

GIT

- Git bash: terminal for git
- Git init: start working inside the current folder
- Git add -all/ -A instead of git add . because 'add .' does not push deletion of files- deleted files may come back on next pull

Cloning a remote repo into git:

```
PS C:\Users\dell\Desktop\git_sample> git clone https://github.com/su-shiii/git_sample.git
Cloning into 'git_sample'...
remote: Enumerating objects: 6, done.
remote: Counting objects: 100% (6/6), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 6 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (6/6), done.
```

```
▼ git_sample
  ≡ one.txt
  ≡ two.txt
```

Git status:

```
PS C:\Users\dell\Desktop\git_sample\git_sample> git status
On branch main
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   one.txt
        deleted:    two

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        two.txt

no changes added to commit (use "git add" and/or "git commit -a")
```

In a subfolder:

```

PS C:\Users\dell\Desktop\git_sample\git_sample> mkdir MyFolder

Directory: C:\Users\dell\Desktop\git_sample\git_sample

Mode                LastWriteTime         Length Name
----                -
d-----          1/12/2026  10:23 PM                MyFolder

PS C:\Users\dell\Desktop\git_sample\git_sample> cd MyFolder
PS C:\Users\dell\Desktop\git_sample\git_sample\MyFolder> New-Item three.txt

Directory: C:\Users\dell\Desktop\git_sample\git_sample\MyFolder

Mode                LastWriteTime         Length Name
----                -
-a-----          1/12/2026  10:24 PM              0 three.txt

```

- Adding changes:

```

PS C:\Users\dell\Desktop\git_sample\git_sample> git status
On branch main
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   one.txt
        deleted:    two

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        MyFolder/
        two.txt

no changes added to commit (use "git add" and/or "git commit -a")
PS C:\Users\dell\Desktop\git_sample\git_sample> git add -A
PS C:\Users\dell\Desktop\git_sample\git_sample> git status
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   MyFolder/three.txt
        modified:   one.txt
        renamed:    two -> two.txt

```

Git reset: rollback changes

```
● PS C:\Users\dell\Desktop\git_sample\git_sample> git reset
Unstaged changes after reset:
M    one.txt
D    two
● PS C:\Users\dell\Desktop\git_sample\git_sample> cd MyFolder
● PS C:\Users\dell\Desktop\git_sample\git_sample\MyFolder> git status
On branch main
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   ../one.txt
        deleted:    ../two

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        ./
        ../two.txt
```

Staging: git add commands

Changes made to 'one.txt' in main directory, and 'three.txt' in sub directory.

- 'add .' was used but it only stages changes in that current directory and its subdirectory
- 'add -A': works for the entire repo. Ps: doesn't stage deletions

```
● PS C:\Users\dell\Desktop\git_sample\git_sample> cd MyFolder
● PS C:\Users\dell\Desktop\git_sample\git_sample\MyFolder> git add .
● PS C:\Users\dell\Desktop\git_sample\git_sample\MyFolder> git status
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   three.txt

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   ../one.txt
```

- Git add *: also works only for current directory (no child directories), and doesn't stage deletions

Here we delete a file and create a new one. Only the new creation is supposed to be staged, not the deletion.

```
● PS C:\Users\dell\Desktop\git_sample\git_sample> git add *
● PS C:\Users\dell\Desktop\git_sample\git_sample> git status
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
(use "git push" to publish your local commits)

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   MyFolder/three.txt
        new file:   four.txt
        deleted:    one.txt
```

But! Turns it does stage the deletion??

Turns out git add * is shell dependent- since I'm using powershell, the '*' is interpreted as paths of everything inside the repo. And when paths are given, 'git add' stages deletions too, checking what is missing from the last time

- Git add *.txt: commits only txt files. Say I made a python and txt file- using this will stage only the txt files.

```
● PS C:\Users\dell\Desktop\git_sample\git_sample> git add *.txt
● PS C:\Users\dell\Desktop\git_sample\git_sample> git status
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
(use "git push" to publish your local commits)

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   MyFolder/three.txt
        new file:   five.txt
        new file:   four.txt
        deleted:    one.txt

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        a.py
```

Commit: to save it to the local repo

```

● PS C:\Users\dell\Desktop\git_sample\git_sample> git commit -m "part 2: commits"
[main d724997] part 2: commits
 5 files changed, 4 insertions(+), 2 deletions(-)
 create mode 100644 a.py
 create mode 100644 five.txt
 create mode 100644 four.txt
 delete mode 100644 one.txt
● PS C:\Users\dell\Desktop\git_sample\git_sample> git status
On branch main
Your branch is ahead of 'origin/main' by 2 commits.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean

```

The working tree is clean and the local repo has changes saved. Now we need to push it to the remote repository.

Reset: git reset HEAD~ will bring everything back to the working directory and then you have to stage and commit them again.

```

● PS C:\Users\dell\Desktop\git_sample\git_sample> git reset HEAD~
Unstaged changes after reset:
 M   MyFolder/three.txt
 D   one.txt
● PS C:\Users\dell\Desktop\git_sample\git_sample> git status
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)

Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
       modified:   MyFolder/three.txt
       deleted:    one.txt

Untracked files:
  (use "git add <file>..." to include in what will be committed)
       a.py
       five.txt
       four.txt

```

Git rm- removes a file and stages that deletion in one go

```

● PS C:\Users\dell\Desktop\git_sample\git_sample> git rm -f a.py
rm 'a.py'

```

Git log: we can see our commit history

```
PS C:\Users\dell\Desktop\git_sample\git_sample> git log --oneline
a3064db (HEAD -> main) Test commit
7ae176e (origin/main, origin/HEAD) Second commit
02d148d Initial commit

PS C:\Users\dell\Desktop\git_sample\git_sample> git log
commit a3064db91ff675886eb9f6585ed558e5d99f3366 (HEAD -> main)
Author: sushiii <suhanazrin.s3@gmail.com>
Date: Tue Jan 13 10:22:08 2026 +0530

    Test commit

commit 7ae176eb2981db6d9df333909187b825a5512009 (origin/main, origin/HEAD)
Author: Suha Nazrin <suhanazrin.s3@gmail.com>
Date: Mon Jan 12 22:18:00 2026 +0530

    Second commit

commit 02d148d5b940684b7738d667e1e1ea8e7cdb011f
Author: Suha Nazrin <suhanazrin.s3@gmail.com>
Date: Mon Jan 12 22:15:09 2026 +0530

    Initial commit
```

Branching:

```
PS C:\Users\dell\Desktop\git_sample\git_sample> git branch
* main

PS C:\Users\dell\Desktop\git_sample\git_sample> git branch branch1
PS C:\Users\dell\Desktop\git_sample\git_sample> git branch
branch1
* main

PS C:\Users\dell\Desktop\git_sample\git_sample> git checkout branch1
Switched to branch 'branch1'

PS C:\Users\dell\Desktop\git_sample\git_sample> git status
On branch branch1
nothing to commit, working tree clean

PS C:\Users\dell\Desktop\git_sample\git_sample> git status
On branch branch1
Untracked files:
  (use "git add <file>..." to include in what will be committed)
    branch_test.txt

nothing added to commit but untracked files present (use "git add" to track)

PS C:\Users\dell\Desktop\git_sample\git_sample> git add .
PS C:\Users\dell\Desktop\git_sample\git_sample> git commit -m "Add file to branch"
[branch1 ab6fc61] Add file to branch
1 file changed, 1 insertion(+)
create mode 100644 branch_test.txt

PS C:\Users\dell\Desktop\git_sample\git_sample> git status
On branch branch1
nothing to commit, working tree clean

PS C:\Users\dell\Desktop\git_sample\git_sample> git checkout main
Switched to branch 'main'
Your branch is ahead of 'origin/main' by 1 commit.
(use "git push" to publish your local commits)
```

Merging branches:

- First we create a new file six.txt to main, then we try to merge main to branch1
- We switch to main, then merge branch1 with main (no changes from prev step)

```
PS C:\Users\dell\Desktop\git_sample\git_sample> git checkout branch1
Switched to branch 'branch1'
PS C:\Users\dell\Desktop\git_sample\git_sample> git merge main -m "Merging main into branch1"
Merge made by the 'ort' strategy.
 six.txt | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 six.txt
PS C:\Users\dell\Desktop\git_sample\git_sample> git checkout main
Switched to branch 'main'
Your branch is ahead of 'origin/main' by 2 commits.
(use "git push" to publish your local commits)
PS C:\Users\dell\Desktop\git_sample\git_sample> git merge -m "Merging branch1 with main"
Already up to date.
```

Merge conflict:

- Changed the content of 'one.txt' in two branches- branch1 and branch2- then tried to merge

```
PS C:\Users\dell\Desktop\git_sample\git_sample> git checkout branch1
Switched to branch 'branch1'
PS C:\Users\dell\Desktop\git_sample\git_sample> git status
On branch branch1
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   one.txt

no changes added to commit (use "git add" and/or "git commit -a")
PS C:\Users\dell\Desktop\git_sample\git_sample> git add .
PS C:\Users\dell\Desktop\git_sample\git_sample> git commit -m "Changed content of branch1"
[branch1 c0907c3] Changed content of branch1
 1 file changed, 1 insertion(+), 1 deletion(-)
PS C:\Users\dell\Desktop\git_sample\git_sample> git status
On branch branch1
nothing to commit, working tree clean
PS C:\Users\dell\Desktop\git_sample\git_sample> git checkout branch2
Switched to branch 'branch2'
PS C:\Users\dell\Desktop\git_sample\git_sample> git status
On branch branch2
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   one.txt

no changes added to commit (use "git add" and/or "git commit -a")
PS C:\Users\dell\Desktop\git_sample\git_sample> git add .
PS C:\Users\dell\Desktop\git_sample\git_sample> git commit -m "Changed content of branch2"
[branch2 a519416] Changed content of branch2
 1 file changed, 1 insertion(+), 1 deletion(-)
PS C:\Users\dell\Desktop\git_sample\git_sample> git merge branch1
Auto-merging one.txt
CONFLICT (content): Merge conflict in one.txt
Automatic merge failed; fix conflicts and then commit the result.
```

Fixed by manually fixing the merge conflict:

```

PS C:\Users\dell\Desktop\git_sample\git_sample> git add .
PS C:\Users\dell\Desktop\git_sample\git_sample> git commit -m "Merge conflict solved"
[branch2 57c5894] Merge conflict solved
PS C:\Users\dell\Desktop\git_sample\git_sample> git branch
  branch1
* branch2
  main
PS C:\Users\dell\Desktop\git_sample\git_sample> git checkout branch1
Switched to branch 'branch1'
PS C:\Users\dell\Desktop\git_sample\git_sample> git merge branch2
Updating c0907c3..57c5894
Fast-forward
 one.txt | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)

```

Switch between commits:

```

PS C:\Users\dell\Desktop\git_sample\git_sample> git log --oneline
57c5894 (HEAD -> main, branch2, branch1) Merge conflict solved
a519416 Changed content of branch2
c0907c3 Changed content of branch1
e2a8369 Merge branch 'branch1' into branch2
152ea66 Changed one.txt
3f32f06 Merging main into branch1
8249f5e w
ab6fc61 Add file to branch
a3064db Test commit
7ae176e (origin/main, origin/HEAD) Second commit
You are in 'detached HEAD' state. You can look around, make experimental
changes and commit them, and you can discard any commits you make in this
state without impacting any branches by switching back to a branch.

```

If you want to create a new branch to retain commits you create, you may do so (now or later) by using -c with the switch command. Example:

```
git switch -c <new-branch-name>
```

Or undo this operation with:

```
git switch -
```

Turn off this advice by setting config variable advice.detachedHead to false

```

HEAD is now at ab6fc61 Add file to branch
PS C:\Users\dell\Desktop\git_sample\git_sample> git checkout main
Previous HEAD position was ab6fc61 Add file to branch
Switched to branch 'main'
Your branch is ahead of 'origin/main' by 9 commits.
(use "git push" to publish your local commits)

```

Show changes between 2 commits:

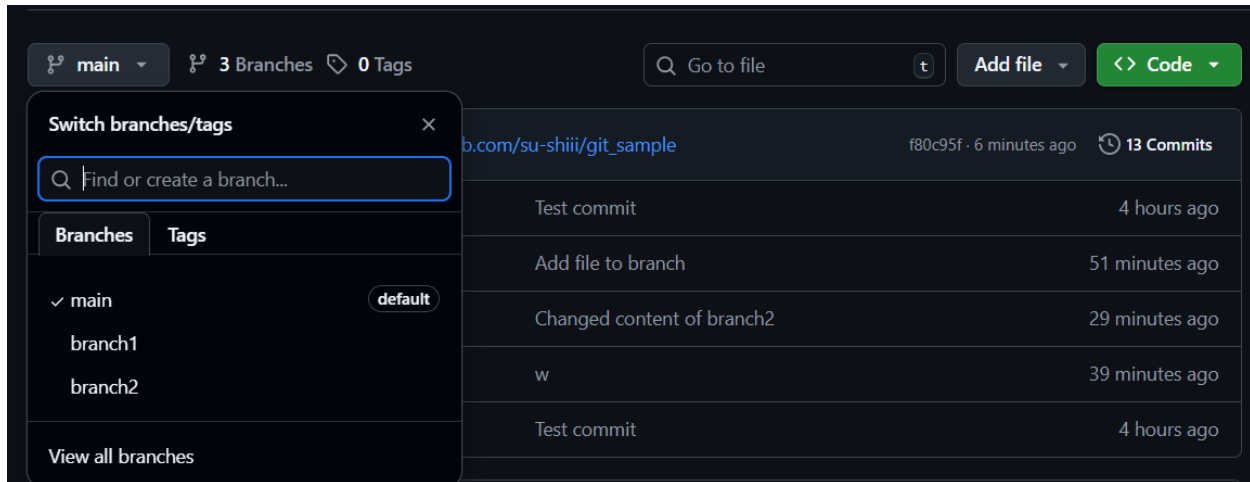
```
PS C:\Users\dell\Desktop\git_sample\git_sample> git diff ab6fc61 57c5894
diff --git a/one.txt b/one.txt
index 5626abf..6f6b544 100644
--- a/one.txt
+++ b/one.txt
@@ -1,1 @@
-one
+meow meow
diff --git a/six.txt b/six.txt
new file mode 100644
index 0000000..b8bbeaf
--- /dev/null
+++ b/six.txt
@@ -0,0 +1 @@
+seis
\ No newline at end of file
```

Push:

Pushed main and then the other side branches to the repository.

```
PS C:\Users\dell\Desktop\git_sample\git_sample> git push origin main
Enumerating objects: 29, done.
Counting objects: 100% (28/28), done.
Delta compression using up to 20 threads
Compressing objects: 100% (18/18), done.
Writing objects: 100% (25/25), 2.01 KiB | 687.00 KiB/s, done.
Total 25 (delta 11), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (11/11), done.
To https://github.com/su-shiii/git_sample.git
 18fd08d..f80c95f main -> main
PS C:\Users\dell\Desktop\git_sample\git_sample> git checkout branch1
Switched to branch 'branch1'
PS C:\Users\dell\Desktop\git_sample\git_sample> git push origin branch1
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'branch1' on GitHub by visiting:
remote:   https://github.com/su-shiii/git_sample/pull/new/branch1
remote:
To https://github.com/su-shiii/git_sample.git
 * [new branch]      branch1 -> branch1
PS C:\Users\dell\Desktop\git_sample\git_sample> git checkout branch2
Switched to branch 'branch2'
PS C:\Users\dell\Desktop\git_sample\git_sample> git push origin branch2
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'branch2' on GitHub by visiting:
remote:   https://github.com/su-shiii/git_sample/pull/new/branch2
remote:
To https://github.com/su-shiii/git_sample.git
 * [new branch]      branch2 -> branch2
```

Repo:



Fetch:

First a change was made to the repo directly, then it was fetched, then merged.

```
PS C:\Users\dell\Desktop\git_sample\git_sample> git fetch
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
Unpacking objects: 100% (3/3), 904 bytes | 129.00 KiB/s, done.
From https://github.com/su-shiii/git_sample
   f80c95f..e83938d  main       -> origin/main
```

The above command just fetched, didn't merge yet.

```
PS C:\Users\dell\Desktop\git_sample\git_sample> git merge
Updating f80c95f..e83938d
Fast-forward
 two.txt | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)
```

Pull: git fetch + merge at the same time.

```
PS C:\Users\dell\Desktop\git_sample\git_sample> git status
On branch main
Your branch is behind 'origin/main' by 1 commit, and can be fast-forwarded.
  (use "git pull" to update your local branch)

nothing to commit, working tree clean
PS C:\Users\dell\Desktop\git_sample\git_sample> git pull
Updating e83938d..951a27f
Fast-forward
 seven.txt | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 seven.txt
PS C:\Users\dell\Desktop\git_sample\git_sample> git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
```

Restore: get the repo back to the previous commit

Stash: temporarily set aside your work in one branch and switch to another.

Pop: to return those files

```
PS C:\Users\dell\Desktop\git_sample\git_sample> git stash
Saved working directory and index state WIP on main: 951a27f Create seven.txt
PS C:\Users\dell\Desktop\git_sample\git_sample> git stash pop
On branch main
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   six.txt

no changes added to commit (use "git add" and/or "git commit -a")
Dropped refs/stash@{0} (2cf6f305996a8a2f5bc5fbf57520c704d5a85486)
```

List of stashes:

```
● PS C:\Users\dell\Desktop\git_sample\git_sample> git stash list
stash@{0}: WIP on branch2: 57c5894 Merge conflict solved
stash@{1}: WIP on branch1: 57c5894 Merge conflict solved
```

Revert: remove the effects of a previous commit- except it doesn't pull back, it creates a new commit which does the job of pulling it back to a previous version

```
PS C:\Users\dell\Desktop\git_sample\git_sample> git revert e83938d
[main 5795851] Revert "Update two.txt"
1 file changed, 1 insertion(+), 1 deletion(-)
```

Rebase:

```
PS C:\Users\dell\Desktop\git_sample\git_sample> git checkout main
Switched to branch 'main'
Your branch is ahead of 'origin/main' by 1 commit.
(use "git push" to publish your local commits)
PS C:\Users\dell\Desktop\git_sample\git_sample> git add .
PS C:\Users\dell\Desktop\git_sample\git_sample> git commit -m "rebase 3"
[main c367e4e] rebase 3
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 nine.txt
PS C:\Users\dell\Desktop\git_sample\git_sample> git checkout feature
Switched to branch 'feature'
PS C:\Users\dell\Desktop\git_sample\git_sample> git rebase main
Successfully rebased and updated refs/heads/feature.
```