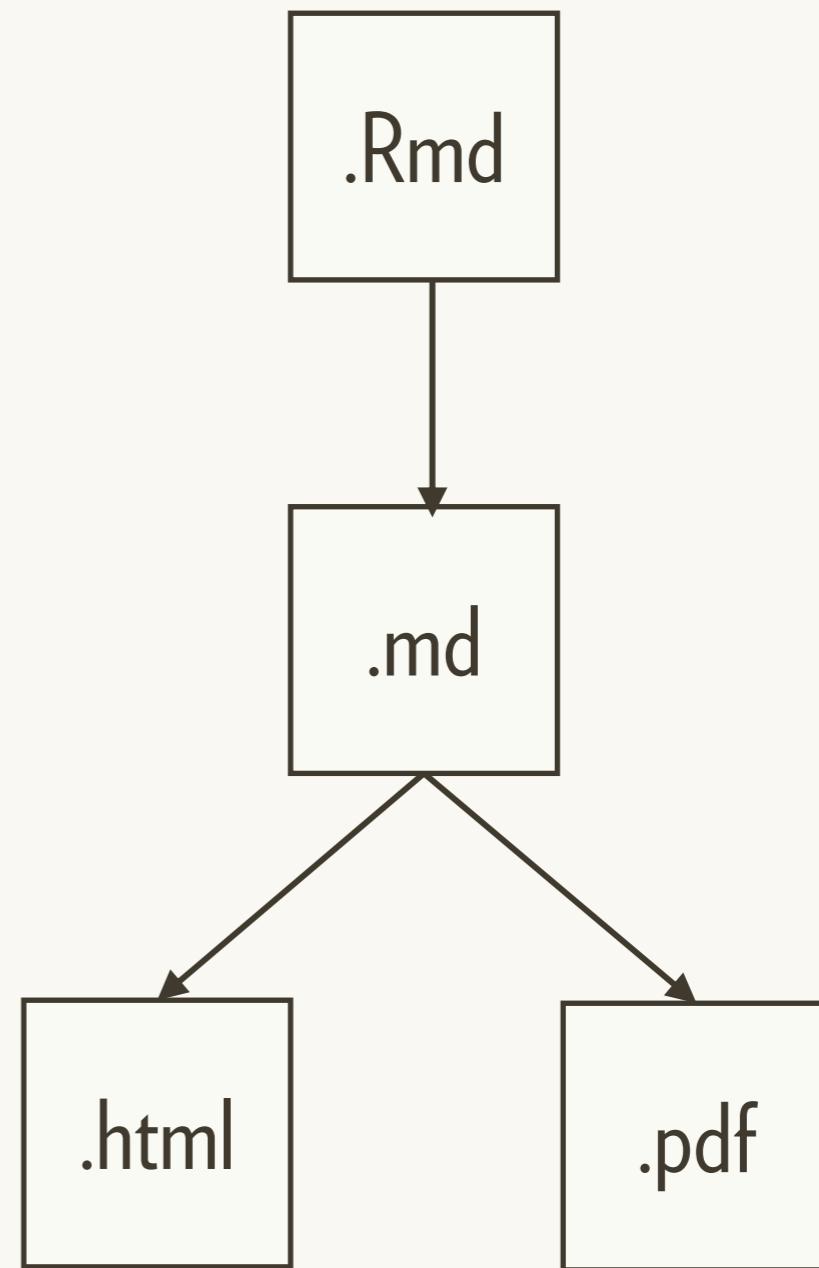
<https://github.com/forwards/workshops/>



Function-level with
roxygen2



Package-level with
rmarkdown



Markdown

I assume you are already familiar with it

Basic markdown formatting

```
# This is a top level heading
```

This is some text. Make text italic with single underscores (or stars). Make it **bold** with double stars (or underscores). Here is a [link to a markdown guide](<http://bit.ly/19fAexE>).

- * This is a list

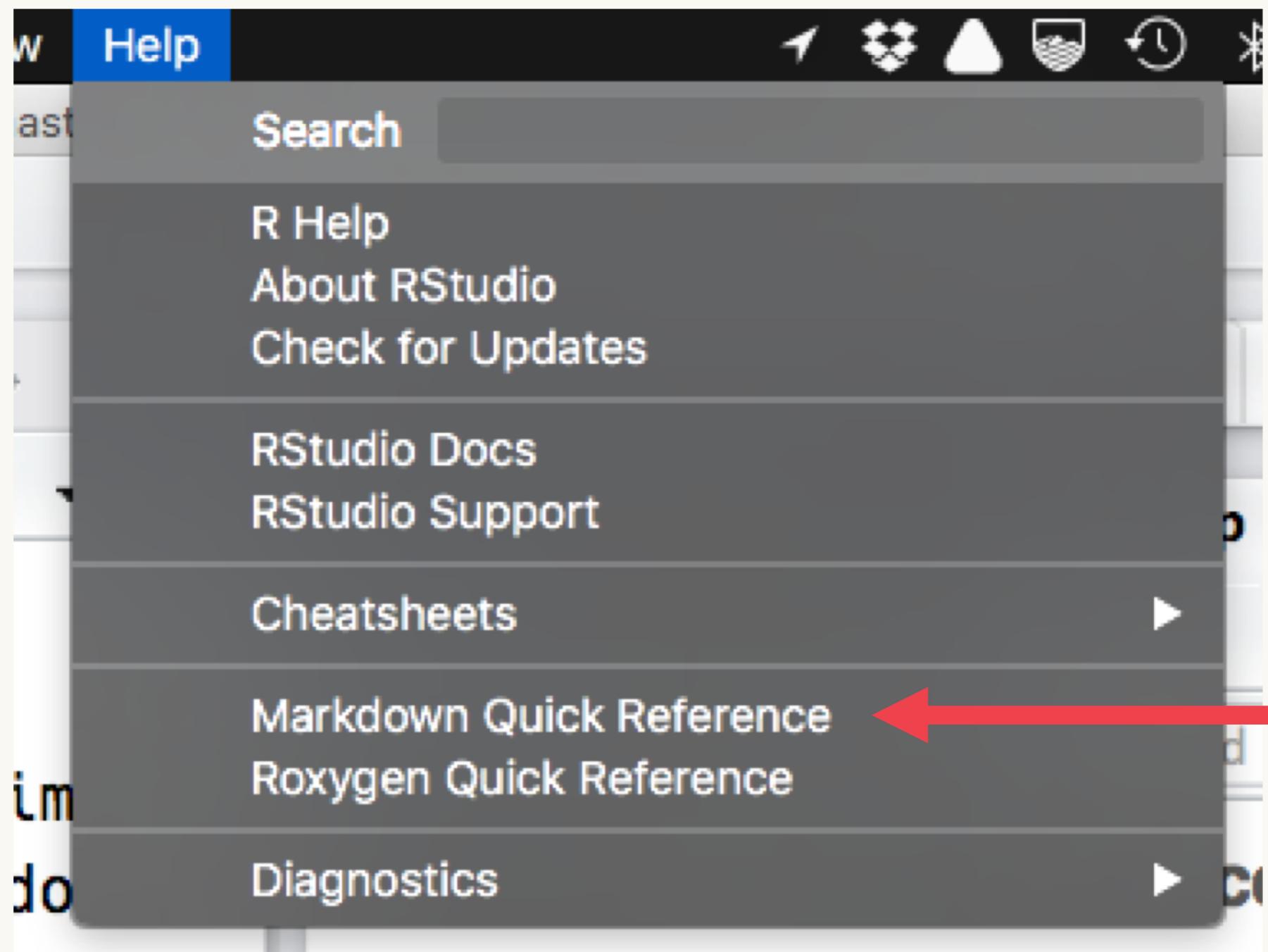
- * This is another item

```
```R
```

```
Some R code
f <- function() x + 1
```
```

```
## This is a secondary heading
```

You can also do `inline code`, numbered lists and quotes and more.



Function documentation with **roxygen2**

Roxygen2

roxygen2

R



You write specially formatted comments in .R

```
#' Add a Column to a Data Frame
#'
#' Allows you to specify the position. Will replace existing variable
#' with the same name if present.
#'
#' @param x A data frame
#' @param name Name of variable to create. If a variable of that name
#'   already exists it will be replaced
#' @param value Values to insert.
#'
#' @param where Position to insert. Use 1 to insert on LHS, or -1 to
#insert on
#'   RHS.
#'
#' @examples
#'
#' df <- data.frame(x = 1:5)
#' add_col(df, "y", runif(5))
#' add_col(df, "y", runif(5), where = 1)
#'
#' add_col(df, "x", 5:1)
```

You write specially formatted comments in .R

```
#' ^ a Frame
#'Roxygen comment
#' Allows you to specify the position. Will replace existing variable
#' with the same name if present.
#
#'@param x variable to create. If a variable of that name
#'@param r already exists it will be replaced
#'@param value Values to insert.
#'@param where Position to insert. Use 1 to insert on LHS, or -1 to
insert on
#'RHS.
#'@examples
#' df <- data.frame(x = 1:5)
#' add_col(df, "y", runif(5))
#' add_col(df, "y", runif(5), where = 1)
#
#'add_col(df, "x", 5:1)
```

Roxygen translates to .Rd

```
% Generated by roxygen2: do not edit by hand
% Please edit documentation in R/add_col.R
\name{add_col}
\alias{add_col}
\title{Add a Column to a Data Frame}
\usage{
add_col(x, name, value, where = -1)
}
\arguments{
\item{x}{A data frame}

\item{name}{Name of variable to create. If a variable of that name
already exists it will be replaced}

\item{value}{Values to insert.}

\item{where}{Position to insert. Use 1 to insert on LHS, or -1 to insert on
RHS.}
}
\description{
Allows you to specify the position. Will replace existing variable
with the same name if present.
}
```

*In almost all cases you
can ignore these files*

`add_col {hadcol}`

R translates to
.html for viewing

Add a Column to a Data Frame

Description

Allows you to specify the position. Will replace existing variable with the same name if present.

Usage

```
add_col(x, name, value, where = -1)
```

Arguments

`x` A data frame

`name` Name of variable to create. If a variable of that name already exists it will be replaced

`value` Values to insert.

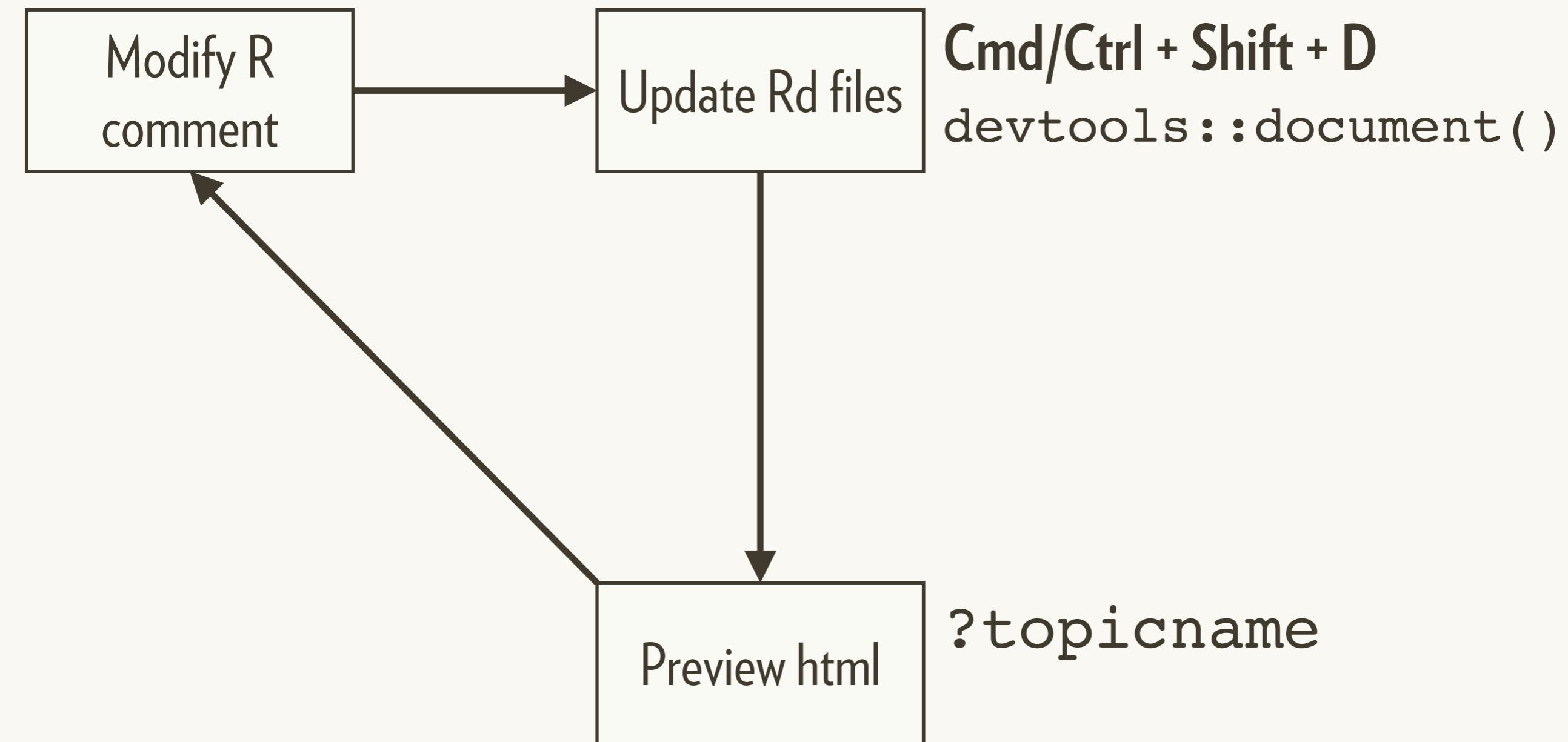
`where` Position to insert. Use 1 to insert on LHS, or -1 to insert on RHS.

Examples

```
df <- data.frame(x = 1:5)
add_col(df, "y", runif(5))
add_col(df, "y", runif(5), where = 1)

add_col(df, "x", 5:1)
```

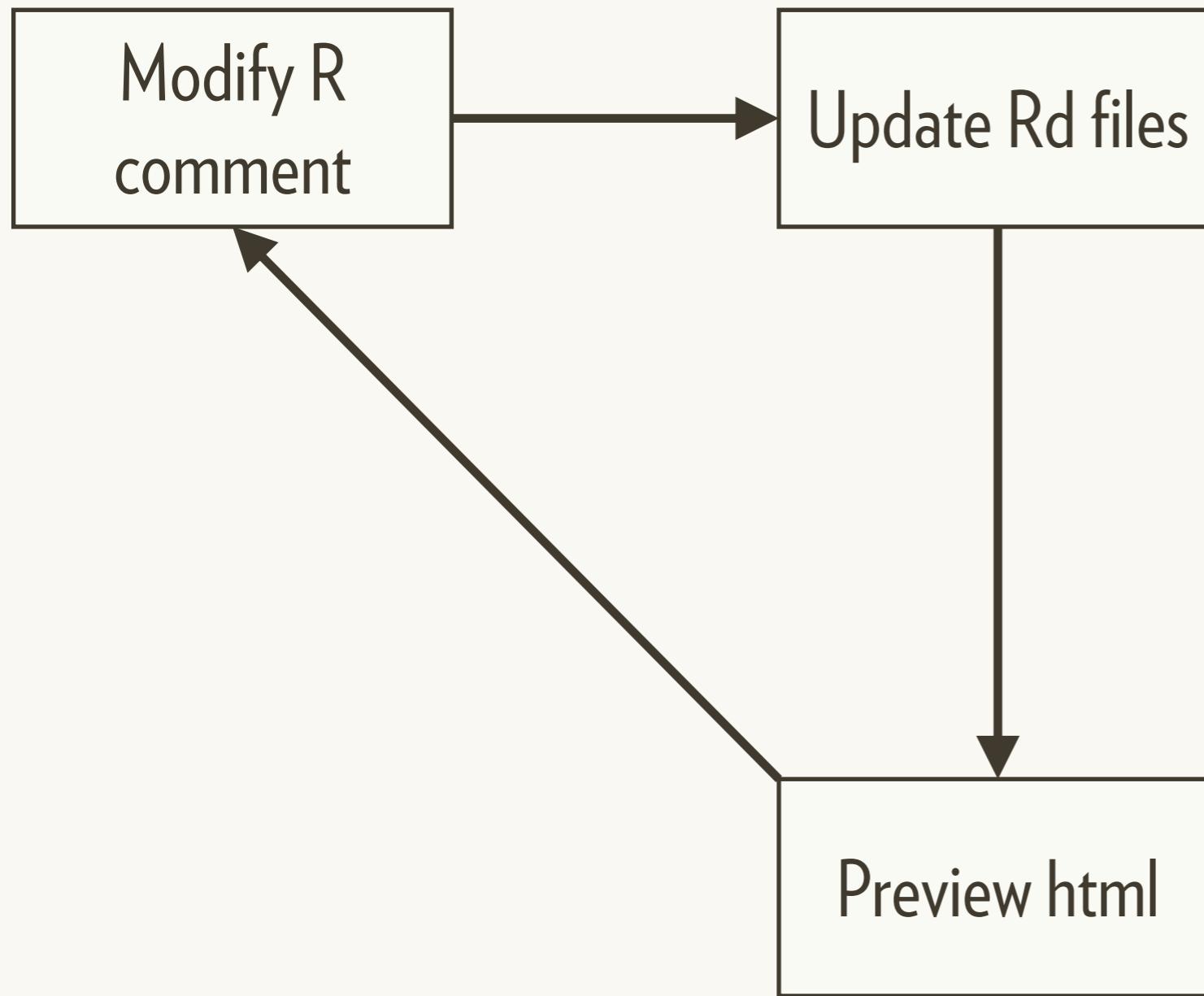
Documentation workflow



Two caveats

1. You must have loaded the package with
`load_all()` at least once.
Check for message "Using development documentation..."
2. This technique only builds individual files
so links do not work.

Documentation workflow



Cmd/Ctrl + Shift + D
`devtools::document()`

NB: You must have loaded the package
with `load_all()` at least once

?topicname
Only shows single file,
so links do not work

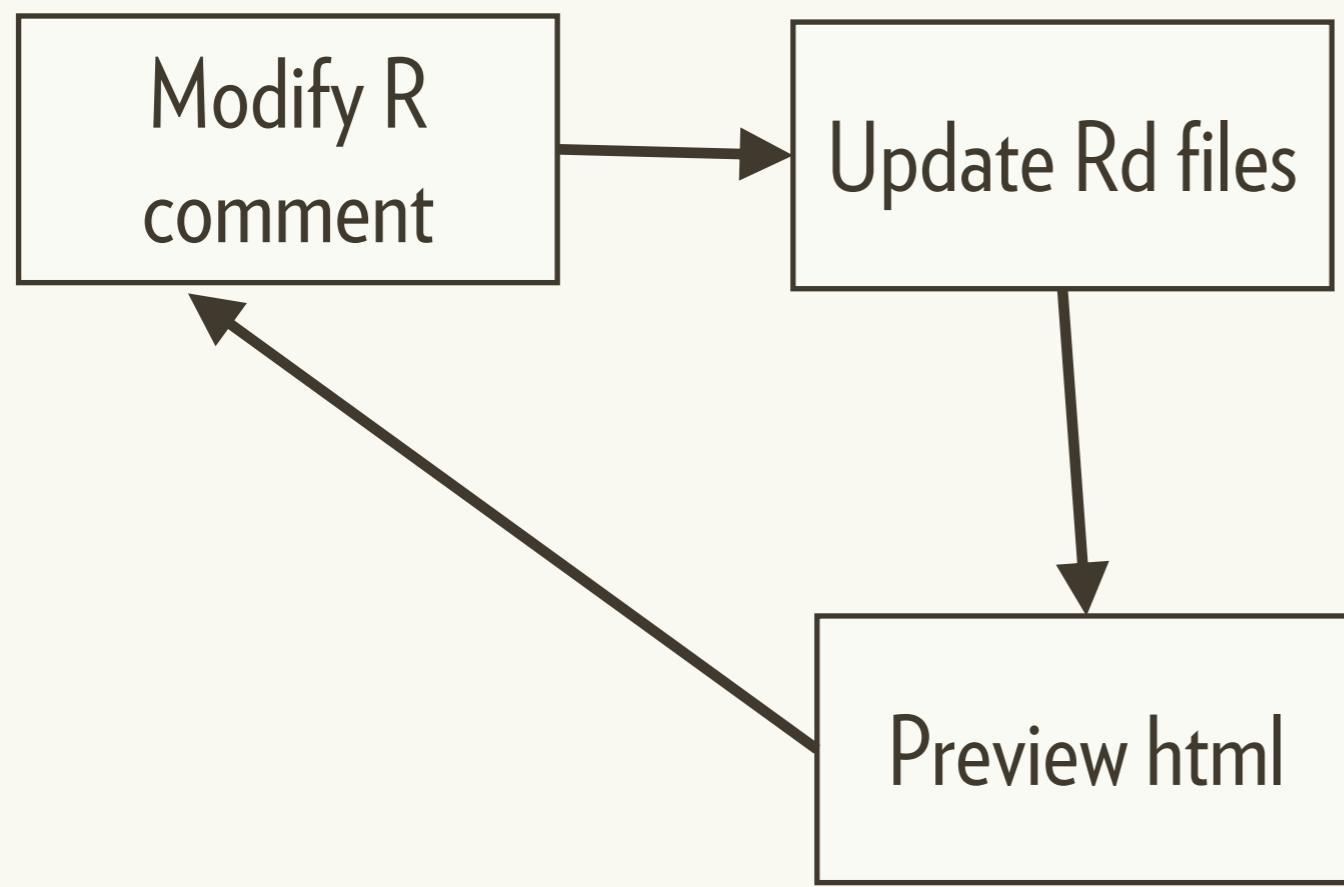
Change working directory/project to:

[hadcol]

<https://github.com/forwards/hadcol>

Your turn

- Fix the typos in the documentation for `add_col`.
- Run the documentation workflow to check your work



Cmd/Ctrl + Shift + D
`devtools::document()`
NB: You must have loaded the package
with `load_all()` at least once

?topicname
Only shows single file,
so links do not work



First sentence is the title

Sum of Vector Elements

Description

`sum` returns the sum of all the values present in its arguments.

Next paragraph is the description

Usage

```
sum(..., na.rm = FALSE)
```

Arguments

... numeric or complex or logical vectors.

`na.rm` logical. Should missing values (including `NaN`) be removed?

Details

Everything else is the details

ectly or via the [Summary](#) group
ld be unnamed, and dispatch is

If `na.rm` is `FALSE` an `NA` or `NaN` value in any of the arguments will cause a value of `NA` or

The description block

First sentence is the **title**

```
#' Sum of vector elements  
#'  
#' \code{sum} returns the sum of all the values present in its arguments.  
#'  
#' This is a generic function: methods can be defined for it directly or via the  
#' \code{\link{Summary}} group generic. For this to work properly, the arguments  
#' \code{...} should be unnamed, and dispatch is on the first argument.
```

Next paragraph is the **description**

Everything else is the **details**

There are five **tags** you'll use for most functions

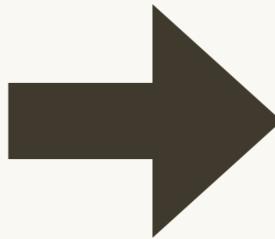
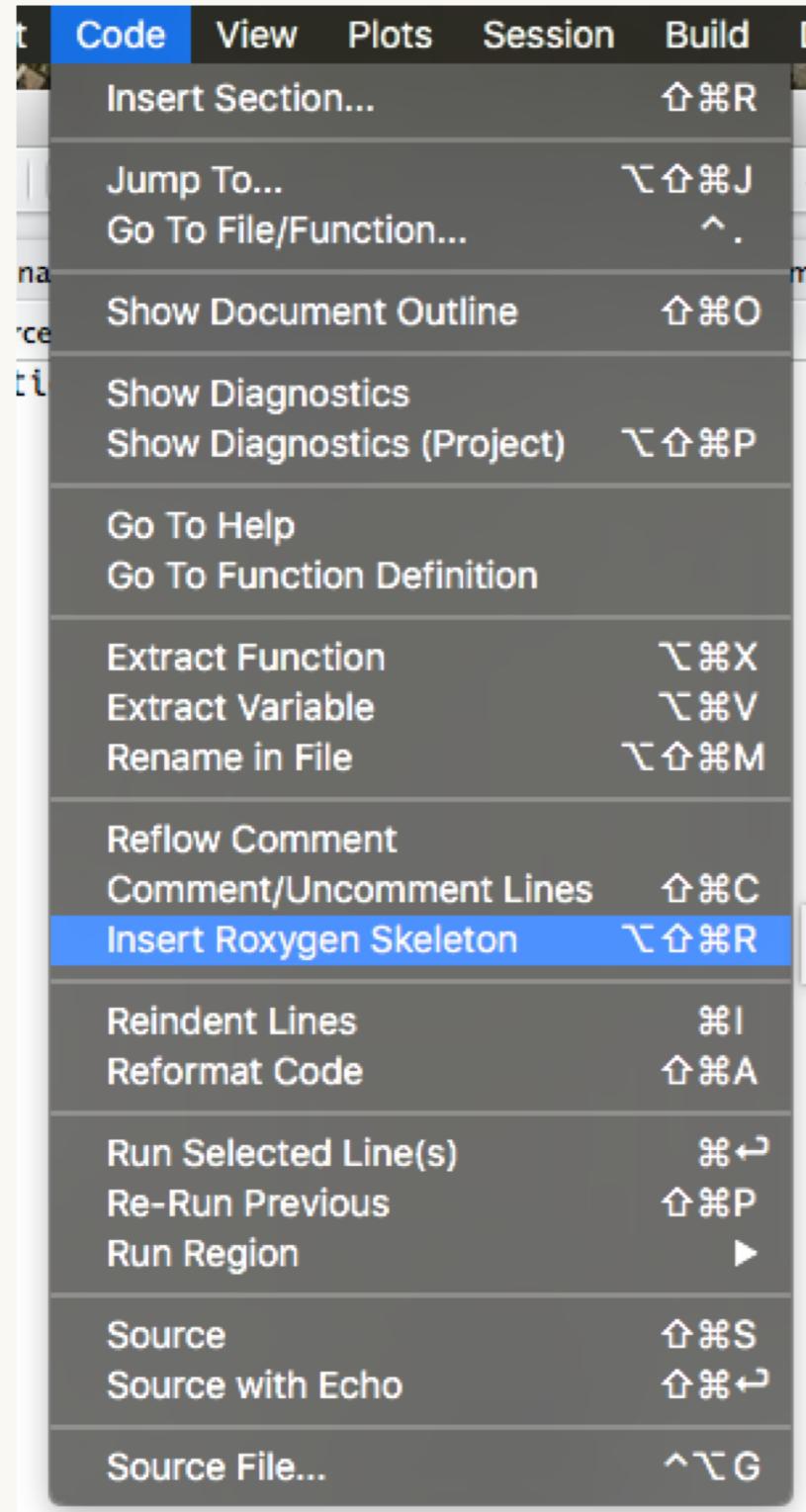
| Tag | Purpose |
|------------|--|
| @param arg | Describe inputs |
| @examples | Show how the function works.
(Usual RStudio shortcuts work) |
| @seealso | Pointers to related functions |
| @return | Describe outputs (value) |
| @export | Is this a user-visible function? |

Your turn

Document `add_cols()`.

(See next slide for hint)

RStudio helps you remember



```
#' Title
#'
#' @param x
#' @param y
#' @param z
#'
#' @return
#' @export
#'
#' @examples
fun <- function(x, y, z) {  
}  
}
```

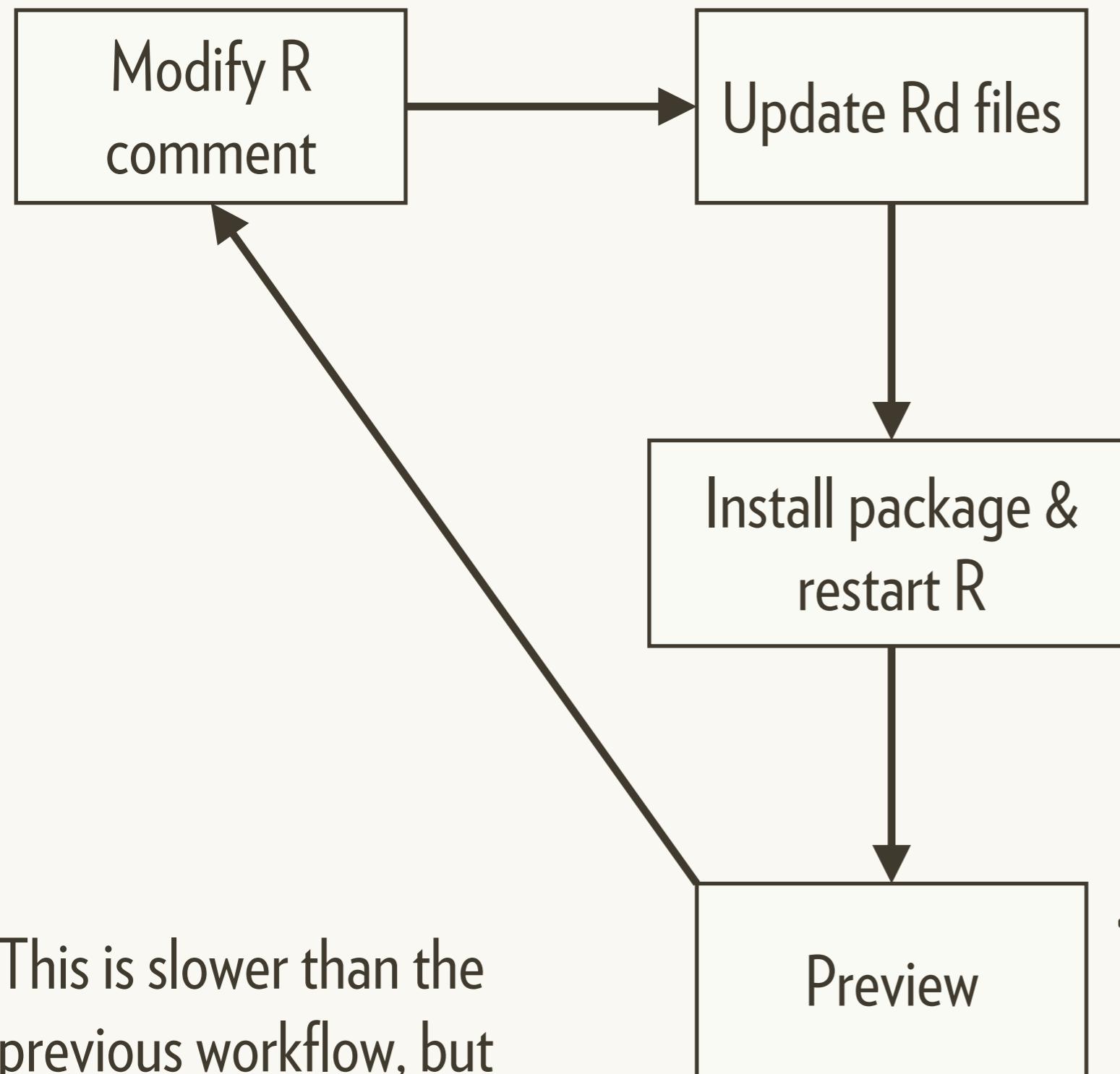
Use markdown for formatting

```
# Activate by running
# use_roxygen_md()

**bold**, _italic_, `code`  
  

* [func() ]
* [pkg::func() ]
* [link text][func() ]
* [link text][pkg::func() ]
```

Documentation workflow 2



Cmd/Ctrl + Shift + D
`devtools::document()`

Cmd/Ctrl + Shift + B

 **Install and Restart**

?topicname

This is slower than the previous workflow, but there are fewer caveats

Your turn

- Make real link to `cbind()`
- Add a see also section (`@seealso`) to `add_col()` and `add_cols()` that links them together.
- What happens if you add `@family xyz` to both?

roxygen2 comes with other tools to reduce duplication

```
# Document multiple functions in the same file
#' @rdname add_col

# Inherit the parameter descriptions from
# another function
#' @inheritParams add_col

# Inherit everything from another topic
#' @inherit add_col

# Inherit selectively
#' @inherit add_col parameters return references
#'   title description details
#'   sections seealso
```

Read online about how to document other objects

Data

<http://r-pkgs.had.co.nz/data.html#documenting-data>

Classes & methods

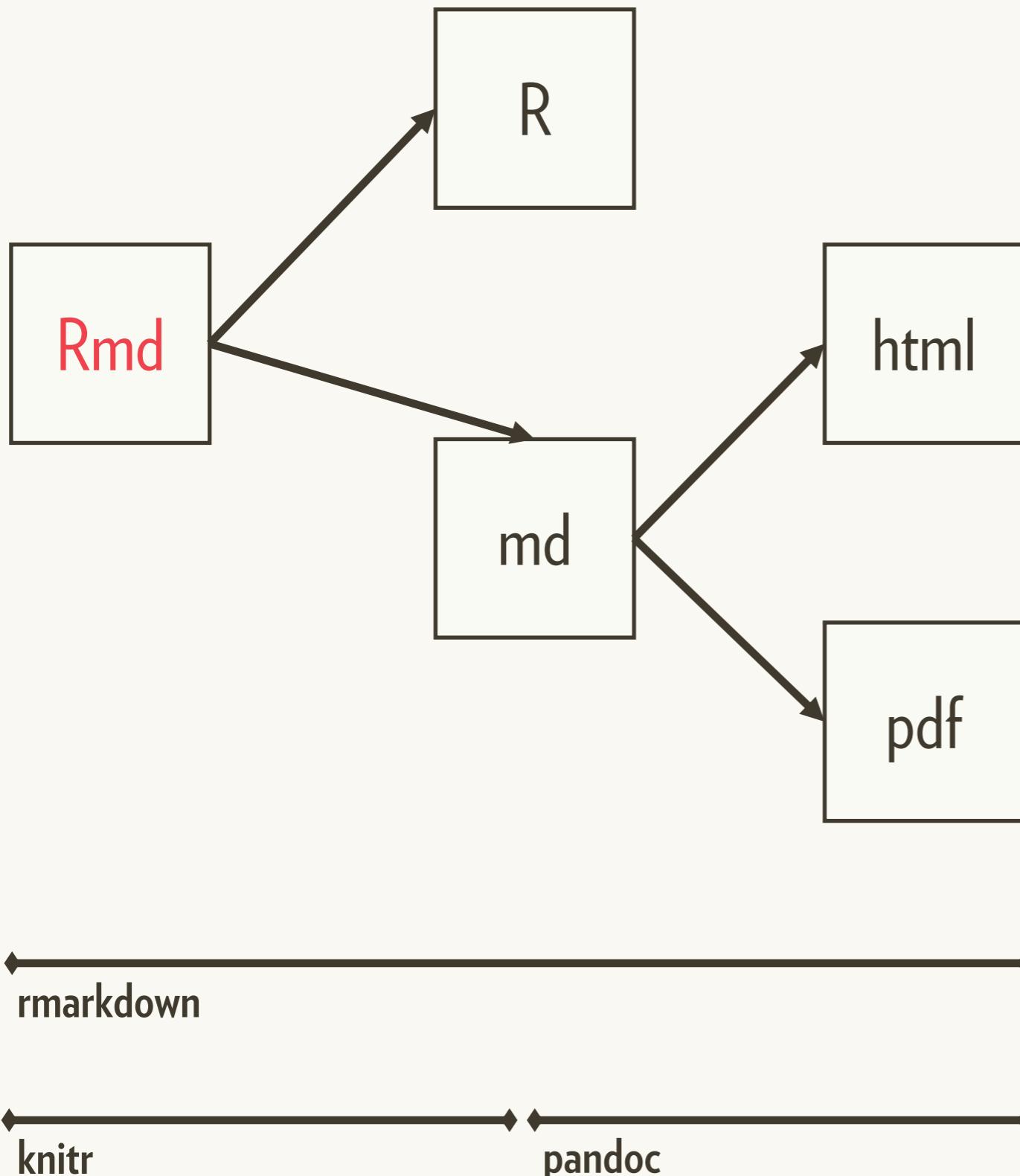
<http://r-pkgs.had.co.nz/man.html#man-classes>

Packages

<http://r-pkgs.had.co.nz/man.html#man-packages>

Package documentation with
rmarkdown

Vignettes



Lets you combine prose and code to explain your how you package works.

The hard part is the writing, not the technology!

Easiest way to get started is with `use_vignette()`

```
usethis::use_vignette("name")
```

```
# Adds to DESCRIPTION
```

```
Suggests: knitr
```

```
VignetteBuilder: knitr
```

```
# Creates vignettes/
```

```
# Drafts vignettes/name.Rmd
```

Vignette = Rmarkdown + special metadata

```
---
```

```
title: "Vignette Title"
author: "Vignette author"
date: ``r Sys.Date()``
output: rmarkdown::html_vignette
vignette: >
  %\VignetteIndexEntry{Vignette Title}
  %\VignetteEngine{knitr::rmarkdown}
  %\VignetteEncoding{UTF-8}
```

```
---
```

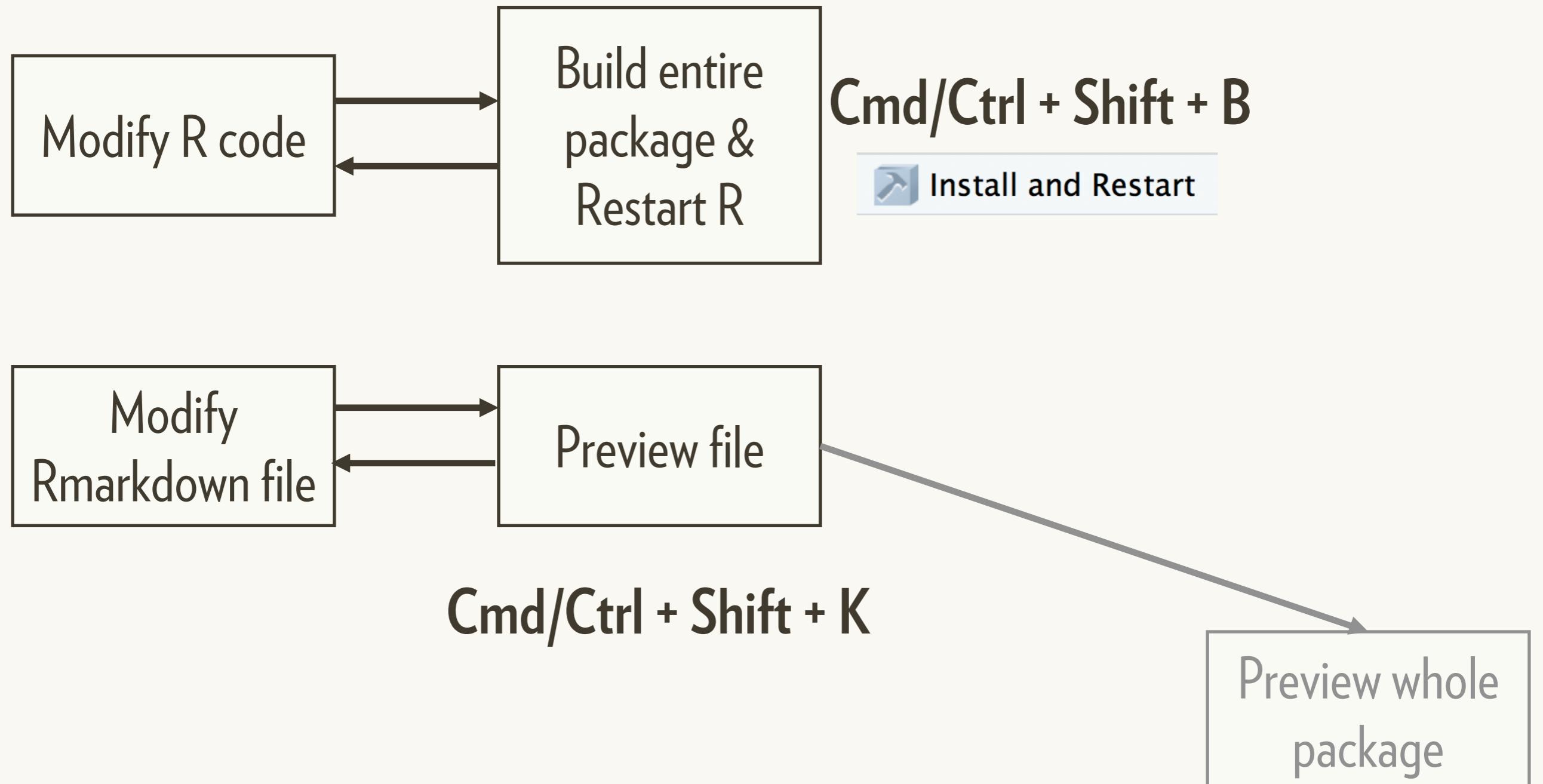
Special output format for vignettes

Special metadata needed by R

Vignettes are long form documentation commonly included in packages. Because they are part of the distribution of the package, they need to be as compact as possible. The `html_vignette` output type provides a custom style sheet (and tweaks some options) to ensure that the resulting html is as small as possible. The `html_vignette` format:

...

Vignette workflow



`devtools::install(build_vignettes = TRUE)`

`browseVignettes()`

Your turn

Create a vignette that shows how to use `add_col()` for adding and removing.

Fix the “vignette title”

README

If sharing with others, include a readme

```
# Your choice: but often useful to include  
# results of running code  
usethis::use_readme_md()  
usethis::use_readme_rmd()  
  
# For public projects this should include a  
# brief overview, instructions on how to  
# install, and a few examples. For private  
# projects, this is a great place to jot down  
# notes
```

NEWS

Also good idea to track changes

```
usethis::use_news_md()
```

Package website with `pkgdown`

Build a Website

`pkgdown::build_site()`

<http://pkgdown.r-lib.org/>

Home page with “home” icon:

- automatically generated from one of the following four files: index.Rmd; README.Rmd; index.md; README.md

Reference:

- The function reference generates one page for each .Rd file in man/, by default generate an overall index, which is an alphabetically ordered list of functions.

If the files are available in the package:

Articles:

- automatically build the .Rmd vignettes

News:

- if NEWS.md is present

Get Started

- if you have an article with the same name as the package.

A link to your your github repo (if listed in the DESCRIPTION url field).

*More options are
available!*

This work is licensed under the
**Creative Commons Attribution-Noncommercial 3.0
United States License.**

To view a copy of this license, visit

<http://creativecommons.org/licenses/by-nc/3.0/us/>