



Coffee Lectures

# Quarto

New tool for authoring scientific publications,  
slides, websites and interactive articles

Dr. Teresa Kubacka, ETH Library  
30 November 2022, Webinar via Zoom

# Content:

- How Quarto can help you do reproducible science
- Easily create beautiful articles, blogs, books, slides using Quarto

# What is Quarto?



Quarto is an open-source  
scientific and technical publishing system  
built on Pandoc, a Swiss-army knife document converter

## Why name “Quarto”?

<https://www.folger.edu/publishing-shakespeare/diy-quarto>

## Who is behind it?

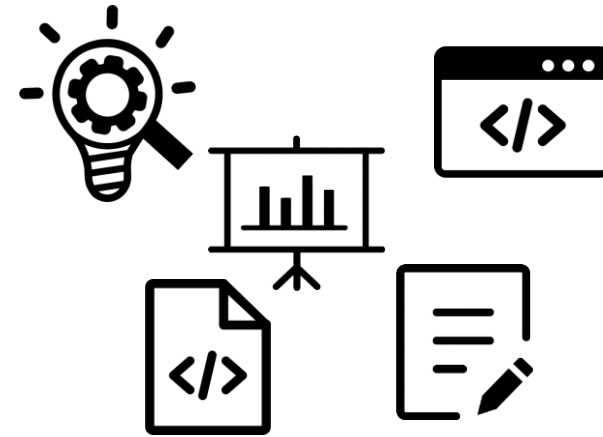


## How old is it?

- V1.0 in summer 2022
- active development
- open source community
- rapidly gaining on popularity

# Why bother?

1. You do your research analysis in Python
2. Present on a conference in Powerpoint
3. Write a blog post in Hugo / HTML+Javascript
4. Write a journal article in LaTeX and DOCX



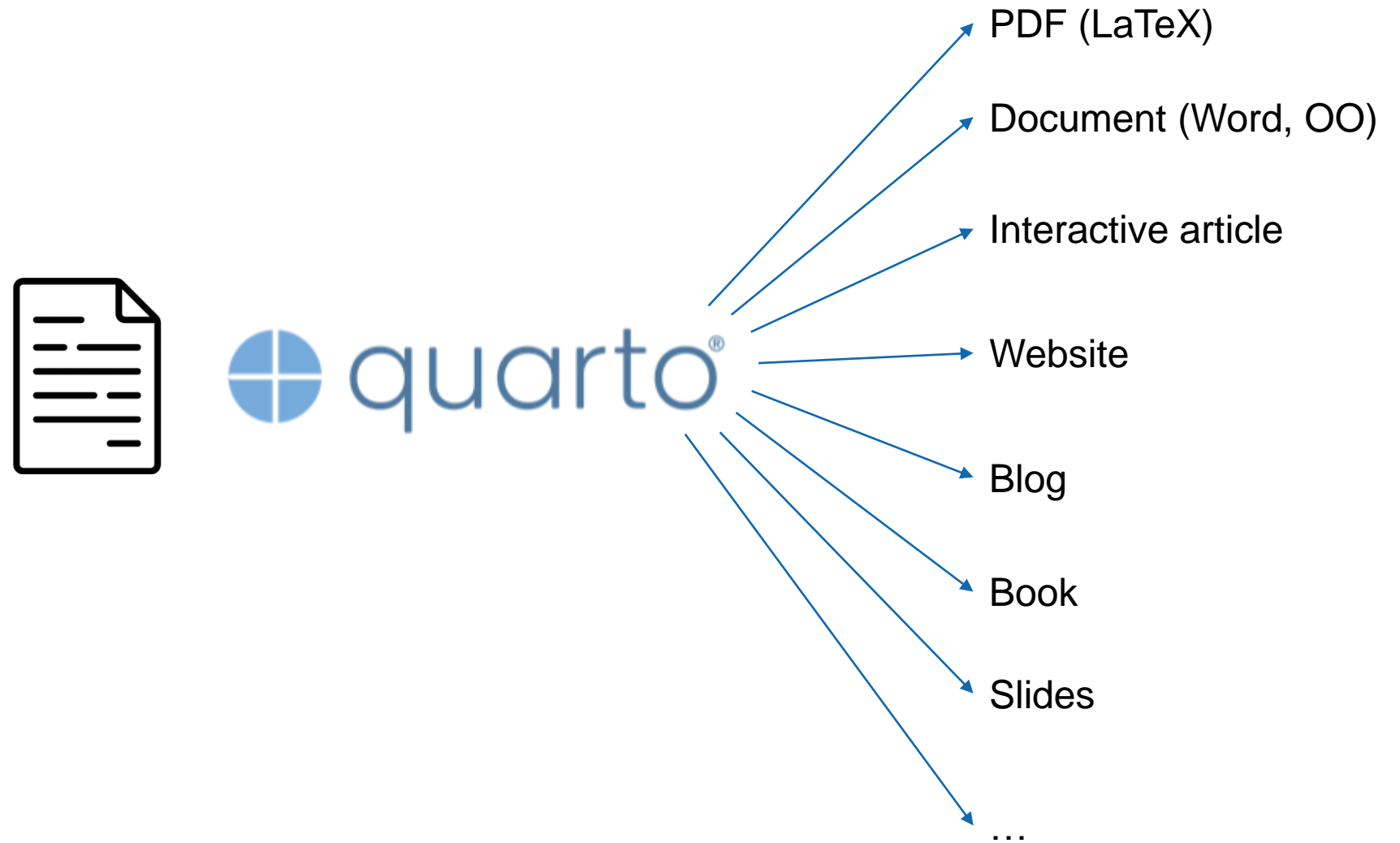
<https://boyter.org/2016/04/collection-orly-book-covers/>

...Then you discover that you need to change something in your analysis and re-do all of that!

**Quarto to the rescue!**

<https://knowyourmeme.com/memes/subcultures/webcomic-name>

# Quarto: **single source publishing**



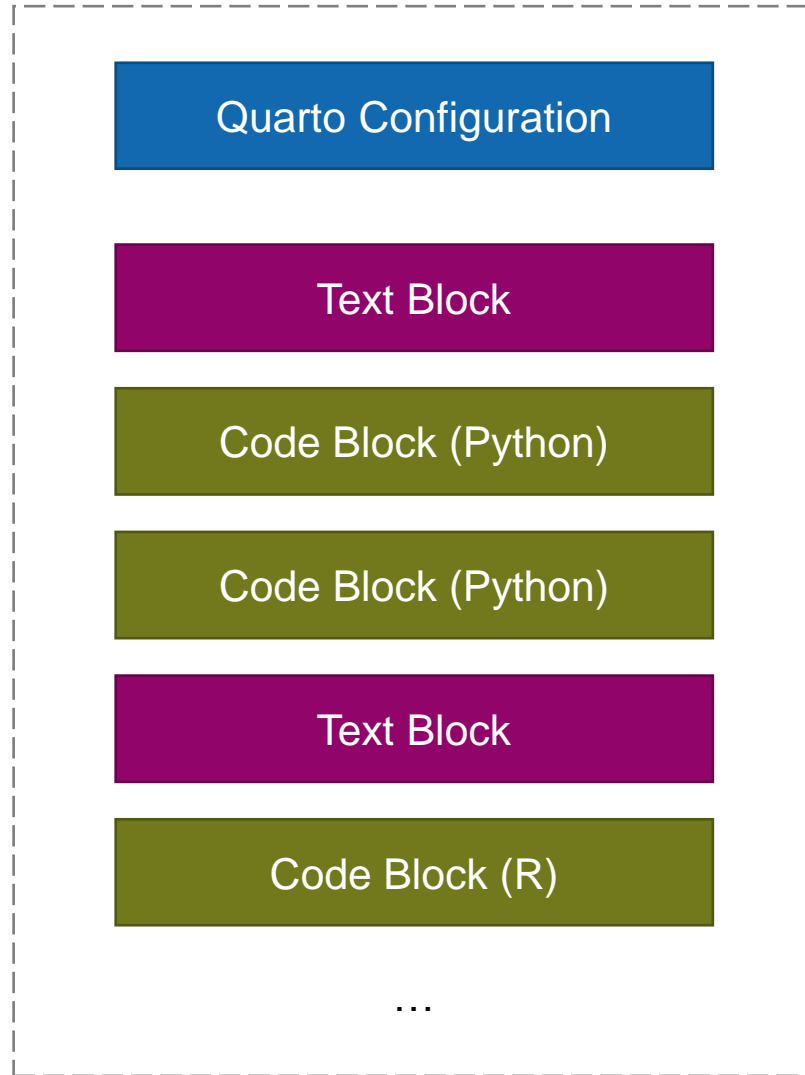
# Quarto: language-agnostic single source publishing



# Quarto: tool and language-agnostic single source publishing



# Files that consist of blocks of code and text



**Quarto configuration** of the exported output

**Text** as markdown

**Code cells** that share variables

Mix and match programming languages

Reproducibility: re-use the same codebase

Adapted to scientific publishing:

- Equations and math content via LaTeX
- Citations via BibTeX
- Cross-references
- Rich design



# Rich design: beautiful and usable outputs

Many design elements:

- figures
- tables
- block diagrams
- margin notes
- footnotes
- expandable blocks
- computational outputs
- code cells
- TOC and referenceable sections
- ...

```
mtcars2 <- mtcars
mtcars2$am <- factor(
  mtcars$am, labels = c('automatic', 'manual')
)
ggplot(mtcars2, aes(hp, mpg, color = am)) +
  geom_point() + geom_smooth() +
  theme(legend.position = 'bottom')
```

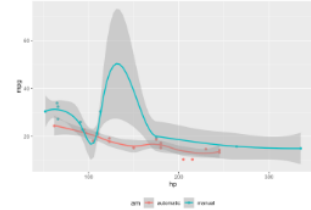


Figure 1: MPG vs horsepower, colored by transmission.

Note the use of the `fig-cap` chunk option to provide a figure caption. You can adjust the proportions of figures using the `fig-width` and `fig-height` chunk options. These are specified in inches, and will be automatically scaled down to fit within the handout margin.

## Arbitrary Margin Content

You can include anything in the margin by placing the class `.column-margin` on the element. See an example on the right about the first fundamental theorem of calculus.

We know from the first fundamental theorem of calculus that for  $x$  in  $[a, b]$ :

$$\frac{d}{dx} \left( \int_a^x f(u) du \right) = f(x).$$

## Full Width Figures

You can arrange for figures to span across the entire page by using the chunk option `fig-column: page-right`.

```
ggplot(diamonds, aes(carat, price)) + geom_smooth() +
  facet_grid(~ cut)
```

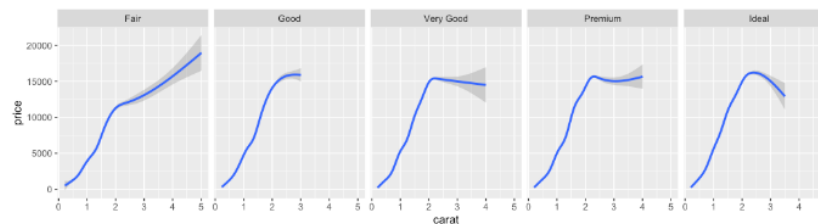
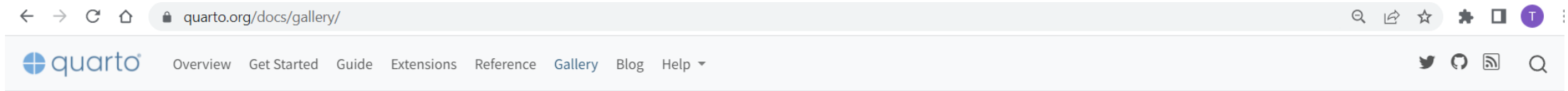


Figure 2: A full width figure.

# Check out the official gallery



### HTML with advanced layout </>

#### Margin Figures

Images and graphics play an integral role in Tuhr's work. To place figures in the margin you can use the Quarto chunk option `layout: margin`. For example:

```
1 library(ggplot2)
2 intxand2 = intxare
3 intxand2 = factor(
4   intxand2, labels = c("automatic", "manual"))
5
6 apply(intxand2, MARGIN = "margin", FUN = intxare) +
7   geom_smooth() = geom_smooth()
8   theme(layout.position = "margin")
```

Note the use of the `layout: margin` option to provide a figure caption. You can adjust the proportions of figures using the `fig-width` and `fig-height` chunk options. These are specified in inches, and will be automatically scaled down to fit within the provided margin.

#### Arbitrary Margin Content

You can include anything in the margin by placing the class `colspan: margin` on the element. See an example on the right about the first fundamental theorem of calculus.

We know from the first fundamental theorem of calculus that for  $x$  in  $(a, b)$

$$\frac{d}{dx} \left( \int_a^x f(t) dt \right) = f(x)$$

### PDF with advanced layout </>

#### Arbitrary Margin Content

You can include anything in the margin by placing the class `colspan: margin` on the element. See an example on the right about the first fundamental theorem of calculus.

#### Full Width Figures

You can arrange for figures to span across the entire page by using the chunk option `fig-column: page-right`.

We know from the first fundamental theorem of calculus that for  $x$  in  $(a, b)$

$$\frac{d}{dx} \left( \int_a^x f(t) dt \right) = f(x)$$

```
1 ggplot(diamonds, aes(carat, price)) + geom_smooth() +
2   facet_grid(~ cut)
```

Figure 2: A full width figure.

## Presentations

Create presentations (slide show) in a variety of formats.

### RevealJS HTML presentations </>

1 / 28

**Quarto Presentations**  
Create beautiful interactive slide decks with Reveal.js

<https://quarto.org>

### Beamer PDF presentations </>

Pop Songs and Political Science

Steven V. Miller  
Department of Political Science

### PowerPoint presentations </>

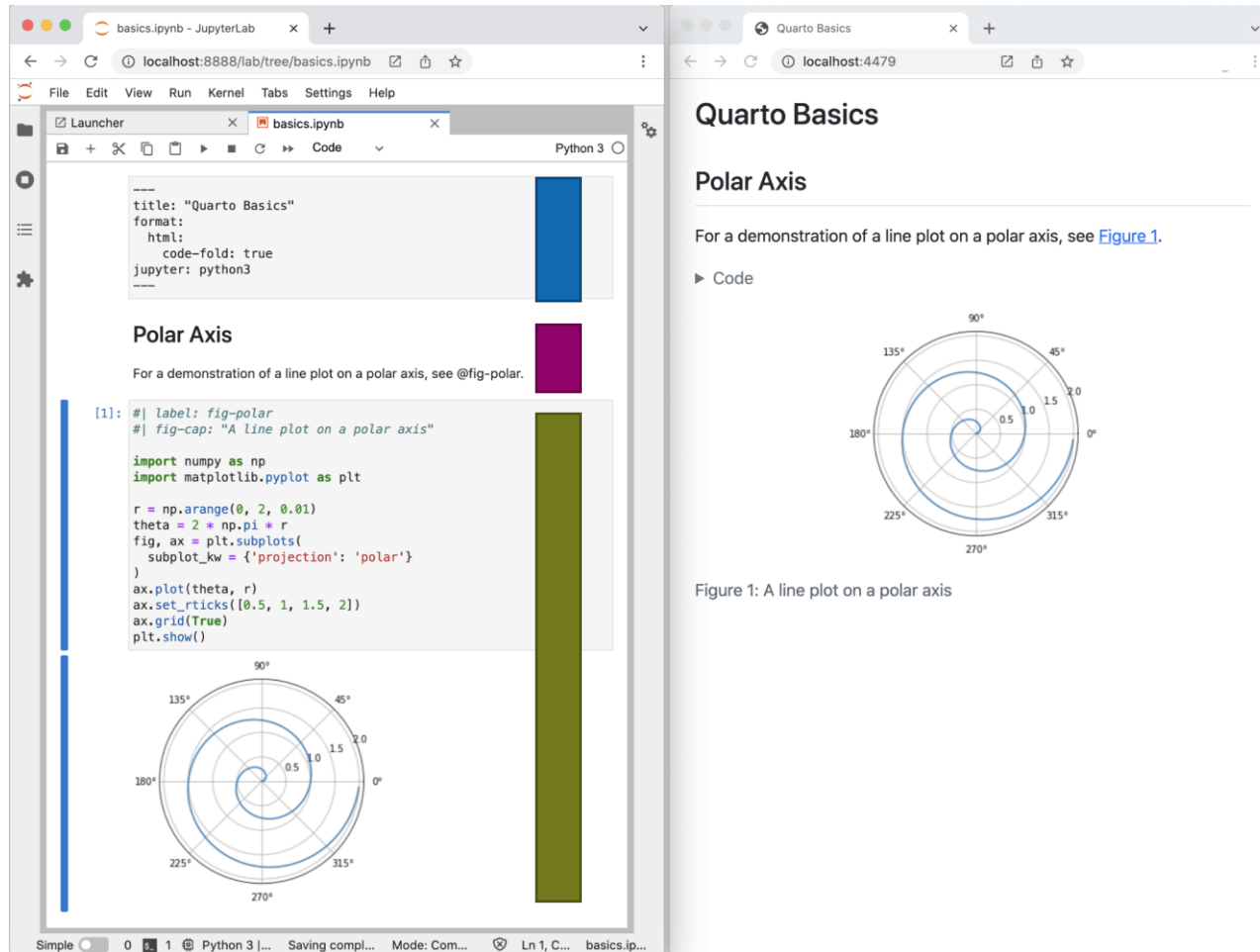
Best Practices for Administering RStudio in Production

Nathan Stephens

<https://quarto.org/docs/gallery/>

# And how does it look in practice?

Use a Jupyter Notebook / Rmarkdown file



The image shows two side-by-side browser windows. The left window is a JupyterLab interface for a file named 'basics.ipynb'. It displays a code cell with the following content:

```
---  
title: "Quarto Basics"  
format:  
  html:  
    code-fold: true  
  jupyter: python3  
---
```

Below the code is a section titled "Polar Axis" with the text "For a demonstration of a line plot on a polar axis, see @fig-polar." and a code cell:

```
[1]: ## label: fig-polar  
## fig-cap: "A line plot on a polar axis"  
  
import numpy as np  
import matplotlib.pyplot as plt  
  
r = np.arange(0, 2, 0.01)  
theta = 2 * np.pi * r  
fig, ax = plt.subplots(  
  subplot_kw = {'projection': 'polar'}  
)  
ax.plot(theta, r)  
ax.set_rticks([0.5, 1, 1.5, 2])  
ax.grid(True)  
plt.show()
```

Below the code is a polar plot showing a spiral line. The plot has concentric circles at radii 0.5, 1.0, 1.5, and 2.0, and radial lines at angles 0°, 45°, 90°, 135°, 180°, 225°, 270°, and 315°. The spiral starts at the origin and winds outwards.

The right window is a Quarto document titled "Quarto Basics" with a section titled "Polar Axis". It contains the text "For a demonstration of a line plot on a polar axis, see [Figure 1](#)." and a code cell:

```
▶ Code
```

Below the code is a polar plot identical to the one in the Jupyter Notebook. Below the plot is the caption "Figure 1: A line plot on a polar axis".

<https://quarto.org/docs/get-started/hello/jupyter.html>

<https://quarto.org/docs/get-started/hello/vscode.html>

# And how does it look in practice?

Use a Jupyter Notebook / Rmarkdown file

The image shows two side-by-side browser windows. The left window is a JupyterLab interface for a file named 'basics.ipynb'. It displays a code cell with the following content:

```
---  
title: "Quarto Basics"  
format:  
  html:  
    code-fold: true  
  jupyter: python3  
---
```

Below the code is a section titled "Polar Axis" with the text "For a demonstration of a line plot on a polar axis, see @fig-polar." and a code cell:

```
[1]: ## label: fig-polar  
## fig-cap: "A line plot on a polar axis"  
  
import numpy as np  
import matplotlib.pyplot as plt  
  
r = np.arange(0, 2, 0.01)  
theta = 2 * np.pi * r  
fig, ax = plt.subplots(  
  subplot_kw = {'projection': 'polar'}  
)  
ax.plot(theta, r)  
ax.set_rticks([0.5, 1, 1.5, 2])  
ax.grid(True)  
plt.show()
```

The rendered output of the code cell is a polar plot showing a spiral. The plot has concentric circles at radii 0.5, 1.0, 1.5, and 2.0, and radial lines at angles 0°, 45°, 90°, 135°, 180°, 225°, 270°, and 315°. The spiral starts at the origin and winds outwards.

The right window shows a Quarto document titled "Quarto Basics". It has a section titled "Polar Axis" with the text "For a demonstration of a line plot on a polar axis, see [Figure 1](#)." Below this is the same polar plot as in the Jupyter Notebook. The Quarto document also includes a code block for the same code as in the Jupyter Notebook.

<https://quarto.org/docs/get-started/hello/jupyter.html>

<https://quarto.org/docs/get-started/hello/vscode.html>

# And how does it look in practice?

Use a Jupyter Notebook / Rmarkdown file

The image shows two side-by-side windows. The left window is a JupyterLab interface displaying a Jupyter Notebook. The notebook has a title "Quarto Basics" and a code cell with the following Python code:

```
[1]: #!/ label: fig-polar
#!/ fig-cap: "A line plot on a polar axis"

import numpy as np
import matplotlib.pyplot as plt

r = np.arange(0, 2, 0.01)
theta = 2 * np.pi * r
fig, ax = plt.subplots(
    subplot_kw = {'projection': 'polar'}
)
ax.plot(theta, r)
ax.set_rticks([0.5, 1, 1.5, 2])
ax.grid(True)
plt.show()
```

Below the code cell, a polar plot is displayed. The plot has a radial axis from 0 to 2 and an angular axis from 0° to 315° in 45° increments. A blue spiral line starts at the origin and winds outwards. The plot is titled "Polar Axis" and has a caption "For a demonstration of a line plot on a polar axis, see @fig-polar.".

The right window is a Quarto document titled "Quarto Basics". It has a section titled "Polar Axis" with the text "For a demonstration of a line plot on a polar axis, see [Figure 1](#)." Below the text is a "Code" button and the same polar plot as in the Jupyter Notebook. The plot is captioned "Figure 1: A line plot on a polar axis".

<https://quarto.org/docs/get-started/hello/jupyter.html>

<https://quarto.org/docs/get-started/hello/vscode.html>

# And how does it look in practice?

Use a Jupyter Notebook / Rmarkdown file

The image shows two side-by-side windows. The left window is a JupyterLab interface for a file named 'basics.ipynb'. It displays a code cell with the following Python code:

```
[1]: #!/ label: fig-polar
#!/ fig-cap: "A line plot on a polar axis"

import numpy as np
import matplotlib.pyplot as plt

r = np.arange(0, 2, 0.01)
theta = 2 * np.pi * r
fig, ax = plt.subplots(
    subplot_kw = {'projection': 'polar'}
)
ax.plot(theta, r)
ax.set_rticks([0.5, 1, 1.5, 2])
ax.grid(True)
plt.show()
```

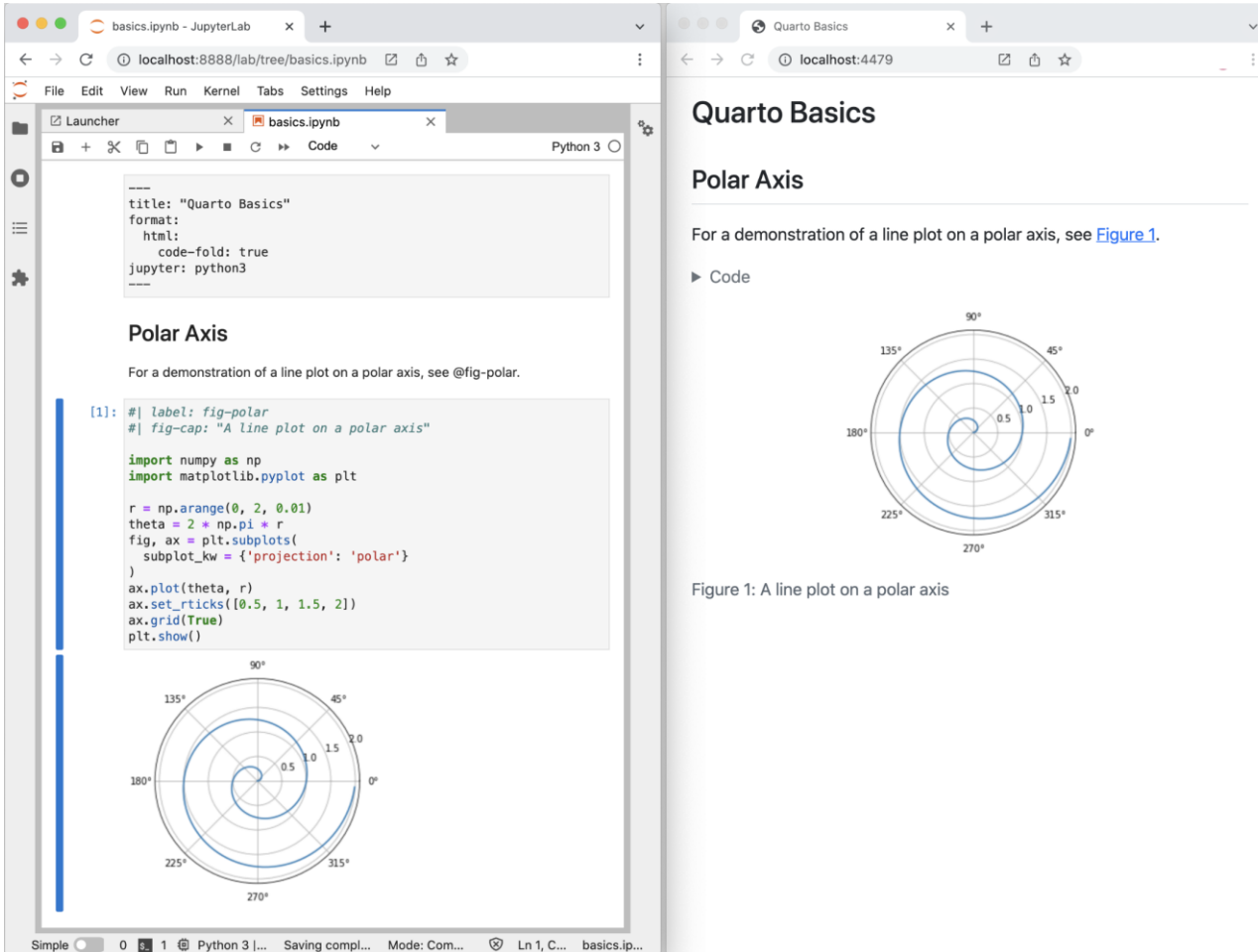
Below the code, a polar plot is rendered, showing a spiral line on a polar coordinate system with radial ticks at 0.5, 1.0, 1.5, and 2.0, and angular ticks at 0°, 45°, 90°, 135°, 180°, 225°, 270°, and 315°. The right window is a Quarto document titled 'Quarto Basics' with a section 'Polar Axis'. It contains the text: 'For a demonstration of a line plot on a polar axis, see [Figure 1](#).' Below this text is a 'Code' button and the same polar plot as seen in the JupyterLab window. A caption below the plot reads 'Figure 1: A line plot on a polar axis'. Green arrows point from the code in the JupyterLab window to the code button and the plot in the Quarto document, and from the plot in the JupyterLab window to the plot in the Quarto document.

<https://quarto.org/docs/get-started/hello/jupyter.html>

<https://quarto.org/docs/get-started/hello/vscode.html>

# And how does it look in practice?

Use a Jupyter Notebook / Rmarkdown file



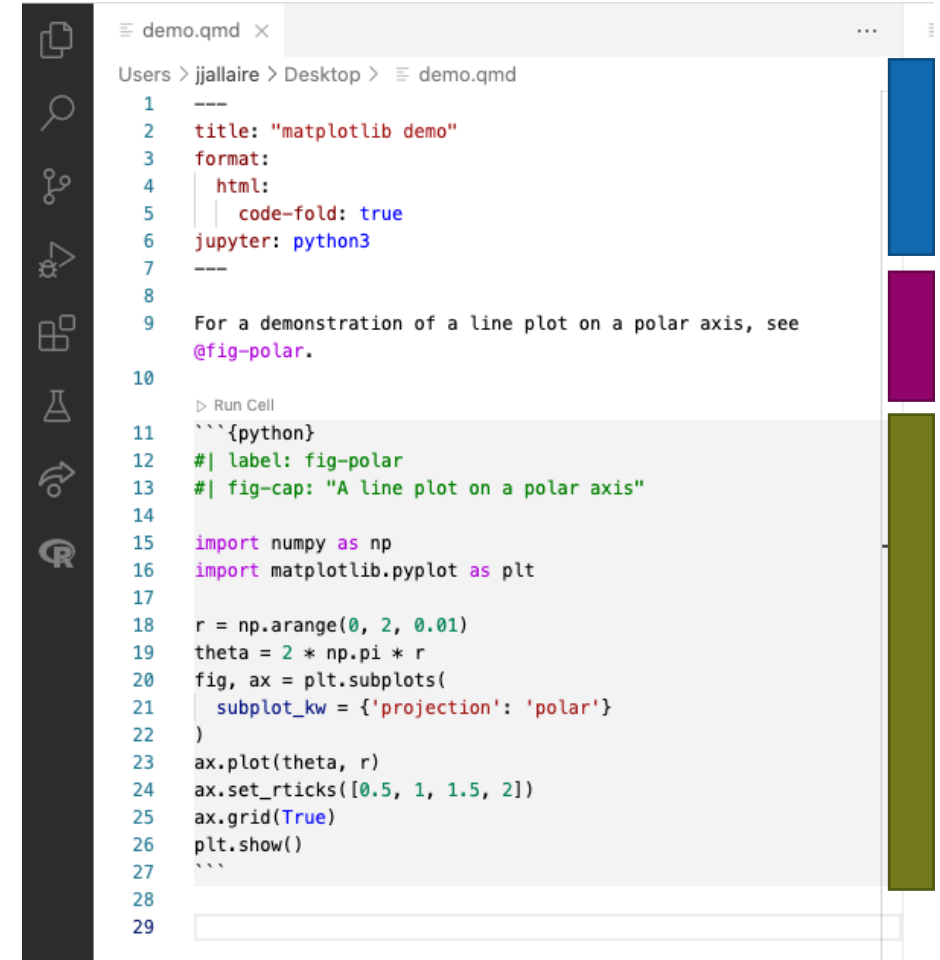
The screenshot shows a Jupyter Notebook interface with a browser window at localhost:8888. The notebook contains a code cell with the following Python code:

```
---  
title: "Quarto Basics"  
format:  
  html:  
    code-fold: true  
  jupyter: python3  
---  
  
Polar Axis  
For a demonstration of a line plot on a polar axis, see @fig-polar.  
  
[1]: ##| label: fig-polar  
##| fig-cap: "A line plot on a polar axis"  
  
import numpy as np  
import matplotlib.pyplot as plt  
  
r = np.arange(0, 2, 0.01)  
theta = 2 * np.pi * r  
fig, ax = plt.subplots(  
  subplot_kw = {'projection': 'polar'}  
)  
ax.plot(theta, r)  
ax.set_rticks([0.5, 1, 1.5, 2])  
ax.grid(True)  
plt.show()
```

Below the code is a polar plot showing a spiral line. The plot has a radial axis from 0 to 2 and an angular axis from 0° to 315° in 45° increments. The spiral starts at the origin and winds outwards.

<https://quarto.org/docs/get-started/hello/jupyter.html>

Or a Quarto markdown file .qmd



The screenshot shows a Quarto markdown file interface with a browser window at localhost:4479. The file contains the following Quarto code:

```
demo.qmd  
Users > jjallaire > Desktop > demo.qmd  
1 ---  
2 title: "matplotlib demo"  
3 format:  
4   html:  
5     code-fold: true  
6   jupyter: python3  
7 ---  
8  
9 For a demonstration of a line plot on a polar axis, see  
10 @fig-polar.  
11  
12 ▶ Run Cell  
13 {python}  
14 ##| label: fig-polar  
15 ##| fig-cap: "A line plot on a polar axis"  
16  
17 import numpy as np  
18 import matplotlib.pyplot as plt  
19  
20 r = np.arange(0, 2, 0.01)  
21 theta = 2 * np.pi * r  
22 fig, ax = plt.subplots(  
23   subplot_kw = {'projection': 'polar'}  
24 )  
25 ax.plot(theta, r)  
26 ax.set_rticks([0.5, 1, 1.5, 2])  
27 ax.grid(True)  
28 plt.show()  
29
```

The code is identical to the Jupyter Notebook example, and it produces the same polar plot of a spiral line.

<https://quarto.org/docs/get-started/hello/vscode.html>

# And how does it look in practice?

Use a Jupyter Notebook / Rmarkdown file

Or a Quarto markdown file .qmd

The screenshot shows a JupyterLab interface with a browser window displaying a Quarto Basics file. The code cell contains the following Python code:

```
---  
title: "Quarto Basics"  
format:  
  html:  
    code-fold: true  
  jupyter: python3  
---  
  
Polar Axis  
  
For a demonstration of a line plot on a polar axis, see @fig-polar.  
  
[1]: ## label: fig-polar  
## fig-cap: "A line plot on a polar axis"  
  
import numpy as np  
import matplotlib.pyplot as plt  
  
r = np.arange(0, 2, 0.01)  
theta = 2 * np.pi * r  
fig, ax = plt.subplots(  
  subplot_kw={'projection': 'polar'}  
)  
ax.plot(theta, r)  
ax.set_rticks([0.5, 1, 1.5, 2])  
ax.grid(True)  
plt.show()
```

The output of the code is a polar plot showing a spiral curve. The plot has a radial axis from 0 to 2 and an angular axis from 0 to 360 degrees. The spiral starts at the origin and winds outwards.

The screenshot shows a Quarto markdown file (.qmd) with the following code:

```
1 ---  
2 title: "matplotlib demo"  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23 ax.plot(theta, r)  
24 ax.set_rticks([0.5, 1, 1.5, 2])  
25 ax.grid(True)  
26 plt.show()  
27  
28  
29
```

To see the output, open the terminal and run

`quarto render`  
or  
`quarto preview`

...and that's it!

<https://quarto.org/docs/get-started/hello/jupyter.html>

<https://quarto.org/docs/get-started/hello/vscode.html>



# Easy and customizable

Quarto is very easy to use if you don't want to dig into details

```
Terminal
```

```
quarto render authoring.qmd
```



```
Terminal
```

```
quarto publish gh-pages
```



And offers you customization when you need it:

- Advanced customization with HTML/CSS/JS and Lua
- Custom extensions and templates, e.g.:
  - Display macro-molecules with molstar
  - Templates for most of scientific publishers and preprint servers
- Various deployment configurations if you need a custom one

Go and try it out!



<https://quarto.org/>

 [mcanouil / awesome-quarto](https://github.com/mcanouil/awesome-quarto) Public

<https://github.com/mcanouil/awesome-quarto>

Dr. Teresa Kubacka  
Data Scientist, Knowledge Management Group  
teresa.kubacka@library.ethz.ch  
kom@library.ethz.ch

ETH Zürich  
ETH-Bibliothek, Knowledge Management  
HG H11.1  
Rämistrasse 101  
8092 Zürich

[www.library.ethz.ch/coffee-lectures](http://www.library.ethz.ch/coffee-lectures)