a1a # a1b # tooling

Git Version Control: Getting Started

TCSS 305 Programming Practicum

This guide introduces Git, the industry-standard version control system, and GitHub, where you'll submit your assignments. You'll learn why version control matters and master the essential workflow: clone, commit, push.

Why Version Control?

You've probably experienced the "versioning nightmare":

```
final_project.docx
final_project_v2.docx
final_project_v2_FINAL.docx
final_project_v2_FINAL_really_final.docx
final_project_v2_FINAL_really_final_USE_THIS_ONE.docx
```

This happens because we naturally want to:

- **Keep a history** of what changed and when
- **Go back** to earlier versions if something breaks
- **Track who** changed what (in team projects)

Version control systems solve this problem properly. Instead of endless file copies, you keep one set of files and the system tracks every change automatically.

> **Important**
>
> Version control is not optional in professional software development. Every company, every open-source project, every serious development team uses it. Learning it now prepares you for internships and careers.

What is Git?

Git is a distributed version control system created by Linus Torvalds in 2005 (the same person who created Linux). It's now the dominant version control system in the software industry.

Key characteristics:

Feature	Meaning
Distributed	Every developer has a complete copy of the project history
Local-first	Most operations work offline on your own machine
Fast	Designed for performance, even with huge projects
Free	Open-source and free to use

Git tracks changes to your files over time, letting you:

- See what changed between any two points in history
- Restore previous versions of files
- Understand when and why changes were made

Git Is Not GitHub

This is a common point of confusion. They are related but different:

	Git	GitHub
What it is	A tool that runs on your computer	A website/service that hosts Git repositories
Where it runs	Locally on your machine	On GitHub's servers (the cloud)
Purpose	Track changes to files	Share repositories, collaborate, submit assignments
Cost	Free, open-source	Free for basic use, paid plans for extras

Analogy: Git is like a word processor (the tool you use), while GitHub is like Google Drive (a place to store and share your documents).

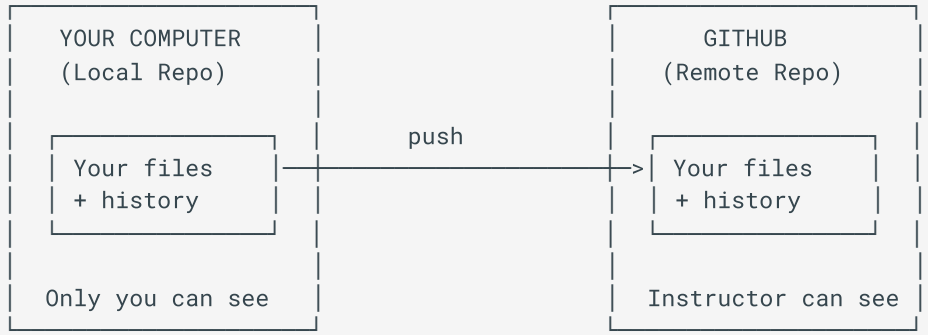
GitHub Classroom

GitHub Classroom is GitHub with educational features added. When you accept an assignment:

1. GitHub creates a private repository for you
2. The starter code is automatically copied in
3. Your instructor can see your repository (for grading)
4. You submit by pushing your work to this repository

Local vs. Remote Repositories

Understanding the relationship between your computer and GitHub is essential.



Repository	Location	Who Can See It
Local	On your computer	Only you
Remote	On GitHub's servers	You + anyone you grant access (instructors for grading)

> **Warning**
>
> Changes on your local machine are invisible to your instructor until you **push** them to GitHub. "I committed it" is not the same as "I pushed it."

Essential Terminology

Before diving into the workflow, learn these terms:

Term	Definition
Repository (repo)	A project folder tracked by Git, including all files and their complete history
Clone	Copy a remote repository to your local machine (done once per project)
Commit	Save a snapshot of your changes to local history (like a checkpoint in a game)
Push	Upload your local commits to the remote repository on GitHub
Working directory	The actual files on your computer that you edit

> **Tip**
>
> Think of commits like save points in a video game. You can always return to any previous save point. Make commits whenever you reach a working state.

The Basic Workflow

For TCSS 305 assignments, you'll follow this simple workflow:

Step 1: Clone (Once Per Assignment)

When you accept a GitHub Classroom assignment, you receive a URL. Clone it to get the starter code:

```
Clone from GitHub → Local copy on your machine
```

You only do this once per assignment.

Step 2: Edit Files in Your IDE

Open the project in IntelliJ and write your code. Git tracks which files you've changed, but changes are not saved to history yet.

Step 3: Commit (Save Checkpoint)

When you reach a good stopping point (code compiles, a feature works), commit your changes:

```
Edit files → Stage changes → Commit with message
```

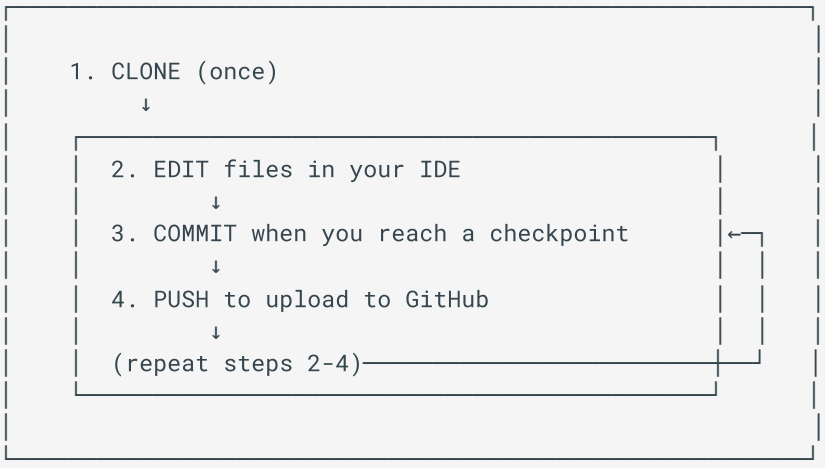
A commit creates a permanent checkpoint in your local history. You can make many commits before pushing.

Step 4: Push (Upload to GitHub)

Send your commits to GitHub so your instructor can see them:

```
Local commits → Push → Visible on GitHub
```

The Cycle



> **Tip**
>
> Commit often. A good rule: if you can describe what you just did in one sentence, it's worth a commit. "Added constructor validation" or "Fixed price calculation bug" are good commit-sized changes.

Authentication: Why GitHub Asks for Credentials

Git operations fall into two categories:

Operation Type	Examples	Requires Authentication?
Local	Commit, view history, compare versions	No
Remote	Clone (private repo), push, pull	Yes

When you push to GitHub, you're proving you have permission to modify that repository.

> **Warning**
>
> GitHub no longer accepts account passwords for Git operations. You'll need to use one of these methods:
>
> - **Personal Access Token** (recommended) — configure once in IntelliJ, works seamlessly
> - **GitHub Desktop** — handles authentication automatically, good alternative
> - **SSH keys** — a more secure method for advanced users
>
> For TCSS 305, we recommend a **Personal Access Token** configured in IntelliJ because it integrates directly with the IDE. See the troubleshooting section in your assignment for setup instructions. GitHub Desktop is a solid alternative if you prefer a visual tool.

Verifying Your Submission

After pushing, always verify your work is visible on GitHub:

1. Open your browser and go to github.com
2. Navigate to your assignment repository
3. Check that you see your latest changes
4. Verify the commit timestamp matches when you pushed

The Golden Rule

If it's not on GitHub, it's not submitted.

Your instructor grades what's on GitHub, not what's on your computer. Always verify your push succeeded before considering an assignment complete.

What to Look For

On your GitHub repository page, check:

- Your files appear with recent "Last commit" timestamps
- Click on files to verify the content is what you expect
- The commit count has increased

Common Mistakes

Mistake 1: Forgetting to Push

Symptom: You committed your work, but GitHub shows old code.

What happened: Commits are local until pushed. Your work is saved on your machine but not uploaded.

Fix: Push your commits.

Mistake 2: Committing but Not Pushing (Before Deadline)

Symptom: You thought you submitted, but GitHub shows commits from hours/days ago.

What happened: You made commits but forgot to push, or the push failed.

Fix: Check GitHub after every push. Make "check GitHub" part of your submission routine.

Mistake 3: Editing Files Outside the Repository

Symptom: Git doesn't see your changes.

What happened: You opened files from a different location or copied files outside the project folder.

Fix: Always edit files within the cloned project folder. Use IntelliJ's project view to navigate.

Tip

Make it a habit: after every work session, push your changes. Even if the assignment isn't due, having your work on GitHub means you have a backup.

Summary

Concept	Key Point
Version Control	Tracks changes over time, eliminates file-copy chaos
Git	The tool that runs on your computer, works offline
GitHub	The website that hosts your repositories in the cloud
Clone	Copy remote repo to local (once per project)
Commit	Save a checkpoint to local history
Push	Upload commits to GitHub
Verify	Always check GitHub.com to confirm your submission

Gen AI & Learning: Version Control Commands

AI coding assistants can help you with Git commands and troubleshoot issues. However, understanding the fundamental concepts (local vs. remote, commit vs. push) is essential because AI suggestions assume you know what you're trying to accomplish. If an AI suggests running `git push`, you should understand that this uploads your local commits to GitHub, not just "saves your work."

Further Reading

External Resources

- GitHub Docs: Hello World - GitHub's beginner tutorial
- Git Handbook - GitHub's Git overview
- GitHub Desktop Documentation - Using the desktop application

---

References

Introductory Resources:

- Pro Git Book — Free online book by Scott Chacon and Ben Straub, Chapter 1-2 for beginners
- GitHub Docs: Getting Started — Official GitHub documentation

Historical Context:

- Torvalds, L. (2007). Git: A Stupid Content Tracker — Original Git documentation
- A Short History of Git — Why Git was created

Tooling:

- GitHub Desktop — GUI application for Git operations
- GitHub Classroom — Educational features for assignments

This guide is part of TCSS 305 Programming Practicum, School of Engineering and Technology, University of Washington Tacoma.