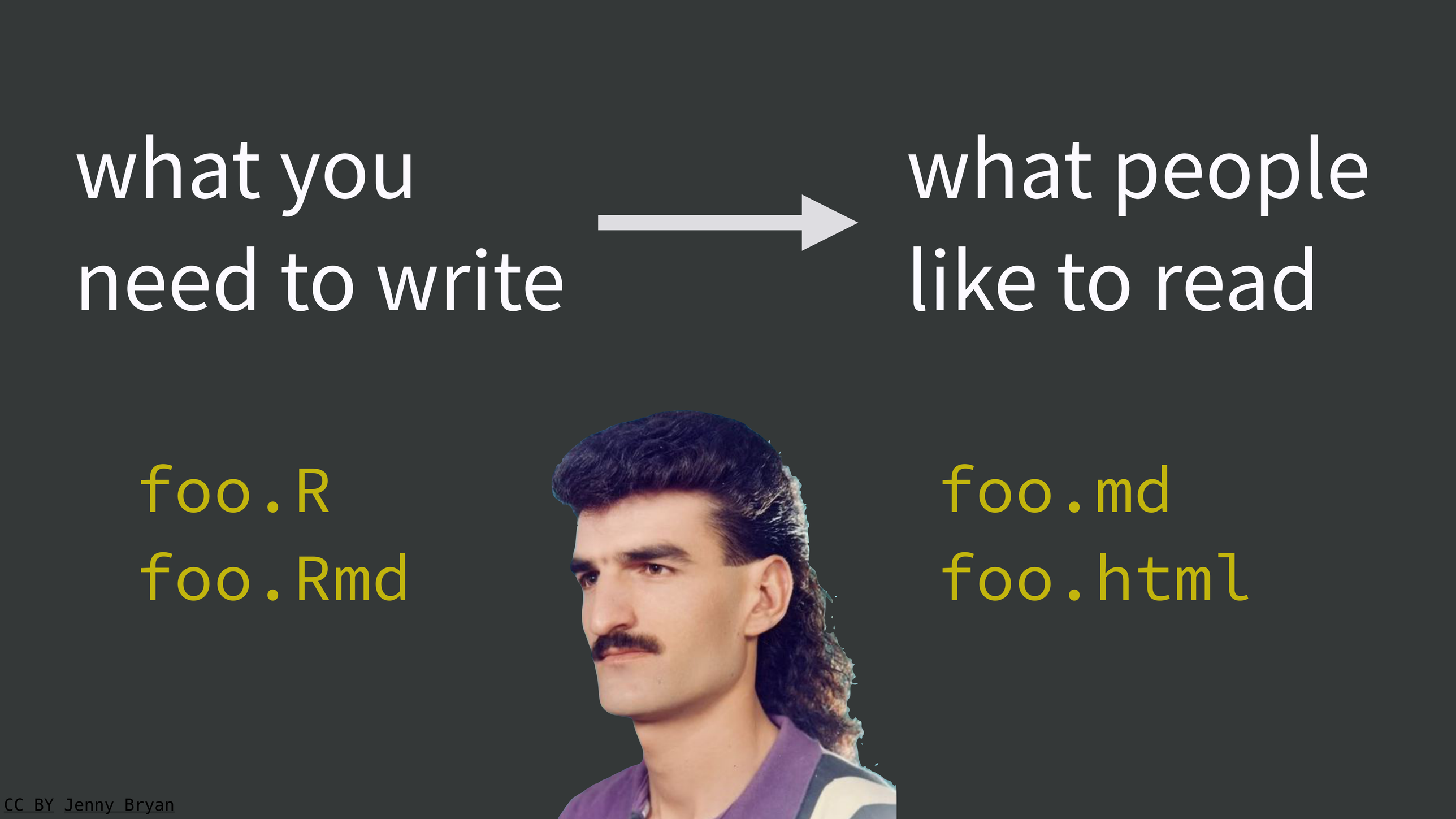
Vignettes and RMarkdown

what you
need to write



what people
like to read

`foo.R`
`foo.Rmd`

`foo.md`
`foo.html`

As I've glanced through your `classes` repos, I've seen some… poorly formatted documents. For my sanity (and for good practice toward making vignettes) I'd like you to clean these up.

Your end goal:

- **At least six files**
    - Three should be .Rmd files (the business in the front)
    - Three should be GitHub-flavored markdown (the party in the back). The one with spatial stuff can use `eval=FALSE` as a chunk option
- **At least two more commits**
    - Use good commit messages! Some commit messages I would like to see from some of you:
        - "switch headers and text"
        - "add packages to startup chunk"
        - "format list"
        - "knit to .md"

Structure of an Rmd file

- ***Minimum requirement: File name ends with .Rmd***
- ***A little richer: first several lines are YAML markup***
  - *Connect to other systems, e.g. Shiny, blowdown, book down*
  - *Set style and document output format*

Text and headers

- ***Text can be plain text or decorated as*** *italic* ***or*** **bold**
- ***Headers use #s***

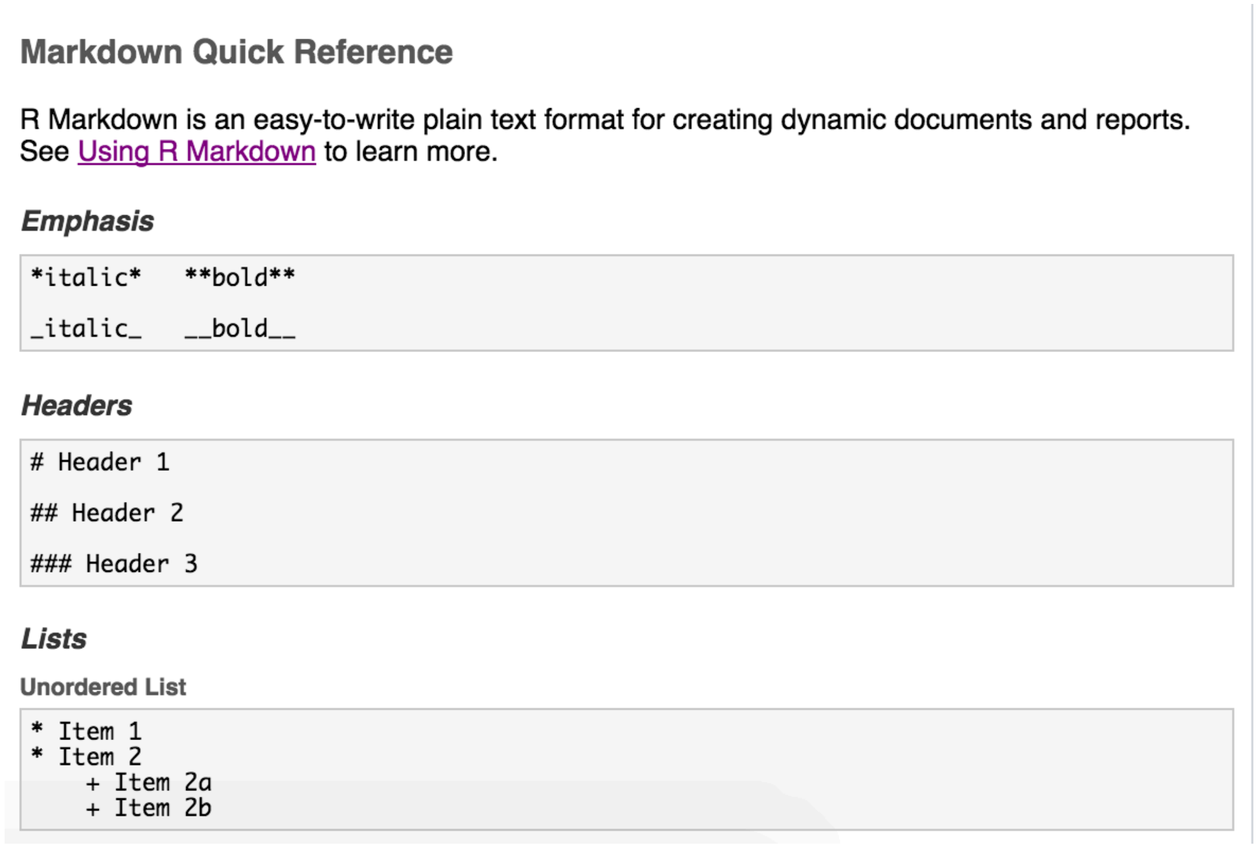
  # Header 1

  ## Header 2

  ### Header 3

Markdown Quick Reference

In RStudio: Help ➡ Markdown Quick Reference



Markdown Quick Reference

R Markdown is an easy-to-write plain text format for creating dynamic documents and reports.
See Using R Markdown to learn more.

Emphasis

```
*italic*   **bold**

_italic_   __bold__
```

Headers

```
# Header 1

## Header 2

### Header 3
```

Lists

Unordered List

```
* Item 1
* Item 2
    + Item 2a
    + Item 2b
```

R Markdown : : CHEAT SHEET

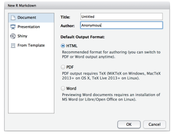
What is R Markdown?

.Rmd files • An R Markdown (.Rmd) file is a record of your research. It contains the code that a scientist needs to reproduce your work along with the narration that a reader needs to understand your work.

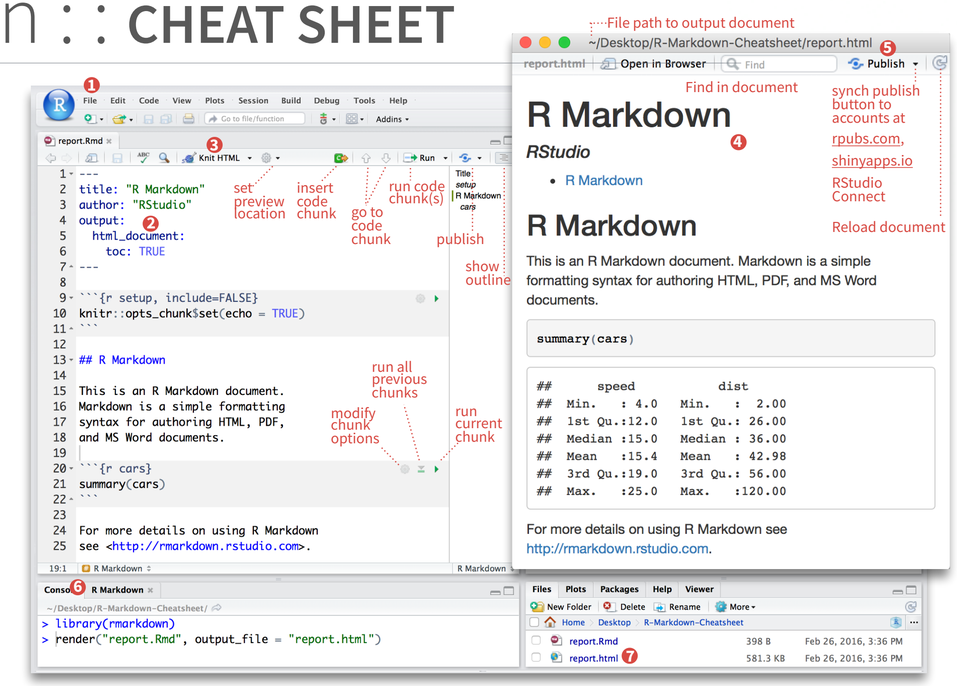
Reproducible Research • At the click of a button, or the type of a command, you can rerun the code in an R Markdown file to reproduce your work and export the results as a finished report.

Dynamic Documents • You can choose to export the finished report in a variety of formats, including html, pdf, MS Word, or RTF documents; html or pdf based slides, Notebooks, and more.

Workflow



1. **Open a new .Rmd file** at File ▶ New File ▶ R Markdown. Use the wizard that opens to pre-populate the file with a template
2. **Write document** by editing template
3. **Knit document to create report**; use knit button or **render()** to knit
4. **Preview Output** in IDE window
5. **Publish** (optional) to web server
6. **Examine build log** in R Markdown console
7. **Use output file** that is saved along side .Rmd



render

Use rmarkdown::**render()** to render/knit at cmd line. Important args:

input - file to render **output_format**	**output_options** - List of render options (as in YAML)	**output_file** **output_dir**	**params** - list of params to use	**envir** - environment to evaluate code chunks in	**encoding** - of input file

.rmd Structure

YAML Header
Optional section of render (e.g. pandoc) options written as key:value pairs (YAML).

At start of file

Between lines of `---`

Text
Narration formatted with markdown, mixed with:

Code Chunks
Chunks of embedded code. Each chunk:

Begins with ```` ```{r} ````

ends with ```` ``` ````

R Markdown will run the code and append the results to the doc.

It will use the location of the .Rmd file as the **working directory**

Parameters

Parameterize your documents to reuse with different inputs (e.g., data, values, etc.)

1. **Add parameters** • Create and set parameters in the header as sub-values of params

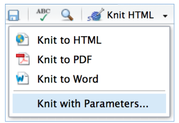
```
---
params:
  n: 100
  d: !r Sys.Date()
---
```

2. **Call parameters** • Call parameter values in code as params$<name>

```
Today's date
is `r params$d`
```

3. **Set parameters** • Set values wth Knit with parameters or the params argument of render():

render("doc.Rmd", params = list(n = 1, d = as.Date("2015-01-01"))



Interactive Documents

Turn your report into an interactive Shiny document in 4 steps

1. Add runtime: shiny to the YAML header.
2. Call Shiny input functions to embed input objects.
3. Call Shiny render functions to embed reactive output.
4. Render with rmarkdown::run or click Run Document in RStudio IDE

```
---
output: html_document
runtime: shiny
---

```{r, echo = FALSE}
numericInput("n",
  "How many cars?", 5)

renderTable({
  head(cars, input$n)
})
```
```

How many cars? 5

	speed	dist
1	4.00	2.00
2	4.00	10.00
3	7.00	4.00
4	7.00	22.00
5	8.00	16.00

Embed a complete app into your document with shiny::**shinyAppDir()**

NOTE: *Your report will rendered as a Shiny app, which means you must choose an html output format, like **html_document**, and serve it with an active R Session.*

Embed code with knitr syntax

INLINE CODE

Insert with **`` `r <code>` ``**. Results appear as text without code.

Built with `` `r getRversion()` `` ➡ Built with 3.2.3

CODE CHUNKS

One or more lines surrounded with **```{r}** and **```**. Place chunk options within curly braces, after **r**. Insert with

```{r echo=TRUE}
getRversion()
```

```
getRversion()
## [1] '3.2.3'
```

GLOBAL OPTIONS

Set with knitr::**opts_chunk$set()**, e.g.

```{r include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
```

IMPORTANT CHUNK OPTIONS

cache - cache results for future knits (default = FALSE)

cache.path - directory to save cached results in (default = "cache/")

child - file(s) to knit and then include (default = NULL)

collapse - collapse all output into single block (default = FALSE)

comment - prefix for each line of results (default = '##')

dependson - chunk dependencies for caching (default = NULL)

echo - Display code in output document (default = TRUE)

engine - code language used in chunk (default = 'R')

error - Display error messages in doc (TRUE) or stop render when errors occur (FALSE) (default = FALSE)

eval - Run code in chunk (default = TRUE)

fig.align - 'left', 'right', or 'center' (default = 'default')

fig.cap - figure caption as character string (default = NULL)

fig.height, fig.width - Dimensions of plots in inches

highlight - highlight source code (default = TRUE)

include - Include chunk in doc after running (default = TRUE)

message - display code messages in document (default = TRUE)

results (default = 'markup')
'asis' - passthrough results
'hide' - do not display results
'hold' - put all results below all code

tidy - tidy code for display (default = FALSE)

warning - display code warnings in document (default = TRUE)

Options not listed above: R.options, aniopts, autodep, background, cache.comments, cache.lazy, cache.rebuild, cache.vars, dev, dev.args, dpi, engine.opts, engine.path, fig.asp, fig.env, fig.ext, fig.keep, fig.lp, fig.path, fig.pos, fig.process, fig.retina, fig.scap, fig.show, fig.showtext, fig.subcap, interval, out.extra, out.height, out.width, prompt, purl, ref.label, render, size, split, tidy.opts

RStudio

Numbered sections

my-first-rmd.Rmd

Knit | Insert | Run

```
1  ---
2  title: "Starting with Rmd"
3  author: "Danny Kaplan et al."
4  date: "1/15/2019"
5  output:
6    html_document:
7      number_sections: true
8  ---
9
10 # Section 1
11
12 ## Section 1.1
13
14 ## Section 1.2
15
16 ### Section 1.2.1
17
18 ### Section 1.2.2
19
```

4:12 | Starting with Rmd | R Markdown

Files | Plots | Packages | Help | Viewer

Publish

Starting with Rmd

Danny Kaplan et al.

1/15/2019

1 Section 1

1.1 Section 1.1

1.2 Section 1.2

1.2.1 Section 1.2.1

1.2.2 Section 1.2.2

Tips

Show / hide document outline

Links

- ***A link can be a plain http address or can underlie a phrase:***
  - http://rmarkdown.rstudio.com/
  - [R Markdown website](http://rmarkdown.rstudio.com/)
- ***Long URLs with, e.g. query parameters, work just as well.***

Images

- *Including an image is very similar to hyperlinking*
- *Images can be on the web:*

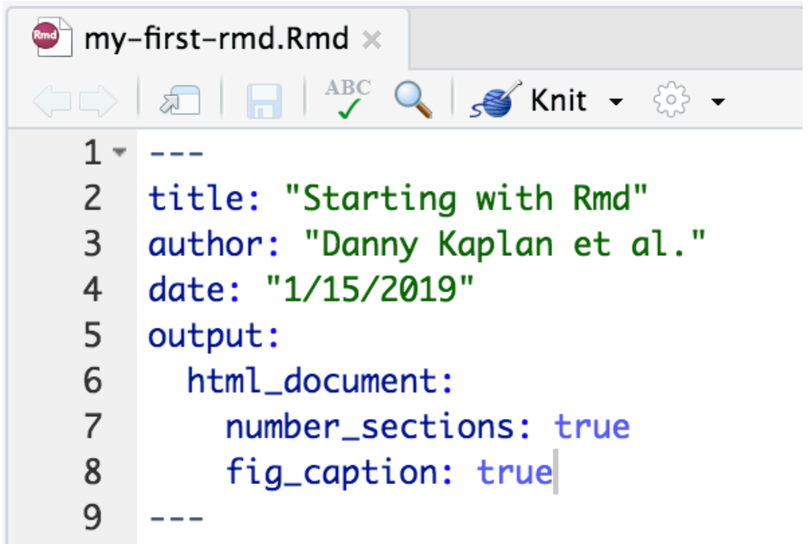
  ```
  ![RStudio logo](https://www.rstudio.com/wp-content/uploads/2014/04/rmarkdown.png)
  ```

- *Or they can be locally stored, e.g. in a directory "images"*

  ```
  ![RStudio logo](images/rmarkdown.png)
  ```

Tips

- *To improve the accessibility of your document, always add* alt text *to your images.*
- *To print the* alt text *underneath the image as a caption,*
  - *use* fig_caption: true *in the YAML,*
  - *make sure there is a line break before the figure call.*



Reference style links and images

- ***Links***
  - *A* [linked phrase][id]
  - *At the bottom of the document:* [id]: http://example.com/ "Title"
- ***Images***
  - ![alt text][id]
  - *At the bottom of the document:* [id]: figures/img.png “Title"
- ***Useful if you’ll be linking to the same target/image multiple times throughout the document***

Math text

- *If you already know some LaTeX, you're good to go*
- *Equations can be inline:*

```
- $\bar{x} \ sim N(\mu,\frac{\sigma}
  {\sqrt{n}})$
```

- *And equations can be centered in a new line:*

```
$$\bar{x} \sim N (\mu,\frac{\sigma}
{\sqrt{n}})$$
```

- Equations can be inline: $\bar{x} \sim N\left(\mu, \frac{\sigma}{\sqrt{n}}\right)$

- And equations can be centered in a new line:

$$\bar{x} \sim N\left(\mu, \frac{\sigma}{\sqrt{n}}\right)$$

Tables

- *Tables are often a bit of a pain…*
- *Dashes separate the header row from content cells, and pipes separate the columns*
- *Colons can be used to align columns*

```
| Column A | Column B | Column C  |
|----------|:--------:|----------:|
| left     |  center  |     right |
| aligned  |  aligned |   aligned |
| text     |   text   |      text |
```

Column A	Column B	Column C
left	center	right
aligned	aligned	aligned
text	text	text

Tips

- *The outer pipes (|) on a Markdown table are optional.*
- *You don't need to make the raw Markdown line up prettily.*
- *You can use inline Markdown within tables.*
- *For complicated tables, use R packages e.g. kable & kableExtra*

Tips

- *Keep your text to max ~80 characters across, especially if you use a version control system (like git)*
- *Starting a list? Leave an empty line before the first item on your list*
- *Need to test out bits of markdown code without knitting the entire document, use another document with bits and pieces of code to test out*