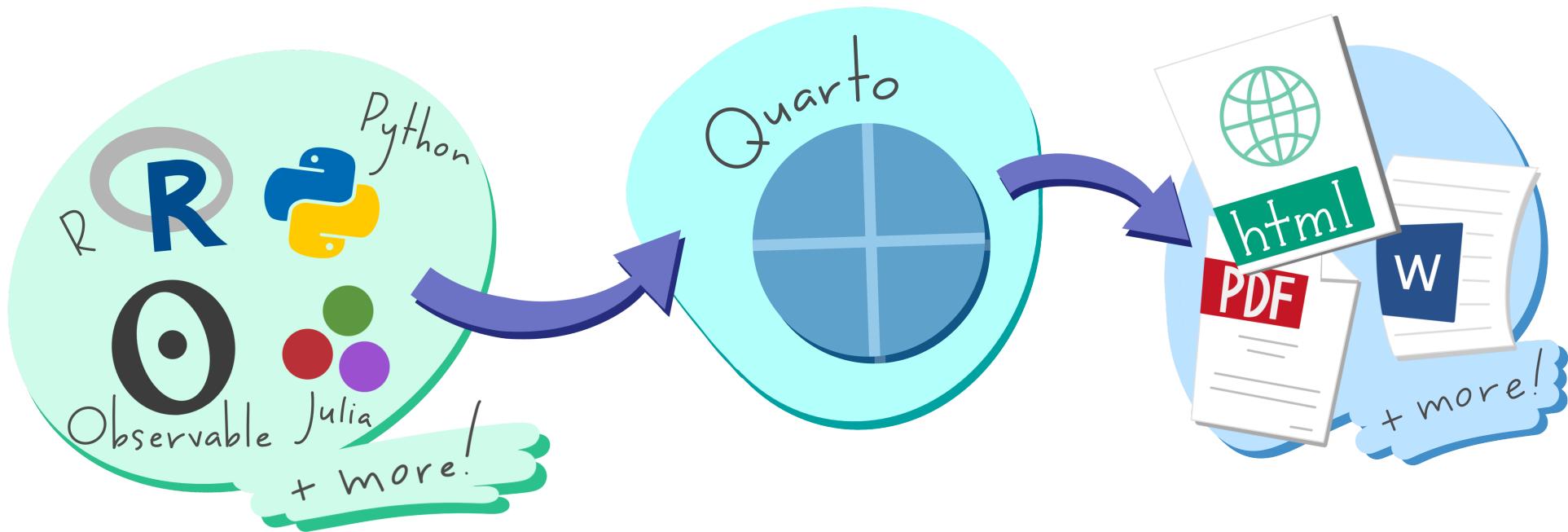


Dynamic documents in Python

reproducible research with Quarto

2025-07-05



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

```
---
```

```
title: "ggplot2 demo"
author: "Norah Jones"
date: "5/22/2021"
format:
  html:
    fig-width: 8
    fig-height: 4
    code-fold: true
```

```
---
```

```
## Air Quality
```

```
@fig-airquality further explores the impact of
temperature on ozone level.
```

```
```{r}
#| label: fig-airquality
#| fig-cap: Temperature and ozone level.
#| warning: false
```

```
library(ggplot2)
```

```
ggplot(airquality, aes(Temp, Ozone)) +
 geom_point() +
 geom_smooth(method = "loess")
```
``
```

Source

ggplot2 demo

Norah Jones

May 22nd, 2021

Output

Air Quality

[Figure 1](#) further explores the impact of temperature on ozone level.

► Code

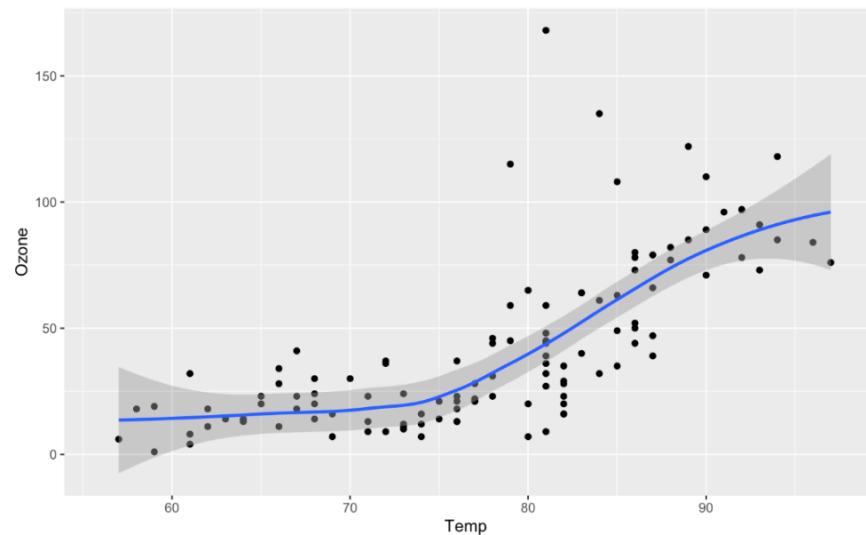
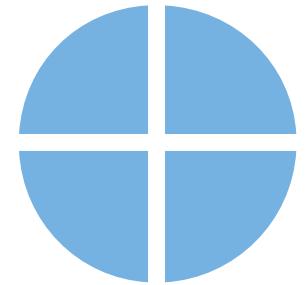


Figure 1: Temperature and ozone level.

Quarto

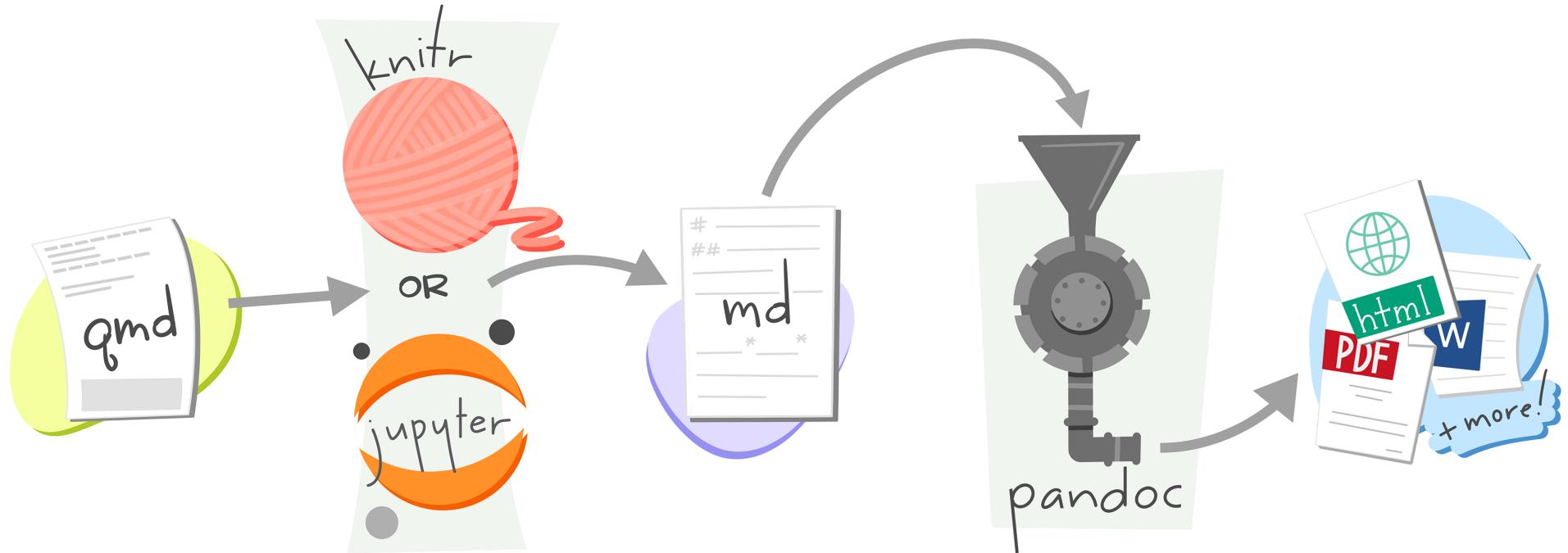


Dynamic: code and text in same document

Reproducible: re-run your analysis, re-render your document

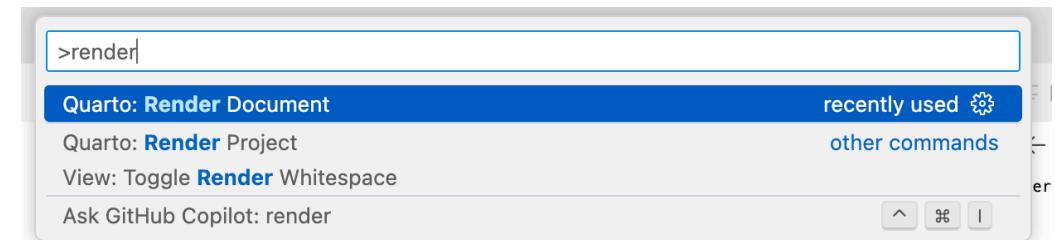
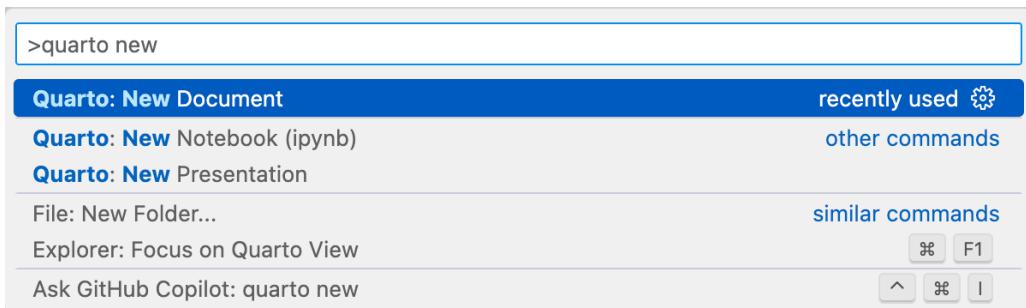
Flexible: output to different formats easily

Rendering



Your Turn 1

Create a new Quarto file. Use the Command Palette (Cmd/ctrl + Shift + P) in VS Code to create a new Quarto document. Save the file and run the Render command.



```
1  ---  
2  title: "Untitled"  
3  format: html  
4  jupyter: python3  
5  ---  
6  
7  ## Quarto  
8  
9  Quarto enables you to weave together content and executable code into a finished document.  
10 To learn more about Quarto see <https://quarto.org>.  
11  
12 ## Running Code  
13  
14 When you click the **Render** button a document will be generated that includes both content and the output of embedded code.  
15 You can embed code like this:  
16  
17  ▷ Run Cell | Run Next Cell  
18  ``{python}  
19  1 + 1  
20  ...  
21  
22  You can add options to executable code like this  
23  
24  ▷ Run Cell | Run Above  
25  ``{python}  
26  #| echo: false  
27  2 * 2  
28  ...  
29  
30  The `echo: false` option disables the printing of code (only output is displayed).  
31
```

← document metadata

→ plain text

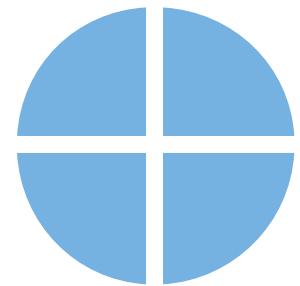
code chunks (cells)

Quarto

Prose

Code

Metadata

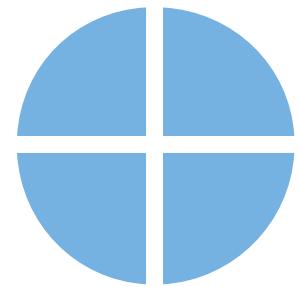


Quarto

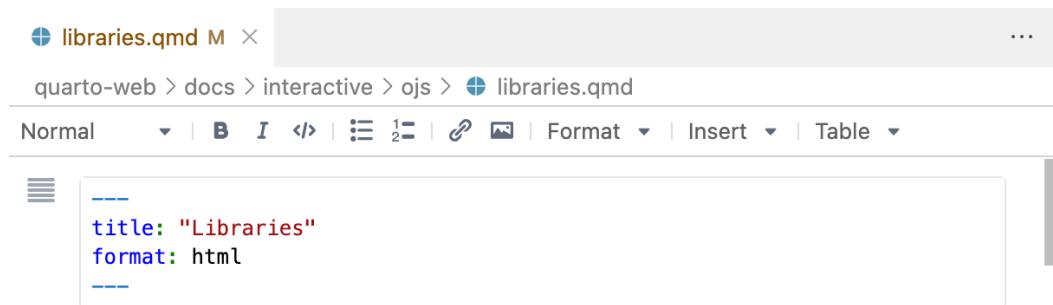
Prose = *Markdown*

Code

Metadata



the Visual Editor



The screenshot shows the Quarto Visual Editor interface. At the top, there's a header bar with a file icon, the text "libraries.qmd M X", and a three-dot menu icon. Below the header is a breadcrumb navigation path: "quarto-web > docs > interactive > ojs > libraries.qmd". The main area is a code editor with a "Normal" toolbar above it. The toolbar includes icons for bold (B), italic (I), code (C), alignment (A), font size (F), and other document operations. The code editor contains the following YAML front matter:

```
----  
title: "Libraries"  
format: html  
----
```

Overview

There are three types of library you'll generally use with OJS:

1. [Observable core libraries](#) automatically available in every document.
2. Third-party JavaScript libraries from [npm](#) and [ObservableHQ](#).
3. Custom libraries you and/or your colleagues have created

In this document we'll provide a high-level overview of the core libraries and some examples of using third-party libraries ([D3](#) and [Arquero](#)). Creating your own libraries is covered in the article on [Code Reuse](#).

Basic Markdown Syntax

italic **bold**

italic __bold__

Basic Markdown Syntax

```
# Header 1
```

```
## Header 2
```

```
### Header 3
```

Basic Markdown Syntax

`http://example.com`

`[linked phrase](http://example.com)`

*Learn more about Markdown Syntax with the
ten-twenty minute tutorial on markdown at
<https://commonmark.org/help/tutorial>.*

Your Turn 2 (open exercises.qmd)

Read this short introduction to the Visual Editor:

<https://quarto.org/docs/visual-editor/vscode/>

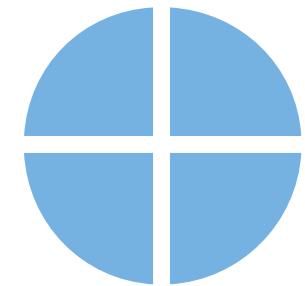
Use the Visual Editor to stylize the text from the Gapminder website below. Experiment with bolding, italicizing, making lists, etc.

Quarto

Prose

Code = *Python code chunks*

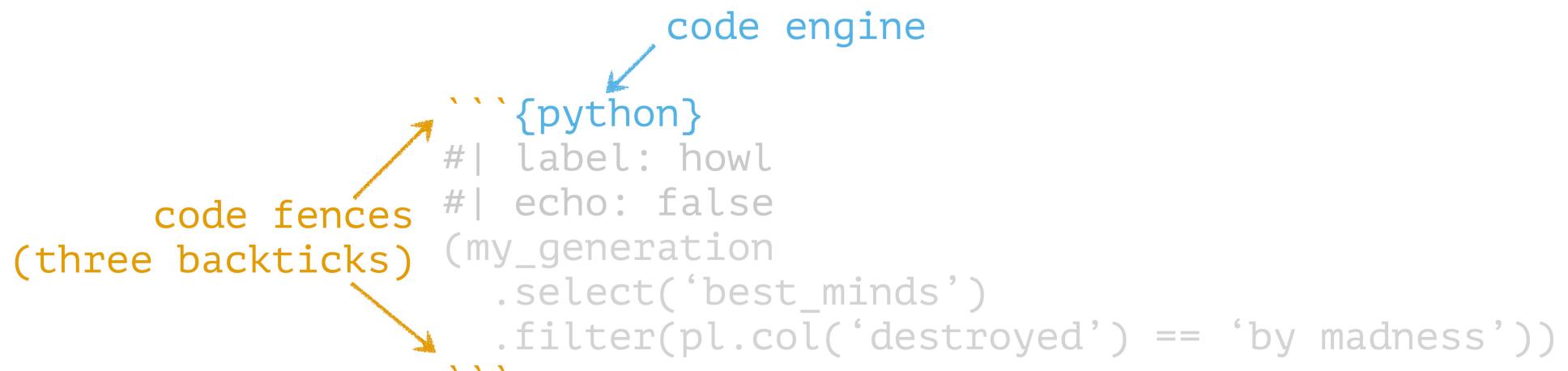
Metadata



Code chunks

```
```{python}
#| label: howl
#| echo: false
(my_generation
 .select('best_minds')
 .filter(pl.col('destroyed') == 'by madness'))
```
```

Code chunks



Code chunks

```
chunk options
```{python}
#| label: howl
#| echo: false
(my_generation
 .select('best_minds')
 .filter(pl.col('destroyed') == 'by madness'))
```
option values
```

Code chunks

code chunk comments

```
```{python}
#| label: howl
#| echo: false
(my_generation
 .select('best_minds')
 .filter(pl.col('destroyed') == 'by madness'))
````
```

Chunk options

| Option | Effect |
|----------------------------------|--|
| <code>include: false</code> | run the code but don't print it or results |
| <code>eval: false</code> | don't evaluate the code |
| <code>echo: false</code> | run the code and output but don't print code |
| <code>message: false</code> | don't print messages (e.g. from a function) |
| <code>warning: false</code> | don't print warnings |
| <code>fig-cap: "Figure 1"</code> | caption output plot with “Figure 1” |

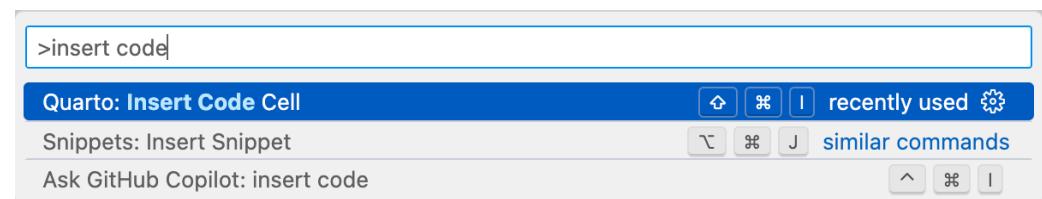
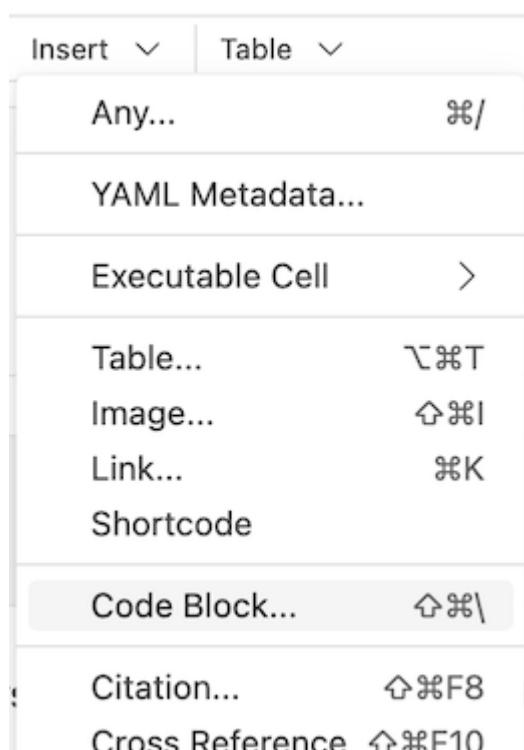
See the Quarto documentation

Other languages

First-class support for R, Julia, and Observable
JS

Supports Jupyter notebooks

Insert code chunks with cmd/ctrl + alt/option + I, Command Palette, or Visual Editor



Start typing and press tab to autocomplete chunk options

▷ Run Cell | Run Next Cell | Run Above

```
```{python}
#| label: setup
#| echo
import echo
import polars.selectors as cs
import seaborn as sns
import matplotlib.pyplot as plt
from gapminder import gapminder
from great_tables import GT
```

▷ Run Cell | Run Next Cell | Run Above

```
```{python}
#| label: setup
#| echo:
import po $\diamond$  false
import po $\diamond$  fenced
import se $\diamond$  true
import matplotlib.pyplot as plt
from gapminder import gapminder
from great_tables import GT
```

Your Turn 3

Create a code chunk. You can type it in manually, use the keyboard short-cut (Cmd/Ctrl + Option/Alt + I), or use the “Insert” button above. Put the following code in it:

```
1 GT(gapminder.head())
```

Render the document

Your Turn 4

Add `echo: false` to the code chunk you created and re-render. What's the difference in the output?

Inline Code

who wept at the romance of the
streets with their pushcarts full of
onions and bad music,
who sat in boxes breathing in the
darkness under the bridge, and rose
up to build `{python}
instruments.select('harpsichord')` in
their lofts,
who coughed on the `{python}
len(floors)` floor of Harlem crowned
with flame under the tubercular sky
surrounded by orange crates of
theology,
who scribbled all night rocking and
rolling over lofty incantations which
in the yellow morning were stanzas of
gibberish,

Inline Code

single
backtick
+ {python}

who wept at the romance of the
streets with their pushcarts full of
onions and bad music,
who sat in boxes breathing in the
darkness under the bridge, and rose
up to build ` {python}
instruments.select('harpsichord')` in
their lofts,
who coughed on the ` {python}
len(floors)` floor of Harlem crowned
with flame under the tubercular sky
surrounded by orange crates of
theology,
who scribbled all night rocking and
rolling over lofty incantations which
in the yellow morning were stanzas of
gibberish,

any Python
code

Your Turn 5

Remove `eval: false` so that Quarto evaluates the code.

Fill in the column name to use `.select()` and `to_numpy()` to get the the number of unique `years` in `gapminder` as a scalar.

Save the results as `n_years`.

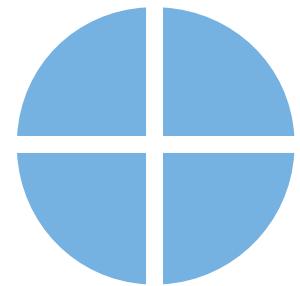
Use inline code to describe the data set in the text below the code chunk and re-render.

Quarto

Prose

Code

Metadata = YAML



YAML Metadata

```
1 ---  
2 title: "Howl"  
3 author: "Allen Ginsberg"  
4 date: "March 18, 1956"  
5 format:  
6   html: default  
7   pdf:  
8     toc: true  
9     number-sections: true  
10 ---
```

Output formats

Format	Outputs
html	HTML
pdf	PDF
word	Word .docx
odt	OpenOffice .odt
gfm	GitHub-flavored Markdown
revealjs	Reveal Slides (HTML)
beamer	Beamer Slides (PDF)
pptx	Powerpoint Slides

Your Turn 6

Set figure chunk options to the code chunk below, such as `dpi`, `fig-width`, and `fig-height`.

Add your name to the YAML header using `author: Your Name`.

Change `format: html` to use the `toc: true` and `code-fold: true` options and re-render

Parameters

```
1 #| tags: [parameters]
2 param1 = x
3 param2 = y
4 data = df
```

Calling parameters in Python

```
1 param1
2 param2
3 data
```

From the Command Line

```
1 quarto render document.qmd -P param1:0.2 -P param2:0.3
```

Your Turn 7

Change the `params` option in the YAML header to use a different continent. Re-render.

```
1 gapminder_filtered = gapminder.filter(pl.col("continent") == continent)
2
3 sns.lineplot(
4     data=gapminder_filtered,
5     x="year",
6     y="lifeExp",
7     hue="country",
8     palette="Spectral",
9     linewidth=1,
10    legend=False
11 )
12 plt.title(f"Continent: {continent}")
13 sns.set_theme(style="whitegrid")
```

Bibliographies and citations

Bibliography files: .bib, Zotero, others

Citation styles: .csl

[@citation-label]

Visual Editor's citation wizard can help!

Including bibliography files in YAML

```
1 ---  
2 bibliography: file.bib  
3 csl: file.csl  
4 ---
```

the Visual Editor can also manage this for you.

Your turn 8

Cite Causal Inference in text below. Using the citation wizard, find the right citation under My sources > Bibliography.

Add the American Journal of Epidemiology CSL to the YAML using csl: `aje.csl`

Re-render

Make cool stuff in Quarto!

- Books
- Blogs
- These slides!

See the Gallery for inspiration

Resources

Quarto Documentation: A comprehensive but friendly introduction to Quarto. Written in Quarto!

