Code Commenting & Documentation

Ayako Okuhama
Lisa Weissburg
Nazia Binte Ali
Kelsey Kinderknecht
Prahlad Govinda Krishnan
Khondoker Nazmoon Nabi

Outline of Presentation

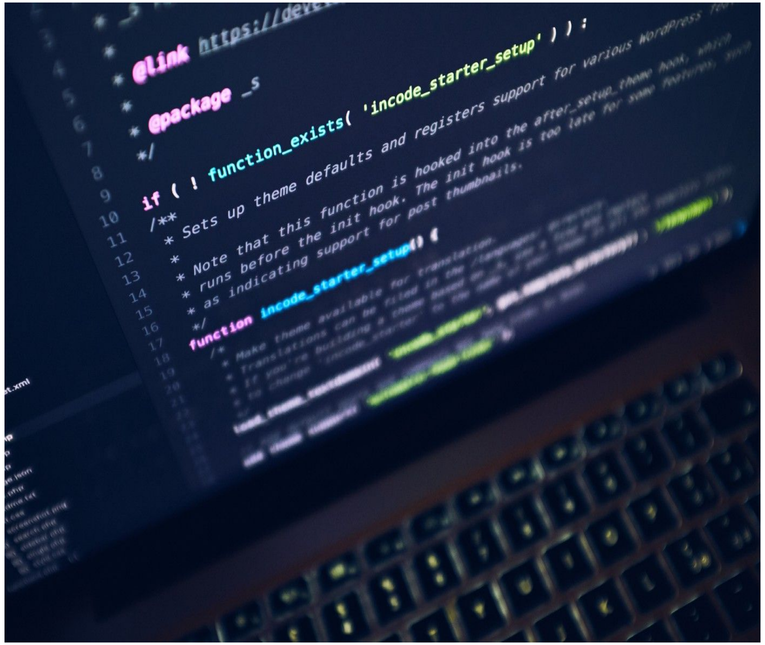
Code commenting

What is Code Commenting?

The practice of sprinkling short, normally single-line notes throughout your code. These notes are called comments.

They explain how your program works, and your intentions behind it.

Why Code Commenting?

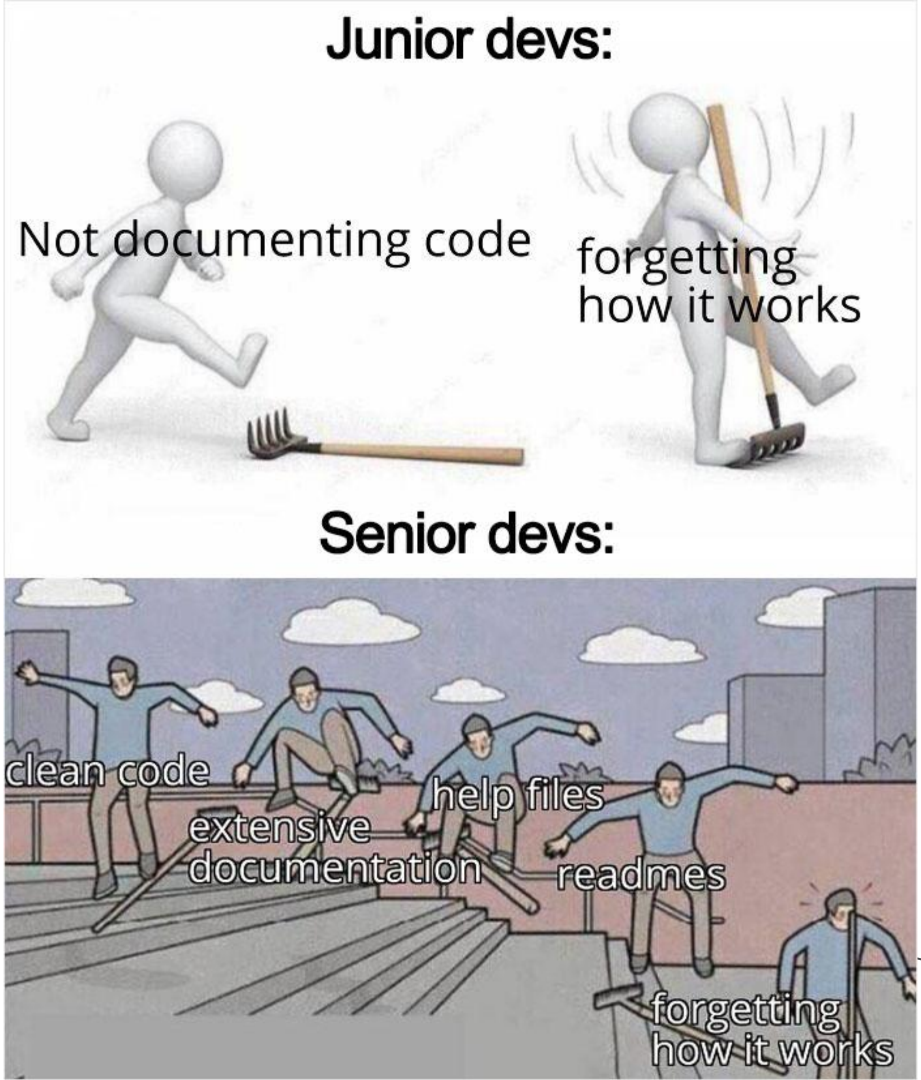


- Save time
- Help yourself, future developers, or collaborators understand what is going on
- Help debug
- Easily come back to certain parts of code
- Show your colleagues and other developers how awesome you are

https://medium.com/swlh/10-reasons-why-you-should-place-comments-in-your-code-96783a69ceda

“Code tells you how, comments tell you why.”

- Jeff Atwood, co-founder of Stack Overflow

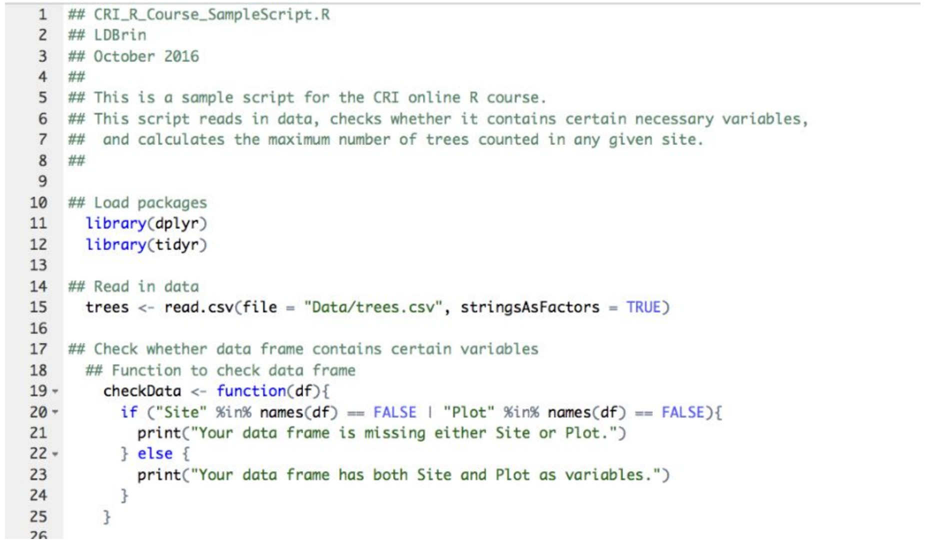

Junior devs:
Not documenting code
forgetting how it works
Senior devs:
clean code
extensive documentation
help files
readmes
forgetting how it works

Best practices

Things to do, for ease of following scripts:

The basics:

- Use a # sign followed by a space
- Use comments to mark off sections of the code
- Provide enough commentary for you and someone else to know why you are doing what you are doing



```
## CRI_R_Course_SampleScript.R
## LDBrin
## October 2016
##
## This is a sample script for the CRI online R course.
## This script reads in data, checks whether it contains certain necessary variables,
##  and calculates the maximum number of trees counted in any given site.
##

## Load packages
  library(dplyr)
  library(tidyr)

## Read in data
  trees <- read.csv(file = "Data/trees.csv", stringsAsFactors = TRUE)

## Check whether data frame contains certain variables
  ## Function to check data frame
    checkData <- function(df){
      if ("Site" %in% names(df) == FALSE | "Plot" %in% names(df) == FALSE){
        print("Your data frame is missing either Site or Plot.")
      } else {
        print("Your data frame has both Site and Plot as variables.")
      }
    }
```

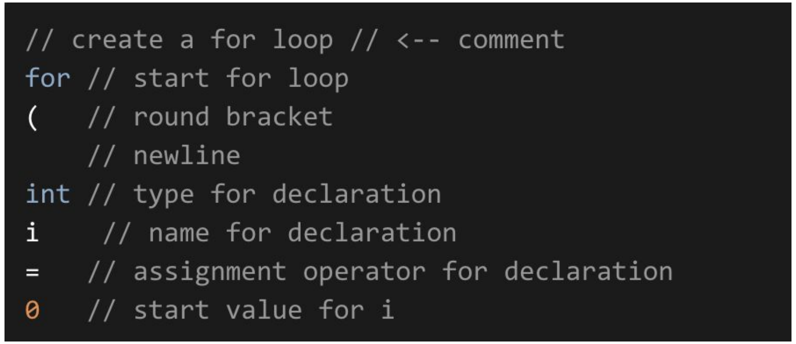
Best practices!

- Avoid duplicating the code with comments
- Concise, informative and easy to understand
- Instead of writing lengthy comments, codes should be improved and simplified

Source: https://stackoverflow.blog/2021/12/23/best-practices-for-writing-code-comments/

Things to <u>not</u> do:

- Avoid OVER comment, to prevent clutter



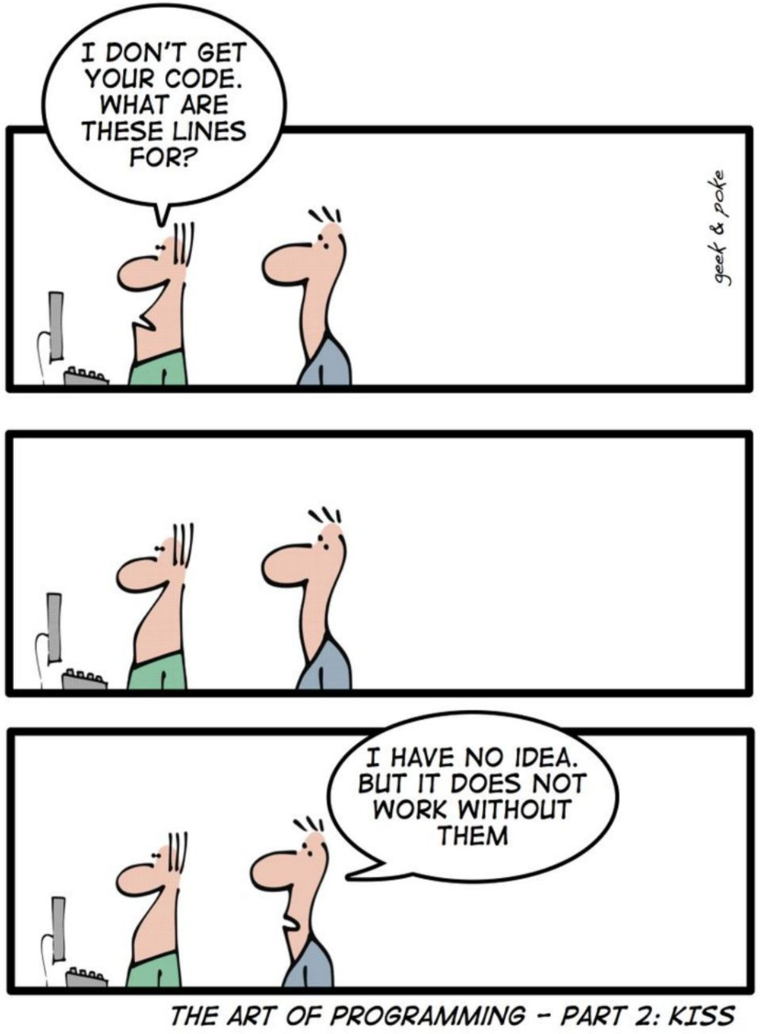
```
// create a for loop // <-- comment
for // start for loop
(   // round bracket
    // newline
int // type for declaration
i    // name for declaration
=   // assignment operator for declaration
0   // start value for i
```

Source: https://stackoverflow.blog/2021/12/23/best-practices-for-writing-code-comments/

Best practices!

- If using code from someone else, provide links to the original source for reference as a comment
- Include links to external references to better understand the code, if still learning what the code does
- Add comments when fixing bugs

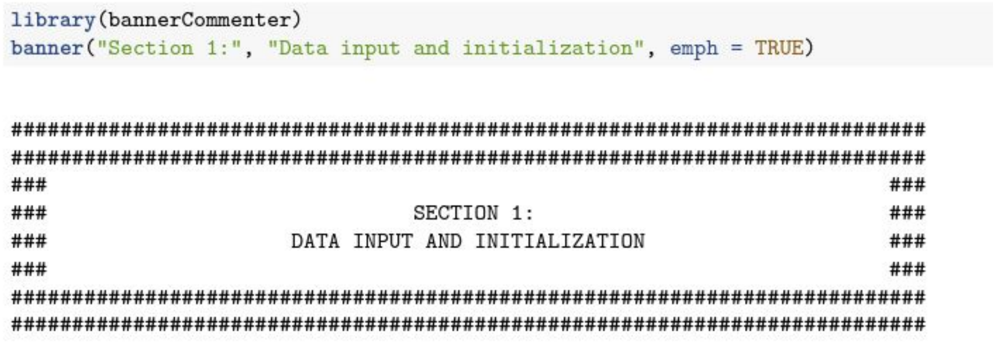
Source: https://stackoverflow.blog/2021/12/23/best-practices-for-writing-code-comments/


I DON'T GET YOUR CODE. WHAT ARE THESE LINES FOR?
geek & poke
I HAVE NO IDEA. BUT IT DOES NOT WORK WITHOUT THEM
THE ART OF PROGRAMMING – PART 2: KISS

Source:
https://stackoverflow.blog/2021/12/23/best-practices-for-writing-code-comments/

Things to do, for ease of following scripts:

- Annotate the beginning of script file with a comment that gives the reader description of what the code does
- For more sophistication and formatting, use package ***bannerCommenter***

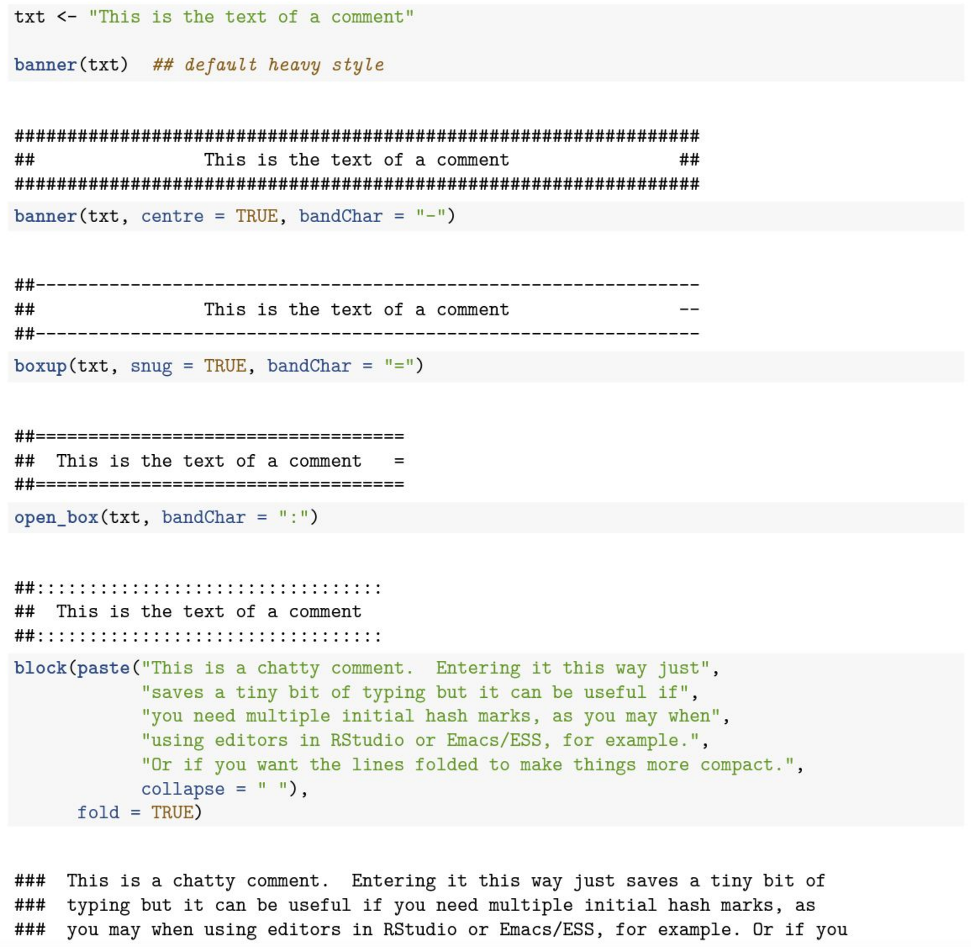


```r
library(bannerCommenter)
banner("Section 1:", "Data input and initialization", emph = TRUE)
```

```
###########################################################################
###########################################################################
###                                                                     ###
###                             SECTION 1:                              ###
###                    DATA INPUT AND INITIALIZATION                    ###
###                                                                     ###
###########################################################################
###########################################################################
```

Source: https://cran.r-project.org/web/packages/bannerCommenter/vignettes/Banded_comment_maker.pdf

Other uses of the bannerCommenter package



```
txt <- "This is the text of a comment"

banner(txt)  ## default heavy style
```

```
##################################################################
##                This is the text of a comment                 ##
##################################################################
```

```
banner(txt, centre = TRUE, bandChar = "-")
```

```
##----------------------------------------------------------------
##                This is the text of a comment                 --
##----------------------------------------------------------------
```

```
boxup(txt, snug = TRUE, bandChar = "=")
```

```
##===================================
##  This is the text of a comment   =
##===================================
```

```
open_box(txt, bandChar = ":")
```

```
##:::::::::::::::::::::::::::::::::
##  This is the text of a comment
##:::::::::::::::::::::::::::::::::
```

```
block(paste("This is a chatty comment.  Entering it this way just",
            "saves a tiny bit of typing but it can be useful if",
            "you need multiple initial hash marks, as you may when",
            "using editors in RStudio or Emacs/ESS, for example.",
            "Or if you want the lines folded to make things more compact.",
            collapse = " "),
      fold = TRUE)
```

```
###  This is a chatty comment.  Entering it this way just saves a tiny bit of
###  typing but it can be useful if you need multiple initial hash marks, as
###  you may when using editors in RStudio or Emacs/ESS, for example. Or if you
```

Source: https://cran.r-project.org/web/packages/bannerCommenter/vignettes/Banded_comment_maker.pdf

Documentation

“Documentation is a love letter that you write to your future self.”

- Damian Conway, computer scientist

What is Code Documentation?

- Process through which programmers document their code
- Text and/ or pictures that describes what the code does and how it can be used
- Can be in the form of a simple explanation of the code, a comprehensive user handbook, or illustrative images
- Commentary on how the software works, includes use cases, and cites any relevant sources

https://www.madcapsoftware.com/blog/write-code-documentation/#:~:text=What%20Is%20Code%20Documentation%3F,readability%2C%20reproducibility%2C%20and%20usability

https://www.r-bloggers.com/2019/04/writing-r-documentation-simplified/

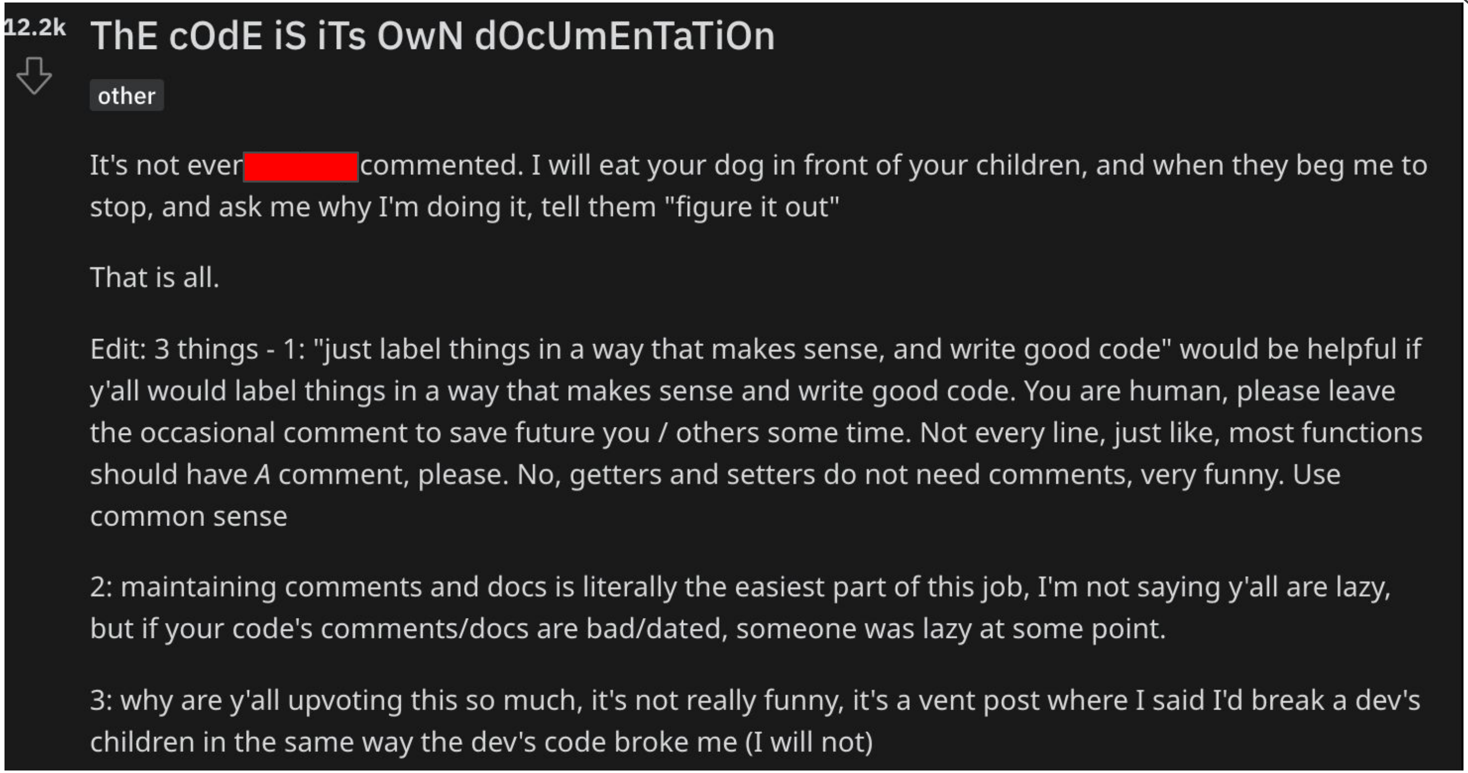
Why Code Documentation?

- Enhances readability, reproducibility and usability
- Communication to both technical and non-technical audiences
- Makes the code easier to maintain and update
- Aids in learning from mistakes
- Saves time in the long run

https://www.madcapsoftware.com/blog/write-code-documentation/#:~:text=What%20Is%20Code%20Documentation%3F,readability%2C%20reproducibility%2C%20and%20usability

https://www.r-bloggers.com/2019/04/writing-r-documentation-simplified/

https://www.linkedin.com/pulse/why-documentation-important-software-development-alexander-ryan/



12.2k

ThE cOdE iS iTs OwN dOcUmEnTaTiOn

other

It's not ever [redacted] commented. I will eat your dog in front of your children, and when they beg me to stop, and ask me why I'm doing it, tell them "figure it out"

That is all.

Edit: 3 things - 1: "just label things in a way that makes sense, and write good code" would be helpful if y'all would label things in a way that makes sense and write good code. You are human, please leave the occasional comment to save future you / others some time. Not every line, just like, most functions should have *A* comment, please. No, getters and setters do not need comments, very funny. Use common sense

2: maintaining comments and docs is literally the easiest part of this job, I'm not saying y'all are lazy, but if your code's comments/docs are bad/dated, someone was lazy at some point.

3: why are y'all upvoting this so much, it's not really funny, it's a vent post where I said I'd break a dev's children in the same way the dev's code broke me (I will not)

https://www.reddit.com/r/ProgrammerHumor/comments/yr9j4b/the_code_is_its_own_documentation/

Best practices

Dos and Don'ts of Code Documentation

✓ **DO**

Choose a documentation tool that can be updated easily, has version control capabilities, and can be accessed by current/future collaborators

✓ **DO**

Document code as you write it

✓ **DO**

Use documentation to increase usability, transparency, and reproducibility of code

✗ **DON'T**

Keep your code documentation in a file only you have access to

✗ **DON'T**

Forget about code documentation until you get a request for it

✗ **DON'T**

Repeat what has been said in your code comments

Sources:

1. Wilson G, Aruliah DA, Brown CT, Chue Hong NP, Davis M, Guy RT, et al. (2014) Best Practices for Scientific Computing. PLoS Biol 12(1): e1001745. https://doi.org/10.1371/journal.pbio.1001745
2. Berkeley Library. "How to Write a Good Documentation." University of California. https://guides.lib.berkeley.edu/how-to-write-good-documentation
3. Bryan J. (2017). Excuse me, do you have a moment to talk about version control? PeerJ Preprints 5:e3159v2 https://doi.org/10.7287/peerj.preprints.3159v2

Examples

What and How to document!

What ?

- Author, Date, Version
- Packages
- Projects/Datasets
- Functions
- Classes
- Any tricky lines of codes
- References

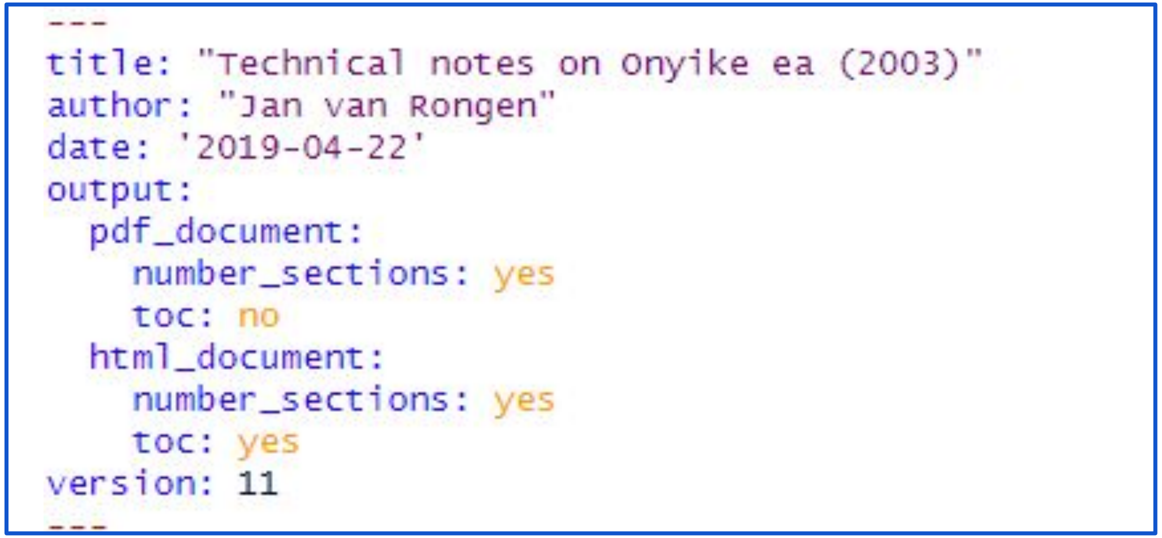
How?

- Write vignette with **R markdown**
- Write comments alongside the codes using **roxygen2** syntax for **man/files.** (link)
    - Make project using package **usethis** & **devtools**
- Create a website with **pkgdown**

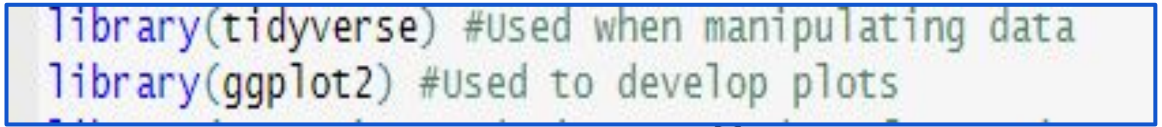
Source: https://stackoverflow.blog/2021/12/23/best-practices-for-writing-code-comments/
https://combine-australia.github.io/r-pkg-dev/documenting-functions.html#adding-documentation

Example: R markdown file

Title,
Author,
Date,
Version,
Output,
Package



```
---
title: "Technical notes on Onyike ea (2003)"
author: "Jan van Rongen"
date: '2019-04-22'
output:
  pdf_document:
    number_sections: yes
    toc: no
  html_document:
    number_sections: yes
    toc: yes
version: 11
---
```

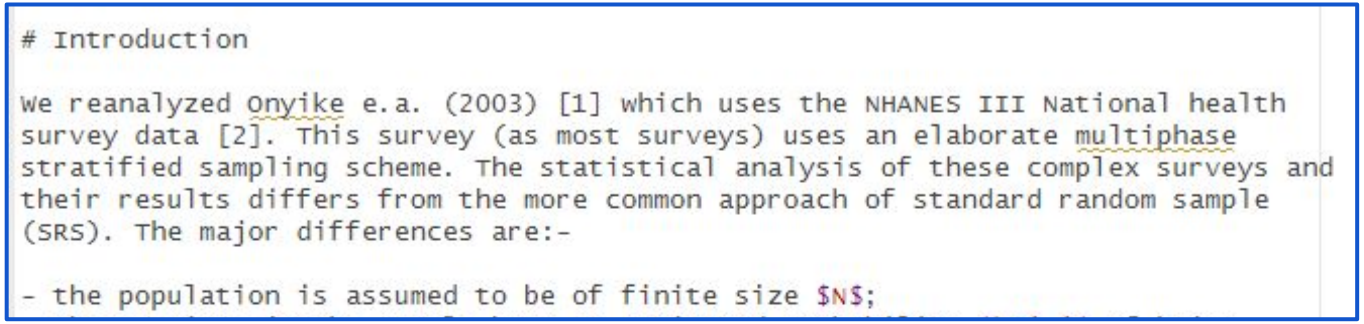


```
library(tidyverse) #Used when manipulating data
library(ggplot2) #Used to develop plots
```

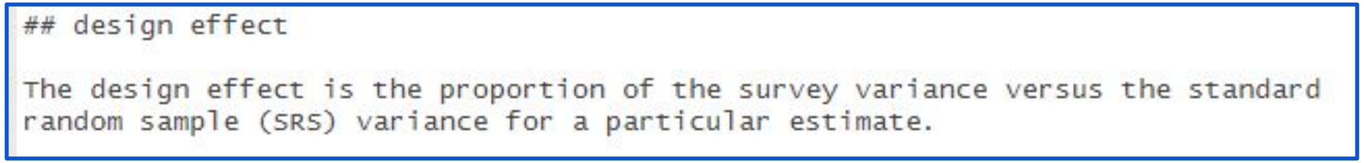
Source: Brown et al 2018

Example: R markdown file

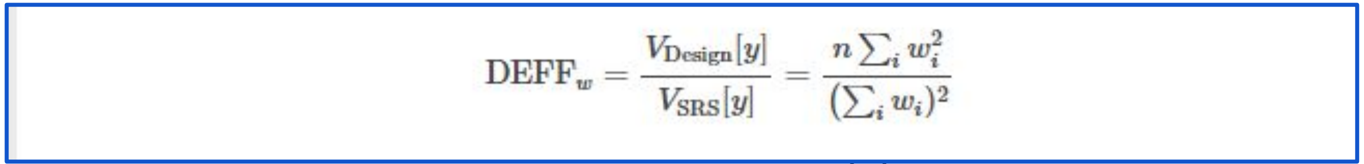
Project,
Data,
Function,
Sample codes,
Additional notes,
Reference



```
# Introduction

We reanalyzed Onyike e.a. (2003) [1] which uses the NHANES III National health
survey data [2]. This survey (as most surveys) uses an elaborate multiphase
stratified sampling scheme. The statistical analysis of these complex surveys and
their results differs from the more common approach of standard random sample
(SRS). The major differences are:-

- the population is assumed to be of finite size $N$;
```



```
## design effect

The design effect is the proportion of the survey variance versus the standard
random sample (SRS) variance for a particular estimate.
```



$$\mathrm{DEFF}_w = \frac{V_{\mathrm{Design}}[y]}{V_{\mathrm{SRS}}[y]} = \frac{n\sum_i w_i^2}{(\sum_i w_i)^2}$$

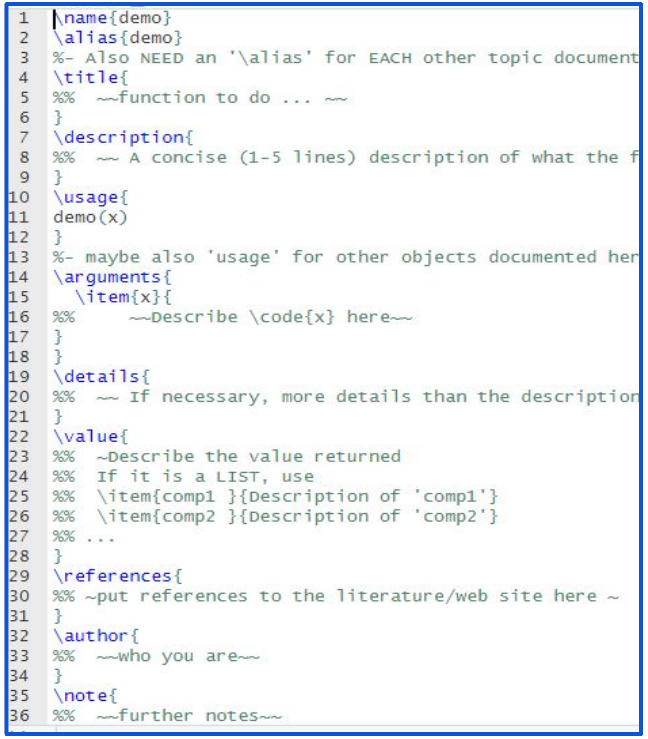
Source: Brown et al 2018

Example: roxygen2

How?

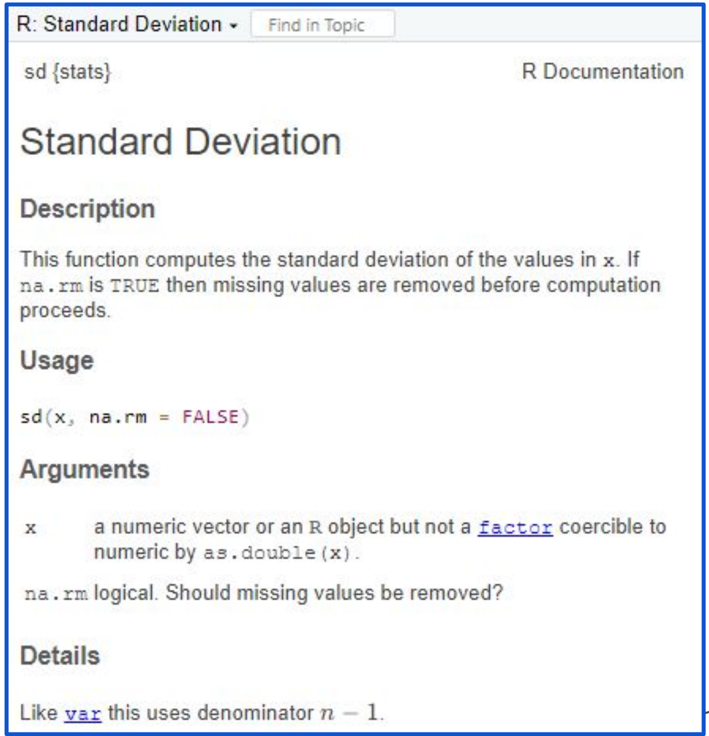
File >
Newfile >
R Documentation file
>
devtools::document()
>
.Rd file>
Install and build
packages >
man file

R documentation file



```
\name{demo}
\alias{demo}
%- Also NEED an '\alias' for EACH other topic document
\title{
%%  ~~function to do ... ~~
}
\description{
%%  ~~ A concise (1-5 lines) description of what the f
}
\usage{
demo(x)
}
%- maybe also 'usage' for other objects documented her
\arguments{
  \item{x}{
%%     ~~Describe \code{x} here~~
}
}
\details{
%%  ~~ If necessary, more details than the description
}
\value{
%%  ~Describe the value returned
%%  If it is a LIST, use
%%  \item{comp1 }{Description of 'comp1'}
%%  \item{comp2 }{Description of 'comp2'}
%% ...
}
\references{
%% ~put references to the literature/web site here ~
}
\author{
%%  ~~who you are~~
}
\note{
%%  ~~further notes~~
```

Man (manual) file



sd {stats} R Documentation

Standard Deviation

Description

This function computes the standard deviation of the values in `x`. If `na.rm` is `TRUE` then missing values are removed before computation proceeds.

Usage

```
sd(x, na.rm = FALSE)
```

Arguments

`x`	a numeric vector or an R object but not a factor coercible to numeric by `as.double(x)`.
`na.rm`	logical. Should missing values be removed?

Details

Like var this uses denominator $n - 1$.

Source: https://combine-australia.github.io/r-pkg-dev/documenting-functions.html#adding-documentation

Thank you!

Do you have any questions?