Introduction & Setup

Session 1

Mickaël CANOUIL, *Ph.D.*

pro@mickael.canouil.dev

Independent Contractor

Saturday, the 7th of February, 2026

 |

Introduction

- About Me
- Session 1: Introduction & Setup
- Workshop Objectives
- Learning Objectives

 |

About Me

- mickael.canouil.fr
- pro@mickael.canouil.dev
- @mcanouil
- mickaelcanouil
- @mickael.canouil.fr
- @MickaelCanouil@fosstodon.org

- *I am Mickaël CANOUIL, I hold a Ph.D. in Biostatistics, with over a decade of academic experience in the genetics of type 2 diabetes and obesity.*
- *My research has contributed to understanding the genetic and molecular mechanisms underlying metabolic diseases, with publications in leading journals.*
- *Currently, I work as a consultant in biostatistics, applying my expertise to diverse projects in multi-omics and data analysis.*
- *I am also deeply involved in the Quarto ecosystem, developing extensions and tools that enhance reproducibility and scientific communication.*
- *My contributions, including Quarto Wizard and various Quarto extensions, aim to streamline workflows for researchers and data scientists.*
- *You can explore my work on GitHub, my projects, and my publications.*

Session 1: Introduction & Setup

 |

Session 1: Introduction & Setup

- **Welcome & Overview**
  - Workshop objectives and participant goals.
  - Quarto overview: purpose, strengths, and reproducibility benefits.
  - What Quarto is and how it works with Pandoc.
  - Brief look at the Quarto Guide.

Session 1: Introduction & Setup

- **Welcome & Overview**
  - Workshop objectives and participant goals.
  - Quarto overview: purpose, strengths, and reproducibility benefits.
  - What Quarto is and how it works with Pandoc.
  - Brief look at the Quarto Guide.
- **Installation & Environment Setup**
  - Installing Quarto (installer, Homebrew, Chocolatey).
  - Verify installation with `quarto check`.
  - Quarto as a command-line interface.

Session 1: Introduction & Setup

- **Welcome & Overview**
  - Workshop objectives and participant goals.
  - Quarto overview: purpose, strengths, and reproducibility benefits.
  - What Quarto is and how it works with Pandoc.
  - Brief look at the Quarto Guide.
- **Quarto Projects**
  - Project types: default, website, blog, book, manuscript.
  - Creating projects with `quarto create project`.
  - Writing with various editors (VS Code, RStudio, Jupyter, etc.).
- **Installation & Environment Setup**
  - Installing Quarto (installer, Homebrew, Chocolatey).
  - Verify installation with `quarto check`.
  - Quarto as a command-line interface.

Workshop Objectives

 |

Workshop Objectives

- **Familiarise** participants with the Quarto ecosystem.

 |

Workshop Objectives

- **Familiarise** participants with the Quarto ecosystem.
- **Empower** attendees to create reproducible documents and interactive publications.

 |

Workshop Objectives

- **Familiarise** participants with the Quarto ecosystem.
- **Empower** attendees to create reproducible documents and interactive publications.
- **Guide** learners through embedding code, adjusting outputs, and publishing projects.

 |

Workshop Objectives

- **Familiarise** participants with the Quarto ecosystem.
- **Empower** attendees to create reproducible documents and interactive publications.
- **Guide** learners through embedding code, adjusting outputs, and publishing projects.
- **Encourage** hands-on practice with interactive sessions.

 |

Learning Objectives

 |

Learning Objectives

By the end of this session, participants will be able to:

 |

Learning Objectives

By the end of this session, participants will be able to:

- Explain what Quarto is and how it relates to Pandoc.

 |

Learning Objectives

By the end of this session, participants will be able to:

- Explain what Quarto is and how it relates to Pandoc.
- Install Quarto and verify the installation.

 |

Learning Objectives

By the end of this session, participants will be able to:

- Explain what Quarto is and how it relates to Pandoc.
- Install Quarto and verify the installation.
- Create and render different Quarto project types.

 |

Learning Objectives

By the end of this session, participants will be able to:

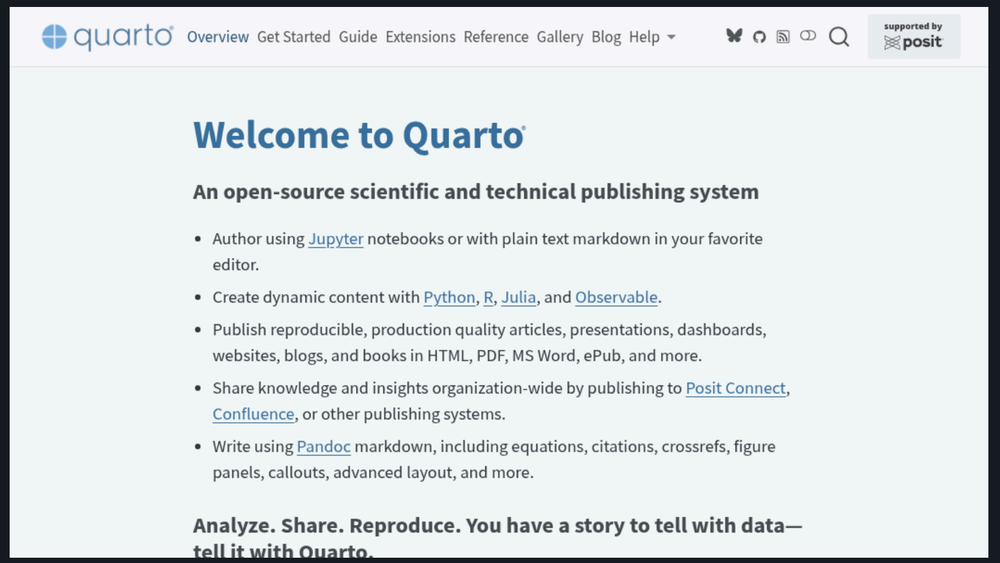
- Explain what Quarto is and how it relates to Pandoc.
- Install Quarto and verify the installation.
- Create and render different Quarto project types.
- Set up an IDE for Quarto development.

 |

- What is Quarto?
- Purpose
- Key Strengths
- Reproducibility Benefits
- What is Pandoc?
- How does Quarto work?
- A Guide to Quarto Versions

What is Quarto? 🤔

What is Quarto? 🤔



https://quarto.org/

Purpose

Quarto is an open-source scientific and technical publishing system built on Pandoc:

Purpose

Quarto is an open-source scientific and technical publishing system built on Pandoc:

- Multi-language platform supporting Python, R, Julia, and Observable.js.

Purpose

Quarto is an open-source scientific and technical publishing system built on Pandoc:

- Multi-language platform supporting Python, R, Julia, and Observable.js.
- Creates dynamic, reproducible documents from code and narrative text.

Purpose

Quarto is an open-source scientific and technical publishing system built on Pandoc:

- Multi-language platform supporting Python, R, Julia, and Observable.js.
- Creates dynamic, reproducible documents from code and narrative text.
- Single-source authoring for articles, websites, books, presentations, and dashboards.

Key Strengths

 |

Key Strengths

Multi-format output

Generate HTML, PDF, Word, ePub, and more from one source.

 |

Key Strengths

Multi-format output

Generate HTML, PDF, Word, ePub, and more from one source.

Advanced markdown

Cross-references, equations, citations, callouts, and complex layouts.

 |

Key Strengths

Multi-format output

Generate HTML, PDF, Word, ePub, and more from one source.

Advanced markdown

Cross-references, equations, citations, callouts, and complex layouts.

Tool flexibility

Works with VS Code, RStudio, JupyterLab, or any text editor.

 |

Key Strengths

Multi-format output

Generate HTML, PDF, Word, ePub, and more from one source.

Advanced markdown

Cross-references, equations, citations, callouts, and complex layouts.

Tool flexibility

Works with VS Code, RStudio, JupyterLab, or any text editor.

Project system

Render document collections with shared configurations.

 |

Key Strengths

Multi-format output

Generate HTML, PDF, Word, ePub, and more from one source.

Advanced markdown

Cross-references, equations, citations, callouts, and complex layouts.

Tool flexibility

Works with VS Code, RStudio, JupyterLab, or any text editor.

Project system

Render document collections with shared configurations.

Interactive elements

Embed widgets, Shiny apps, and Observable JS for dynamic content.

 |

Reproducibility Benefits

 |

Reproducibility Benefits

Embedded computation

Code and results stay together, eliminating copy-paste errors.

 |

Reproducibility Benefits

Embedded computation

Code and results stay together, eliminating copy-paste errors.

Version control friendly

Plain text format works seamlessly with Git.

 |

Reproducibility Benefits

Embedded computation

Code and results stay together, eliminating copy-paste errors.

Version control friendly

Plain text format works seamlessly with Git.

Transparent workflows

Complete audit trail from data to final output.

 |

Reproducibility Benefits

Embedded computation
: Code and results stay together, eliminating copy-paste errors.

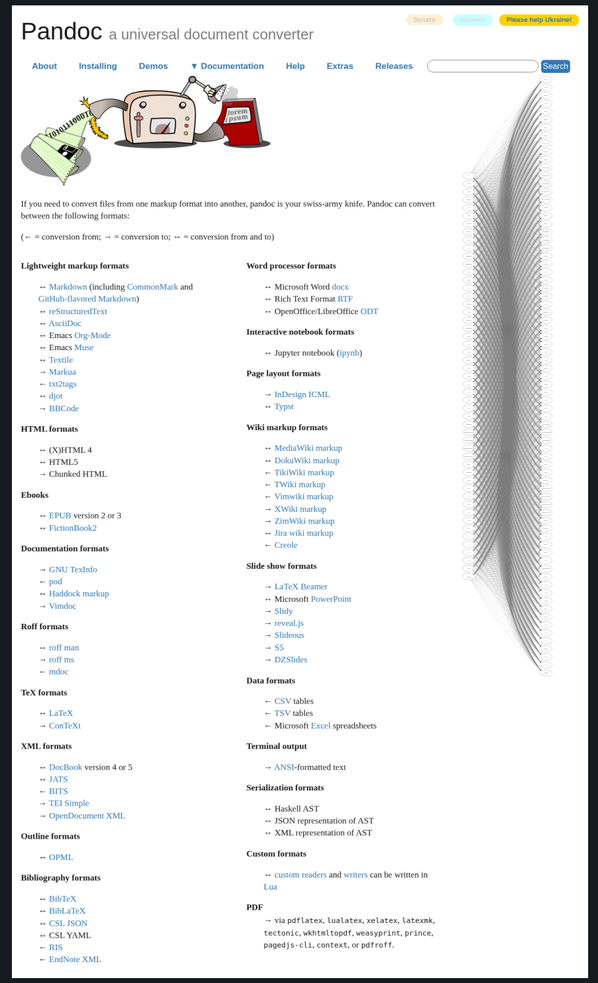
Version control friendly
: Plain text format works seamlessly with Git.

Transparent workflows
: Complete audit trail from data to final output.

Collaborative research
: Shared environments and dependencies ensure consistent results.

 |

What is Pandoc? 🤔



Pandoc a universal document converter

Donate Sponsor Please help Ukraine!

About Installing Demos ▼ Documentation Help Extras Releases Search

If you need to convert files from one markup format into another, pandoc is your swiss-army knife. Pandoc can convert between the following formats:

(← = conversion from; → = conversion to; ↔ = conversion from and to)

Lightweight markup formats

↔ Markdown (including CommonMark and GitHub-flavored Markdown)
↔ reStructuredText
↔ AsciiDoc
↔ Emacs Org-Mode
↔ Emacs Muse
↔ Textile
→ Markua
← txt2tags
↔ djot
→ BBCode

HTML formats

↔ (X)HTML 4
↔ HTML5
→ Chunked HTML

Ebooks

↔ EPUB version 2 or 3
↔ FictionBook2

Documentation formats

→ GNU TexInfo
← pod
↔ Haddock markup
→ Vimdoc

Roff formats

↔ roff man
→ roff ms
← mdoc

TeX formats

↔ LaTeX
→ ConTeXt

XML formats

↔ DocBook version 4 or 5
↔ JATS
← BITS
→ TEI Simple
→ OpenDocument XML

Outline formats

↔ OPML

Bibliography formats

↔ BibTeX
↔ BibLaTeX
↔ CSL JSON
↔ CSL YAML
← RIS
← EndNote XML

Word processor formats

↔ Microsoft Word docx
↔ Rich Text Format RTF
↔ OpenOffice/LibreOffice ODT

Interactive notebook formats

↔ Jupyter notebook (ipynb)

Page layout formats

→ InDesign ICML
↔ Typst

Wiki markup formats

↔ MediaWiki markup
↔ DokuWiki markup
← TikiWiki markup
← TWiki markup
← Vimwiki markup
→ XWiki markup
→ ZimWiki markup
↔ Jira wiki markup
← Creole

Slide show formats

→ LaTeX Beamer
↔ Microsoft PowerPoint
→ Slidy
→ reveal.js
→ Slideous
→ S5
→ DZSlides

Data formats

← CSV tables
← TSV tables
← Microsoft Excel spreadsheets

Terminal output

→ ANSI-formatted text

Serialization formats

↔ Haskell AST
↔ JSON representation of AST
↔ XML representation of AST

Custom formats

↔ custom readers and writers can be written in Lua

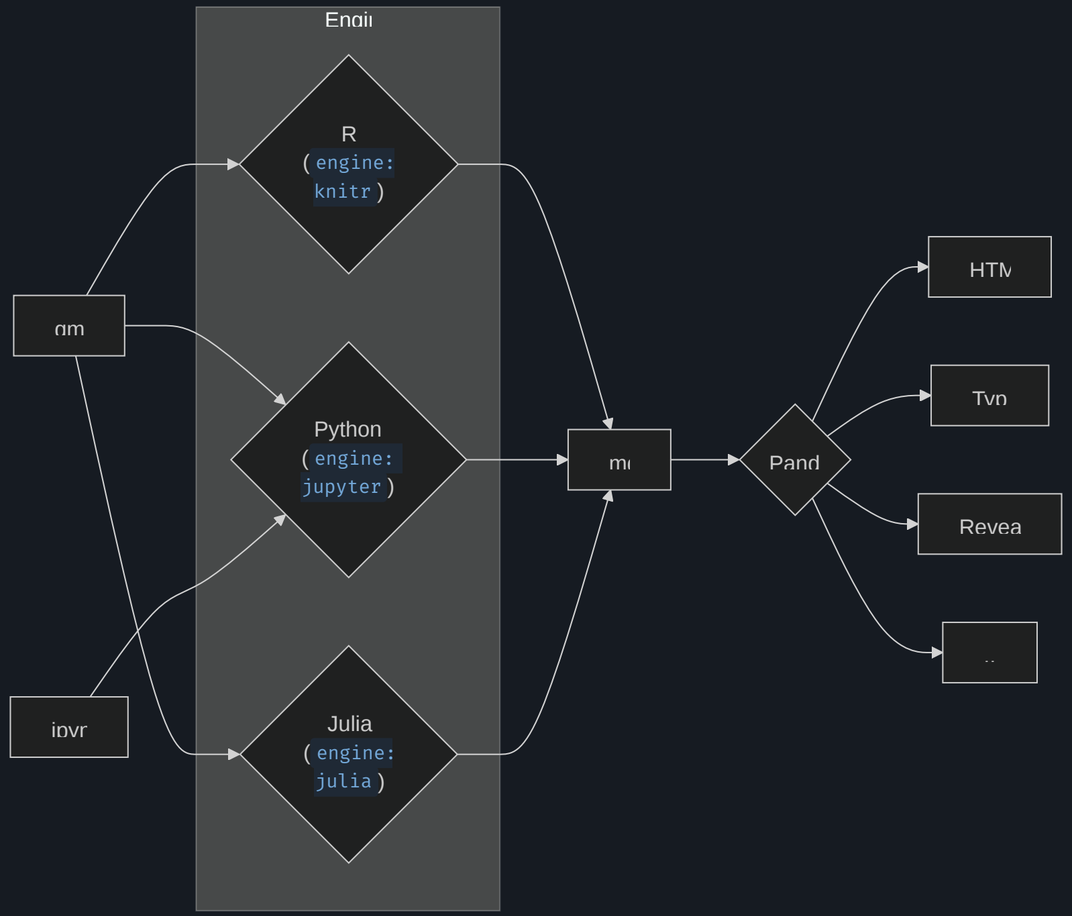
PDF

→ via `pdflatex`, `lualatex`, `xelatex`, `latexmk`, `tectonic`, `wkhtmltopdf`, `weasyprint`, `prince`, `pagedjs-cli`, `context`, or `pdfroff`.

Pandoc (pandoc.org):

- is a **free software**, released under the GPL by John MacFarlane.
- is a **command line interface** (CLI).
- allows to **convert to and between multiple formats**.
- **understands Markdown** syntax.
- **understands LaTeX math** and macros.
- handles **citations and bibliographies**.

How does Quarto work? 🤔



A Guide to Quarto Versions

Quarto follows a versioning scheme similar to Semantic Versioning with three release types: Pre-release, Release Candidate (RC), and Release.

> **Tip**
>
> Always use the latest stable release for production work. See the Quarto Changelog for the full release history.

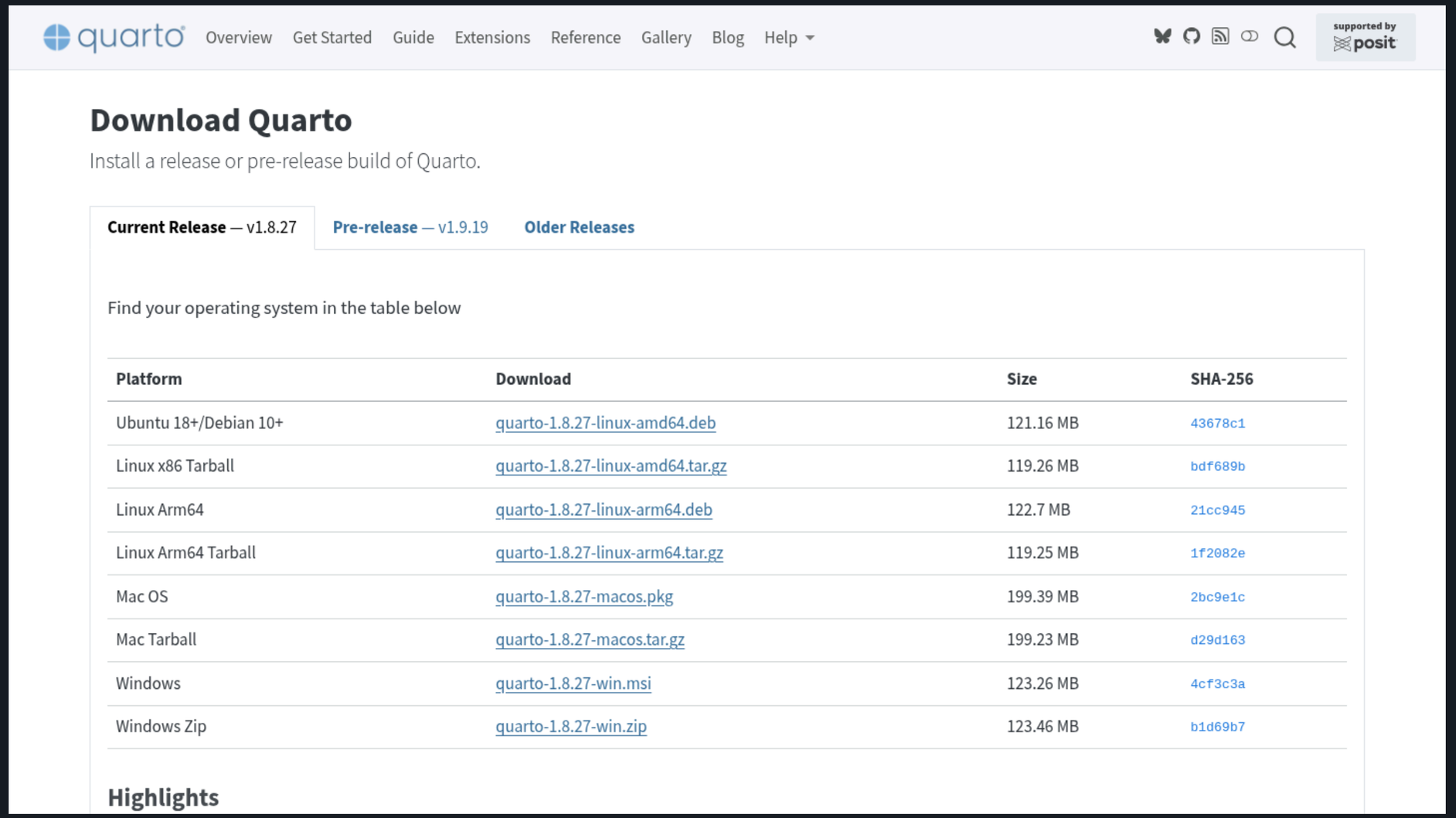
 |

- Downloading and installing Quarto
- Using GitHub Codespaces
- Quarto: a command line interface
- Checking Quarto is installed

Getting started

 |

Downloading and installing Quarto



quarto Overview Get Started Guide Extensions Reference Gallery Blog Help

supported by posit

Download Quarto

Install a release or pre-release build of Quarto.

Current Release — v1.8.27 | **Pre-release** — v1.9.19 | **Older Releases**

Find your operating system in the table below

Platform	Download	Size	SHA-256
Ubuntu 18+/Debian 10+	quarto-1.8.27-linux-amd64.deb	121.16 MB	43678c1
Linux x86 Tarball	quarto-1.8.27-linux-amd64.tar.gz	119.26 MB	bdf689b
Linux Arm64	quarto-1.8.27-linux-arm64.deb	122.7 MB	21cc945
Linux Arm64 Tarball	quarto-1.8.27-linux-arm64.tar.gz	119.25 MB	1f2082e
Mac OS	quarto-1.8.27-macos.pkg	199.39 MB	2bc9e1c
Mac Tarball	quarto-1.8.27-macos.tar.gz	199.23 MB	d29d163
Windows	quarto-1.8.27-win.msi	123.26 MB	4cf3c3a
Windows Zip	quarto-1.8.27-win.zip	123.46 MB	b1d69b7

Highlights

https://quarto.org/

Downloading and installing Quarto

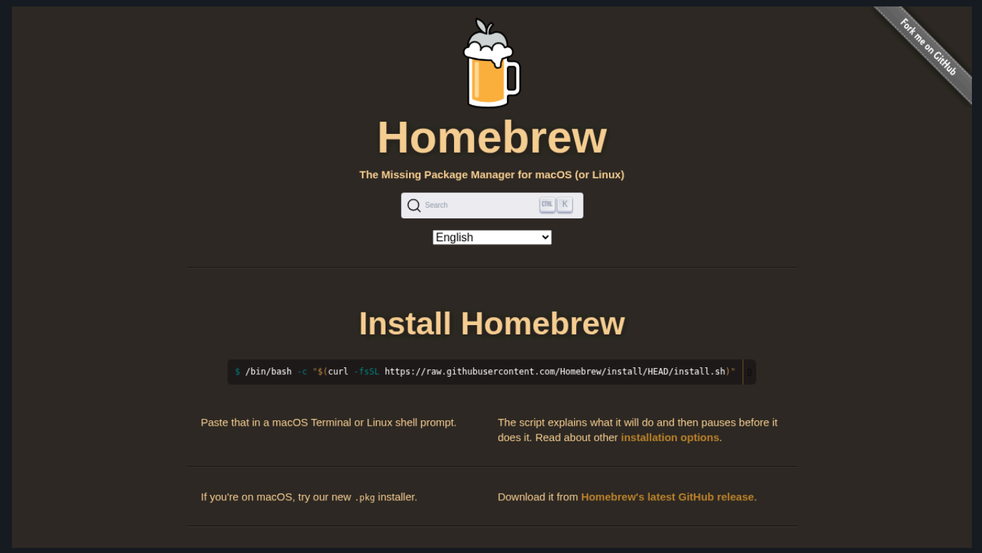
Additionally, you can download and install Quarto using:

Bash

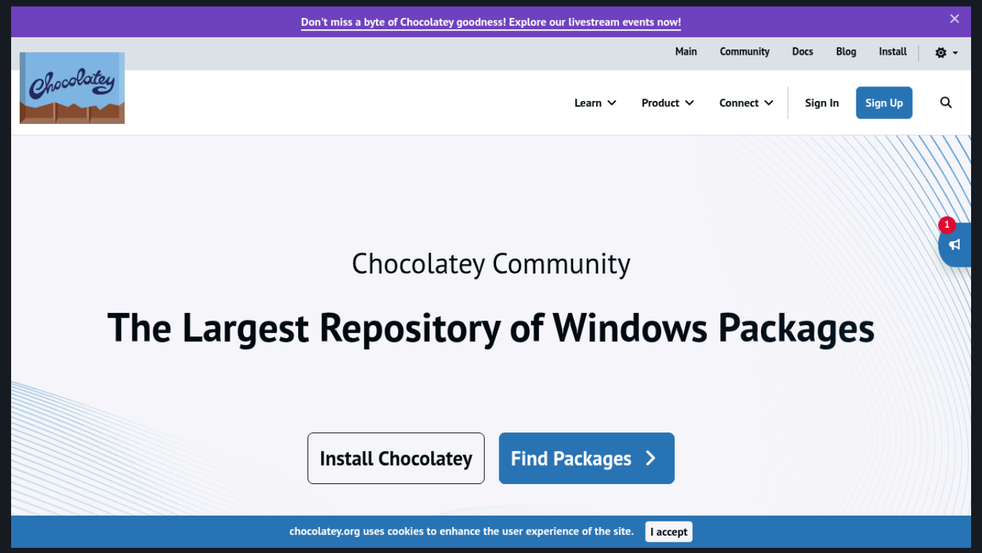
```bash
brew install quarto
```

PowerShell

```powershell
choco install quarto
```



https://brew.sh/



https://community.chocolatey.org/

Using GitHub Codespaces

A pre-configured development environment is available via GitHub Codespaces with all required tools pre-installed: Quarto, R, Python, Julia, TinyTeX, and supporting utilities.

Quarto: a command line interface

BASH

```bash
quarto --help
```

```
Usage:   quarto
Version: 1.9.18

Description:

  Quarto CLI

Options:

  -h, --help     - Show this help.
  -V, --version  - Show the version number for this program.

Commands:

  render     [input] [args ... ]     - Render files or projects to various document types.
  preview    [file] [args ... ]      - Render and preview a document or website project.
  serve      [input]                 - Serve a Shiny interactive document.
  create     [type] [commands ... ]  - Create a Quarto project or extension
  use        <type> [target]         - Automate document or project setup tasks.
  add        <extension>             - Add an extension to this folder or project
  update     [target ... ]           - Updates an extension or global dependency.
  remove     [target ... ]           - Removes an extension.
  convert    <input>                 - Convert documents to alternate representations.
  pandoc     [args ... ]             - Run the version of Pandoc embedded within Quarto.
  typst      [args ... ]             - Run the version of Typst embedded within Quarto.
  run        [script] [args ... ]    - Run a TypeScript, R, Python, or Lua script.
  list       <type>                  - Lists an extension or global dependency.
  install    [target ... ]           - Installs a global dependency (TinyTex or Chromium).
  uninstall  [tool]                  - Removes an extension.
  tools                              - Display the status of Quarto installed dependencies
  publish    [provider] [path]       - Publish a document or project to a provider.
  check      [target]                - Verify correct functioning of Quarto installation.
  call                               - Access functions of Quarto subsystems such as its rendering engines.
  help       [command]               - Show this help or the help of a sub-command.
```

Checking Quarto is installed

BASH

```bash
quarto check
```

SH

```
Quarto 1.8.24
[v] Checking environment information ...
      Quarto cache location: /home/vscode/.cache/quarto
[v] Checking versions of quarto binary dependencies ...
      Pandoc version 3.6.3: OK
      Dart Sass version 1.87.0: OK
      Deno version 2.3.1: OK
      Typst version 0.13.0: OK
[v] Checking versions of quarto dependencies......OK
[v] Checking Quarto installation......OK
      Version: 1.8.24
      Path: /opt/quarto/bin

[v] Checking tools.....................OK
      TinyTeX: (external install)
      Chromium: (not installed)

[v] Checking LaTeX.....................OK
      Using: TinyTex
      Path: /home/vscode/.TinyTeX/bin/x86_64-linux
      Version: 2025

```

Checking Quarto is installed

BASH

```bash
quarto check
```

SH

```
Quarto 1.8.24
[√] Checking environment information ...
      Quarto cache location: /home/vscode/.cache/quarto
[√] Checking versions of quarto binary dependencies ...
      Pandoc version 3.6.3: OK
      Dart Sass version 1.87.0: OK
      Deno version 2.3.1: OK
      Typst version 0.13.0: OK
[√] Checking versions of quarto dependencies......OK
[√] Checking Quarto installation......OK
      Version: 1.8.24
      Path: /opt/quarto/bin

[√] Checking tools.....................OK
      TinyTeX: (external install)
      Chromium: (not installed)

[√] Checking LaTeX.....................OK
      Using: TinyTex
      Path: /home/vscode/.TinyTeX/bin/x86_64-linux
      Version: 2025

```

1 Quarto version and depende

Checking Quarto is installed

BASH

```bash
quarto check
```

SH

```
[v] Checking versions of quarto binary dependencies...
      Pandoc version 3.6.3: OK
      Dart Sass version 1.87.0: OK
      Deno version 2.3.1: OK
      Typst version 0.13.0: OK
[v] Checking versions of quarto dependencies......OK
[v] Checking Quarto installation......OK
      Version: 1.8.24
      Path: /opt/quarto/bin

[v] Checking tools.....................OK
      TinyTeX: (external install)
      Chromium: (not installed)

[v] Checking LaTeX.....................OK
      Using: TinyTex
      Path: /home/vscode/.TinyTeX/bin/x86_64-linux
      Version: 2025

[v] Checking Chrome Headless.....................OK
      Using: Chrome found on system
      Path: /usr/bin/google-chrome
      Source: PATH
```

Tools installed by Quarto (*e.g.*, `quarto install`).

2

3

4

Checking Quarto is installed

BASH

```bash
quarto check
```

SH

```
[✓] Checking versions of quarto dependencies......OK
[✓] Checking Quarto installation......OK
      Version: 1.8.24
      Path: /opt/quarto/bin

[✓] Checking tools.....................OK
      TinyTeX: (external install)
      Chromium: (not installed)

[✓] Checking LaTeX.....................OK
      Using: TinyTex
      Path: /home/vscode/.TinyTeX/bin/x86_64-linux
      Version: 2025

[✓] Checking Chrome Headless.....................OK
      Using: Chrome found on system
      Path: /usr/bin/google-chrome
      Source: PATH

[✓] Checking basic markdown render....OK

[✓] Checking Python 3 installation....OK
      Version: 3.13.7
```

2

LaTeX installation detected by Quarto.

3

4

5

Checking Quarto is installed

BASH

```bash
quarto check
```

SH

```
[v] Checking tools.....................OK
      TinyTeX: (external install)
      Chromium: (not installed)

[v] Checking LaTeX.....................OK
      Using: TinyTex
      Path: /home/vscode/.TinyTeX/bin/x86_64-linux
      Version: 2025

[v] Checking Chrome Headless...................OK
      Using: Chrome found on system
      Path: /usr/bin/google-chrome
      Source: PATH

[v] Checking basic markdown render....OK

[v] Checking Python 3 installation....OK
      Version: 3.13.7
      Path: /usr/local/python/current/bin/python3
      Jupyter: 5.8.1
      Kernels: julia-1.11, python3

[v] Checking Jupyter engine render     OK
```

2

3

Chromium-based browser installation detected by Quarto.

4

5

Checking Quarto is installed

BASH

```bash
quarto check
```

SH

```
      Version: 2025
22
23 [v] Checking Chrome Headless.....................OK
24       Using: Chrome found on system
25       Path: /usr/bin/google-chrome
26       Source: PATH
27
28 [v] Checking basic markdown render....OK
29
30 [v] Checking Python 3 installation....OK
31       Version: 3.13.7
32       Path: /usr/local/python/current/bin/python3
33       Jupyter: 5.8.1
34       Kernels: julia-1.11, python3
35
36 [v] Checking Jupyter engine render....OK
37
38 [v] Checking R installation...........OK
39       Version: 4.5.1
40       Path: /opt/R/4.5.1/lib/R
41       LibPaths:
42         - /home/vscode/R/x86_64-pc-linux-gnu-library/4.5
43         - /opt/R/4.5.1/lib/R/library
```

4

Python installation and computing dependencies detected by Quarto.

5

6

Checking Quarto is installed

BASH

```bash
quarto check
```

SH

```
      Source: PATH

[v] Checking basic markdown render....OK

[v] Checking Python 3 installation....OK
      Version: 3.13.7
      Path: /usr/local/python/current/bin/python3
      Jupyter: 5.8.1
      Kernels: julia-1.11, python3

[v] Checking Jupyter engine render....OK

[v] Checking R installation...........OK
      Version: 4.5.1
      Path: /opt/R/4.5.1/lib/R
      LibPaths:
        - /home/vscode/R/x86_64-pc-linux-gnu-library/4.5
        - /opt/R/4.5.1/lib/R/library
      knitr: 1.50
      rmarkdown: 2.29

[v] Checking Knitr engine render......OK
```

5

R installation and computing dependencies detected by Quarto.

6

- Creating a new Quarto project
- Quarto Projects: default
- Quarto Projects: website
- Quarto Projects: blog
- Quarto Projects: book
- Quarto Projects: manuscript
- Writing with your favourite editor
- Writing using the Visual Editor

Quarto Projects

 |

Creating a new Quarto project

```bash
quarto create project
```

 |

Creating a new Quarto project

Bash

```bash
quarto create project
```

BASH

```
? Type
❯ default
  website
  blog
  manuscript
  book
  confluence
```

 |

Quarto Projects: `default`

Bash

```
quarto create project default Default
```

```
_demo/Default
├── Default.qmd
`-- _quarto.yml

0 directories, 2 files
```

 |

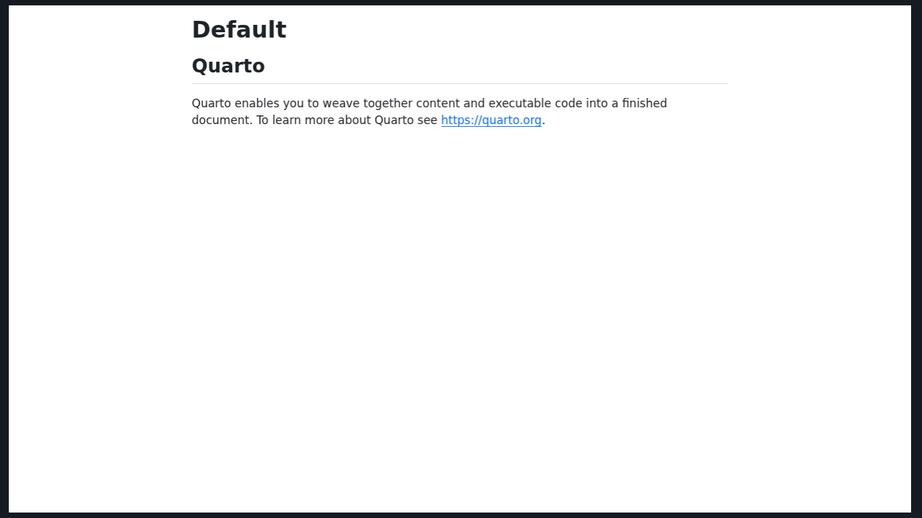
Quarto Projects: `default`

Bash

```
quarto create project default Default
```

```
_demo/Default
├── Default.qmd
`-- _quarto.yml

0 directories, 2 files
```



Default

Quarto

Quarto enables you to weave together content and executable code into a finished document. To learn more about Quarto see https://quarto.org.

Quarto Projects: `default`

 |

Quarto Projects: `default`

- `_quarto.yml` : project configuration file.

Quarto

```yaml
project:
  title: "Default"
```

 |

Quarto Projects: `default`

- `_quarto.yml` : project configuration file.

Quarto

```yaml
project:
  title: "Default"
```

- `Default.qmd` : default Quarto document.

Quarto

```yaml
---
title: "Default"
---

## Quarto

Quarto enables you to weave together content and executable code into a finished doc
```

 |

Quarto Projects: `website`

Bash

```bash
quarto create project website Website
```

```
_demo/Website
├── _quarto.yml
├── about.qmd
├── index.qmd
`-- styles.css

0 directories, 4 files
```

 |

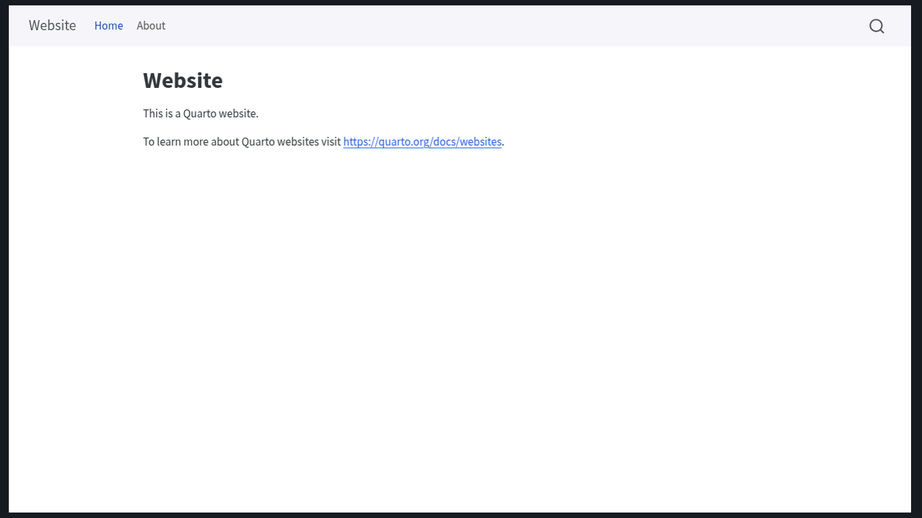
Quarto Projects: `website`

Bash

```
quarto create project website Website
```

```
_demo/Website
├── _quarto.yml
├── about.qmd
├── index.qmd
`-- styles.css

0 directories, 4 files
```



Quarto Projects: `blog`

Bash

```
quarto create project blog Blog
```

```
_demo/Blog
├── _quarto.yml
├── about.qmd
├── index.qmd
├── posts
│   ├── _metadata.yml
│   ├── post-with-code
│   │   ├── image.jpg
│   │   └── index.qmd
│   └── welcome
│       ├── index.qmd
│       └── thumbnail.jpg
├── profile.jpg
└── styles.css

3 directories, 10 files
```

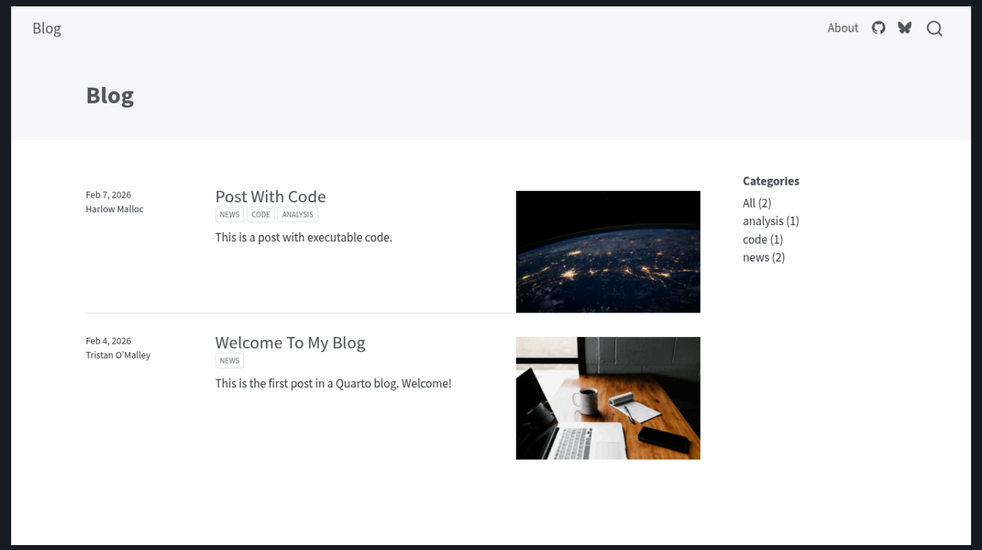
Quarto Projects: `blog`

Bash

```
quarto create project blog Blog
```

```
_demo/Blog
├── _quarto.yml
├── about.qmd
├── index.qmd
├── posts
│   ├── _metadata.yml
│   ├── post-with-code
│   │   ├── image.jpg
│   │   └── index.qmd
│   └── welcome
│       ├── index.qmd
│       └── thumbnail.jpg
├── profile.jpg
└── styles.css

3 directories, 10 files
```



Quarto Projects: book

Bash

```bash
quarto create project book Book
```

```
_demo/Book
├── _quarto.yml
├── cover.png
├── index.qmd
├── intro.qmd
├── references.bib
├── references.qmd
`-- summary.qmd

0 directories, 7 files
```

 |

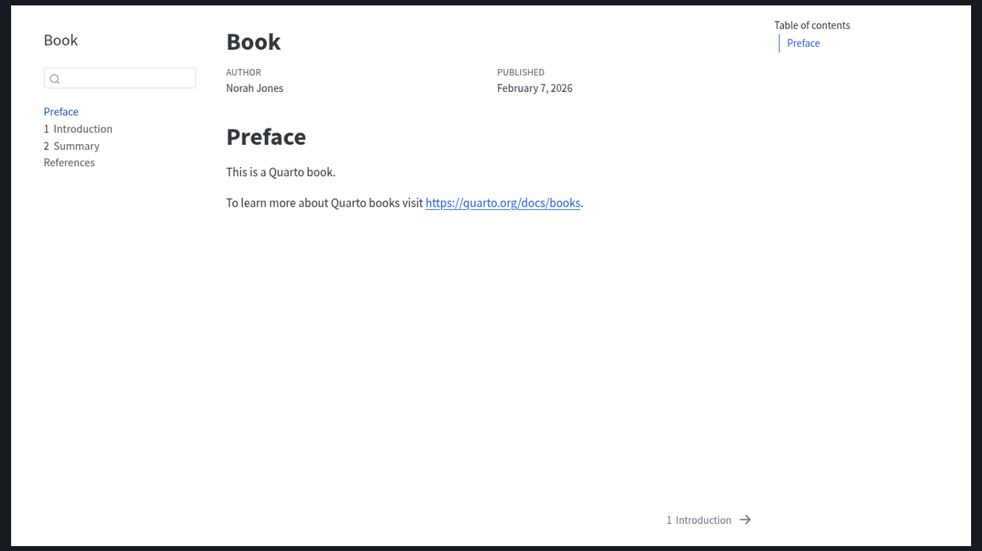
Quarto Projects: book

Bash

```
quarto create project book Book
```

```
_demo/Book
├── _quarto.yml
├── cover.png
├── index.qmd
├── intro.qmd
├── references.bib
├── references.qmd
`-- summary.qmd

0 directories, 7 files
```



Quarto Projects: `manuscript`

Bash

```bash
quarto create project manuscript Manuscript
```

```
_demo/Manuscript
├── _quarto.yml
├── index.qmd
`-- references.bib

0 directories, 3 files
```

 |

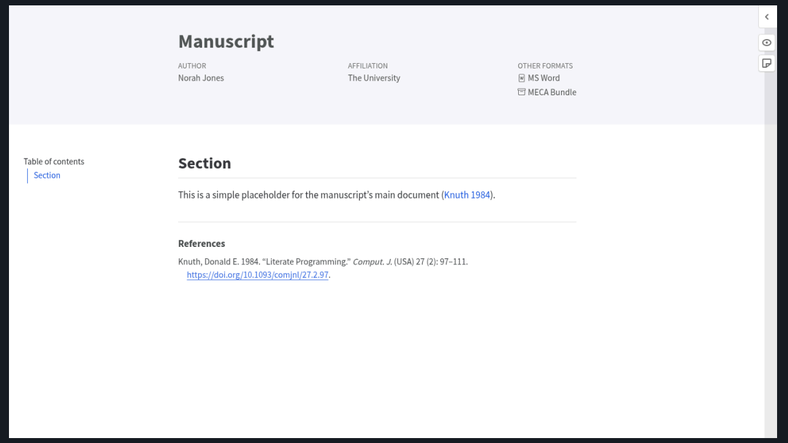
Quarto Projects: `manuscript`

Bash

```
quarto create project manuscript Manuscript
```

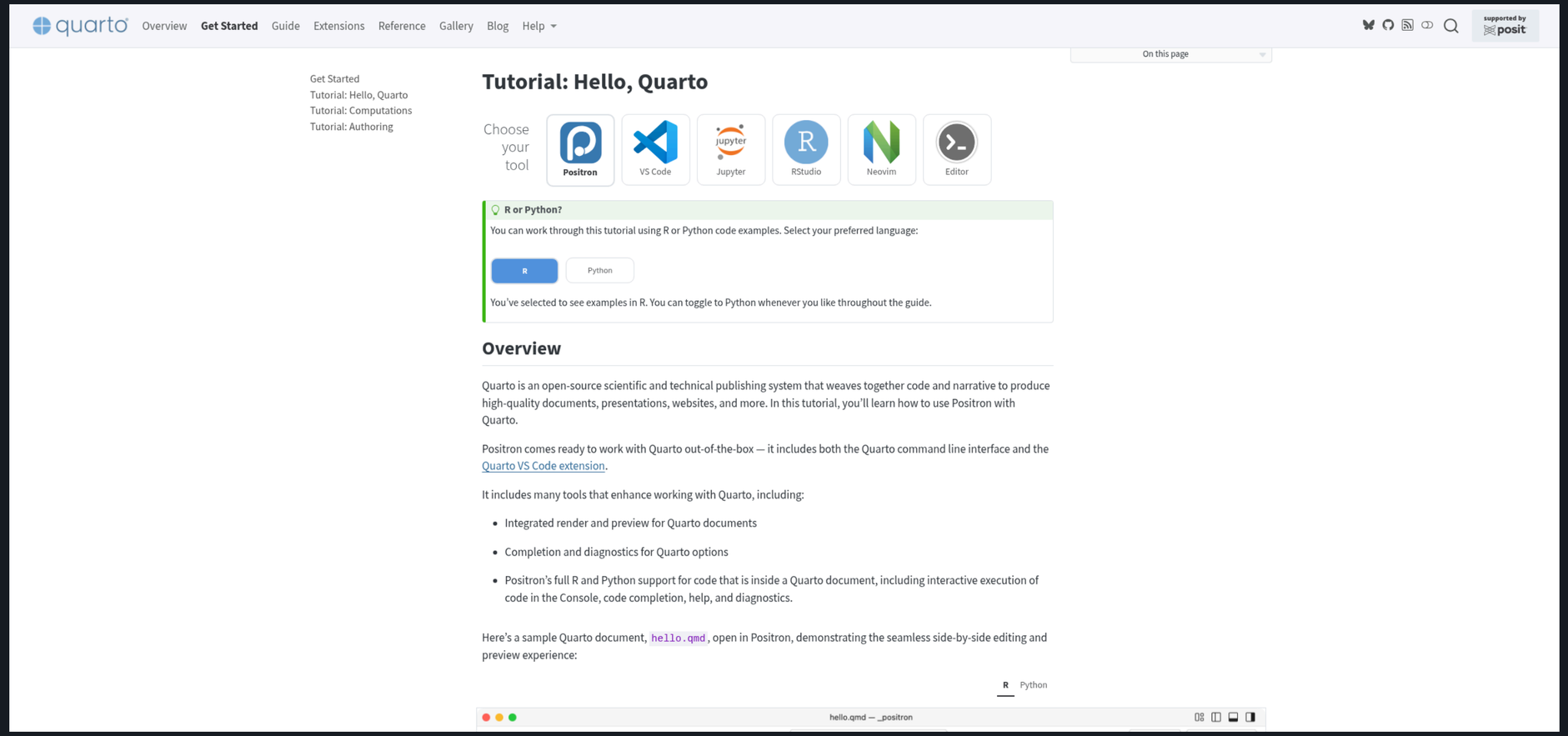
```
_demo/Manuscript
├── _quarto.yml
├── index.qmd
`-- references.bib

0 directories, 3 files
```



Writing with your favourite editor

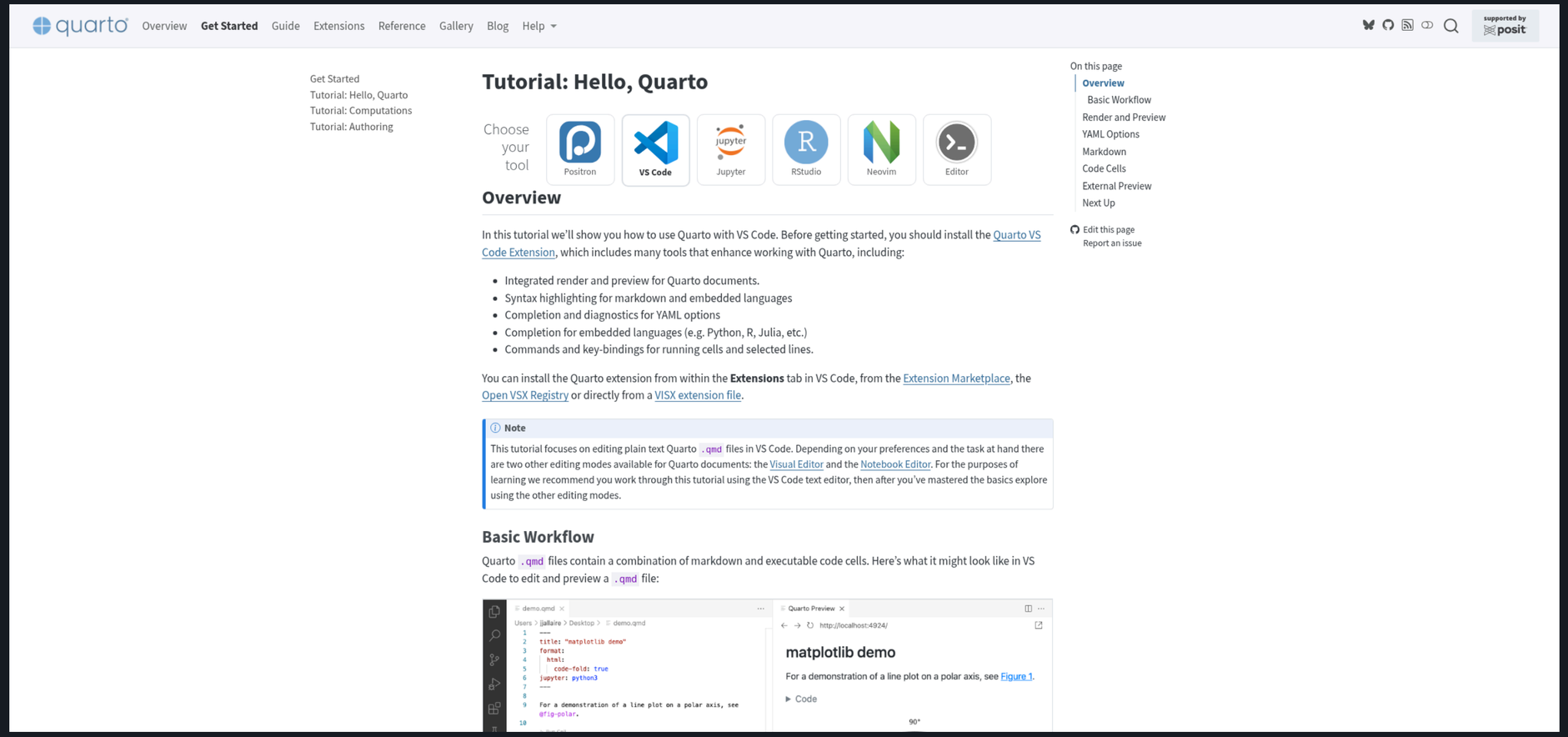
POSITRON | VSCODE | JUPYTER | RSTUDIO | NEOVIM | TEXT-EDITOR



https://quarto.org/

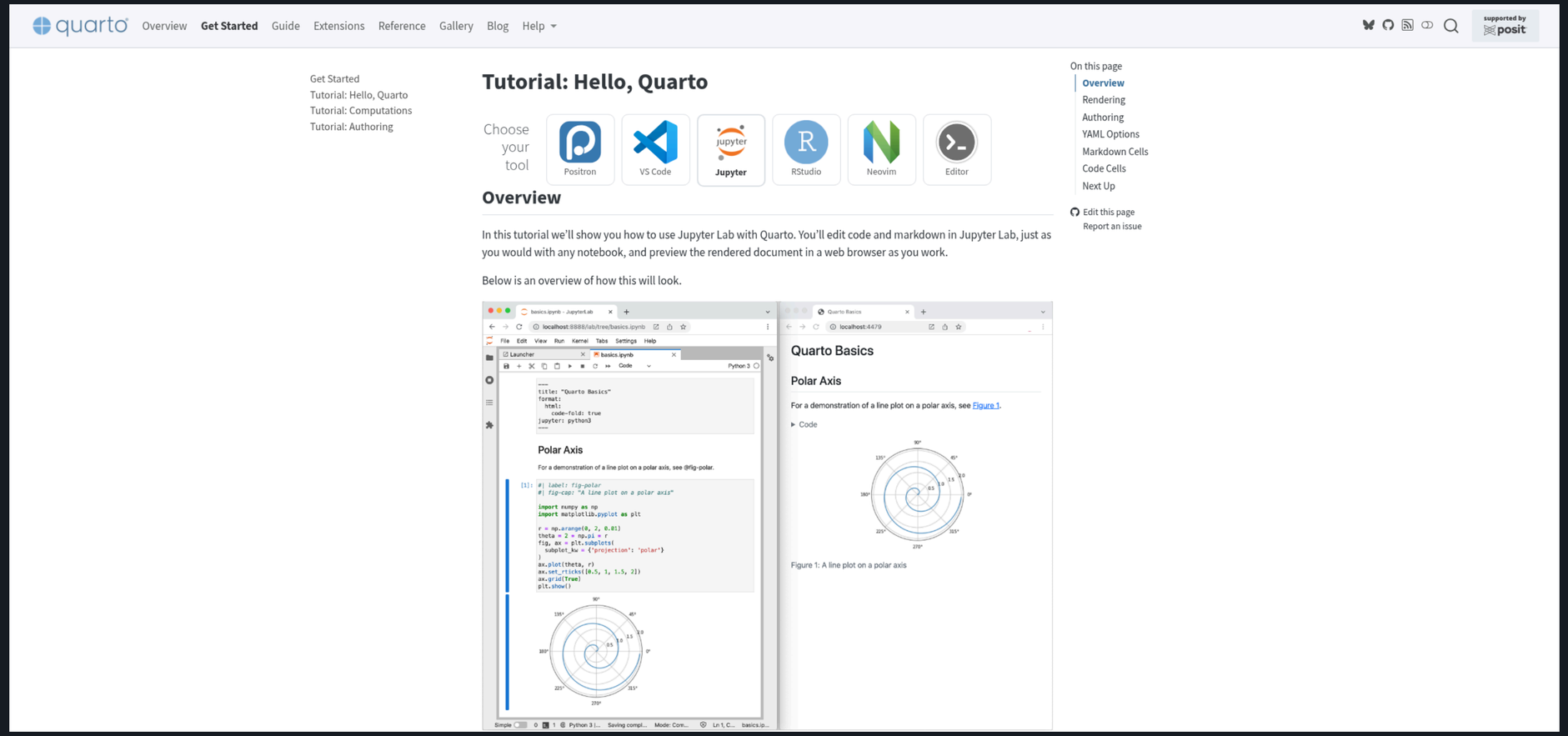
Writing with your favourite editor

POSITRON | VSCODE | JUPYTER | RSTUDIO | NEOVIM | TEXT-EDITOR



Writing with your favourite editor

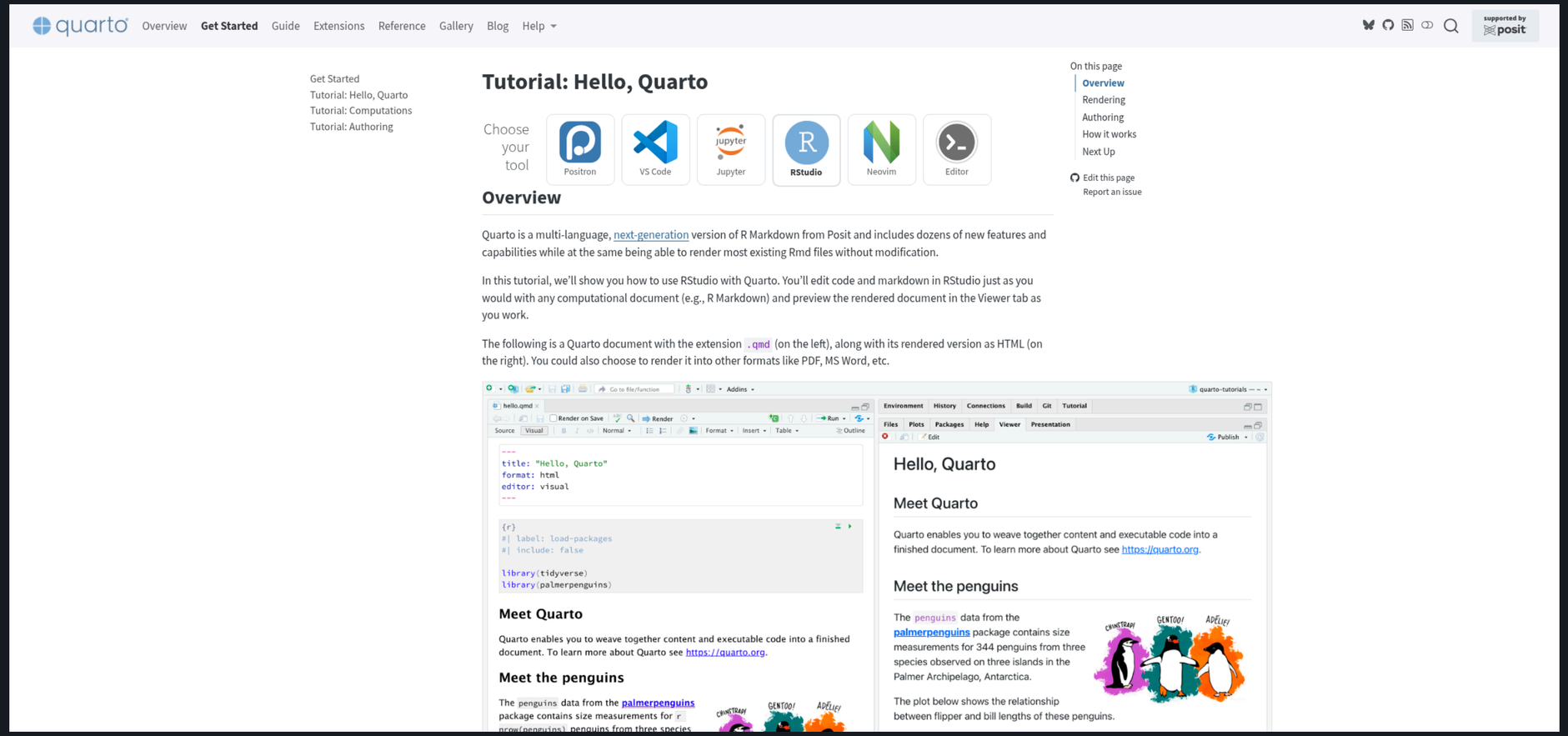
POSITRON VSCODE **JUPYTER** RSTUDIO NEOVIM TEXT-EDITOR



https://quarto.org/

Writing with your favourite editor

POSITRON VSCODE JUPYTER **RSTUDIO** NEOVIM TEXT-EDITOR



https://quarto.org/

Writing with your favourite editor

POSITRON VSCODE JUPYTER RSTUDIO **NEOVIM** TEXT-EDITOR

quarto Overview **Get Started** Guide Extensions Reference Gallery Blog Help

supported by posit

Get Started
Tutorial: Hello, Quarto
Tutorial: Computations
Tutorial: Authoring

Tutorial: Hello, Quarto

Choose your tool: Positron | VS Code | Jupyter | RStudio | **Neovim** | Editor

On this page
- **Overview**
  - Basic Workflow
- Render and Preview
- YAML Options
- Code Cells
- Next Up
- Next Up

Edit this page
Report an issue

Overview

In this tutorial we'll show you how to use Quarto with Neovim. While we cover the basics here, you will also want to review the article on Using Neovim with Quarto to learn more about installing, using, and customizing Neovim for Quarto.

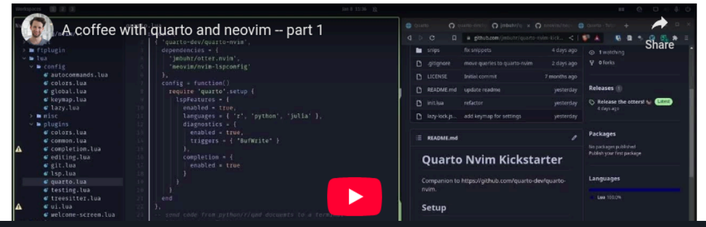
If you already have Neovim configured to your liking, you may only want to add the quarto-nvim plugin and only refer to this guide for inspiration and seeing the possibilities. But if you are entirely new to Neovim or want to simply try out a configuration already set up for data science with Quarto, you should head over to this kickstarter configuration. This is also what we will be using for this tutorial.

Note

Neovim is a highly customizable editor. So much so that Neovim core member TJ Devries has recently coined the term Personal Development Environments (PDE)[1] to separate the concept from Integrated Development Environments (IDEs) such as VS Code and RStudio.

Out of the box neovim is fairly minimal. To work efficiently and get all the nice features, you have to configure it. You have to make it your own. If this approach sounds enticing to you, read on. Welcome to the rabbit hole.

You can also watch this video for a quick guide to getting started with the kickstarter configuration alongside this write-up.

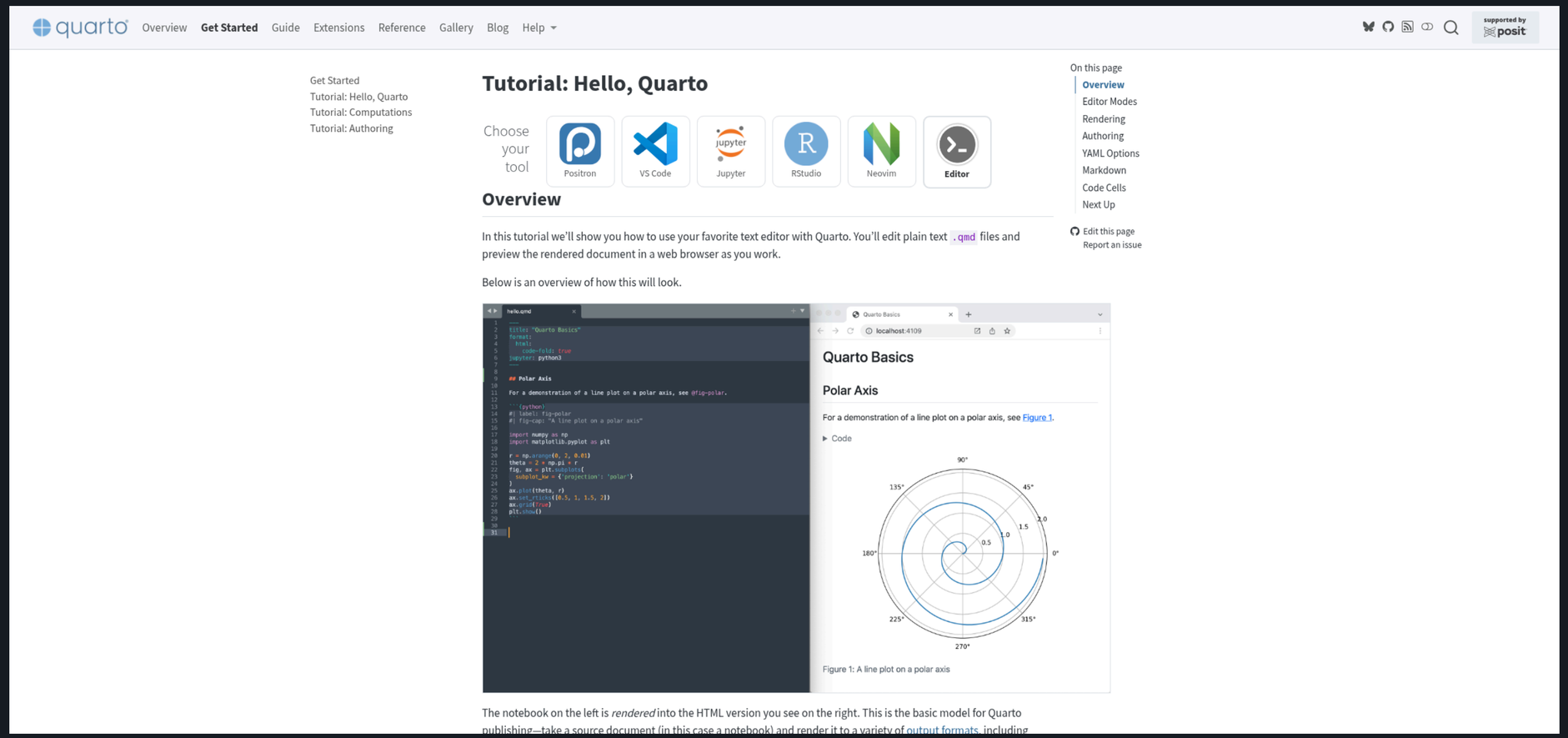


https://quarto.org/

Mickaël CANOUIL

Writing with your favourite editor

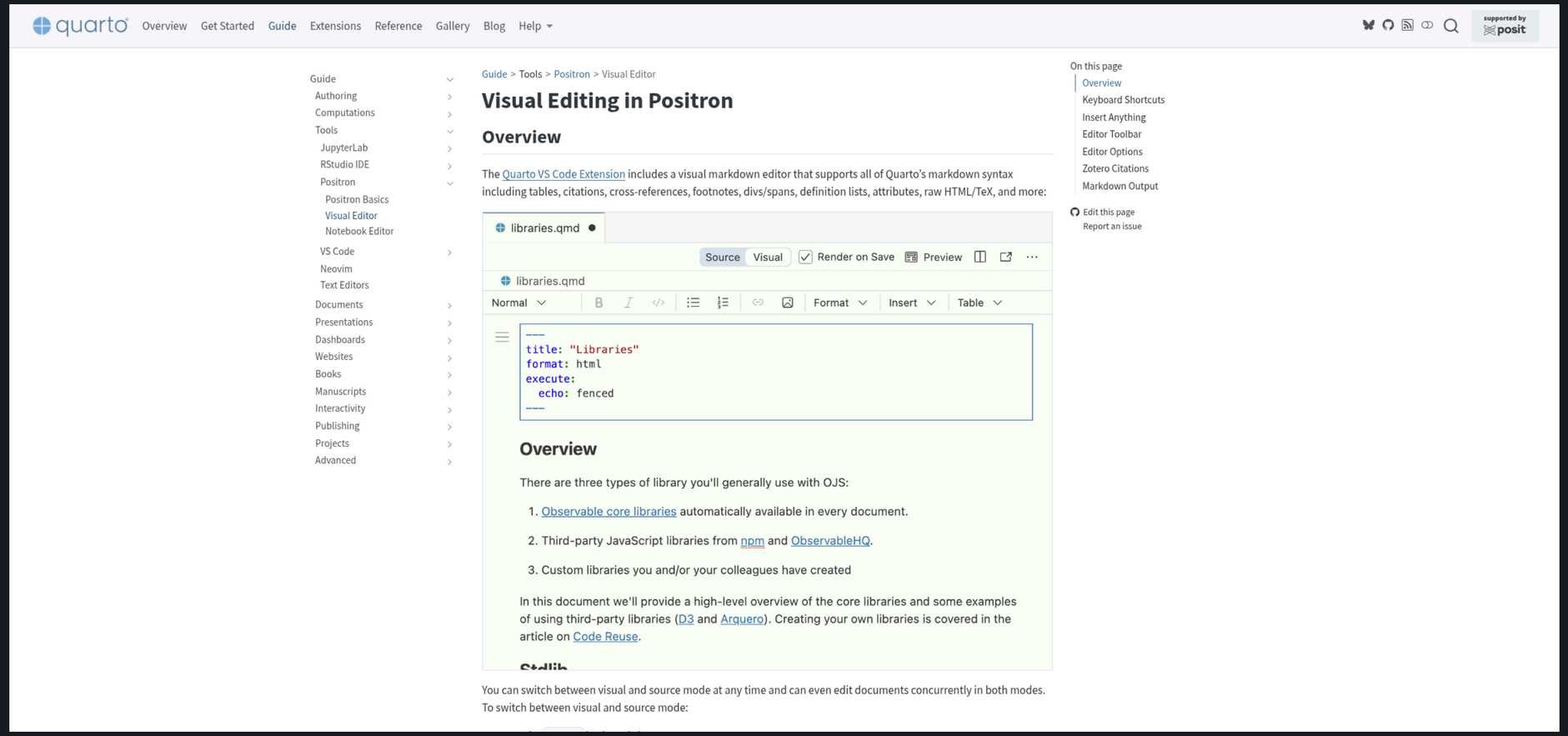
POSITRON VSCODE JUPYTER RSTUDIO NEOVIM TEXT-EDITOR



https://quarto.org/

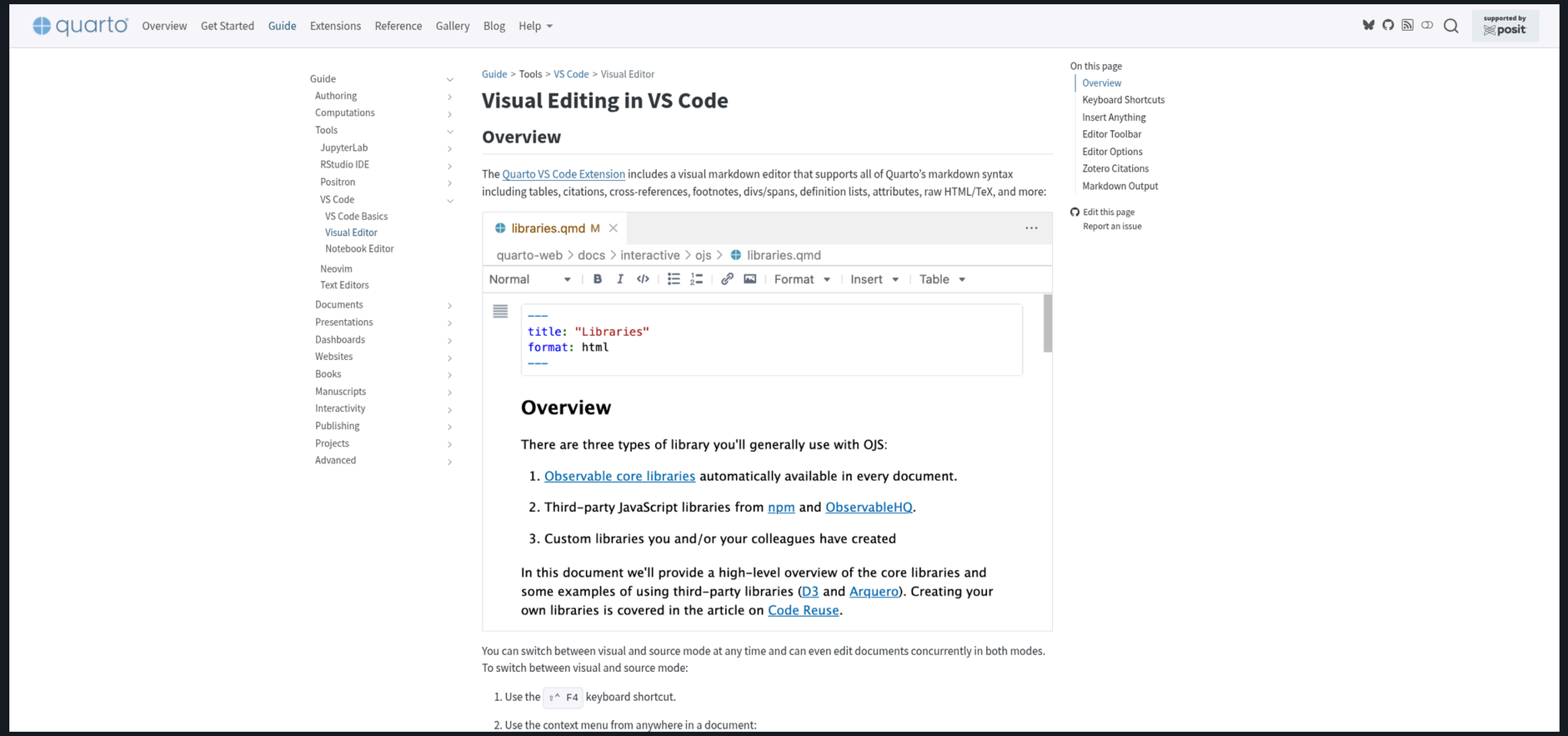
Writing using the Visual Editor

POSITRON | VSCODE | RSTUDIO



Writing using the Visual Editor

POSITRON | **VSCODE** | RSTUDIO



Mickaël CANOUIL

Writing using the Visual Editor

POSITRON VSCODE **RSTUDIO**

quarto Overview Get Started Guide Extensions Reference Gallery Blog Help

Guide
- Authoring
- Computations
- Tools
  - JupyterLab
  - RStudio IDE
    - RStudio Basics
    - Visual Editor
      - Editor Basics
      - Technical Writing
      - Content Editing
      - Shortcuts & Options
      - Markdown Output
  - Positron
  - VS Code
  - Neovim
  - Text Editors
- Documents
- Presentations
- Dashboards
- Websites
- Books
- Manuscripts
- Interactivity
- Publishing
- Projects
- Advanced

Guide > Tools > RStudio IDE > Visual Editor

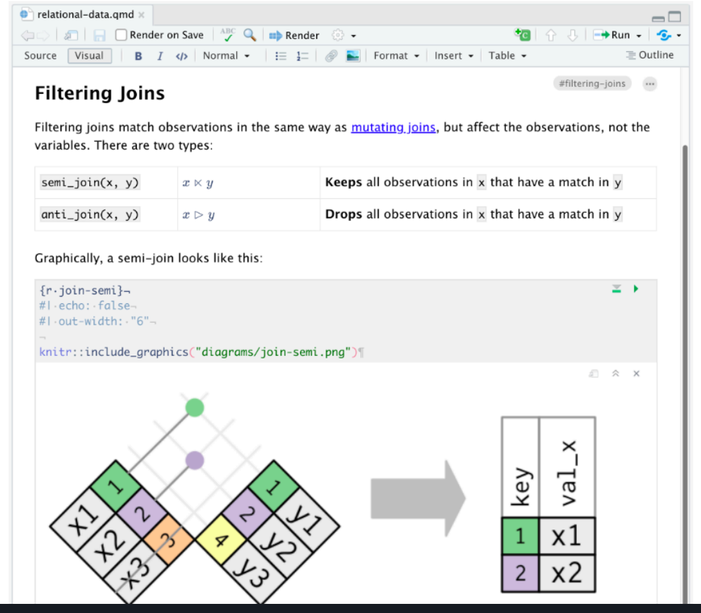
Visual Editing in RStudio

Overview

The Quarto visual editor provides a WYSIWYM editing interface for all of Pandoc markdown, including tables, citations, cross-references, footnotes, divs/spans, definition lists, attributes, raw HTML/TeX, and more. The visual editor also includes support for executing code cells and viewing their output inline:

On this page
- Overview
  - Switching Modes
- Getting Started
- Using the Editor
- Learning More

Edit this page
Report an issue



https://quarto.org/

- Exercise Overview
- Part 1: Project Creation
- Part 2: IDE Features Exploration
- Part 3: Document Customisation
- Part 4: Rendering
- Success Criteria

Hands-On Exercise: Creating Your First Quarto Project

 |

Exercise Overview

Objective: Create a personal Quarto document that demonstrates basic features, project setup skills, and IDE integration

Example Code: Exercises

```bash
tar -xzf "01-exercises.tar.gz" -C "01-introduction-setup"
```

IDE Configuration

Before starting, ensure your development environment is ready:

- **VS Code**: Install the Quarto extension (`Ctrl/Cmd + Shift + X` , search “Quarto”).
- **Positron**: Built-in Quarto support is included (no additional extensions needed).
- Open the Command Palette (`Ctrl/Cmd + Shift + P`) and type “Quarto” to verify available commands.

Part 1: Project Creation

1. **Choose Your Adventure**: Create a project using the CLI or your IDE.
2. **Explore the project structure** in the IDE's file explorer panel.

 |

Part 2: IDE Features Exploration

Discover your IDE's Quarto capabilities:

- **File recognition**: Notice `.qmd` files have distinctive icons and syntax highlighting.
- **Document outline**: Open a `.qmd` file and check the Outline/Structure panel (shows headings and sections).
- **Quarto assist panel**: Look for a Quarto-specific panel that provides contextual help.
- **Command palette power**: `Ctrl/Cmd + Shift + P` → type "Quarto" to see all available commands.
- **Integrated features**: Visual editor mode, preview panels, terminal integration.
- **Smart editing**: Try auto-completion in YAML headers, fenced divs, and code blocks.

Part 3: Document Customisation

Modify the main document (`index.qmd`) to include:

1. **Update the YAML header** with your information:
   - ▶ Code
2. **Add content** to the document body.

Part 4: Rendering

Try different rendering approaches:

1. **Command Line** (classic method):

```bash
quarto render
```

2. **IDE Integration** (modern workflow):
   - Command Palette (`Ctrl/Cmd + Shift + P`) - search for “Quarto Preview” and/or “Quarto Render” commands.
   - Use keyboard shortcut if you configured one
   - Notice IDE-specific feedback and error handling

 |

Success Criteria

You've successfully completed the exercise if you can:

- Configure your IDE for optimal Quarto development.
- Create a new Quarto project using the CLI or your IDE.
- Navigate project files efficiently in your IDE.
- Utilise IDE features like syntax highlighting and command palette.
- Modify documents with IDE assistance (auto-completion, outline view).
- Render documents using both CLI and IDE methods.

Mickaël CANOUIL, *Ph.D.*

Independent Contractor

pro@mickael.canouil.dev

Website	GitHub	LinkedIn	Bluesky	Mastodon
mickael.canouil.fr	@mcanouil	MickaelCanouil	@mickael.canouil.fr	@MickaelCanouil

 |