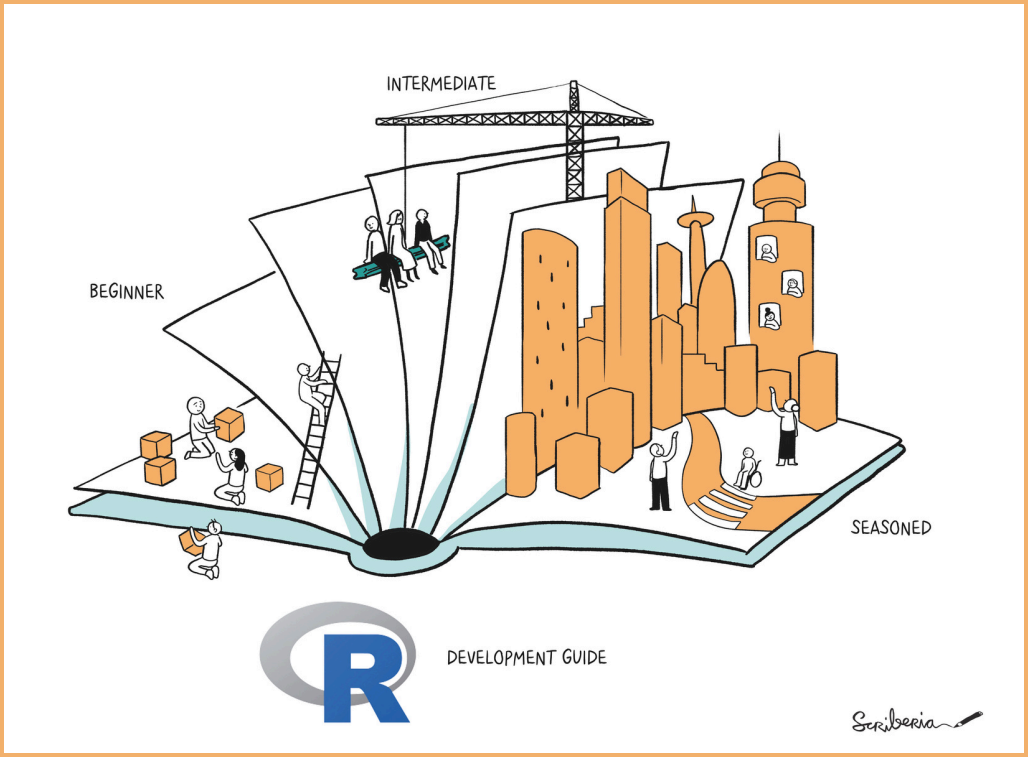
R Development Guide



R Development Guide. This illustration is created by Scriberia with The Turing Way community, used under a CC-BY 4.0 licence. DOI: https://zenodo.org/records/13882307

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? 4

2 Introduction 5

3 What This Book Offers 6
3.1 For BU Faculty Use Only 6

4 Introduction 7

5 Overview 8

6 Define Learning Outcomes 9

7 Choose Your Infrastructure 10

8 Use Quarto to Unify Content 11
8.1 Organize Your Repository 11

9 Plan for Iteration 13

10 Summary 14

11 Introduction to Git and GitHub 15

12 Overview 16

13 Learning Objectives 17

14 Understanding Version Control 18
14.1 What is Version Control? 18
14.2 Git: A Distributed Version Control System 18
14.3 GitHub: A Platform for Collaboration 18

15 Setting Up Git 19
15.1 Installing Git 19
15.2 Configuring Git 19

16 Creating a GitHub Account 20

17 Initializing a Git Repository **21**
17.1 Creating a New Repository on GitHub 21
17.2 Cloning the Repository Locally 21

18 Making Your First Commit **22**

19 Exploring GitHub's Web Interface **23**

20 Summary **24**

21 Summary **25**

References **26**

1 Designing an Online Course: Where Do I Start?

2 Introduction

As data and computation continue to reshape disciplines across the university, faculty are increasingly called upon to integrate code, collaboration, and scalable infrastructure into their teaching. Yet, despite this growing demand, instructors often lack structured guidance on how to develop and maintain reproducible instructional materials across tools, platforms, and modalities.

Teach Reproducibly is a faculty-facing resource developed by the **Metropolitan College**, in partnership with **BU ELIVE Services** and **BU Virtual**, to support instructors in building robust, version-controlled, and scalable course content for use in **Boston University's Blackboard Ultra environment**.

This book provides practical guidance for instructors teaching in data-rich and computation-heavy domains — including analytics, computer science, statistics, engineering, public health, and beyond — who are committed to developing high-quality, maintainable instructional workflows.

3 What This Book Offers

Rather than focusing solely on curriculum content, this book emphasizes **how content is developed, maintained, and shared reproducibly**. It introduces foundational tools and design principles that support instructional reuse, transparency, and collaboration, including:

- Literate programming environments with **Quarto**
- Source control and team workflows using **Git and GitHub**
- Collaborative assignments via **GitHub Classroom**
- Cloud-based computing with **AWS Academy** and **VS Code.dev**
- Browser-native runtime environments such as **WebR** and **Pyodide**
- Seamless integration with **Blackboard Ultra** and BU's LMS ecosystem

This resource supports faculty in developing modular instructional pipelines, enabling version-controlled teaching materials, reproducible research-in-practice demonstrations, and seamless cloud-backed student exercises. Each module aligns with BU's pedagogical standards and technical infrastructure.

3.1 For BU Faculty Use Only

This resource is intended for **internal instructional development and support** within Boston University. It is not licensed for external reproduction, distribution, or commercial use.

For inquiries, access to editable templates, or instructional support, please contact:

- **BU ELIVE Services**: elive@bu.edu
- **BU Virtual Instructional Design**: buv@bu.edu
- **Metropolitan College Academic Affairs**: metacad@bu.edu

4 Introduction

Designing an Quarto-based Course

5 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.

6 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.

7 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

8 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

8.1 Organize Your Repository

Structure a course repo like so:

```
teach-reproducibly/
  index.qmd               # Course overview/homepage
  syllabus.qmd            # Syllabus details
  deliverables.qmd        # Assignment & grading structure
  schedule.qmd            # Weekly/module timeline
  M1/
      M1_LN1.qmd          # Lecture Note 1
      M1_LN2.qmd
      M1_P1.qmd           # Presentation 1
      M1_P2.qmd
      M1_T1.qmd           # Tutorial 1
      M1_T2.qmd
      M1_Lab1.qmd
      M1_Lab2.qmd
      M1_A1.qmd           # Assignment
      M1_Proj1.qmd        # Mini project
      M01-highlights.qmd
      M1L01_figures/
      M1L02_figures/
  _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.

9 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.

10 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.

11 Introduction to Git and GitHub

12 Overview

In the realm of modern education, especially within data-centric disciplines, the ability to manage and collaborate on code is paramount. Git and GitHub serve as foundational tools that facilitate version control, collaborative development, and reproducible research. This chapter aims to introduce these tools, providing educators with the skills necessary to integrate them into their teaching workflows.

13 Learning Objectives

By the end of this chapter, you will be able to:

- Understand the concepts of version control and its significance in educational contexts.
- Differentiate between Git and GitHub and comprehend their respective roles.
- Install and configure Git on your local machine.
- Create a GitHub account and set up your user profile.
- Perform basic Git operations: initializing a repository, committing changes, and pushing to GitHub (GitHub 2024).

14 Understanding Version Control

14.1 What is Version Control?

Version control is a system that records changes to a file or set of files over time, enabling you to recall specific versions later. In educational settings, this allows instructors to:([arxiv.org][2])

- Track the evolution of course materials.
- Collaborate with colleagues on curriculum development.
- Revert to previous versions of teaching resources when necessary.([docs.github.com][3])

14.2 Git: A Distributed Version Control System

Git is a distributed version control system that allows multiple contributors to work on a project simultaneously without interfering with each other's work. Each contributor has a complete copy of the repository, including its history, on their local machine. This setup enhances collaboration and ensures data integrity.([docs.github.com][4])

> For more information, refer to the official Git documentation: About Git.([docs.github.com][4])

14.3 GitHub: A Platform for Collaboration

GitHub is a cloud-based platform that hosts Git repositories, providing tools for collaboration, code review, and project management. In an educational context, GitHub can be used to:

- Distribute assignments and course materials.
- Facilitate student collaboration on projects.
- Manage feedback and revisions efficiently.([docs.github.com][5], [en.wikipedia.org][6])

> Learn more about GitHub here: About GitHub.([docs.github.com][7])

15 Setting Up Git

15.1 Installing Git

To begin using Git, you need to install it on your local machine.

- **Windows**: Download the installer from Git for Windows and follow the setup instructions.
- **macOS**: Install Git using Homebrew:

  ```
  brew install git
  ```

- **Linux**: Use your distribution's package manager. For example, on Ubuntu:

  ```
  sudo apt-get install git
  ```

  Detailed installation instructions can be found here: Set up Git.([docs.github.com][8])

15.2 Configuring Git

After installation, configure your Git username and email address. These details will be associated with your commits.

```
git config --global user.name "Your Name"
git config --global user.email "your_email@bu.edu"
```

To verify your configuration:

```
git config --list
```

16 Creating a GitHub Account

To utilize GitHub's features, create an account:

1. Navigate to GitHub's Sign Up Page.
2. Enter your email address, create a username and password.
3. Follow the prompts to complete the setup process.

After account creation, you can personalize your profile by adding a profile picture, bio, and other details.

17 Initializing a Git Repository

17.1 Creating a New Repository on GitHub

1. Log in to your GitHub account.
2. Click on the "+" icon in the top-right corner and select "New repository".
3. Provide a repository name, e.g., `teaching-materials`.
4. Optionally, add a description.
5. Choose the repository's visibility (public or private).
6. Initialize the repository with a README file.
7. Click "Create repository".([docs.github.com][1])

   For a step-by-step guide, refer to: Quickstart for repositories.([docs.github.com][1])

17.2 Cloning the Repository Locally

To work on your repository locally:

1. Navigate to your repository on GitHub.
2. Click on the "Code" button and copy the URL.
3. Open your terminal and run:

   ```
   git clone https://github.com/yourusername/teaching-materials.git
   ```

4. Navigate into the cloned directory:

   ```
   cd teaching-materials
   ```

18 Making Your First Commit

Let's add a new file to your repository:

1. Create a new file named `lesson1.md` and add some content.
2. Stage the file for commit:
   ```
   git add lesson1.md
   ```
3. Commit the file with a message:
   ```
   git commit -m "Add Lesson 1 materials"
   ```
4. Push the changes to GitHub:
   ```
   git push origin main
   ```

Your new file is now part of the repository on GitHub.

19 Exploring GitHub's Web Interface

GitHub's web interface provides various features to manage your repository:

- **Code**: View and manage your files.
- **Issues**: Track tasks, enhancements, and bugs.
- **Pull Requests**: Propose changes and collaborate with others.
- **Actions**: Automate workflows.
- **Projects**: Organize and prioritize your work.

  For a comprehensive overview, visit: Hello World - GitHub Docs.([docs.github.com][9])

20 Summary

In this chapter, we've covered the essentials of Git and GitHub, focusing on their applications in educational settings. By setting up Git, creating a GitHub account, and performing basic operations, you're now equipped to manage and collaborate on teaching materials effectively.

Note: For visual learners, GitHub provides illustrative guides and tutorials to further enhance understanding. It's recommended to explore these resources to reinforce the concepts covered in this chapter.

21 Summary

In summary, this book has no content whatsoever.

References

GitHub. 2024. "Quickstart for Repositories - GitHub Docs." https://docs.github.com/en/repositories/creating-and-managing-repositories/quickstart-for-repositories.