Minimal Git Workflow

Simple and effective git workflow for individual and team projects.

Repository and Worktree

- **Repository**: The Git database (history, branches, tags). Stored in `.git`.
- **Worktree**: The directory of files you edit. Your current branch's snapshot, plus any uncommitted changes.

Local and Remote Repository

- **Local**: Your machine. Full history and working copy live here.
- **Remote**: Hosted copy (e.g. GitHub). Shared source of truth; others push and pull from it.
- You sync between them with push and pull.

Core Concepts

- **Staging**: Mark changes to include in the next snapshot (`git add`).
- **Commit**: Save a snapshot of staged changes in local history (`git commit`).
- **Push**: Send your commits from local to remote (`git push`).
- **Pull**: Bring remote commits into your local repo and working tree (`git pull`).

Basic Workflow

1. Clone: `git clone <url>`
2. Branch: `git switch -c <branch-name>`
3. Make changes
4. Stage: `git add .` or `git add <file>`
5. Commit: `git commit -m "Descriptive message"`
6. Pull main, merge
7. Push: `git push -u origin <branch>`
8. Create Pull Request
9. Merge after approval

Branch Naming

- `feature/<name>` - New features
- `fix/<name>` - Bug fixes
- `docs/<name>` - Documentation

Commit Messages

- Start with verb (Add, Fix, Update)
- Be specific and concise
- Reference issues: "Fix login, closes #42"

GitHub: Existing Repository

```
git clone https://github.com/owner/repo.git
cd repo
git remote add upstream https://github.com/owner/repo.git
git remote -v
```

Sync with Upstream

```
git fetch upstream
git switch main
git merge upstream/main
git push origin main
```

Issue-Based Workflow

1. Create issue on GitHub (title, description, labels)
2. Create branch from issue (Development section)
3. Make changes, commit with "closes #42"
4. Push, create PR (auto-links to issue)
5. Address review, merge

Common Commands

Command	Description
`git status`	File status
`git diff`	Changes
`git log`	History
`git reset HEAD~1`	Undo last commit (keep changes)
`git stash`	Store changes
`git switch -`	Previous branch

Best Practices

- Commit often with clear messages
- Pull regularly to avoid conflicts
- Focused branches per feature/fix
- Start with issue first
- Reference issues in commits
- Use `git switch` not `git checkout` (Git 2.23+)