KGW Portfolio Notes

Kevin G. Williams

2025-11-10

Table of contents

Preface

This is a Quarto book. My portfolio. If I can make it work.

To learn more about Quarto books visit https://quarto.org/docs/books.

```
1 + 1
```

```
[1] 2
```

1 Introduction

This is a book created from markdown and executable code.

See Knuth (1984) for additional discussion of literate programming.

```
1 + 1
```

```
[1] 2
```

Part I

Part 1

2 Quarto Building Notes

2.1 Main Quarto Sources

Main Quarto Guide:

https://quarto.org/docs/guide/

> **!** Important
>
> As you preview your book, chapters will be rendered and updated. However, if you make changes to global options (e.g. `_quarto.yml` or included files) you need to fully re-render your book to have all of the changes reflected. Consequently, you should always fully `quarto render` your site before deploying it, even if you have already previewed changes to some pages with the preview server.

2.1.1 Publishing

When you are ready to publish the book, use the `render` command to render all output formats:

Listing 2.1 `Terminal`

```
quarto render
```

If you pass no arguments to `quarto render`, all formats will be rendered. You can also render individual formats via the `--to` argument:

Listing 2.2 `Terminal`

```
quarto render           # render all formats
quarto render --to pdf  # render PDF format only
```

The output of your book will be written to the `_book` sub-directory of your book project:

Listing 2.3 `Terminal`

```
mybook/
  _book/
    index.html # and other book files
    mybook.pdf
    mybook.epub
```

See the documentation on Publishing Websites for details on how to publish books to GitHub Pages, Netlify, and other services. Note that in that documentation the `output-dir` may be referred to as `_site`: for publishing books you should use `_book` rather than `_site`.

2.2 To Publish to Quarto Pub

2.3 Publish Command

The `quarto publish` command is the easiest way to publish locally rendered content. From the directory where your project is located, execute the `quarto publish` command for Quarto Pub:

Listing 2.4 `Terminal`

```
quarto publish quarto-pub
```

If you haven't published to Quarto Pub before, the publish command will prompt you to authenticate. After confirming that you want to publish, your content will be rendered and deployed, and then a browser opened to view your site.

2.4 toc table of contents info

https://quarto.org/docs/output-formats/html-basics.html
You can customize the table of contents (TOC) for HTML output formats using the `toc` option in your document or project YAML. Here are some common configurations:

- `toc: true` - Enables the TOC with default settings.
- `toc: false` - Disables the TOC.
- `toc: {depth: 2}` - Sets the depth of the TOC to 2 levels.
- `toc: {numbered: true}` - Enables numbered entries in the TOC.

- `toc: {float: true}` - Makes the TOC float alongside the content.
- `toc: {position: left}` - Positions the TOC on the left side of the page.
- `toc: {collapse: true}` - Allows sections in the TOC to be collapsible. You can combine these options as needed. For example, to create a floating, numbered TOC with a depth of 3 levels, you would use:

```
toc:
  depth: 3
  numbered: true
  float: true
```

2.5 Tags

#tag:projects #tag:data #todo

You can add tags to your Quarto documents using the `tags` field in the YAML front matter. Tags help categorize and organize your content, making it easier to find related documents. Here's how you can add tags:

```
---
title: "My Document"
tags: [data, analysis, R]
---
```

In this example, the document is tagged with "data", "analysis", and "R".
You can also add tags in a more structured way using a list:

```
---
title: "My Document"
tags:
  - data
  - analysis
  - R
---
```

Tags can be used for filtering and searching within your Quarto projects, especially when you have multiple documents. They can also be useful for generating tag clouds or lists of related content if you are building a website or blog with Quarto.
More Quarto Resources
- Quarto Official Documentation: https://quarto.org/docs/
- Quarto GitHub Repository: <

2.5.1 Invisible tags when rendered

You can add invisible tags to your Quarto documents by using HTML comments in the YAML front matter. This way, the tags will not be displayed in the rendered output but will still be present in the source code for organizational purposes. Here's how you can do it:

```
---
title: "My Document"
<!--
tags: [data, analysis, R]
-->
---
```

In this example, the tags "data", "analysis", and "R" are included in an HTML comment, making them invisible in the rendered document.
You can also use a list format within the HTML comment:

```
---
title: "My Document"
<!--
tags:
  - data
  - analysis
  - R
-->
---
```

This method allows you to keep your tags organized without displaying them in the final output.

3 Hexwall

3.1 Hex file resources

3.1.1 Shiny App for creating hex

https://connect.thinkr.fr/hexmake/

3.2 FIX - show examples of RRL hex

3.3 FIX - get all of these downloaded

https://github.com/rstudio/hex-stickers

3.4 FIX - fix warning for using deprecated purrr in hexwall.R file

Warning: `invoke()` was deprecated in purrr 1.0.0. Please use `exec()` instead. This warning is displayed once every 8 hours. Call `lifecycle::last_lifecycle_warnings()` to see where this warning was generated.

3.5 load hexwall.R function

```
source("~/Documents/r-studio-and-git/KGW_Portfolio_Notes/hexwall.R")
```

```
Linking to ImageMagick 6.9.13.29
Enabled features: cairo, fontconfig, freetype, heic, lcms, pango, raw, rsvg, webp
Disabled features: fftw, ghostscript, x11
```

```
test <- hexwall("~/Documents/r-studio-and-git/KGW_Portfolio_Notes/my_stickers", sticker_row_s
```

```
Warning: `invoke()` was deprecated in purrr 1.0.0.
i Please use `exec()` instead.
```

```
test
```

3.6 FIX - getting to export to PDF

3.7 FIX - getting webshot and phantomjs working

source("~/Documents/r-studio-and-git/my_hex_stickers/hexwall/hexwall.R") ## call hexwall function and assign to "test" ## this sometimes get an error – try adjusting sticker_row_size test <- hexwall("~/Documents/r-studio-and-git/my_hex_stickers/my_stickers", sticker_row_size = 7, sticker_width = 200) test

png("~/Documents/r-studio-and-git/my_hex_stickers/hexwall/samplehex/test123.png") hexwall("~/Documents/r-studio-and-git/my_hex_stickers/hexwall/samplehex", sticker_row_size = 4, sticker_width = 200) image_write(test, "~/Documents/r-studio-and-git/my_hex_stickers/hexwall/sample dev.off()

hex_table <- data.table

3.8 FIX - data.table is not working properly – needs to be fixed

It seems to be reading, but not displaying in data.table format correctly.
It works within R, but not in Quarto when rendered.

3.9 FIX - data.table giving a warning about reading in last line

```
#library(data.table)
hex_table <- datatable(read.csv("~/Documents/r-studio-and-git/my_book_again/my_stickers_data_
hex_table
```

```
file:////private/var/folders/sf/d810pt617h181j949xmh0yvh0000gn/T/RtmpTwdt2T/file7936c4955b/wi
```

Show 10 entries Search:

	Name	Type	Downloaded	Added_To_Hexwall	Official.	Source	Notes	Last_Updat
1	Rkaggle	Package	Y	Y	Y	https://github.com/benyamindsmith/RKaggle	N/A	11/9/25
2	RRL	Company	Y	Y	N	Selfmade	N/A	Unknown
3	Leaflet	Package	N	N	N	https://r-graph-gallery.com/package/leaflet.html	No official hex image	11/9/25

Showing 1 to 3 of 3 entries Previous 1 Next

```
#hex_table_as_tibble <- as_tibble(hex_table)
#hex_table_as_tibble
```

4 Projects - OpenAI

4.1 use ellmer package

```
library(ellmer)
#chat <- chat_openai()
```

Sys.setenv(OPENAI_API_KEY = '<your key here')

#> Using model = "gpt-4.1". chat$chat(" What is the difference between a tibble and a data frame? Answer with a bulleted list ")

chat$chat("Who created R?", echo = FALSE)

4.2 get prompt working with openai

5 Projects - Creating Projects

5.0.1 Creating a new quarto document with connection to github repository

1) click on new project
2) new directory
3) Choose project type: Example: Quarto Book
4)) Directory name: YOUR-PROJECT-NAME
5) Create a git repository: check box
6) Create project

Notes here: https://happygitwithr.com/existing-github-last

Now, to connect to your GITHUB repository, follow these steps:
In console:
usethis::use_git() # to make initial comment
Then: usethis::use_github() This will create a new repository on GitHub and link it to your local Git repository. It will also push your initial commit to the remote repository on GitHub.

README.md will need to be created. You can created one from github respository page. "Add Page" button.
Then pull it down to your local repository.

5.0.2 Creating a New Project

To create a new project, follow these steps: Make this your working directory: ~/Documents/r-studio-and-git

In console: library(usethis)

usethis::create_project("~/Documents/r-studio-and-git/YOUR-PROJECT-NAME") This will create a new directory for your project and open it in RStudio. It will also create a default R project file (.Rproj) for you. It will also create a readme file for you. It will also create a .gitignore file for you.

5.0.3 Setting Up Version Control with Git

To set up version control with Git for your new project, follow these steps: In console: usethis::use_git() This will initialize a Git repository in your project directory. It will also create a .git directory to track changes. It will also make an initial commit with the existing files in your project. ### Connecting to GitHub To connect your local Git repository to a remote repository on GitHub, follow these steps: First, create a new repository on GitHub without a README, .gitignore, or license. Then, in console: usethis::use_github() This will create a new repository on GitHub and link it to your local Git repository. It will also push your initial commit to the remote repository on GitHub. ### Summary You have now created a new R project, set up version control with Git, and connected it to a remote repository on GitHub. You can now start working on your project and use Git to track changes and collaborate with others.

5.0.4 add starting files

usethis::use_r("import") # for R scripts under "R" folder usethis::use_r("tidy") # for R scripts under "R" folder

5.0.5 add data folder

usethis::use_data_raw() # for raw data under "data-raw" usethis::use_directory("DATA") # for processed data under "data"

5.0.6 fixing and updating README.md file with github info

When README.md file was first created, it does not render properly on GitHub.
To fix this, follow these steps:
1. Open README.md file in RStudio.
2. Add the following YAML header at the top of the file:

```
---
output: github_document
---
```

3. Save the file.
4. In console, run the following command to render the README.md file: rmarkdown::render("README.md")

5. Commit and push the changes to GitHub. This will update the README.md file to render properly on GitHub with the specified title, author, and date.
   You will need to render the README.md file each time you make changes to it.

Part II

Part 2

6 Projects - Hex

- Get Hexwall working
- Get Hexwall working with maps
- Make a RRL Hex with Shiny App
- Make a RRL Hex with hexSticker package
- Get a Shiny Hexwall working
- Get a Shiny Hexwall working with maps
- Get a Shiny Hexwall working with multiple map choices
- Get an updated Hexwall working with an input of a hex

- Data loaded

- Need to preprocess

- Pending

- Some progress
- No progress
- Stalled
- New
- Ongoing
- Completed

- Scheduled
- Important
- Private
- Public
- Testing
- Maintenance
- Documentation
- Ideas
- Design
- Analysis
- Visualization
- Automation
- Research

- Integration
- Configuration
- Cleanup
- Exploration
- Organization
- Calculation
- Foundation
- Experimentation
- Toolbox
- Security
- Networking
- Cloud
- Hardware
- Packaging
- Development
- Testing
- Design
- Documentation
- Deployment
- Maintenance
- Collaboration
- Management
- Presentation

7 RRL.com Stuff

7.0.1 As of 20251010

www.wordpress.com account to use is: User: kwill1992rrl which links to kwill1992@hotmail.com There is another one (at least one other), which is User: kwill1992 and links to kevin.williams@rrldataanalytics.com The kwill1992 one does not have any websites linked to it. The kwill1992rrl account has both the old one @ www.rrldataanalytics.com and the one in development at the numbered IP address.

To get to the linked sites at www.wordpress.com, go to "managed blogs" button and this will come up: https://wordpress.com/sites

Only one is default site at any given time. You still need to login with user account to edit each website. Each website has a "user" and "kwill1992" user. Different passwords for each for each website.

8 Concepts

8.1 data.table vs DT

The `data.table` package is a powerful R package for data manipulation and aggregation, known for its speed and efficiency with large datasets. It provides an enhanced version of data frames with additional features like fast grouping, indexing, and in-place updates.
On the other hand, `DT` is an R package that provides an interface to the JavaScript library DataTables. It is primarily used for creating interactive tables in R Markdown documents and Shiny applications. `DT` allows users to create sortable, searchable, and paginated tables with ease, enhancing the user experience when working with tabular data in web applications.

8.1.0.1 for a quarto document and output as a website, use DT for a data table.

8.2 Link to kaggle datasets to downloaded data directly into R using the kaggle API

library(RKaggle) # for interacting with Kaggle API)
using RKaggle to download the dataset from Kaggle
Example:
superstore_dataset <- get_dataset("vivek468/superstore-dataset-final")

github for RKaggle:
https://github.com/benyamindsmith/RKaggle

8.3 Leaflet

The `leaflet` package in R is a powerful tool for creating interactive maps. It allows users to visualize spatial data with various layers, markers, and pop-ups. The package is built on top of the Leaflet JavaScript library, providing a user-friendly interface for R users to create dynamic maps that can be embedded in R Markdown documents, Shiny applications, or viewed in RStudio. With `leaflet`, users can easily add tiles, polygons, and other geographical features to their maps, making it a versatile choice for geospatial data visualization.

https://rstudio.github.io/leaflet/index.html
https://r-charts.com/spatial/interactive-maps-leaflet/
https://r-graph-gallery.com/package/leaflet.html
https://bookdown.org/nicohahn/making_maps_with_r5/docs/leaflet.html https://leafletjs.com/reference.htm
https://www.geeksforgeeks.org/r-language/leaflet-package-in-r/
https://www.jla-data.net/eng/leaflet-in-r-tips-and-tricks/

8.4 Publishing

https://quarto.org/docs/publishing/quarto-pub.html
https://quarto.org/docs/publishing/

9 Summary

In summary, this book has no content whatsoever.

```
1 + 1
```

```
[1] 2
```

Errors and Warnings

Github

- to fix a git push commit holding up
  - rm .git/index.lock [in Terminal]

Render to PDF if using html widgets

- if rendering to PDF from HTML widgets (e.g., `leaflet`, `DT`), use the `webshot2` package
  - install with `install.packages("webshot2")`
  - ensure that `webshot2::install_phantomjs()` has been run once to install PhantomJS

References

Knuth, Donald E. 1984. "Literate Programming." *Comput. J.* 27 (2): 97–111. https://doi.org/10.1093/comjnl/27.2.97.