Version Control: Git and GitHub

CS2113 – Software Development Project

Masoud Hamad

School of Computing Communication and Media Studies

Academic Year 2025

Outline

What is Version Control?

Definition

A system for storing code that enables tracking, sharing, and managing changes.

Benefits:

- Store backups of current and older versions
- Enable collaboration without meeting in person
- Track who made what changes
- Recover previous project states
- Identify when bugs were introduced

Git and GitHub

Git

- Created by Linus Torvalds
- Distributed version control
- Runs locally on your machine
- Free and open source

GitHub

- Cloud hosting for Git repos
- Collaboration features
- Issues, PRs, Projects
- Industry standard

Initial Setup

```
# Configure your identity
git config --global user.name "Your␣Name"
git config --global user.email "email@example.com"
git config --global init.defaultBranch main
```

Important

Use the same email as your GitHub account!

Starting a Project

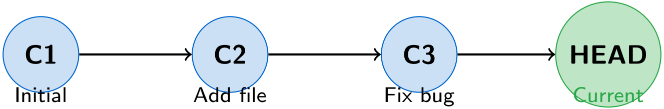
```
# Initialize a new Git repository
git init

# Check status
git status
```

What happens?

Creates a `.git` folder storing all project metadata and history.

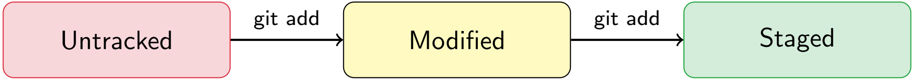
Understanding Commits



A Commit is...

A bundle of changes – a snapshot of your project at a point in time.

File States in Git



- **Untracked:** New files unknown to Git
- **Modified:** Changed but not staged
- **Staged:** Ready for commit

The Commit Workflow

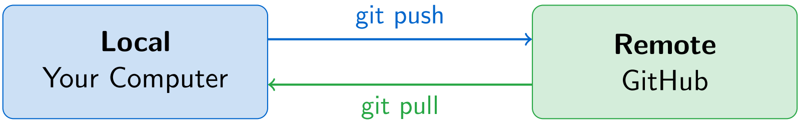
```
# 1. Make changes to files

# 2. Stage changes
git add filename.txt
git add .  # Add all changes

# 3. Create commit
git commit -m "Descriptive␣message"

# 4. Check status
git status
```

Local vs Remote Repository



Local
Your Computer

git push

git pull

Remote
GitHub

Connecting to GitHub

```
# Add remote repository
git remote add origin https://github.com/user/repo.git

# Push to remote (first time)
git push -u origin main

# Push subsequent changes
git push

# Pull changes from remote
git pull
```

Cloning a Repository

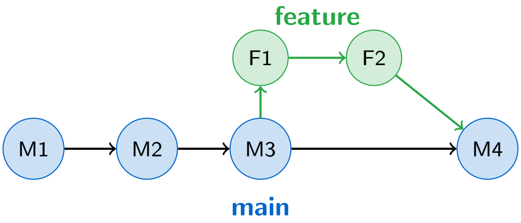
```
# Clone existing repository
git clone https://github.com/user/repo.git

# Clone into specific folder
git clone https://github.com/user/repo.git folder-name
```

Note

To push to a cloned repo, you need collaborator access!

Understanding Branches



Purpose

Branches allow parallel development without affecting the main code.

Branch Commands

```
# Create new branch
git branch feature-name

# Switch to branch
git checkout feature-name

# Create and switch (shortcut)
git checkout -b feature-name

# List all branches
git branch

# Merge branch into current
git merge feature-name
```

Merge Conflicts

When two people change the same lines:

```
<<<<<<< HEAD
Your local changes
=======
Changes from remote
>>>>>>> abc123def
```

Resolution:

1. Edit file – choose/combine changes
2. Remove conflict markers
3. `git add filename`
4. `git commit`

Git Best Practices

1. **Pull before starting** any development
2. Make **small, focused commits** with clear messages
3. **Push frequently** to backup your work
4. Use **branches** for new features
5. **Never commit sensitive data** (passwords, keys)
6. **Communicate** with team about file ownership

Warning

Once pushed, secrets are in history forever!

Command Reference

Command	Description
`git init`	Initialize repository
`git clone <url>`	Clone remote repo
`git status`	Check current status
`git add <file>`	Stage changes
`git commit -m "msg"`	Create commit
`git push`	Push to remote
`git pull`	Pull from remote
`git log`	View history

Questions?

Questions?