

# Styling documents and building extensions with Quarto

*19 December 2023*

Dr Nicola Rennie

 [nrennie.rbind.io](https://nrennie.rbind.io)

 [nrennie](#)

 [nicola-rennie](#)



# About me

Lecturer in Health Data Science at  
[Lancaster Medical School](#).

Academic background in statistics, with  
experience in data science consultancy.

Blog about R and data science at  
[nrennie.rbind.io/blog](http://nrennie.rbind.io/blog).



# What is this talk about?

- How to style Quarto documents with built-in options
- How to style Quarto documents using Quarto extensions
- How to build your own Quarto extension



# Styling your Quarto documents



# Why define a document style?

## **Consistency**

Documents from different authors look the same.

---

## **Professionalism**

Default options usually look like the default options.

---

## **Accessibility**

Information can often be presented in better ways when designed actively.



# Starting with a plain document

Quarto is an open-source scientific and technical publishing system that allows you to combine text, images, code, plots, and tables in a fully-reproducible document. Quarto has support for multiple languages including R, Python, Julia, and Observable. It also works for a range of output formats such as PDFs, HTML documents, websites, presentations, ...

```
1 ---
2 title: "My document"
3 format: html
4 ---
5
6 Content goes here.
```

## My document

Content goes here.



# Built-in themes for HTML output

Quarto includes 25 themes from the [Bootstrap](#) project for HTML output.

See [quarto.org/docs/output-formats/html-themes.html](https://quarto.org/docs/output-formats/html-themes.html) for a complete list.

There are also 11 built-in themes for Revealjs presentations.



# Built-in themes for HTML output

```
1 ---
2 title: "My document"
3 format:
4   html:
5     theme: slate
6 ---
7
8 Content goes here.
```

My document

Content goes here.

# Built-in options for HTML output

```
1 ---
2 title: "My document"
3 format:
4   html:
5     backgroundColor: lightblue
6 ---
7
8 Content goes [here] (https://github.co
```

See

[quarto.org/docs/reference/formats/html](https://quarto.org/docs/reference/formats/html)

for all HTML options.

My document

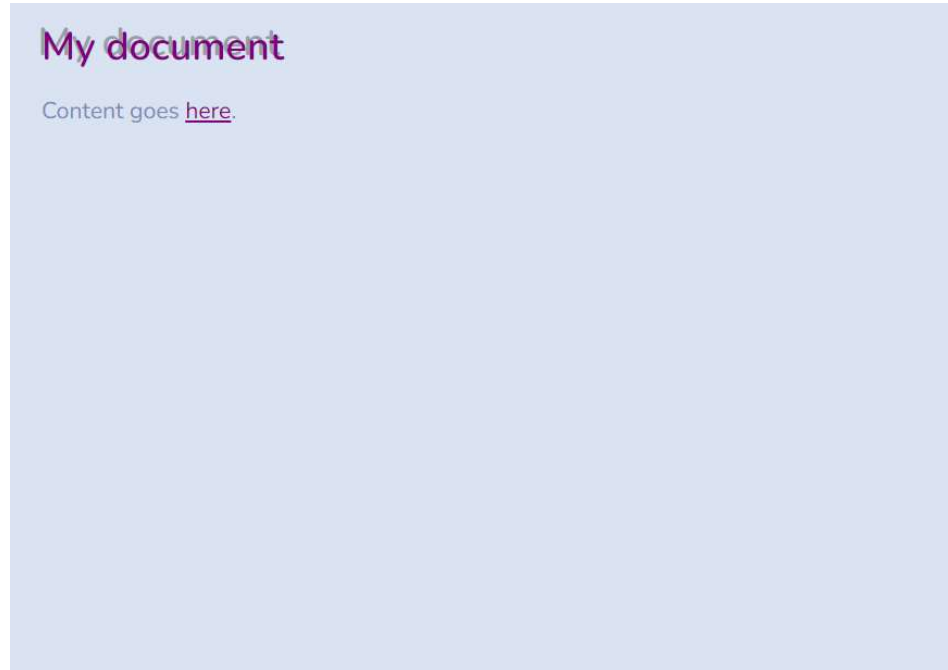
Content goes [here](#).

# Adding your own CSS styling

```
1 ---
2 title: "My document"
3 format:
4   html:
5     theme: [morph, styles.scss]
6 ---
7
8 Content goes [here](https://github.co
```

## styles.scss

```
1 /*-- scss:defaults --*/
2 $link-color: #800080;
3
4 /*-- scss:rules --*/
5 h1, h2, h3, h4, h5, h6 {
6   text-shadow: -3px -3px 0 rgba(0, 0, 0, 0.5);
7 }
```

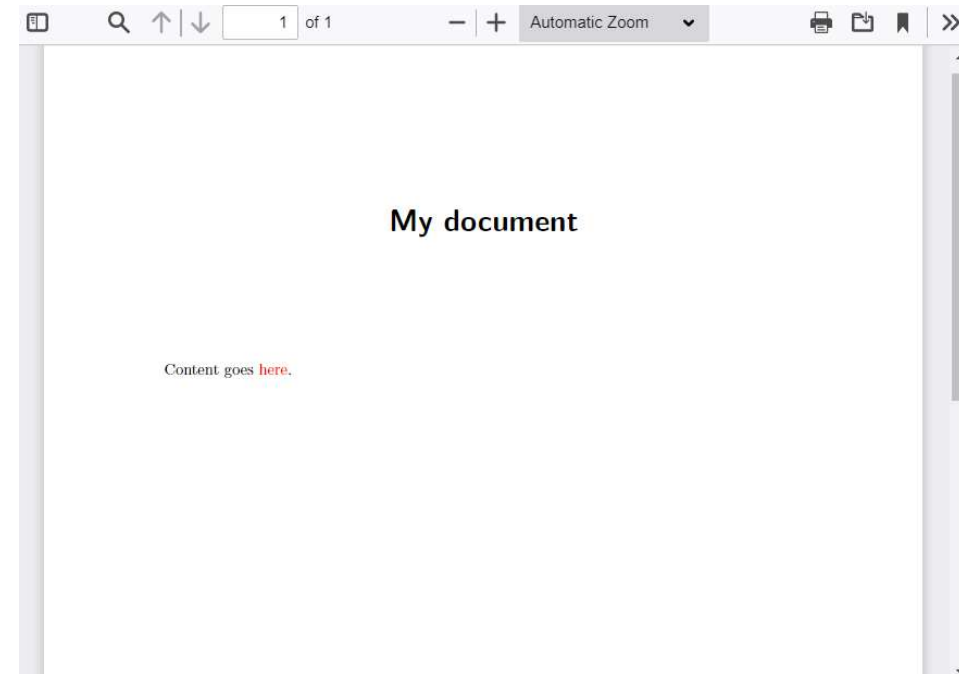


# Built-in options for PDF output

```
1 ---
2 title: "My document"
3 format:
4   pdf:
5     urlcolor: "red"
6 ---
7
8 Content goes [here] (https://github.co
```

See

[quarto.org/docs/reference/formats/pdf](https://quarto.org/docs/reference/formats/pdf) for  
all PDF options.



# Adding your own LaTeX styling

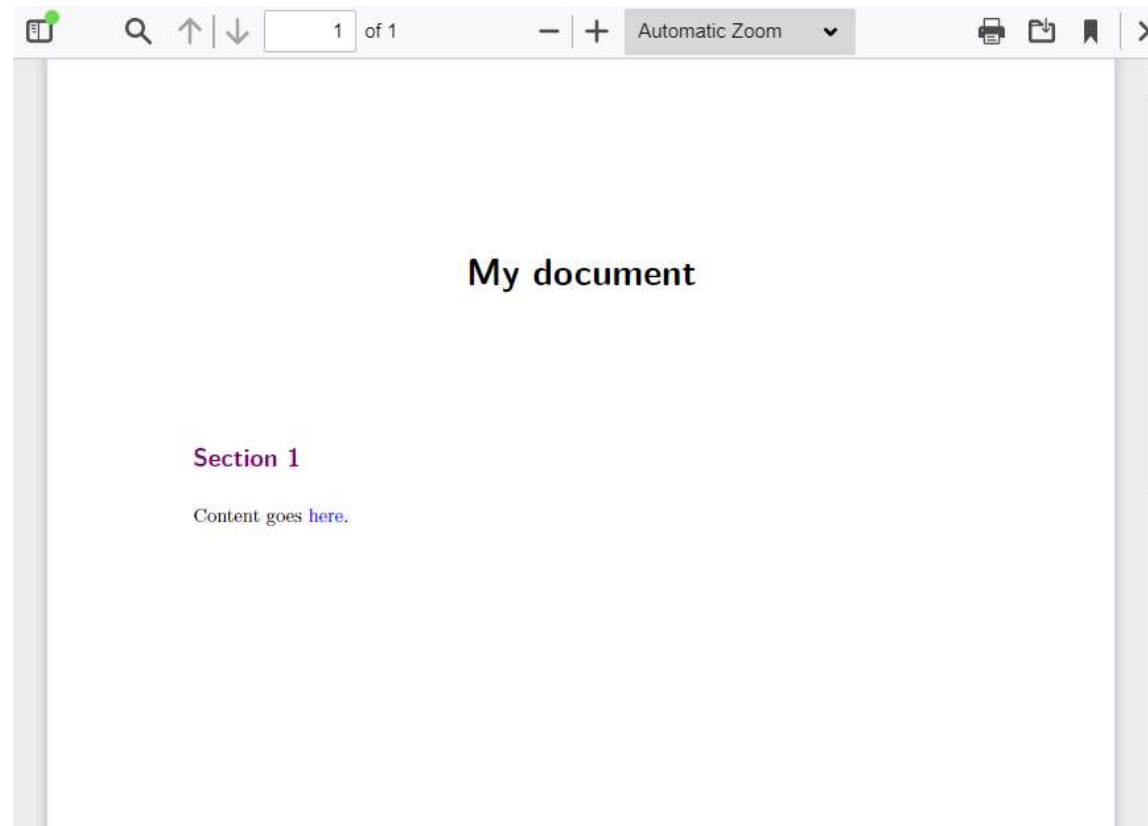
## document.qmd

```
1 ---
2 title: "My document"
3 format:
4   pdf:
5     include-in-header:
6       - "styles.tex"
7 ---
8
9 # Section 1
10
11 Content goes [here] (https://github.com)
```

## styles.tex

```
1 % load packages
2 \usepackage{xcolor}
3 \usepackage{sectsty}
4
5 %% Let's define some colours
6 \definecolor{MyPurple}{HTML}{800080}
7
8 %% Section font colour
9 \sectionfont{\color{MyPurple}}
```

# Adding your own LaTeX styling



In the upcoming Quarto 1.4 release, you can also create PDFs using Typst (no LaTeX!).

# Using Quarto extensions

Extensions are a powerful way to modify and extend the behavior of Quarto. Extensions can be used to add styling to your documents.

Extensions aren't just used to change document styling.

There are also extensions to:

- Embed shinylive applications in a Quarto document
- Embed HTML forms in documents
- Use Font Awesome icons in HTML and PDF outputs
- ...





# Installing Quarto extensions

There are different ways to distribute Quarto extensions. Many are distributed via GitHub.

To install an extension, run the following in the command line:

```
1 quarto install extension username/repository
```

For example, to install my **PrettyPDF** extension:

```
1 quarto install extension nrennie/PrettyPDF
```

# Using Quarto extensions

This creates an `_extensions` folder in your current directory.

You then need to edit the YAML at the top of the Quarto document, to tell it to use the extension:

```
1 ---
2 format: PrettyPDF-pdf
3 ---
```

Options works as normal:

```
1 ---
2 format:
3   PrettyPDF-pdf:
4     keep-tex: true
5 ---
```

# Using Quarto extensions

Need to reformat to a completely different style?

```
1 ---  
2 format: PrettyPDF-pdf  
3 ---
```

Just change the YAML!

```
1 ---  
2 format: PrettierPDF-pdf  
3 ---
```

# Installing Quarto templates

Some Quarto extensions (especially style extensions) come with a template file that changes the YAML for you.

```
1 quarto use template username/repository
```

For example, to use my **PrettyPDF** template:

```
1 quarto use template nrennie/PrettyPDF
```

This creates an **\_extensions** folder in your current directory **and** a **.qmd** file with the correct YAML.

# Where do I find Quarto extensions?

A few suggestions...

[quarto.org/docs/extensions](https://quarto.org/docs/extensions)

[github.com/quarto-journals](https://github.com/quarto-journals)

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



# Building your own Quarto extension



# Why build your own extension?

## **Reduce repetition**

Don't copy and paste commonly used YAML options between documents.

---

## **Share with others**

Everybody in an organisation can create similar outputs without having to adjust the style themselves.

---

## **Version control style**

Changes to designs can be tracked alongside code.



# What are the components of a (style) extension?

The essentials...

- `_extensions` folder
- `_extension.yml` file

The *nice to haves*...

- `template.qmd` file
- Other files e.g. logos or fonts



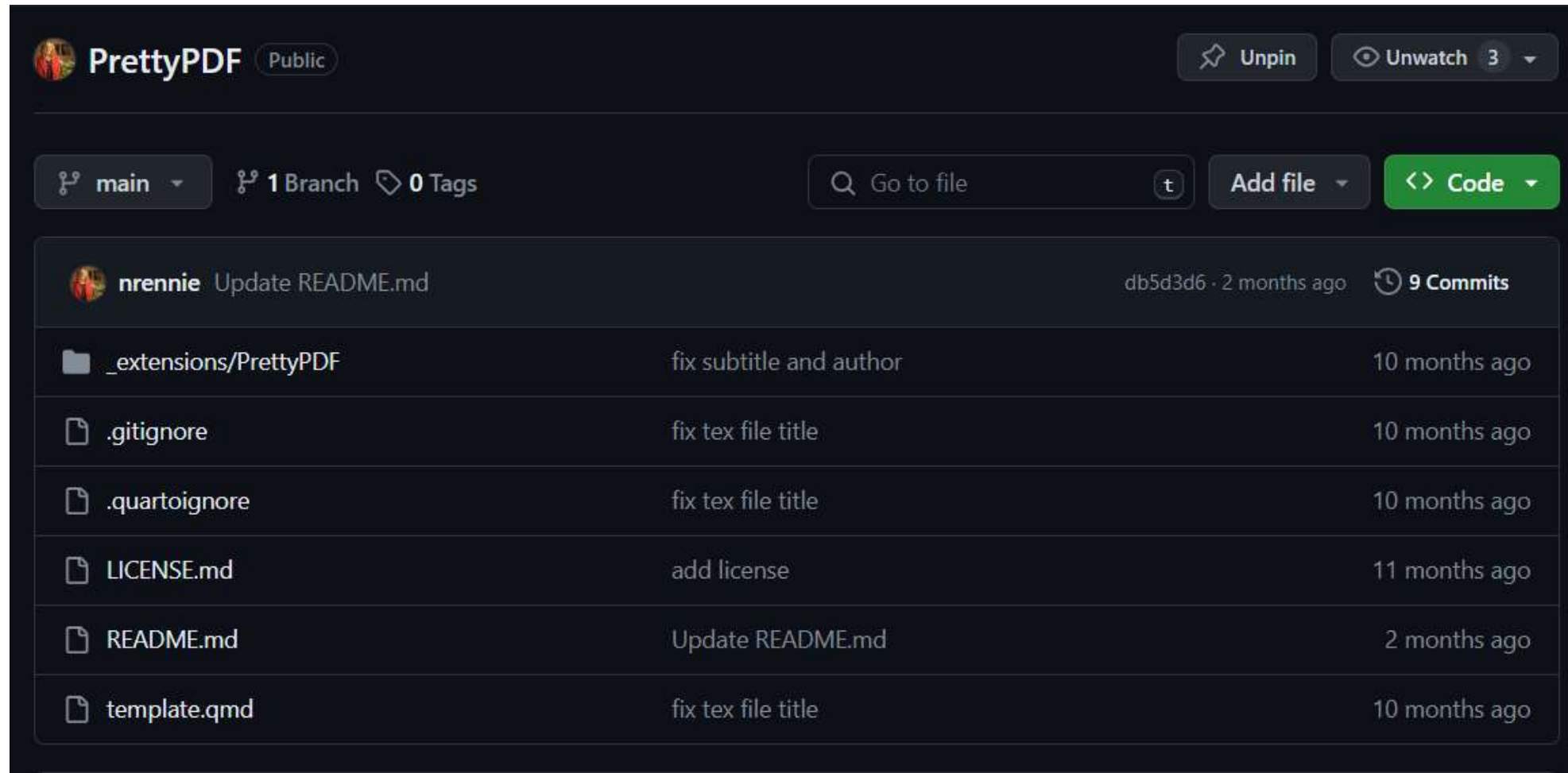


# What are the components of a (style) extension?

The folder structure might look like this:

```
1 |─ template.qmd
2 |─ _extensions/
3   |─ Extension Name/
4     |─ _extension.yml
5     |─ Other files...
```

# What are the components of a (style) extension?



The screenshot shows the GitHub interface for the 'PrettyPDF' repository. At the top, the repository name 'PrettyPDF' is displayed with a 'Public' badge. To the right are buttons for 'Unpin' and 'Unwatch' (with a count of 3). Below this, the current branch is 'main', with '1 Branch' and '0 Tags' indicated. A search bar 'Go to file' and buttons for 'Add file' and 'Code' are also present.

The commit history section shows a commit by 'nrennie' titled 'Update README.md' with commit hash 'db5d3d6' and '9 Commits' in total. Below this, a list of files and folders is shown, each with its commit message and the time since the last commit.

File/Folder	Commit Message	Time
_extensions/PrettyPDF	fix subtitle and author	10 months ago
.gitignore	fix tex file title	10 months ago
.quartoignore	fix tex file title	10 months ago
LICENSE.md	add license	11 months ago
README.md	Update README.md	2 months ago
template.qmd	fix tex file title	10 months ago

# The `template.qmd` file

```
1 ---
2 title: "Pretty PDFs with Quarto"
3 format: PrettyPDF-pdf
4 ---
5
6 ## Quarto
7
8 Quarto enables you to weave together content and executable code into a finished
9 document. To learn more about Quarto see <https://quarto.org>.
10
11 ### Running Code
12
13 When you click the **Render** button a document will be generated that includes
14 content and the output of embedded code. You can embed code like this:
15
16 ```{r}
17 1 + 1
18 ```
```

# The `_extension.yml` file

```
1 title: PrettyPDF
2 author: Nicola Rennie
3 version: 0.0.1
4 contributes:
5   formats:
6     pdf:
7       include-in-header:
8         - "PrettyPDF.tex"
9       include-before-body:
10        - "pagestyle.tex"
11       toc: false
12       code-block-bg: light
13       linkcolor: highlight
14       urlcolor: highlight
```

# The PrettyPDF.tex file

```
1 % load packages
2 \usepackage{geometry}
3 \usepackage{xcolor}
4 \usepackage{eso-pic}
5 \usepackage{fancyhdr}
6 \usepackage{sectsty}
7 \usepackage{fontspec}
8 \usepackage{titlesec}
9
10 %% Set page size with a wider right margin
11 \geometry{a4paper, total={170mm,257mm}, left=20mm, top=20mm, bottom=20mm, right=20mm}
```

# The PrettyPDF.tex file

```
1 %% Let's define some colours
2 \definecolor{light}{HTML}{E6E6FA}
3 \definecolor{highlight}{HTML}{800080}
4 \definecolor{dark}{HTML}{330033}
5
6 %% style the chapter/section fonts
7 \chapterfont{\color{dark}\fontsize{20}{16.8}\selectfont}
8 \sectionfont{\color{dark}\fontsize{20}{16.8}\selectfont}
9 \subsectionfont{\color{dark}\fontsize{14}{16.8}\selectfont}
10 \titleformat{\subsection}
11     {\sffamily\Large\bfseries}{\thesection}{1em}{}[{\titlerule[0.8pt]}]
```

# The PrettyPDF.tex file

```
1 %% Let's add the border on the right hand side
2 \AddToShipoutPicture{%
3     \AtPageLowerLeft{%
4         \put(\LenToUnit{\dimexpr\paperwidth-3cm},0){%
5             \color{light}\rule{3cm}{\LenToUnit\paperheight}%
6         }%
7     }%
8     % logo
9     \AtPageLowerLeft{% start the bar at the bottom right of the page
10         \put(\LenToUnit{\dimexpr\paperwidth-2.25cm},27.2cm){% move it to the top
11             \color{light}\includegraphics[width=1.5cm]{_extensions/nrennie/Pret
12         }%
13     }%
14 }
```

# Check it works!

After pushing code to GitHub, install the extension:

```
1 quarto use template nrennie/PrettyPDF
```

Render the document:

```
1 quarto render document.qmd
```



# Check it works!

## PRETTY PDFS WITH QUARTO

### Quarto

---

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

### Running Code

When you click the **Render** button a document will be generated that includes both content and the output of embedded code. You can embed code like this:

```
1 + 1
```

```
[1] 2
```

You can add options to executable code like this

```
[1] 4
```

The `echo: false` option disables the printing of code (only output is displayed).



# Tips for building extensions

*Things I learnt the hard way...*

- Things get confusing when you build and extension with same name as your GitHub username
- (Font) paths are hard
- You'll likely want multiple output formats in one extension (e.g. book and pdf)



# Useful links

Slides: [nrennie.rbind.io/talks/bplim-quarto-extensions](https://nrennie.rbind.io/talks/bplim-quarto-extensions)

Blog: [nrennie.rbind.io/blog/making-pretty-pdf-quarto](https://nrennie.rbind.io/blog/making-pretty-pdf-quarto)

PrettyPDF extension: [github.com/nrennie/PrettyPDF](https://github.com/nrennie/PrettyPDF)

Building extensions guide: [quarto.org/docs/extensions/creating](https://quarto.org/docs/extensions/creating)



Building your own Quarto extension is an effective way to streamline the process of styling your documents.

 [nrennie.rbind.io](https://nrennie.rbind.io)

 [nrennie](https://github.com/nrennie)

 [nicola-rennie](https://www.linkedin.com/in/nicola-rennie)



