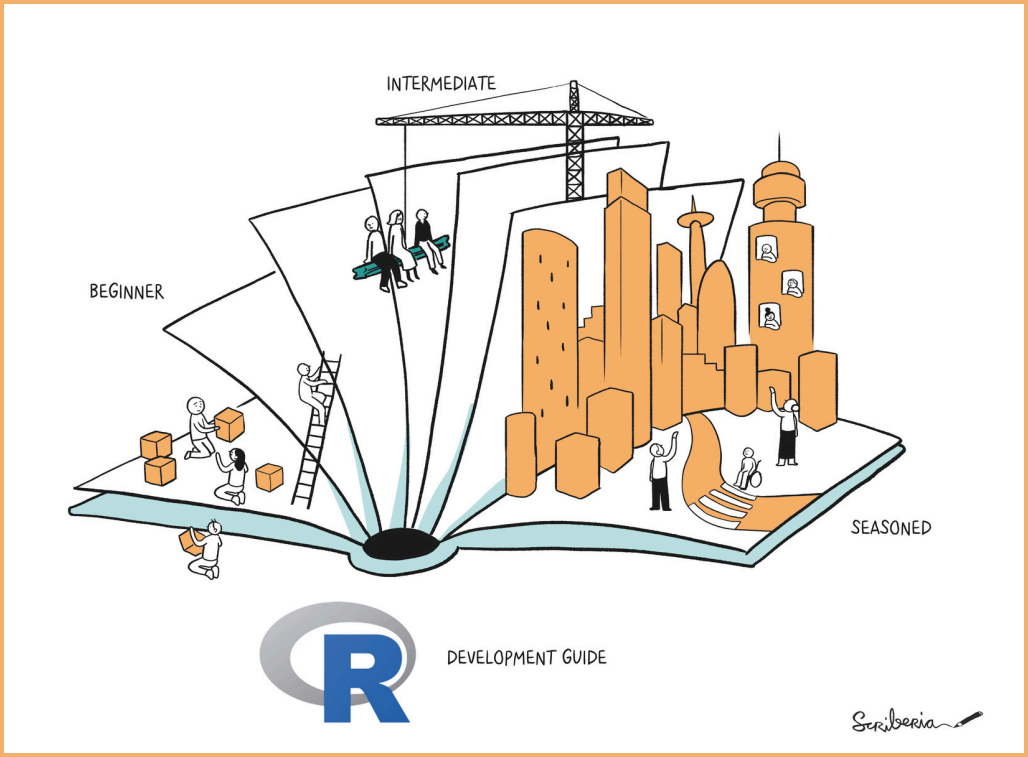
R Development Guide



collaboratively authored by the

R Contribution Working Group

Building Reproducible and Interactive Analytics Courses

Table of contents

1 Designing an Online Course: Where Do I Start? 3

2 Overview 4

3 Define Learning Outcomes 5

4 Choose Your Infrastructure 6

5 Use Quarto to Unify Content 7

5.1 4. Organize Your Repository 7

6 Plan for Iteration 8

7 Summary 9

8 Introduction 10

10 Summary 12

References 13

1 Designing an Online Course: Where Do I Start?

2 Overview

Designing an effective online course starts not with technology, but with intention. The goal is to align instructional strategies with learning outcomes, supported by scalable and reproducible tools. This chapter walks you through establishing a reproducible, cloud-friendly online course infrastructure.

3 Define Learning Outcomes

Start with the end in mind. Ask yourself: - What should students be able to *do* by the end of the course? - How will you know they've learned it? - What evidence of learning will students produce?

Use **Bloom's Taxonomy** to scaffold outcomes across cognitive levels, from remembering to creating.

4 Choose Your Infrastructure

Pick your teaching stack based on your audience, goals, and support capacity:

Tool	Use Case	Features
GitHub Classroom	Code-driven assignment workflows	Version control, autograding, feedback
Posit Cloud	R/Python course projects	No install, team projects, reproducibility
AWS Academy	Cloud labs and AI/ML workflows	SageMaker, EC2, real-world industry tooling
VSCode.dev / Codespaces	Lightweight online IDE	GitHub native integration, no setup required

5 Use Quarto to Unify Content

Create reproducible lecture notes, websites, assignments, and slides with Quarto:

```
quarto create-project mycourse --type website
```

Then add:

- `index.qmd` for homepage/overview
- `syllabus.qmd`, `assignments/`, `modules/` for modular content

5.1 4. Organize Your Repository

Structure a course repo like so:

```
mycourse/
  index.qmd
  syllabus.qmd
  assignments/
      hw01.qmd
  modules/
      module01.qmd
      module02.qmd
  _quarto.yml
```

Tips:

- Use **GitHub Issues** for class Q&A and announcements.
- Have students submit via **pull requests**.
- Use **GitHub Actions** for automated feedback, grading, or rendering.

6 Plan for Iteration

Your first version won't be perfect. Plan to revise:

- Module sequencing or timing
- Tooling constraints (e.g., firewall, browser compatibility)
- Based on student feedback and usage analytics

Use a `CHANGELOG.md` and commit messages to document changes over time.

7 Summary

Start with outcomes. Choose infrastructure intentionally. Use Quarto and GitHub to scaffold your course reproducibly. Keep it flexible, open, and iterative. You're not just teaching content—you're modeling professional workflows.

8 Introduction

This is a book created from markdown and executable code.

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

9

10 Summary

In summary, this book has no content whatsoever.

References

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