TECH TIP — INTERMEDIATE

How to Use Git to Roll Back a Mistake

The undo button that actually works

From Nolan Law Firm — nemolegal.com/tools

Why This Matters

Git is version control — it tracks every change you make to your code. If you break something, delete the wrong file, or ship a bad update, Git lets you go back to any previous state. This is the most important thing Git does, and most people never learn it until they're in a panic.

First — Understand Where You Are

```
git status # what files have changed

git log --oneline # list recent commits with their hash (short ID)
```

Every commit has a hash — a short ID like `a3f2c1d`. You'll use these to navigate history.

Scenario 1 — Undo Changes Before You Committed

You edited a file and want to throw away your changes:

```
git checkout -- filename.py # discard changes to one file

git checkout -- . # discard ALL uncommitted changes (no undo)
```

> **Warning:** git checkout -- . is permanent. Uncommitted changes are gone. Make sure you mean it.

Scenario 2 — Undo the Last Commit (Keep the Changes)

You committed too soon but want to keep your work:

```
git reset HEAD~1 # undo last commit, keep files as they are
```

Scenario 3 — Undo the Last Commit (Throw Away Changes)

You want to fully erase the last commit and its changes:

```
git reset --hard HEAD~1 # undo last commit AND discard all its changes
```

Scenario 4 — Revert to a Specific Older Commit

Something broke three commits ago and you want to go back:

```
git log --oneline # find the commit hash you want

git revert a3f2c1d # safe: creates a new commit that undoes it


# OR — if you want to hard reset to that exact state:

git reset --hard a3f2c1d # dangerous: rewrites history
```

> **Safe vs. dangerous:** git revert is safe — it adds a new commit that undoes the change and preserves history. git reset --hard rewrites history. On a shared repo, never use reset --hard on commits that others have pulled.

Scenario 5 — Look at an Old Version Without Changing Anything

```
git show a3f2c1d:filename.py # see the old version of one file

git diff HEAD~3 HEAD # see what changed in last 3 commits
```

Quick Reference

Situation	Command
Discard unsaved changes in one file	git checkout -- filename
Discard ALL unsaved changes	git checkout -- .
Undo last commit, keep files	git reset HEAD~1
Undo last commit, delete files	git reset --hard HEAD~1
Safely undo a past commit	git revert <hash>
Hard reset to a past commit	git reset --hard <hash>
See history	git log --oneline
See old file version	git show <hash>:filename

Using AI to Help With This You don't have to fully understand this to use it. Here are prompts that work:

```
"I broke something in my Git repo. Here is git log --oneline: [paste]. I need to get back to how it was at [commit]. Walk me through the exact commands."
```

```
"I accidentally deleted a file and committed the deletion. How do I restore it?"
```

```
"Explain the difference between git revert and git reset --hard and which one I should use for my situation: [describe situation]."
```
