Git Tags and Stashing - Quick Guide

Git Tags

What Are Git Tags?

Git tags are markers pointing to specific commits, often used for versioning releases.

Types of Git Tags:

1. Lightweight Tag: A simple pointer to a commit.

2. Annotated Tag: A full object with metadata (preferred for releases).

Creating Tags:

- Annotated Tag: git tag -a v1.0 -m "Release 1.0"

- Lightweight Tag: git tag v1.0

Viewing Tags:

- List all tags: git tag

- List matching pattern: git tag -l "v1.*"

Checking Out a Tag:

- git checkout v1.0 (Note: Detached HEAD state)

Pushing Tags:

- Single tag: git push origin v1.0

- All tags: git push --tags

Deleting Tags:

- Local tag: git tag -d v1.0

- Remote tag: git push origin --delete tag v1.0

Tagging Past Commits:

- git tag -a v0.9 abc123 -m "Old release"

Git Tags and Stashing - Quick Guide

Use Case: Great for versioned releases in GitHub.

Git Stashing

What is Git Stashing?

Temporarily saves changes you don't want to commit yet.

Basic Usage:

- Stash all changes: git stash
- Stash with message: git stash push -m "message"
- View stashes: git stash list
- Apply latest stash: git stash apply
- Apply and remove stash: git stash pop

Stashing Specific Files:

- git stash push path/to/file.py

Managing Stashes:

- Drop specific stash: git stash drop stash@{0}
- Clear all stashes: git stash clear

Use Case:

Ideal for switching branches or pulling changes without committing work-in-progress.