Documentation

September 2019

Angela Li
Samantha Toet

Workshop materials: `bit.ly/cville_pkg`

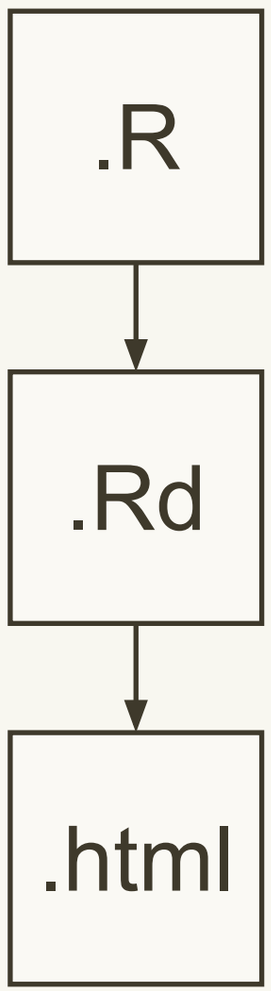
Why?

People need instructions to use new things!

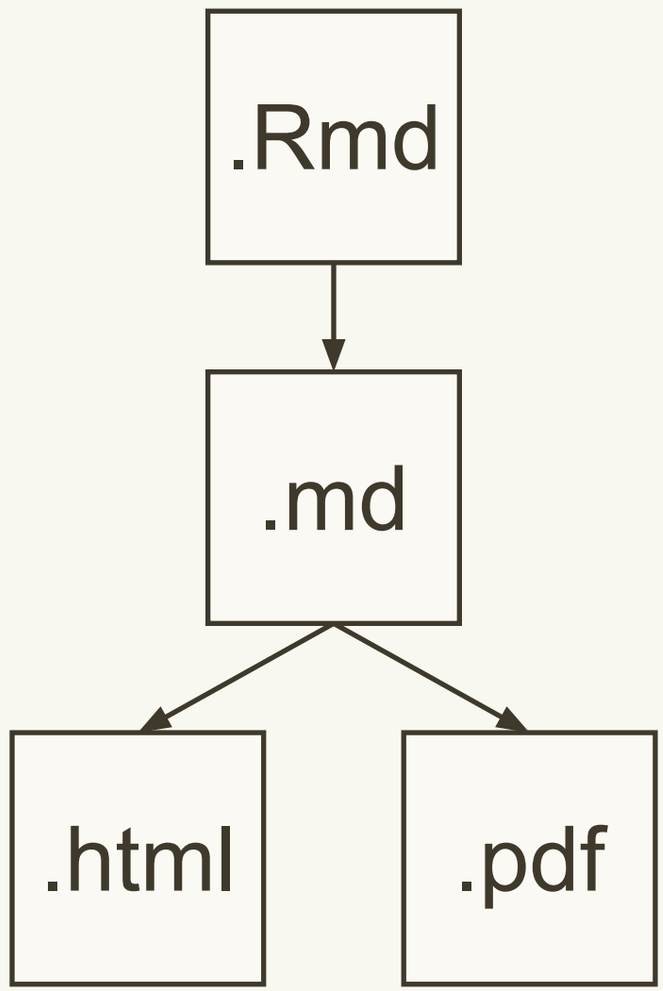
You might want instructions to remind you how your tools work too.

Documentation is the way you preserve the information about your tools.

Function-level with roxygen2



Package-level with rmarkdown

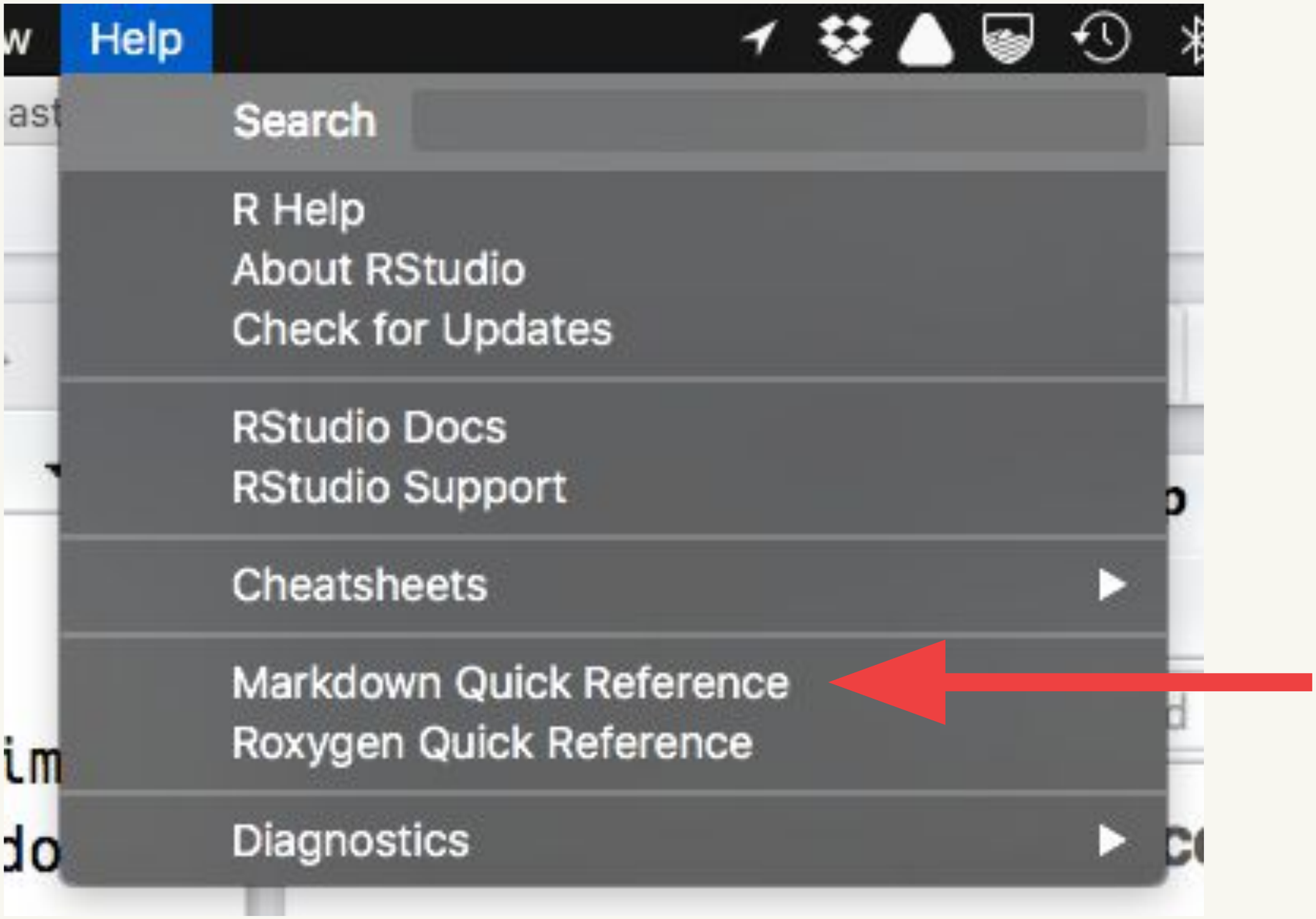


Markdown

Quick overview

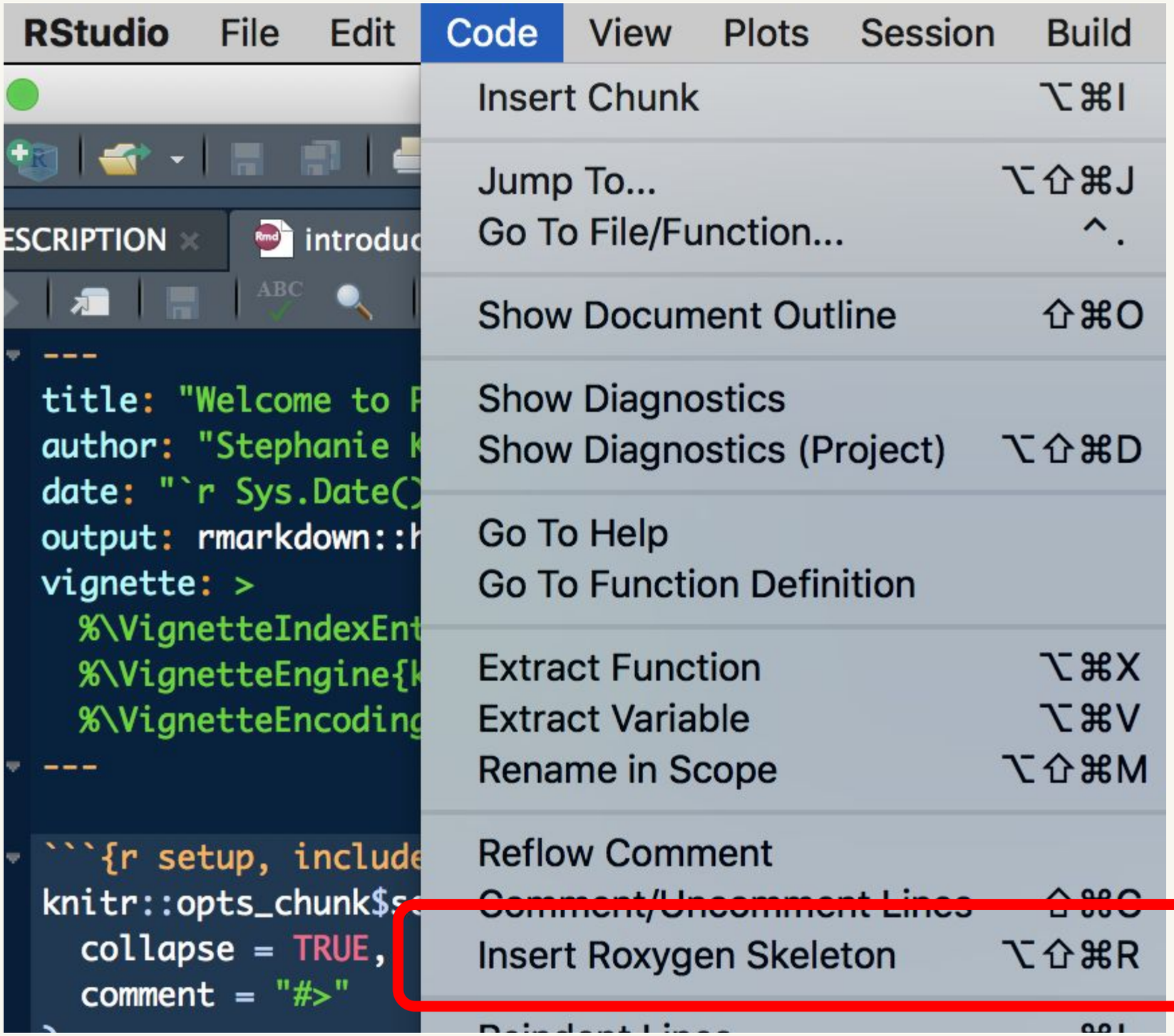
Basic markdown formatting

````
# 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.
````


Help
Search
R Help
About RStudio
Check for Updates
RStudio Docs
RStudio Support
Cheatsheets
Markdown Quick Reference
Roxygen Quick Reference
Diagnostics

Document Each Function

Roxygen2



Roxygen allows us to explain the function's parts...

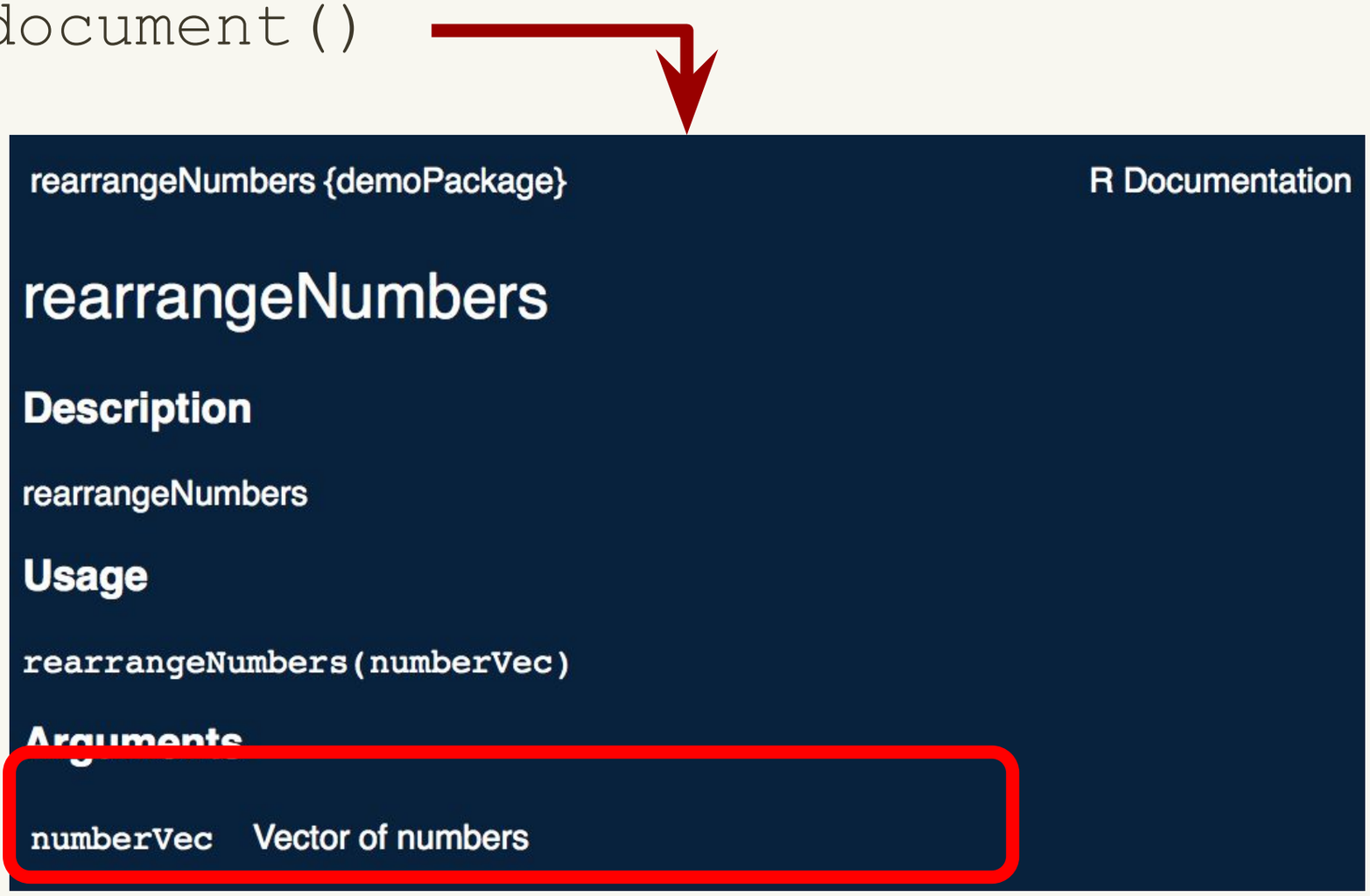
http://r-pkgs.had.co.nz/man.html

You write specially formatted comments in .R

```
#' @param numberVec Vector of numbers
```

Starts with the type of element, then you describe it in reasonable human language

`devtools::document()`



Your turn

Open up a .R function file, either from the provided examples or one of your own.

Does it have a Roxygen header?
If so, can you find a parameter and read the documentation?

Roxygen examples

One of the most useful things to put in your Roxygen header is a working example of your function, so the user can see how to run it.

```
#' @examples
#' \dontrun{
#'   FILL IN YOUR CODE HERE
#'   REMEMBER TO COMMENT EACH LINE
#' }
```

Roxygen renders to .Rd in /man/ folder

In almost all cases you can ignore these files

```
% 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.}
```

add_col {hadcol}

Add a column to a data frame

R translates .Rd into .html for viewing

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)
```

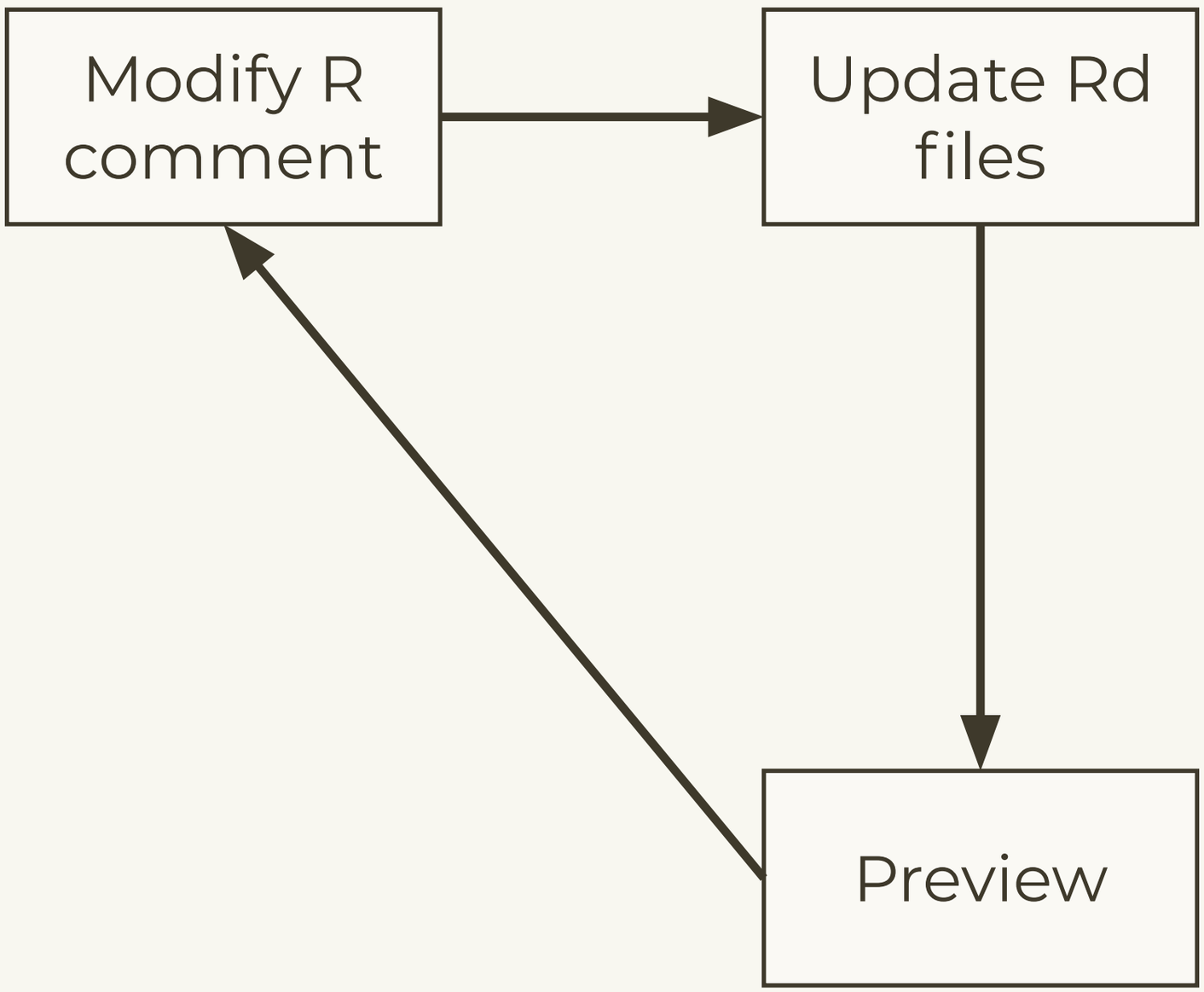
Arguments

`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



Cmd/Ctrl + Shift + D

devtools::document()

?topicname

Your turn

Now, open the .R file for a function and:

- if it does not have a Roxygen header, add one

- if it is one of the functions for this workshop, add a working `example`.

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?

You can use markdown for formatting

```
# Activate by running
# use_roxygen_md()

**bold**,  italic , `code`

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

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