Namit Rampal

SAP ID – 500123236

Batch – B2 DevOps

Lab Exercise 2- Working with Git Reset

Lab Exercise: Git Reset

This lab exercise will guide you through the usage of the git reset command in various scenarios. The git reset command is used to undo changes in the Git history, working directory, or staging area. There are three main modes: **soft**, **mixed**, and **hard**.

Objective

- Learn how to use git reset to modify the commit history, unstage files, or discard changes.
- Understand the differences between --soft, --mixed, and --hard reset modes.

Prerequisites

1. Install Git on your system.

2. Set up a Git repository:

```
git init git-reset-lab

cd git-reset-lab
```

Steps

1. Set Up the Repository

1. Create and commit an initial file:

```
echo "Line 1" > file.txt

git add file.txt

git commit -m "Initial commit: Add Line 1"
```

2. Add a second change:

```
echo "Line 2" >> file.txt

git commit -am "Add Line 2"
```

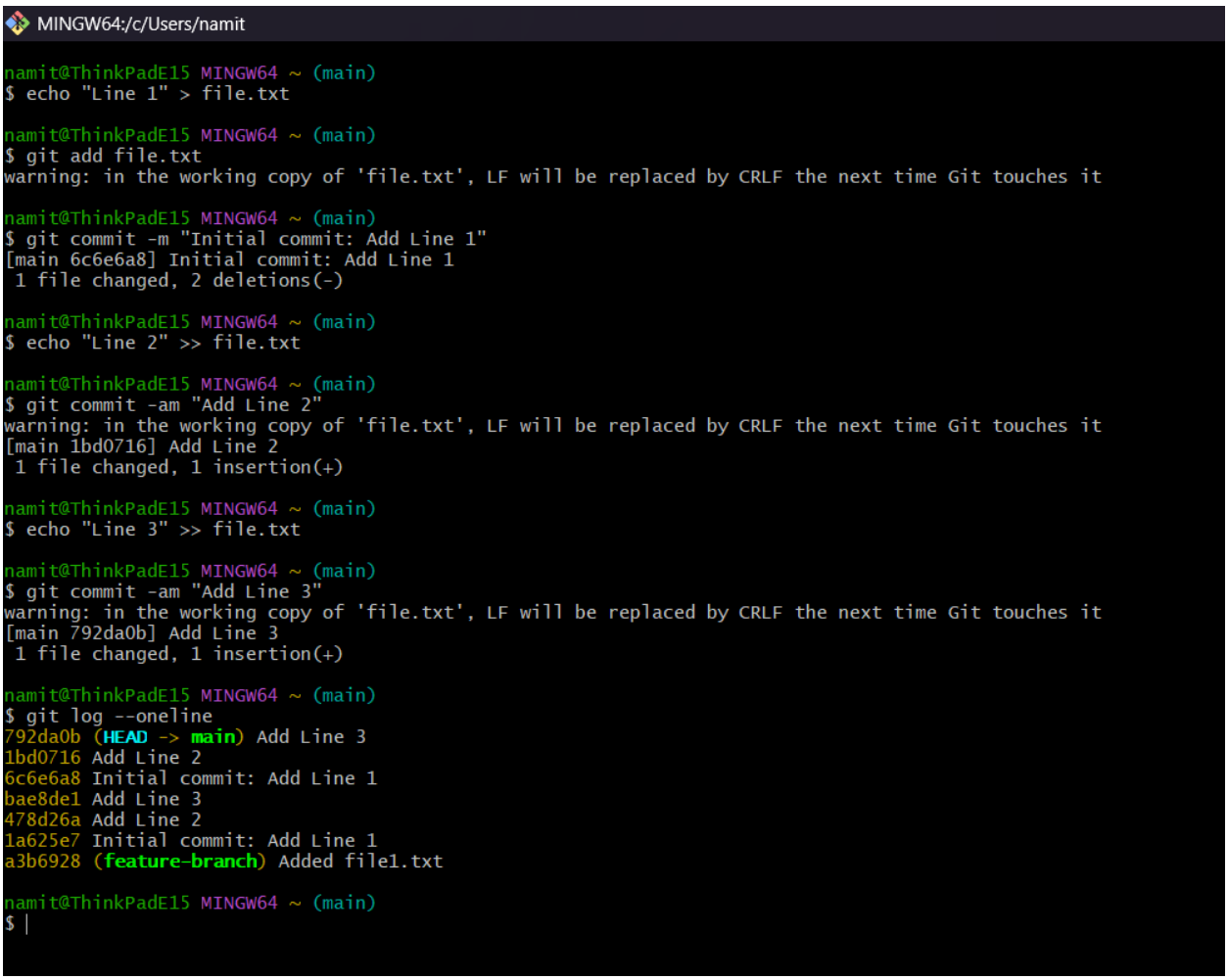
3. Add a third change:

```
echo "Line 3" >> file.txt

git commit -am "Add Line 3"
```

4. Check the commit history:

```
git log --oneline
```

Output:



2. Use git reset --soft

This mode moves the HEAD pointer to an earlier commit but keeps the changes in the staging area.

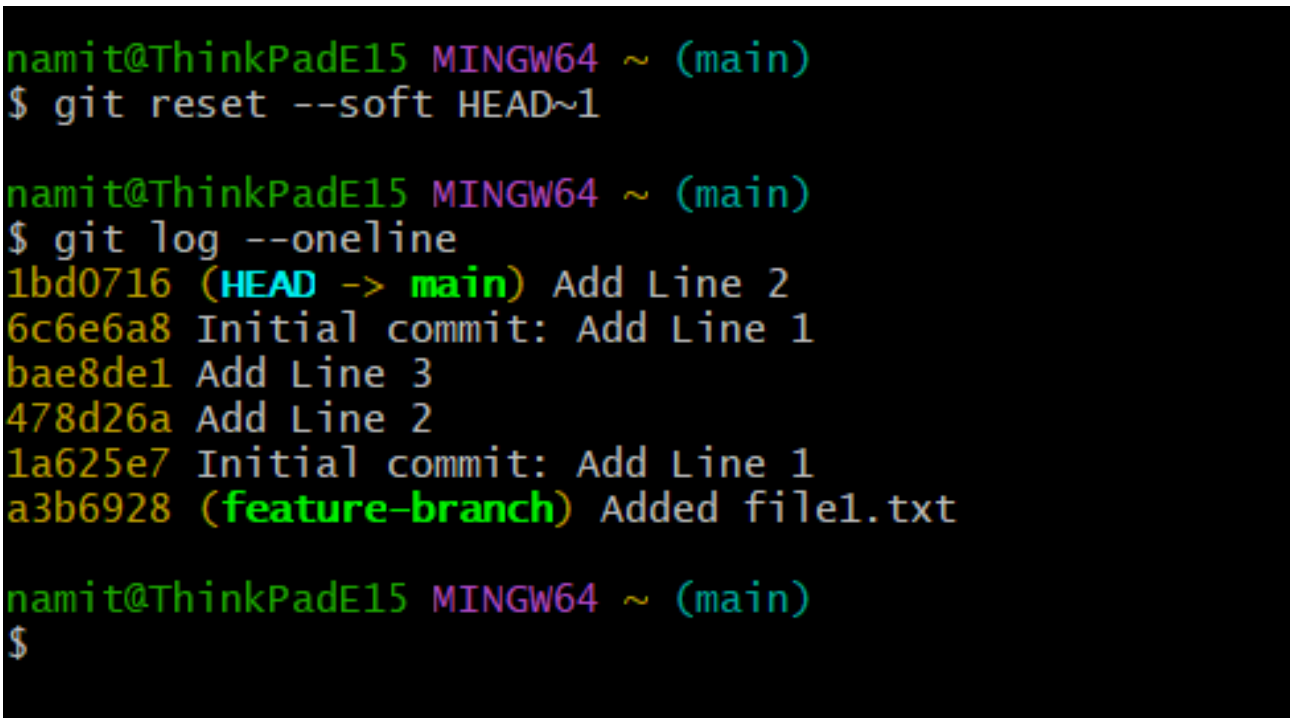
1. Reset to the second commit:

```
git reset --soft HEAD~1
```

2. Check the commit history:

```
git log --oneline
```

Output:



3. Verify the staged changes:

```
git status
```

Output:

```
Changes to be committed:
```

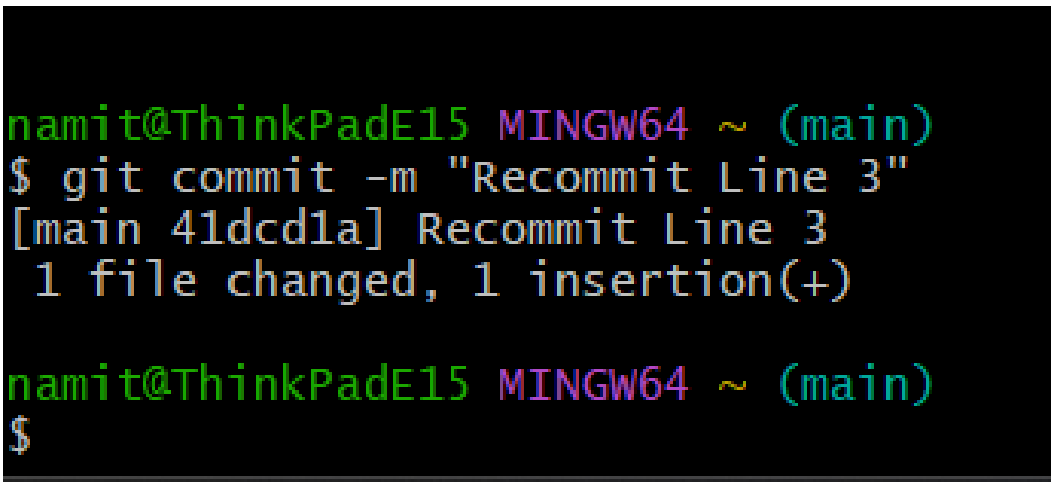
```
modified: file.txt
```

```
namit@ThinkPadE15 MINGW64 ~ (main)
$ git status
warning: could not open directory 'Application Data/': Permission denied
warning: could not open directory 'Cookies/': Permission denied
warning: could not open directory 'Local Settings/': Permission denied
warning: could not open directory 'My Documents/': Permission denied
warning: could not open directory 'NetHood/': Permission denied
warning: could not open directory 'PrintHood/': Permission denied
warning: could not open directory 'Recent/': Permission denied
warning: could not open directory 'SendTo/': Permission denied
warning: could not open directory 'Start Menu/': Permission denied
warning: could not open directory 'Templates/': Permission denied
On branch main
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   file.txt

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        .VirtualBox/
        .anaconda/
        .bash_history
        .conda/
        .condarc
        .continuum/
        .eclipse/
        .gitconfig
        .gitignore
        .idlerc/
        .ipynb_checkpoints/
        .ipython/
        .jupyter/
        .lesshst
        .m2/
        .matplotlib/
        .nbi/
        .p2/
        .packettracer
        .viminfo
        .vscode/
        .zenmap/
        AppData/
        BrawlhallaReplays/
        Cisco Packet Tracer 8.2.2/
        Contacts/
        Documents/
        Downloads/
        Exp_6/
        Favorites/
        Links/
        Music/
        NTUSER.DAT
        NTUSER.DAT{685ad9db-2747-11f0-afc7-dcba70cb53ef}.TM.blf
```

4. If needed, re-commit the changes:

```
git commit -m "Recommit Line 3"
```

Output:



3. Use git reset --mixed

This mode moves the HEAD pointer and unstages the changes but keeps them in the working directory.

1. Reset to the first commit:

```
git reset --mixed HEAD~1
```

2. Check the commit history:

```
git log --oneline
```

Output:

```
namit@ThinkPadE15 MINGW64 ~ (main)
$ git reset --mixed HEAD~1
Unstaged changes after reset:
M       file.txt

namit@ThinkPadE15 MINGW64 ~ (main)
$ git log --oneline
1bd0716 (HEAD -> main) Add Line 2
6c6e6a8 Initial commit: Add Line 1
bae8de1 Add Line 3
478d26a Add Line 2
1a625e7 Initial commit: Add Line 1
a3b6928 (feature-branch) Added file1.txt

namit@ThinkPadE15 MINGW64 ~ (main)
$
```

3. Verify the changes in the working directory:

```
git status
```

Output:

```
namit@ThinkPadE15 MINGW64 ~ (main)
$ git status
warning: could not open directory 'Application Data/': Permission denied
warning: could not open directory 'Cookies/': Permission denied
warning: could not open directory 'Local Settings/': Permission denied
warning: could not open directory 'My Documents/': Permission denied
warning: could not open directory 'NetHood/': Permission denied
warning: could not open directory 'PrintHood/': Permission denied
warning: could not open directory 'Recent/': Permission denied
warning: could not open directory 'SendTo/': Permission denied
warning: could not open directory 'Start Menu/': Permission denied
warning: could not open directory 'Templates/': Permission denied
On branch 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:   file.txt
```

4. If needed, stage and re-commit:

```
git add file.txt

git commit -m "Recommit Line 2 and Line 3"
```

---

4. Use git reset --hard

This mode moves the HEAD pointer and discards all changes in the staging area and working directory.

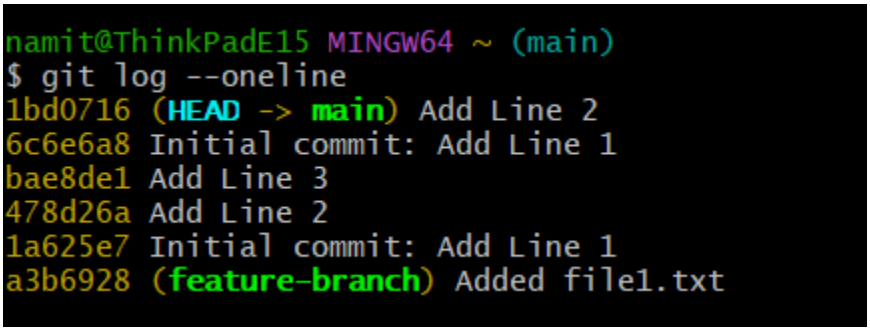
1. Reset to the initial commit:

```
git reset --hard HEAD~1
```

2. Check the commit history:

```
git log --oneline
```

Output:



3. Verify the working directory:

```
cat file.txt
```

```
namit@ThinkPadE15 MINGW64 ~ (main)
$ cat file.txt
Line 1


namit@ThinkPadE15 MINGW64 ~ (main)
$
```

---

5. Use git reset with a Commit Hash

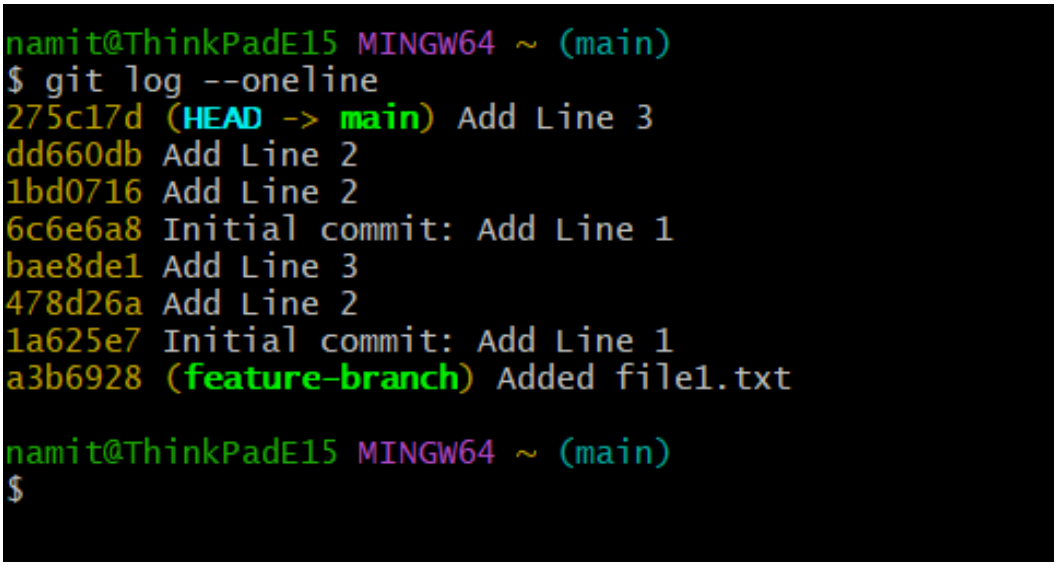
1. Add some changes for demonstration:

```
echo "Line 2" >> file.txt

git commit -am "Add Line 2"

echo "Line 3" >> file.txt

git commit -am "Add Line 3"
```

```
namit@ThinkPadE15 MINGW64 ~ (main)
$ echo "Line 2" >> file.txt
git commit -am "Add Line 2"
echo "Line 3" >> file.txt
git commit -am "Add Line 3"
warning: in the working copy of 'file.txt', LF will be replaced by CRLF the next time Git touches it
[main dd660db] Add Line 2
 1 file changed, 1 insertion(+)
warning: in the working copy of 'file.txt', LF will be replaced by CRLF the next time Git touches it
[main 275c17d] Add Line 3
 1 file changed, 1 insertion(+)

namit@ThinkPadE15 MINGW64 ~ (main)
$
```

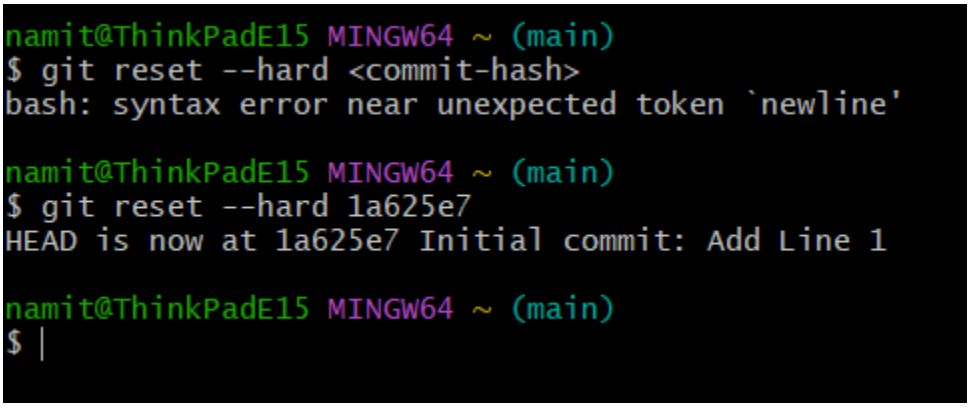
2. Get the commit hash for the initial commit:

```
git log --oneline
```



3. Reset to the initial commit using the hash:

```
git reset --hard <commit-hash>
```



4. Verify the working directory and commit history:

```
git log --oneline
```

```
cat file.txt
```

```
namit@ThinkPadE15 MINGW64 ~ (main)
$ git log --oneline
1a625e7 (HEAD -> main) Initial commit: Add Line 1
a3b6928 (feature-branch) Added file1.txt

namit@ThinkPadE15 MINGW64 ~ (main)
$
```

```
namit@ThinkPadE15 MINGW64 ~ (main)
$ cat file.txt
Line 1

namit@ThinkPadE15 MINGW64 ~ (main)
```

Summary of Commands

Mode	Effect	Command Example
--soft	Moves HEAD, keeps changes staged.	git reset --soft HEAD~1
--mixed	Moves HEAD, unstages changes, keeps them in working dir.	git reset --mixed HEAD~1

Mode	Effect	Command Example
--hard	Moves HEAD, discards all changes in staging and working dir.	git reset --hard HEAD~1