PSYC 259:
Principles of Data Science

Sharing & Reproducibility

Today

1. Complete course evals
2. Reproducible workflows
3. Introducing Quarto
4. *Break*
5. R Markdown tutorial
6. Start homework (if time remains)

Don’t forget: Final project

- Project due 3/19 at 5pm
- Share your private repo with OST + Madison

Research transparency

- Replicability - find same result when rerunning the study (ideally 3rd party)
- Analytic reproducibility - statistical analysis can be confirmed by a 3rd party
- Analytic robustness - result is stable despite changes analytical procedures

Sharing is key to allow replication and reproducibility

- Study protocol, materials
  - Allows replication
- Metadata, raw/summarized data, scripts
  - Allows analytical reproducibility checks by other researchers
  - 3rd party researchers can tweak workflow/analytical approach to determine robustness

Restrictions on sharing

- Sensitive and/or identifiable
  - Depends on IRB/local/institutional guidelines
  - Audio/video can be shared with IRB approval
- De-identified data can usually be shared, but make all efforts to anonymize
  - Especially in a small sample, combinations of birthdate, ZIP code, sex, race/ethnicity could identify participants
  - Consent should indicate what might be shared and with whom; sharing can be optional

Common storage sites

- Open Science Foundation
  - Integrates different types of cloud storage
  - Allows for document registration
- Github (+ Zenodo)
  - Better for code and smaller files
- Databrary
  - Specialized for video
  - Flexible access controls (by participant)

How to make your analysis reproducible

- Documentation helps, but isn't everything
- Use open source software (e.g., R, Python)
- Avoid absolute file paths
- Set random number seed
- Document hardware and software versions

Document your R and package environment for posterity

```
Sys.time()
```

```
## [1] "2022-02-22 09:19:08 PST"
```

```
sessionInfo()
```

```
## R version 4.1.1 (2021-08-10)
## Platform: x86_64-apple-darwin17.0 (64-bit)
## Running under: macOS Catalina 10.15.7
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/4.1/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/4.1/Resources/lib/libRlapack.dylib
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] knitr_1.37      forcats_0.5.1   stringr_1.4.0   dplyr_1.0.7
##  [5] purrr_0.3.4     readr_2.0.2     tidyr_1.1.4     tibble_3.1.6
##  [9] ggplot2_3.3.5   tidyverse_1.3.1
##
## loaded via a namespace (and not attached):
##  [1] tidyselect_1.1.1 xfun_0.29        bslib_0.3.1      haven_2.4.3
```

Groundhog package

If we wanted to load a single package on a single date, instead of running:

```
library("dplyr")
```

we would instead run:

```
groundhog.library("dplyr", "2020-01-01")
```

Host your project on a cloud computing environment

- Lets users open and run your project without having to transfer files to their local computer; preserves software versions
  - RStudio Cloud
  - CodeOcean
  - Binder

CodeOcean example

Published Adapting the coordination of eyes and head for task-specific visual exploration in the... *(John Franchak, Brianna McGee & Gabriell...)*

Edit Original

Capsule File Help

Files | App Panel | Tabs

Core Files

- metadata 659 B
- environment 919 B
- code 24.55 KB
  - calibration_error.R 1.54 KB
  - LICENSE 1.04 KB
  - lss1_summary_analyses.R 8.56 KB
  - lss2_peak_analyses.R 2.78 KB
  - lss2_summary_analyses.R 10.33 KB
  - run 287 B
- data Manage Datasets 2.43 MB
  - calibration_LSS1.csv 11.4 KB
  - calibration_LSS2.csv 5.49 KB
  - LICENSE 18.88 KB
  - summary_stats_LSS1.csv 12.12 KB
  - summary_stats_LSS2_peaks.... 2.37 MB
  - summary_stats_LSS2.csv 15.46 KB
- .gitignore 7 B

Results

- results

Your files will appear in the timeline.

View latest results

Other Files

- readme.txt 738 B

readme.txt

```
1  Data files and analyses for Franchak, McGee, & Blanch QJEP "Adapting the
   coordination of eyes and head for task-specific visual exploration in the
   context of locomotion"
2
3  Files tagged "LSS1" correspond to Study 1 (Eyes only), "LSS2" corresponds to
   Study 2 (eyes, head, and gaze)
4
5  There are 4 R scripts.
6  1) Calibration error reports summary statistics for the calibration error for
   both Study 1 and Study 2
7  2) lss1_summary_analyses reports t-tests for spread, speed, and walking GPS data
   for Study 1
8  3) lss2_summary_analyses reports LMMs (eye vs head) for spread and speed,
   andt-tests for walking GPS data for Study 1
9  4) lss2_peak_analyses reports the head contribution analysis
10
11 Each R script produces the figures used in the publication.
```

Reproducible Run

or launch a cloud workstation

lab | RStudio | jupyter | Shiny

Timeline

Nov 5, 2020
Published **Version 1.0**
Currently viewing

Author ran Nov 5, 2020 00:00:22

- **Published Result**
  - head_contribution.pdf 4.92 KB
  - lss1_position_sd.pdf 11.65 KB
  - lss1_speed.pdf 11.55 KB
  - lss1_walking_stats_c... 24.85 KB
  - lss2_position_sd.pdf 14.61 KB
  - lss2_speed.pdf 11.31 KB
  - lss2_walking_stats_c... 14.92 KB
  - output 24.55 KB

John Franchak committed Nov 5, 2020

Version 1.0

Nov 5, 2020
Created capsule

Reproducibility

Share dynamic documents

- Instead of comments, use R Markdown to mix prose, code, tables, and figures
- From within RStudio, you can render the document in different output formats: html, word, pdf, slides, and more
- The *papaja* package can even render your document as an APA manuscript

R Markdown components

- YAML header – metadata (title, author), output format and options
  - Always comes first
  - Does not use R syntax, pay attention to spacing

```yaml
---
title: "Exploratory Data Analysis"
author: "John Franchak"
date: "2/19/2021"
output:
  html_document:
    toc: true
    toc_float: true
    code_folding: show
---
```

R Markdown components

- Markdown text
  - Special characters identify headings, lists, bold, italic, etc as opposed to using toolbars (i.e. Word)

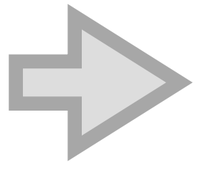
```
### Level 3 heading

Lists are easy to create.

* Top level of an unordered list
  * 2nd level of list using tab
  * 2nd level again
* Back to top level

1. Numbered lists work the same way
1. But just keep using "1".
1. Rmd will figure it out for you
1. See!

Hyperlinks are easy to embded in text, like so: [click
here](https://padlab.ucr.edu). Or if you want the URL to display
like this <https://padlab.ucr.edu>.
```



Level 3 heading

Lists are easy to create.

- Top level of an unordered list
  - 2nd level of list using tab
  - 2nd level again
- Back to top level

1. Numbered lists work the same way
2. But just keep using “1”.
3. Rmd will figure it out for you
4. See!

Hyperlinks are easy to embded in text, like so: click here. Or if you want the URL to display you can do so like this https://padlab.ucr.edu.

Pandoc's Markdown

Write with syntax on the left to create effect on right (after render)

```
Plain text
End a line with two spaces
to start a new paragraph.
*italics* and **bold**
`verbatim code`
sub/superscript^2^~2~
~~strikethrough~~
escaped: \* \_ \\
endash: --, emdash: ---
equation: $A = \pi*r^{2}$
equation block:

$$E = mc^{2}$$

> block quote

# Header1  {#anchor}

## Header 2  {#css_id}

### Header 3  {.css_class}

#### Header 4

##### Header 5

###### Header 6

<!--Text comment-->

\textbf{Tex ignored in HTML}
<em>HTML ignored in pdfs</em>

<http://www.rstudio.com>
[link](www.rstudio.com)
Jump to [Header 1](#anchor)
image:

![Caption](smallorb.png)
```

Plain text
End a line with two spaces
to start a new paragraph.
italics and **bold**
`verbatim code`
sub/superscript$^2_2$
~~strikethrough~~
escaped: * _ \
endash: –, emdash: —
equation: $A = \pi * r^2$
equation block:

$$E = mc^2$$

> block quote

Header1

Header 2

Header 3

Header 4

Header 5

Header 6

HTML ignored in pdfs

http://www.rstudio.com
link
Jump to Header 1
image:

Caption

```
* unordered list
  + sub-item 1
  + sub-item 2
    - sub-sub-item 1

* item 2

    Continued (indent 4 spaces)

1. ordered list
2. item 2
  i) sub-item 1
     A.  sub-sub-item 1

(@)  A list whose numbering

continues after

(@)  an interruption

Term 1

:  Definition 1

| Right | Left | Default | Center |
|------:|:-----|---------|:------:|
|  12 | 12 |  12 |  12 |
| 123 | 123 | 123 | 123 |
|  1 |  1 |  1 |  1 |

- slide bullet 1
- slide bullet 2

(>- to have bullets appear on click)

horizontal rule/slide break:

***

A footnote [^1]

[^1]: Here is the footnote.
```

- unordered list
  - sub-item 1
  - sub-item 2
    - sub-sub-item 1
- item 2

  Continued (indent 4 spaces)

1. ordered list
2. item 2
   i. sub-item 1
      A. sub-sub-item 1

(@) A list whose numbering

continues after

(@) an interruption

Term 1

Definition 1

Right	Left	Default	Center
12	12	12	12
123	123	123	123
1	1	1	1

- slide bullet 1
- slide bullet 2

(>- to have bullets appear on click)

horizontal rule/slide break:

A footnote [1]

1. Here is the footnote. ↩

R Markdown components

- Code chunks
  - Lines in between ```{r} and ```
  - Settings to tell R to show code/ warnings/results/plots

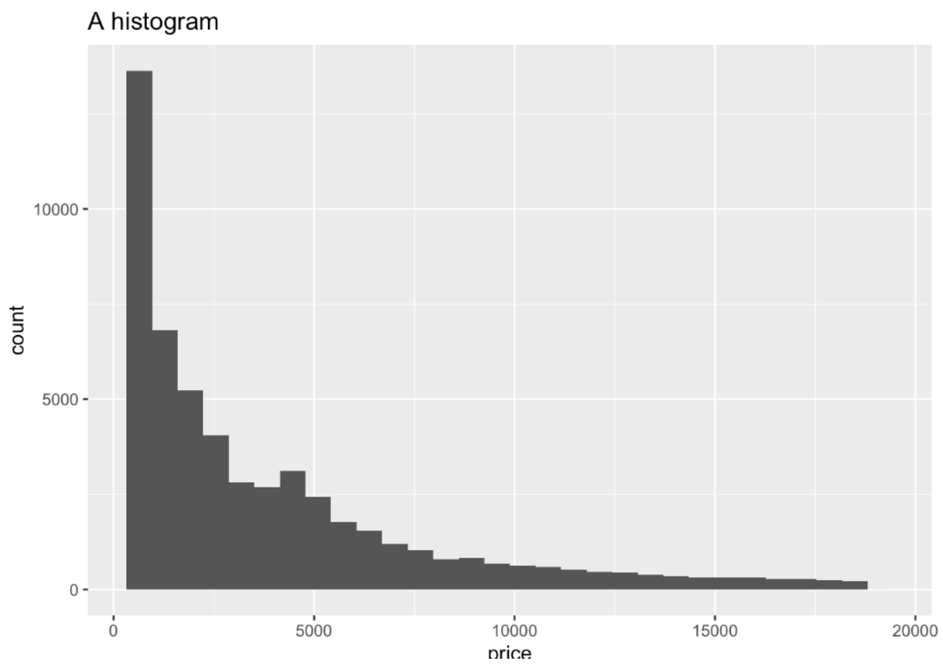
````
### Figures

A code chunk that produces a figure will insert that
figure into the output document.
```{r warning = FALSE, message=FALSE}
ds <- diamonds
ggplot(ds, aes(x = price)) + geom_histogram() +
ggtitle("A histogram")
```
````

Figures

A code chunk that produces a figure will insert that figure into the output document.

```
ds <- diamonds
ggplot(ds, aes(x = price)) + geom_histogram() + ggtitl
e("A histogram")
```



R Markdown : : CHEAT SHEET

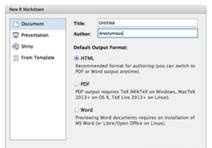
What is R Markdown?

.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.

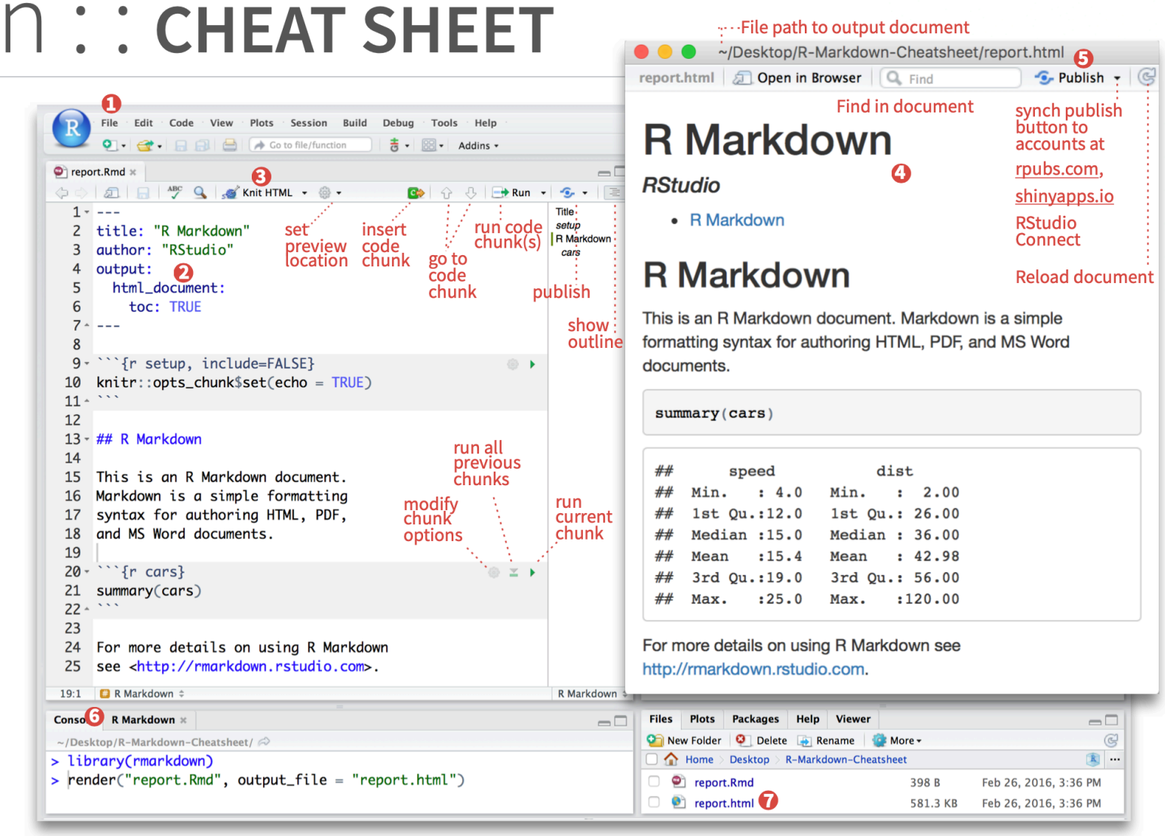
Reproducible Research • At the click of a button, or the type of a command, you can rerun the code in an R Markdown file to reproduce your work and export the results as a finished report.

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



.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 new inputs (e.g., data, values, etc.)

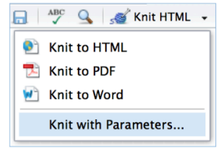
1. **Add parameters** • Create and set parameters in the header as sub-values of params

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

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"))

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

Today's date is `r params$d`



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

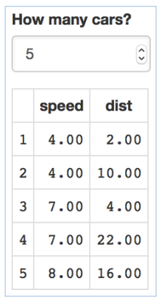
Interactive Documents

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 w 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()**

Publish on RStudio Connect, to share R Markdown documents securely, schedule automatic

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)

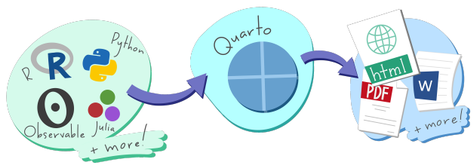
That's Markdown…What's Quarto?

- In 2022 RStudio rebranded as Posit
  - RStudio was both a company and a software product
  - The company is now Posit; the software remains RStudio
  - Posit's goal is to bring open-source software to the scientific and data scientific communities at large
    - Markdown worked in only R
    - Lots of science is done in other languages (e.g., Python)
    - Quarto is Posit's "generalization" of Markdown to other languages

What's Quarto?

- Quarto and Markdown, thus, share many features
  - e.g., lists are made using the same code
  - e.g., headings and subheadings are made in the same way
  - Most Markdown documents can be rendered in Quarto
- Posit plans to maintain Markdown but not issue major developments for it
  - If you want the latest features, you'll probably need to use Quarto
- Let's look at an overview of Quarto

Publish and Share with Quarto : : CHEATSHEET



Author

WRITE AND CODE IN PLAIN TEXT

Author documents as .qmd files or Jupyter notebooks.
Write in a rich Markdown syntax.

Render

GENERATE DOCUMENTS, PRESENTATIONS AND MORE

Produce HTML, PDF, MS Word reveal.js, MS Powerpoint, Beamer Websites, blogs, books...

Publish

SHARE YOUR WORK WITH THE WORLD

Quickly deploy to GitHub Pages, Netlify, Quarto Pub, Posit Cloud, or Posit Connect

GET QUARTO

https://quarto.org/docs/download/

Or use version **bundled with RStudio**

GET STARTED

https://quarto.org/docs/get-started/

Author

SOURCE FILE: hello.qmd

```
---
title: "Hello, Penguins"
format: html
execute:
  echo: false
---

## Meet the penguins

The `penguins` data contains size m
the Palmer Archipelago, Antarctica.

The three species of penguins have
dimensions (@fig-penguins).

```{r}
#| label: fig-penguins
#| fig-cap: "Dimensions of penguins across three
#| warning: false
library(tidyverse, quietly = TRUE)
library(palmerpenguins)
penguins |>
  ggplot(aes(x = flipper_length_mm, y = bill_length_mm)) +
  geom_point(aes(color = species)) +
  scale_color_manual(
    values = c("darkorange", "purple", "cyan4")) +
```

Set format(s) and options
Use YAML Syntax

Write with **Markdown**
RStudio: Help > Markdown Quick Reference
Use **Visual Editor**

Include code
R, Python, Julia, Observable, or any language with a Jupyter kernel

USE A TOOL WITH A RICH EDITING EXPERIENCE

RStudio
Visual Studio Code + Quarto extension

Run code cells as you write

Render with a button or keyboard shortcut

Edit Quarto documents with a **Visual Editor**

Apply formatting in Visual Editor. Saved as Markdown in source.

Insert elements like code cells, cross references, and more.

Normal | Format | Insert | Table

OR ANY TEXT EDITOR

Quarto documents (.qmd) can be edited in any tool that edits text.

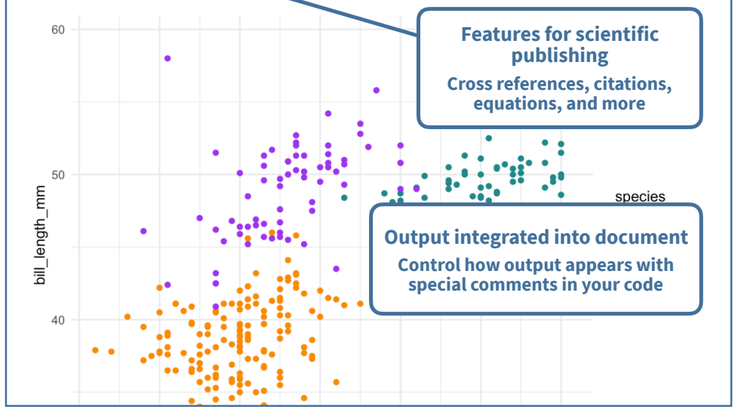
Render

RENDERED OUTPUT: hello.html

Hello, Penguins

Meet the penguins

The three species of penguins have quite distinct distributions of physical dimensions (Figure 1).



Features for scientific publishing
Cross references, citations, equations, and more

Output integrated into document
Control how output appears with special comments in your code

Save, then render to **preview** the document output.

```
Terminal
quarto preview hello.qmd
```

Use **Render** button

Use **Preview** button

The resulting HTML/PDF/MS Word/etc. document will be created and saved in the same directory as the source .qmd file.

BEHIND THE SCENES

When you render a document, Quarto:

1. Runs the code and embeds results and text into an .md file with: **Knitr**, if any {r} cells or, **Jupyter**, if any other cells.
2. Converts the .md file into the output format with Pandoc.

Publish

```
Terminal
quarto publish {venue} hello.qmd
```

{venue}: quarto-pub, connect, gh-pages, netlify, confluence, posit-cloud

Use **Publish** button

Quarto Pub — Free publishing service for Quarto content.

posit Cloud — Cloud-hosted, control access to project and output.

posit Connect — Org-hosted, control access, schedule updates.

Quarto Projects

CREATE WEBSITES, BOOKS, AND MORE

A directory of Quarto documents + a configuration file (_quarto.yml)

See examples at https://quarto.org/docs/gallery/

Get started from the command line:

```
Terminal
quarto create project {type}
```

{type}: default, website, blog, book, confluence, manuscript

Use **File** > **New Project**

Artwork from "Hello, Quarto" keynote by Julia Lowndes and Mine Çetinkaya-Rundel, presented at RStudio Conference 2022. Illustrated by Allison Horst.
```

Include Code

CODE CELLS

Code cells start with ```{language} and end with ```.

Use **Insert Code Chunk/Cell**

```{r}
#| label: chunk-id
library(tidyverse)
```

```{python}
#| label: chunk-id
import pandas as pd
```

Other languages: {julia}, {ojs}

Add code cell options with **#|** comments.

Cell options control **execution**, figures, tables, layout and more. See them all at: https://quarto.org/docs/reference/cells

EXECUTION OPTIONS

OPTION	DEFAULT	EFFECTS
echo	true	false: hide code fenced: include code cell syntax
eval	true	false: don't run code
include	true	false: don't include code or results
output	true	false: don't include results asis: treat results as raw markdown
warning	true	false: don't include warnings in output
error	false	true: include error in output and continue with render

Set execution options at the **cell level**:

```{r}
#| echo: false
```

```{python}
#| echo: false
```

Or, **globally** in the YAML header with the **execute** option:

```
---
execute:
  echo: false
---
```

Set options in code cells with #| comments and YAML syntax:
key: value

INLINE CODE

Use computed values directly in text sections. Code is evaluated at render and results appear as text.

KNITR	JUPYTER	OUTPUT
Value is `r 2 + 2`.	Value is `{python} 2 + 2`.	Value is 4.

Set Format and Options

SET FORMAT OPTIONS

```
---
title: "My Document"
format:
  html:
    code-fold: true
    toc: true
---
```

Indent options 4 spaces

Indent format 2 spaces

MULTIPLE FORMATS

```
---
title: "My Document"
toc: true
format:
  html:
    code-fold: true
  pdf: default
---
```

Top-level options apply to all formats

Common formats: **html, pdf, docx, odt, rtf, gfm, pptx, revealjs, beamer**

Render **all** formats:

Terminal
```
quarto render hello.qmd
```

Render a **specific** format:

Terminal
```
quarto render hello.qmd --to pdf
```

	OPTION	html/revealjs	pdf/beamer	docx/pptx	DESCRIPTION	Also use in code cells
Nav	**toc**	X	X	X	Add a table of contents (true or false)	
Nav	**toc-depth**	X	X	X	Lowest level of headings to add to table of contents (e.g. 2, 3)	
Nav	**anchor-sections**	X			Show section anchors on mouse hover (true or false)	
Style	**highlight-style**	X	X	X	Syntax highlighting theme (e.g. arrow, pygments, kate, zenburn)	
Style	**mainfont, monofont**	X	X		Font name. HTML: sets CSS font-family; LaTeX: via fontspec package	
Style	**theme**	X			Bootswatch theme name (e.g. cosmo, darkly, solar etc.)	
Style	**css**	X			CSS or SCSS file to use to style the document (e.g. "style.css")	
Style	**reference-doc**			X	docx/pptx file containing template styles (e.g. file.docx, file.pptx)	
Style	**include-in-header**	X	X		Files of content to include in header of output document, also **include-before-body, include-after-body**	
Style	**keep-md**	X	X	X	Keep intermediate markdown (true or false), also **keep-ipynb**, **keep-tex**	
LaTeX	**documentclass**		X		LaTeX document class, set document class options with **classoption**	
LaTeX	**pdf-engine**		X		LaTeX engine to produce PDF output (xelatex, pdflatex, lualatex)	
LaTeX	**cite-method**		X		Method used to format citations (citeproc, natbib, biblatex)	
Code	**code-fold**	X			Let readers toggle the display of R code (false, true, or show)	?
Code	**code-tools**	X			Add menu for hiding, showing, and downloading code (true or false)	
Code	**code-overflow**	X			Display of wide code (scroll, or wrap)	?
Figures	**fig-align**	X	X	/	Alignment of figures (default, left, right, or center)	?
Figures	**fig-width, fig-height**	X	X	X	Default width and height for figures in inches	Knitr
Figures	**fig-format**	X	X	X	Format for Matplotlib or R figures (retina, png, jpeg, svg, or pdf)	

Visit https://quarto.org/docs/reference/ to see **all options** by format

Add Content

FIGURES ?

MARKDOWN

```
![CAP](image.png){#fig-LABEL fig-alt="ALT"}
```

COMPUTATION

Or {r}

```
```{python}
#| label: fig-LABEL
#| fig-cap: CAP
#| fig-alt: ALT
{{ plot code here }}
```
```

CROSS REFERENCES

1. **Add labels**
   Code cell: add option label: prefix-LABEL
   Markdown: add attribute #prefix-LABEL
2. **Add references** @prefix-LABEL, e.g.

You can see in **@fig-scatterplot**, that...

Prefix	Renders	Prefix	Renders
fig-	Figure 1	eq-	Equation 1
tbl-	Table 1	sec-	Section 1

TABLES ?

COMPUTATION Output a Markdown table or an HTML table from your code

MARKDOWN

```
|object | radius|
|:------|------:|
|Sun    | 696000|
|Earth  |   6371|

: CAPTION {#tbl-LABEL}
```

Use **Insert Table** in the **Visual Editor**

KNITR

Use knitr::kable() to produce Markdown:

```{r}
#| label: tbl-LABEL
#| tbl-cap: CAPTION
knitr::kable(head(cars))
```

Also see the R packages: gt, flextable, kableExtra.

JUPYTER Add Markdown()to Markdown output:

```{python}
#| label: tbl-LABEL
#| tbl-cap: CAPTION
import pandas as pd, tabulate
from IPython.display import Markdown
df = pd.DataFrame({"A": [1, 2],
                   "B": [1, 2]})
Markdown(df.to_markdown(index=False))
```

CITATIONS

1. Add a bibliography **file** to the YAML header:

```
---
bibliography: references.bib
---
```

2. Add citations: [@citation], or @citation

Use **Insert Citations** dialog in the **Visual Editor**

Build your bibliography file from your Zotero library, DOI, Crossref, DataCite, or PubMed

CALLOUTS

tip

```
::: {.callout-tip}
## Title

Text
:::
```

Instead of **tip** use one of: note, caution, warning, or important.

note · warning · caution · important

SHORTCODES

{{< include _file.qmd >}}
{{< embed file.ipynb#id >}}
{{< video video.mp4 >}}

Getting started with Quarto

- Download & install: https://quarto.org/docs/get-started/
- Once installed, select “New File -> Quarto Document”

R Markdown tutorial

Last homework (due 3/14)

- Take a heavily-commented R script and turn it into an R Markdown report
- Create a new Rmd file and transfer over the comments/code to format them correctly
- An example output is provided