**Reference for Common Functionality of RMarkdown**

*29 January 2021*

**Links:**

Links to RMarkdown resources that could be useful and this document could be based on the content in some of these links.

* <https://rmarkdown.rstudio.com/lesson-1.html>
* <https://rstudio.github.io/visual-markdown-editing/#/markdown>
* <https://www.rstudio.com/wp-content/uploads/2016/03/rmarkdown-cheatsheet-2.0.pdf?_ga=2.114546062.867792584.1611653172-787395666.1608309879>
* <https://rstudio.com/wp-content/uploads/2015/03/rmarkdown-reference.pdf?_ga=2.114546062.867792584.1611653172-787395666.1608309879>
* <https://bookdown.org/yihui/rmarkdown/>
* <https://towardsdatascience.com/ten-awesome-r-markdown-tricks-56ef6d41098>
* <https://rmarkdown.rstudio.com/authoring_pandoc_markdown.html>

**Installation**

install.packages("rmarkdown")

**How It Works**

* Plain text file with .Rmd extension
* 3 types of content:

1. Optional YAML header [---s used]
2. R code chunks [```s used]
3. Text with simple text formatting

* Notebook interface in RStudio
* Rendering Output: use render command or Knit button in RStudio

e.g.:

library(rmarkdown)

render("1-example.Rmd")

* RMarkdown generates a new file with text, code, results from Rmd file. Multiple different possible output format



**Code Chunks**

Add chunks using:

1. Keyboard shortcut (Ctrl+Alt+I)
2. “Add Chunk” +C icon in code editor
3. Using delimiters: ```{r} and ```

5 examples of chunk options:

1. **include = FALSE:** code & results don’t appear in final report
2. **echo = FALSE:** results appear in finished file, but not code
3. **message = FALSE:** messages generated by code don’t appear in final file
4. **warning = FALSE:** warnings generated by code don’t appear in final file
5. **fig.cap = “…”:** adds caption to generated graphical results

Global Options: use knitr::opts\_chunk$set in a chunk

**Inline Code**

Insert code results directly into text part of Rmd file.

Enclose the code with `r `, e.g. `r colorFunc` where colorFunc is a function defined in R.

**Code Languages**

Available language engines:

* Python
* SQL
* Bash
* Rcpp
* Stan
* JavaScript
* CSS

Simply, replace the “r” at the beginning of a chunk with the name of the desired language.

**Parameters**

Rmd files can have multiple parameters which can be set when rendering the document.

Parameters are set with the params field within the YAML header.

Use Parameters in Code: these are available in the code as a read-only list called “params”. Use params$<parameter name> to access a specific parameter.

Set Parameter Values: Use the “params” argument to “render” to give a list of parameter values.

e.g. render("5-parameters.Rmd", params = list(data = "aleutians"))

Alternatively, click “Knit with Parameters”.

**Tables**

By default, tables/matrices displayed as they are in R terminal.

Additional formatting: use knitr::kable function.

Use results='asis' chunk option – ensures raw table output isn’t processed further.

**Markdown Basics**

Format text with “Pandoc’s Markdown” = markup annotation for plain text files.

Some of the things you can include with this markdown:

* Headers
* Lists
* Tables
* Images
* Bold/Italicized/Underlined text
* etc

**Output Formats**

Change format using “output\_format” argument to “render” function.

e.g. render("1-example.Rmd", output\_format = "word\_document")

Default format: can set this in the header of Rmd file (“output” field).

Can also use buttons in RStudio to choose format.

Many different possible output formats.

Output Options: Customize output by passing arguments to the function as sub-values of “output” field.

e.g.:

output:

html\_document:

toc: true

toc\_float: true

**Notebooks**

Render a file to a HTML notebook: output: html\_notebook

nb.html version of file created. HTML rendered version of the notebook with all current output plus a copy of the Rmd notebook itself.

They work well with Version Control.

**Slide Presentations**

Rmd renders to 4 presentation formats:

1. **beamer\_presentation** - PDF presentations with beamer
2. **ioslides\_presentation** - HTML presentations with ioslides
3. **slidy\_presentation** - HTML presentations with slidy
4. **powerpoint\_presentation** - PowerPoint presentation
5. **revealjs::revealjs\_presentation** - HTML presentations with reveal.js

Each slide beginning at a new first or second level header.

Horizontal rule (\*\*\*) = manual slide break.

Incremental bullets with >-

**Dashboards**

Use flexdashboard::flex\_dashboard output format

Level 1 Header (#) = new page

Level 2 Header (##) = new column

Level 3 Header (###) = new box

Further modify elements with attributes, e.g. {.sidebar}

**R Markdown Websites**

rmarkdown::render\_site function: render collection of Rmd files into website.

Requirements for each Website:

1. **\_site.yml file:** global YAML header for site
2. **index.Rmd file:** content for the home page

Execute rmarkdown::render\_site function from within the directory containing your files. This builds “\_site” directory – ready to deploy as static website.

Alternatively, create RStudio Project for the website – a build tab will be added to the IDE.

**Interactive Documents**

To make them interactive, add:

1. Interactive JavaScript visualizations (htmlwidgets)
2. Reactive components (Shiny)

* htmlwidgets can execute Javascript visualization functions, e.g. leaflet maps. Client side interactions.
* shiny creates web apps powered by R code. Add runtime: shiny to the header of a Rmd file. Shiny interactions occur on Server side.

**Cheatsheets**

<https://rmarkdown.rstudio.com/lesson-15.html>