R & RStudio

Table of contents

This section covers installing R/RStudio and tuning the IDE.

- R & RStudio overview
- Four Panes
- Global Options
- Legacy Intro (PDF-to-QMD)

Chapter 1

R vs. RStudio

- **R** is the programming language and runtime (it executes code).
- **RStudio** (Posit IDE) is the graphical environment where you write code, manage files, and render documents.

You'll always use them together: R runs under the hood; RStudio makes it easier to work productively.

Chapter 2

The RStudio interface (four panes)

1. **Source** (top-left): edit scripts and `.qmd` files
2. **Console** (bottom-left): run code interactively
3. **Environment / History** (top-right): see objects in memory, command history
4. **Files / Plots / Packages / Help / Viewer** (bottom-right): navigate files, view plots, load packages, get help

---

Chapter 3

Projects: your #1 best practice

Always work **inside an RStudio Project** (one folder per course or research project).

- Each Project has its own working directory, history, and settings.
- This avoids broken file paths and makes your work reproducible.

Create a Project:
File → New Project → New Directory → New Project

Pro tip: In Preferences → General, set RStudio to **never save/restore `.RData`**. Clean sessions prevent stale objects from carrying over.

Chapter 4

Console vs. Script (and Quarto)

- Use the **Console** for quick experiments.

- Save work you want to keep in a **script** (`.R`) or a **Quarto document** (`.qmd`).

Render `.qmd` with the **Render** button (RStudio toolbar) or in a terminal:

```
quarto render myfile.qmd
```

4.1 Handy keys (Source pane)

- Run line/selection: **Ctrl+Enter** (Win/Linux) or **Cmd+Return** (macOS)
- Insert code chunk in `.qmd`: **Ctrl+Alt+I** (Win/Linux) or **Option+Cmd+I** (macOS)
- Full list: **Tools → Keyboard Shortcuts Help**

Chapter 5

Files and paths (keep them relative)

Inside a Project, paths are **relative to the project root**.

Example:

```r
# Good (portable)
songs <- read.csv("data/songs.csv")

# Bad (breaks on other computers)
songs <- read.csv("C:/Users/Alex/Desktop/songs.csv")
```

Avoid `setwd()` in scripts. The Project sets the working directory automatically.

Chapter 6

Git integration

If your Project is under Git, you'll see a **Git** tab in RStudio. Use it to:

- **Stage** → **Commit** → **Push** changes to GitHub.

If the Git tab is missing: **Tools** → **Project Options** → **Git/SVN** → **Enable version control system: Git** or run:

```
usethis::use_git()
```

Chapter 7

Handy shortcuts

- **Command Palette**: **Ctrl/Cmd+Shift+P**
- **Keyboard Shortcuts Help**: **Alt/Option+Shift+K**
- **Comment/Uncomment**: **Ctrl+Shift+C** (Win/Linux) or **Cmd+Shift+C** (macOS)

Chapter 8

Default settings to adopt (once)

- Disable save/restore of `.RData` (Preferences → General).
- Keep Projects in simple paths (e.g., `Documents/MC451`).
- Enable Git integration early.

Chapter 9

Where this fits with the package

Everything in **mccoursepack**—listing weeks, downloading scaffolds, rendering `.qmd`—assumes you are inside your course Project.

Workflow reminder:

```r
library(mccoursepack)

# list weeks
list_weeks("mc451")

# download week 1 into the project folder
download_week("mc451", 1, dest = ".")
```

Open the downloaded `week_01/` folder, work on `.qmd` files there, and render as needed.