CI:

Continuous integration is process of Automated Build and Automated Tests.

Git Repository:

Git Repository contains Files, History, Config Management by GIT,

.git folder directory will create in Project it is called a git repository.

ls -a - to see hiddenn file

Stage of GIT:

Working Directory

Staging Area

Git Directory

Working Directory: Area of live files, also known as Untracked area of GIT.

Staging Area: Staging Area is when git starts tracking and saving changes that occur in files.

Git Directory/Repository: Also called 'Local Repo', is your .git repo. It's area where GIT save everything.

Remote Repository GITHub:

Remote repositry is stored on a code hosting service like GitHub or on an internal server.

Branch in GIT: Branch in Git is a way to keep developing and coding a new feature or modification to the software and still not affecting the main part of the project.

GIT: is a free and open source version control system.

Version Control System: VCS helps a software team manage changes to source code over time.

Version control software keeps track of every modification to the code in a special kind of database.

Types of VCS:

Centralized Version Control System (CVCS):

Uses a central server to store all files and enables team collaboration.

Works on a single repository to which users can directly access a central server.

Everyone will have their own working copy they will connect to central Repository.

Distributed Version Control System:

Everyone will have their clone copy of Repository on their local machine.

In Distributed VCS, every contributor has alocal copy or clone of the main repository.

User can change and commit local repo without any interfarence.

User can update thier local Repo from the Central Server.

User can update the central server from their repo.

**Installation steps for windows git installation:**

Install GIT on Windows : Text Guide

This guide will show you how to install Git on Windows.

Prerequisites

Administrator privileges

Access to a command-line

Internet Access

Download Git for Windows

1. Browse to the official Git website: https://git-scm.com/downloads

2. Click the download link for Windows and allow the download to complete.

Extract and Launch Git Installer

3. Browse to the download location (or use the download shortcut in your browser). Double-click the file to extract and launch the installer.

4. Allow the app to make changes to your device by clicking Yes on the User Account Control dialog that opens.

5. Review the GNU General Public License, and when you’re ready to install, click Next.

6. The installer will ask you for an installation location. Leave the default, unless you have reason to change it, and click Next.

7. A component selection screen will appear. Leave the defaults unless you have a specific need to change them and click Next.

8. The installer will offer to create a start menu folder. Simply click Next.

9. Select a text editor you’d like to use with Git. Go with the 'Vim' option.

10. This installation step allows you to change the PATH environment. The PATH is the default set of directories included when you run a command from the command line.

Choose Option "Use Git from Git Bash only".

11. The next option relates to server certificates. Most users should use the default. If you’re working in an Active Directory environment, you may need to switch to Windows Store certificates. Click Next.

12. The next selection converts line endings. It is recommended that you leave the default selection. This relates to the way data is formatted and changing this option may cause problems. Click Next.

13. Choose the terminal emulator you want to use. The default MinTTY is recommended, for its features. Click Next.

14. The default options are recommended, however, this step allows you to decide which extra option you would like to enable. If you use symbolic links, which are like shortcuts for the command line, tick the box. Click Next.

15. Depending on the version of Git you’re installing, it may offer to install experimental features. At the time this article was written, the option to include interactive options was offered. Unless you are feeling adventurous, leave them unchecked and click Install.

16. Once the installation is complete, tick the boxes to view the Release Notes or Launch Git Bash, then click Finish.

**How to Launch Git in Windows:**

To launch Git Bash open the Windows Start menu, type git bash, and press Enter (or click the application icon).

Now on Git bash, you can execute any GIT Command. To verify the version, execute the command-

git --version

----------------------------------------------------

Configure Global User for Git:

git config --global user.name "Sanjay Jadhav"

git config --global user.email "spjadhav78@gmail.com"

git config --list

----------------------------------------------------

Create Git Repository and initialise it:

Create Folder called gittraining\_v1 in user profile

pwd

12 ll

13 ll

14 cd gittraining\_v1/x

15 pwd

16 git init

17 ll

18 ls -lrt

19 ls -a

20 cd .git

21 ls -a

22 cd ..

23 pwd

**Commit Code in GIT:**

Commit is used to move your files information (not actual files)from your local working folder i.e. gittraining\_v1 to the staging area.

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/gittraining\_v1 (master)

$ git status

On branch master

No commits yet

Untracked files:

(use "git add <file>..." to include in what will be committed)

first\_file.txt

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

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/gittraining\_v1 (master)

$ git status

On branch master

No commits yet

Untracked files:

(use "git add <file>..." to include in what will be committed)

first\_file.txt

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

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/gittraining\_v1 (master)

$ git add first\_file.txt

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/gittraining\_v1 (master)

$ git status

On branch master

No commits yet

Changes to be committed:

(use "git rm --cached <file>..." to unstage)

new file: first\_file.txt

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/gittraining\_v1 (master)

$ git add .

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/gittraining\_v1 (master)

$ git status

On branch master

No commits yet

Changes to be committed:

(use "git rm --cached <file>..." to unstage)

new file: first\_file.txt

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/gittraining\_v1 (master)

$ git commit -m "First File Commit"

[master (root-commit) 8e6d6a2] First File Commit

1 file changed, 1 insertion(+)

create mode 100644 first\_file.txt

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/gittraining\_v1 (master)

$ git status

On branch master

nothing to commit, working tree clean

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/gittraining\_v1 (master)

$ git status

On branch master

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: first\_file.txt

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

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/gittraining\_v1 (master)

$ git diff

diff --git a/first\_file.txt b/first\_file.txt

index eee2f13..3f707ec 100644

--- a/first\_file.txt

+++ b/first\_file.txt

@@ -1 +1,2 @@

-this is my first file

\ No newline at end of file

+this is my first file

+be the change you want to see

\ No newline at end of file

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/gittraining\_v1 (master)

$ git commit -m "First File Commit edited"

On branch master

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: first\_file.txt

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

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/gittraining\_v1 (master)

$ git add .

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/gittraining\_v1 (master)

$ git commit -m "First File Commit edited"

[master e31451e] First File Commit edited

1 file changed, 2 insertions(+), 1 deletion(-)

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/gittraining\_v1 (master)

$ git status

On branch master

nothing to commit, working tree clean

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/gittraining\_v1 (master)

$

Git Tracking & Git Logs:

$ touch my\_secondfile.txt

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/gittraining\_v1 (master)

$ ll

total 1

-rw-r--r-- 1 spjad 197609 52 Mar 11 21:20 first\_file.txt

-rw-r--r-- 1 spjad 197609 0 Mar 11 21:39 my\_secondfile.txt

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/gittraining\_v1 (master)

$ vi first\_file.txt

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/gittraining\_v1 (master)

$ git status

On branch master

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: first\_file.txt

Untracked files:

(use "git add <file>..." to include in what will be committed)

my\_secondfile.txt

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

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/gittraining\_v1 (master)

$ git add .

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/gittraining\_v1 (master)

$ git status

On branch master

Changes to be committed:

(use "git restore --staged <file>..." to unstage)

modified: first\_file.txt

new file: my\_secondfile.txt

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/gittraining\_v1 (master)

$ git commit -m "My another commit for second file"

[master b1490ec] My another commit for second file

2 files changed, 2 insertions(+), 1 deletion(-)

create mode 100644 my\_secondfile.txt

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/gittraining\_v1 (master)

$ git log

commit b1490ec5d8604ea57ca86210af3e03a6bd088a05 (HEAD -> master)

Author: Sanjay Jadhav <spjadhav78@gmail.com>

Date: Thu Mar 11 21:45:15 2021 +0530

My another commit for second file

commit e31451e0b4f5b9cb4d2ea7c4f487b8fe09d2452f

Author: Sanjay Jadhav <spjadhav78@gmail.com>

Date: Thu Mar 11 21:35:17 2021 +0530

First File Commit edited

commit 8e6d6a21f0b9a2c222c70d01d63771c1d67dfa42

Author: Sanjay Jadhav <spjadhav78@gmail.com>

Date: Thu Mar 11 21:19:30 2021 +0530

First File Commit

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/gittraining\_v1 (master)

$ git log --author="Sanjay Jadhav"

commit b1490ec5d8604ea57ca86210af3e03a6bd088a05 (HEAD -> master)

Author: Sanjay Jadhav <spjadhav78@gmail.com>

Date: Thu Mar 11 21:45:15 2021 +0530

My another commit for second file

commit e31451e0b4f5b9cb4d2ea7c4f487b8fe09d2452f

Author: Sanjay Jadhav <spjadhav78@gmail.com>

Date: Thu Mar 11 21:35:17 2021 +0530

First File Commit edited

commit 8e6d6a21f0b9a2c222c70d01d63771c1d67dfa42

Author: Sanjay Jadhav <spjadhav78@gmail.com>

Date: Thu Mar 11 21:19:30 2021 +0530

First File Commit

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/gittraining\_v1 (master)

$

$ history

1 git --version

2 git help

3 git help pull

4 git help -pull

5 git config --global user

6 git config --global user.name"Sanjay Jadhav"

7 git config --global user.name "Sanjay Jadhav"

8 git config --global user.email "spjadhav78@gmail.com"

9 git config --list

10 history

11 pwd

12 ll

13 ll

14 cd gittraining\_v1/

15 pwd

16 git init

17 ll

18 ls -lrt

19 ls -a

20 cd .git

21 ls -a

22 cd ..

23 pwd

24 history

25 ll

26 git commit

27 git status

28 git status

29 git add first\_file.txt

30 git status

31 git add .

32 git status

33 git commit -m "First File Commit"

34 git status

35 git status

36 git diff

37 git commit -m "First File Commit edited"

38 git add .

39 git commit -m "First File Commit edited"

40 git status

41 git status

42 touch my\_secondfile.txt

43 ll

44 vi first\_file.txt

45 git status

46 git add .

47 git status

48 git commit -m "My another commit for second file"

49 git log

50 git log --author="Sanjay Jadhav"

51 history

52 clear

53 pwd

54 cd ..

55 pwd

56 clear

57 git clone -b main https://github.com/spjadhav78/git\_training\_v3.git

58 ll

59 cd git\_training\_v3

60 ll

61 ls -a

62 git status

63 git pull

64 ll

65 vi testfile.txt

66 git status

67 git commit -m "Commit Message 2"

68 git commit -am "It will add file as well will Commit Message 2 with -am"

69 git push

70 exit

71 git status

72 git --version

73 pwd

Create Git Repo On Github.com

https://github.com/spjadhav78/git\_training\_v3.git

Configure Git Authentication with SSH Key

https://docs.github.com/en/github/authenticating-to-github/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent

ssh-keygen -t ed25519 -C "spjadhav78@gmail.com"

ssh-keygen -t rsa -b 4096 -C "spjadhav78@gmail.com"

-------------------

Here is what I performed step by step (step 2 onwards on GitBash):

Cleaned up my .ssh folder at C:\user\<username>\.ssh\

Generated a new SSH key

$ ssh-keygen -t rsa -b 4096 -C "xyz@abc.com"

Check if any process id(ssh agent) is already running.

$ ps aux | grep ssh

(Optional) If found any in step 3, kill those

$ kill <pids>

Started the ssh agent

$ eval `ssh-agent -s`

Added ssh key generated in step 2 to ssh agent

$ ssh-add ~/.ssh/id\_rsa

cd git\_training\_v3

77 git status

78 ll

79 history

80 ssh-keygen -t ed25519 -C "spjadhav78@gmail.com"

81 ll

82 ssh-keygen -t ed25519 -C "spjadhav78@gmail.com"

83 ssh-keygen -t ed25519 -C "spjadhav78@gmail.com"

84 ll

85 rm -a sshkeyfile.pub

86 rm -f sshkeyfile.pub

87 ll

88 rm -f sshkeyfile

89 ll

90 eval ssh-agent -s

91 ssh-add ~/.ssh/sshkey

92 clip < ~/.ssh/sshkey.pub

93 ssh-add ~/.ssh/sshkey.pub

94 exec ssh-agent bash

95 eval ssh-agent -s

96 ssh-add ~/.ssh/sshkey.pub

97 pwd

98 ;;

99 pwd

100 ll

101 ll

102 ls

103 ssh-add ~/.ssh/id\_rsa

104 ssh-add ~/.ssh/

105 eval `ssh-agent -s`

106 pwd

107 cd /

108 pwd

109 ls -lrt

110 cd bin

111 ls -lrt

112 cd ..

113 cd /c/Users/spjad/git\_training\_v3

114 pwd

115 ls -lrt

116 ssh-keygen -t rsa -b 4096 -C "spjadhav78@gmail.com"

117 ps aux | grep ssh

118 kill 819

119 kill 718

120 kill 747

121 kill 761

122 ps aux | grep ssh

123 eval `ssh-agent -s`

124 ssh-add ~/.ssh/id\_rsa

125 ssh-add ~/.ssh/sshkey1

126 ssh-add ~/.ssh/sshkey1

127 clip < ~/.ssh/sshkey1.pub

128 ssh -T git@github.com

129 git push

130 pwd

131 git status

132 ll

133 ls -lrt

134 ls -lrt

135 git status

136 vi testfile.txt

137 git status

138 git commit -m "Commit Message ssh test connection"

139 git commit -am "Commit Message ssh test connection"

140 git push

141 history

ssh-rsa  spjadhav78@gmail.com

**Merging Branches:**

git branch

143 ll

144 git branch develop

145 git branch

146 git switch develop

147 git branch

148 ll

149 ls

150 touch sampletestfile1.txt

151 touch sampletestfile2.txt

152 echo "This is sample file 1" >> sampletestfile1.txt

153 echo "This is sample file 2" >> sampletestfile2.txt

154 ls

155 git status

156 git add .

157 git status

158 git commit -am "Commit sample file on develop"

159 ls

160 git push

161 git push origin develop

162 git switch main

163 ls

164 touch sampletestfile\_main1.txt

165 ls

166 git commit -am "commit sample file on main"

167 git add .

168 git commit -am "commit sample file on main"

169 git push

170 ls

171 git swith develop

172 git switch develop

173 ls

174 history

175 git pull origin develop

176 ls

177 history

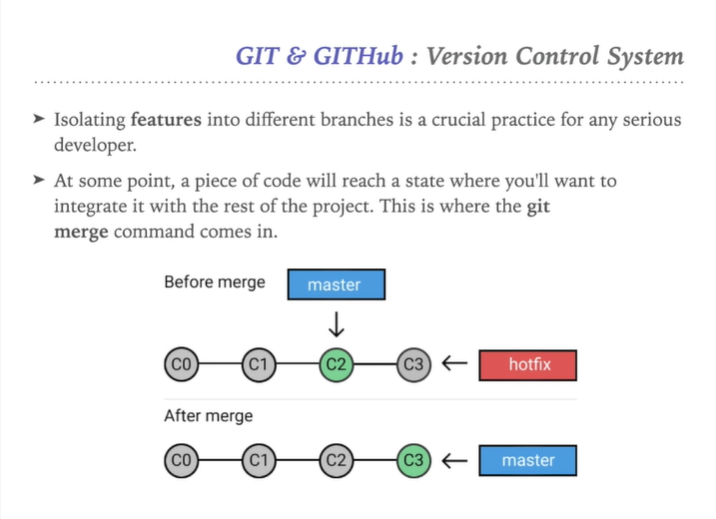
**Merge in branches GIT:**

>Isolating features into different branches is a crucial practice for any serious developer.

There could be multiple kind of development activity but you are working on single project.

Whenever you are working on different task create separate branches for each because your existing incompleting it will not merge unwanted code to main branch.

>At some point, a piece of code will reach a state where you'll want to integrate it with the rest of the project. This is where the git merge command comes in.



189 git clone -b main https://github.com/spjadhav78/git\_training\_v3.git

190 ll

191 cd git\_training\_v3

192 ll

193 git status

194 git switch main

195 ll

196 cls

197 clear

198 history

199 ll

200 git branch

201 git switch develop

202 ll

203 clear

204 ll

205 git switch main

206 ll

207 git pull

208 git merge develop

209 ll

210 git add.

211 git add .

212 git status

213 git commit

214 git push

215 history

**Revert in Git:**

Revert in GIT:

>Revert is all about to Undo the changes, you did in repo.

>In GIT this can be done via RESET & REVERT

>RESET: Practically, user can think of it as a "rollback."

>Reset points local environment back to a previous commit.

>REVERT: Net effect of the git revert command is similar to reset but its approach is different.

>Revert adds a new commit at the end of the chain to "cancel" changes.

>Revert or Reset?

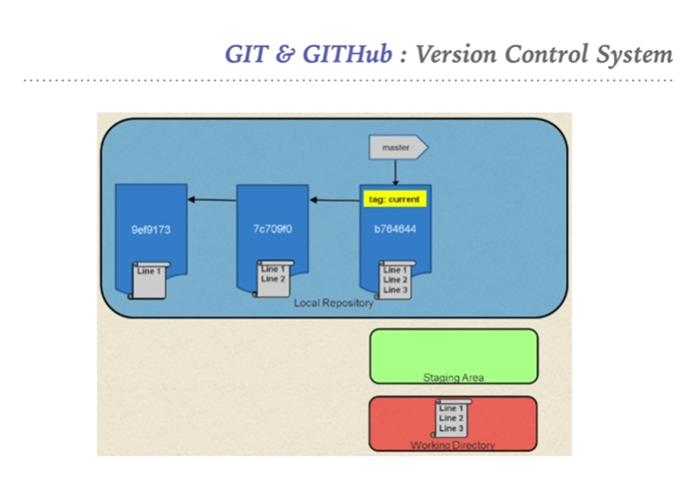
>If user have already pushed commits to the remote repo, a revert is anicer way to cancel out changes.

>Git workflow works well for picking up additional commits at the end of a branch, but can be challenging if a set of commits is no longer seen in the chain when someone reset the branch pointer back.

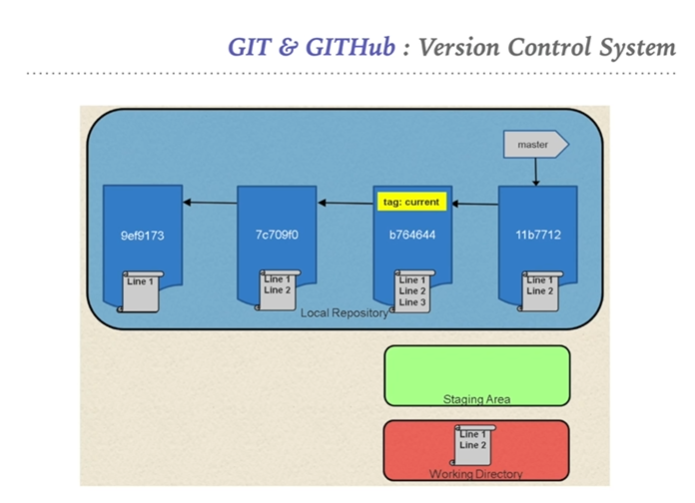
>If Commit in local then Reset is good, If commit is pushed then revert is good option.

git reset --soft baef1bc -> will just revert the git history changes

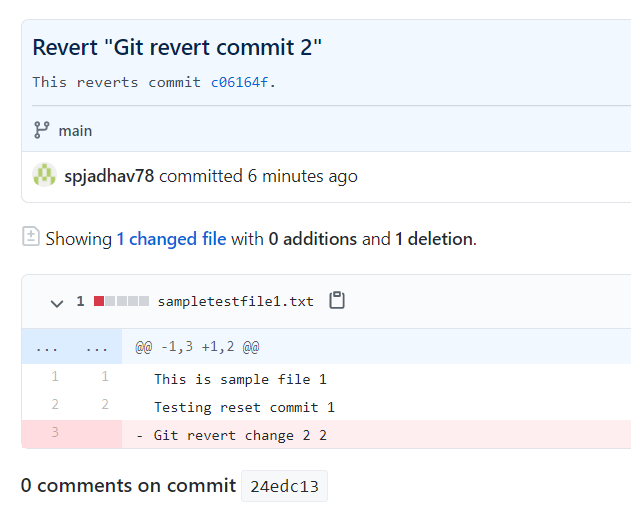
git reset --hard baef1bc -> will revert the history as well it will revert the changes in the file.



Example of Git Reset where we want reset my commit which is done locally on b764



Example of Git Revert where we want revert my commit which is pushed from locally to remote of b764



Example of git revert.

git log --online

218 git log --oneline

219 vi sampletestfile1.txt

220 git commit -am "Commit Reset 1"

221 vi sampletestfile2.txt

222 git commit -am "Commit Reset 2"

223 git log --oneline

224 git reset --soft

225 git reset --soft baef1bc

226 git log --oneline

227 cat sampletestfile2.txt

228 git status

229 git log --oneline

230 git reset --hard baef1bc

231 cat sampletestfile2.txt

232 cat sampletestfile1.txt

233 git log --oneline

234 cat sampletestfile2.txt

235 cat sampletestfile1.txt

236 git log --oneline

237 cat sampletestfile1.txt

238 cat sampletestfile2.txt

239 vi sampletestfile2.txt

240 cat sampletestfile2.txt

241 git commit -am

242 git commit -am "Git revert commit 1"

243 vi sampletestfile1.txt

244 git commit -am "Git revert commit 2"

245 git push

246 git log --oneline

247 git revert HEAD

248 git push

249 git log --oneline

250 cat sampletestfile1.txt

251 history

**Git - Comparison: Compare working Directory & Stage area**

git status

254 ll

255 vi sampletestfile1.txt

256 echo "This is Comparison lecture" >> sampletestfile1.txt

257 echo "This is Comparison lecture" >> sampletestfile2.txt

258 git status

259 git diff sampletestfile1.txt

260 git diff sampletestfile2.txt

261 git diff

262 vi sampletestfile1.txt

263 git status

264 git add .

265 git status

266 git commit -am "Git comparision commit 1"

267 git status

268 git push

269 git status

270 git diff sampletestfile1.txt

271 git status

272 git log --oneline

273 hostory

274 history

**Compare Commits in Git:**

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git diff sampletestfile1.txt

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git diff sampletestfile2.txt

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git commit -am "Sample Commit"

On branch main

Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git log --oneline

b616214 (HEAD -> main, origin/main, origin/HEAD) Git comparision commit 1

24edc13 Revert "Git revert commit 2"

c06164f Git revert commit 2

eada059 Git revert commit 1

baef1bc Commit Reset 1

258a9df (origin/develop, develop) Merge pull request #1 from spjadhav78/main

3c410d6 commit sample file on main

8222f7b Commit sample file on develop

65aab9f Commit Message ssh test connection

81f1441 It will add file as well will Commit Message 2 with -am

7406ae2 Create testfile.txt

5bbb62c Initial commit

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git diff

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git diff b616214 24edc13

diff --git a/sampletestfile1.txt b/sampletestfile1.txt

index 960b9e8..a8723ea 100644

--- a/sampletestfile1.txt

+++ b/sampletestfile1.txt

@@ -1,4 +1,2 @@

This is sample file 1

-Testing reset commit

-Testing revert commit

-This is Comparison lecture

+Testing reset commit 1

diff --git a/sampletestfile2.txt b/sampletestfile2.txt

index e47e128..40e6d5e 100644

--- a/sampletestfile2.txt

+++ b/sampletestfile2.txt

@@ -1,3 +1,2 @@

This is sample file 2

Git revert commit 1

-This is Comparison lecture

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ cat sampletestfile1.txt

This is sample file 1

Testing reset commit

Testing revert commit

This is Comparison lecture

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ cat sampletestfile2.txt

This is sample file 2

Git revert commit 1

This is Comparison lecture

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$

**Concept of Branches in GIT:**

Commands:

>Verify Branch

>Why Branching in GIT

>List GIT Branches

git branch -a

>Create New Branch

git branch <branch\_name>

>Switch Branch in GIT

git checkout <branch\_name>

>Rename Branch

git branch -m <Old\_Name> <New\_Name>

>Delete Branch

git branch -d <branch\_name>

Practice:

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git branch -a

develop

\* main

remotes/origin/HEAD -> origin/main

remotes/origin/develop

remotes/origin/main

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git branch hotfix1

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git branch -a

develop

hotfix1

\* main

remotes/origin/HEAD -> origin/main

remotes/origin/develop

remotes/origin/main

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git status

On branch main

Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git branch -a

develop

hotfix1

\* main

remotes/origin/HEAD -> origin/main

remotes/origin/develop

remotes/origin/main

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ ll

total 4

-rw-r--r-- 1 spjad 197609 64 Mar 11 23:12 README.md

-rw-r--r-- 1 spjad 197609 92 Apr 2 23:07 sampletestfile1.txt

-rw-r--r-- 1 spjad 197609 71 Apr 2 23:04 sampletestfile2.txt

-rw-r--r-- 1 spjad 197609 0 Mar 12 22:35 sampletestfile\_main1.txt

-rw-r--r-- 1 spjad 197609 131 Mar 12 20:05 testfile.txt

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git switch hotfix1

Switched to branch 'hotfix1'

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (hotfix1)

$ git branch -a

develop

\* hotfix1

main

remotes/origin/HEAD -> origin/main

remotes/origin/develop

remotes/origin/main

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (hotfix1)

$ ll

total 4

-rw-r--r-- 1 spjad 197609 64 Mar 11 23:12 README.md

-rw-r--r-- 1 spjad 197609 92 Apr 2 23:07 sampletestfile1.txt

-rw-r--r-- 1 spjad 197609 71 Apr 2 23:04 sampletestfile2.txt

-rw-r--r-- 1 spjad 197609 0 Mar 12 22:35 sampletestfile\_main1.txt

-rw-r--r-- 1 spjad 197609 131 Mar 12 20:05 testfile.txt

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (hotfix1)

$ git branch -m hotfix1 hot\_fix1

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (hot\_fix1)

$ git branch -a

develop

\* hot\_fix1

main

remotes/origin/HEAD -> origin/main

remotes/origin/develop

remotes/origin/main

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (hot\_fix1)

$ git branch -d hot\_fix1

error: Cannot delete branch 'hot\_fix1' checked out at 'C:/Users/spjad/git\_training\_v3'

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (hot\_fix1)

$ git checkout main

Switched to branch 'main'

Your branch is up to date with 'origin/main'.

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git branch -d hot\_fix1

Deleted branch hot\_fix1 (was b616214).

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git branch -a

develop

\* main

remotes/origin/HEAD -> origin/main

remotes/origin/develop

remotes/origin/main

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$

Merge Branches in GIT:

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git branch -a

develop

\* main

remotes/origin/HEAD -> origin/main

remotes/origin/develop

remotes/origin/main

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git branch hotfix1

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git branch -a

develop

hotfix1

\* main

remotes/origin/HEAD -> origin/main

remotes/origin/develop

remotes/origin/main

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git status

On branch main

Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git branch -a

develop

hotfix1

\* main

remotes/origin/HEAD -> origin/main

remotes/origin/develop

remotes/origin/main

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ ll

total 4

-rw-r--r-- 1 spjad 197609 64 Mar 11 23:12 README.md

-rw-r--r-- 1 spjad 197609 92 Apr 2 23:07 sampletestfile1.txt

-rw-r--r-- 1 spjad 197609 71 Apr 2 23:04 sampletestfile2.txt

-rw-r--r-- 1 spjad 197609 0 Mar 12 22:35 sampletestfile\_main1.txt

-rw-r--r-- 1 spjad 197609 131 Mar 12 20:05 testfile.txt

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git switch hotfix1

Switched to branch 'hotfix1'

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (hotfix1)

$ git branch -a

develop

\* hotfix1

main

remotes/origin/HEAD -> origin/main

remotes/origin/develop

remotes/origin/main

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (hotfix1)

$ ll

total 4

-rw-r--r-- 1 spjad 197609 64 Mar 11 23:12 README.md

-rw-r--r-- 1 spjad 197609 92 Apr 2 23:07 sampletestfile1.txt

-rw-r--r-- 1 spjad 197609 71 Apr 2 23:04 sampletestfile2.txt

-rw-r--r-- 1 spjad 197609 0 Mar 12 22:35 sampletestfile\_main1.txt

-rw-r--r-- 1 spjad 197609 131 Mar 12 20:05 testfile.txt

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (hotfix1)

$ git branch -m hotfix1 hot\_fix1

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (hot\_fix1)

$ git branch -a

develop

\* hot\_fix1

main

remotes/origin/HEAD -> origin/main

remotes/origin/develop

remotes/origin/main

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (hot\_fix1)

$ git branch -d hot\_fix1

error: Cannot delete branch 'hot\_fix1' checked out at 'C:/Users/spjad/git\_training\_v3'

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (hot\_fix1)

$ git checkout main

Switched to branch 'main'

Your branch is up to date with 'origin/main'.

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git branch -d hot\_fix1

Deleted branch hot\_fix1 (was b616214).

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git branch -a

develop

\* main

remotes/origin/HEAD -> origin/main

remotes/origin/develop

remotes/origin/main

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git branch -a

develop

\* main

remotes/origin/HEAD -> origin/main

remotes/origin/develop

remotes/origin/main

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git switch develop

Switched to branch 'develop'

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (develop)

$ git branch -a

\* develop

main

remotes/origin/HEAD -> origin/main

remotes/origin/develop

remotes/origin/main

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (develop)

$ vi sampletestfile1.txt

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (develop)

$ cat sampletestfile1.txt

This is sample file 1

This is modified for merge branch in git practice.

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (develop)

$ git status

On branch develop

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: sampletestfile1.txt

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

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (develop)

$ git diff

diff --git a/sampletestfile1.txt b/sampletestfile1.txt

index e129759..a85080a 100644

--- a/sampletestfile1.txt

+++ b/sampletestfile1.txt

@@ -1 +1,2 @@

This is sample file 1

+This is modified for merge branch in git practice.

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (develop)

$ git commit -am "Sample merge on develop"

[develop ca2b21e] Sample merge on develop

1 file changed, 1 insertion(+)

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (develop)

$ git checkout main

Switched to branch 'main'

Your branch is up to date with 'origin/main'.

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ ll

total 4

-rw-r--r-- 1 spjad 197609 64 Mar 11 23:12 README.md

-rw-r--r-- 1 spjad 197609 96 Apr 2 23:51 sampletestfile1.txt

-rw-r--r-- 1 spjad 197609 72 Apr 2 23:51 sampletestfile2.txt

-rw-r--r-- 1 spjad 197609 0 Mar 12 22:35 sampletestfile\_main1.txt

-rw-r--r-- 1 spjad 197609 131 Mar 12 20:05 testfile.txt

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ cat sampletestfile1.txt

This is sample file 1

Testing reset commit

Testing revert commit

This is Comparison lecture

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git log --oneline --graph

\* b616214 (HEAD -> main, origin/main, origin/HEAD) Git comparision commit 1

\* 24edc13 Revert "Git revert commit 2"

\* c06164f Git revert commit 2

\* eada059 Git revert commit 1

\* baef1bc Commit Reset 1

\* 258a9df (origin/develop) Merge pull request #1 from spjadhav78/main

|\

| \* 3c410d6 commit sample file on main

\* | 8222f7b Commit sample file on develop

|/

\* 65aab9f Commit Message ssh test connection

\* 81f1441 It will add file as well will Commit Message 2 with -am

\* 7406ae2 Create testfile.txt

\* 5bbb62c Initial commit

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git checkout develop

Switched to branch 'develop'

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (develop)

$ git log -oneline --graph

fatal: unrecognized argument: -oneline

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (develop)

$ git log --oneline --graph

\* ca2b21e (HEAD -> develop) Sample merge on develop

\* 258a9df (origin/develop) Merge pull request #1 from spjadhav78/main

|\

| \* 3c410d6 commit sample file on main

\* | 8222f7b Commit sample file on develop

|/

\* 65aab9f Commit Message ssh test connection

\* 81f1441 It will add file as well will Commit Message 2 with -am

\* 7406ae2 Create testfile.txt

\* 5bbb62c Initial commit

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (develop)

$ ll

total 4

-rw-r--r-- 1 spjad 197609 64 Mar 11 23:12 README.md

-rw-r--r-- 1 spjad 197609 75 Apr 2 23:52 sampletestfile1.txt

-rw-r--r-- 1 spjad 197609 23 Apr 2 23:52 sampletestfile2.txt

-rw-r--r-- 1 spjad 197609 0 Mar 12 22:35 sampletestfile\_main1.txt

-rw-r--r-- 1 spjad 197609 131 Mar 12 20:05 testfile.txt

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (develop)

$ cat sampletestfile2.txt

This is sample file 2

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (develop)

$ git checkout main

Switched to branch 'main'

Your branch is up to date with 'origin/main'.

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ ll

total 4

-rw-r--r-- 1 spjad 197609 64 Mar 11 23:12 README.md

-rw-r--r-- 1 spjad 197609 96 Apr 2 23:53 sampletestfile1.txt

-rw-r--r-- 1 spjad 197609 72 Apr 2 23:53 sampletestfile2.txt

-rw-r--r-- 1 spjad 197609 0 Mar 12 22:35 sampletestfile\_main1.txt

-rw-r--r-- 1 spjad 197609 131 Mar 12 20:05 testfile.txt

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ cat sampletestfile2.txt

This is sample file 2

Git revert commit 1

This is Comparison lecture

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git checkout develop

Switched to branch 'develop'

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (develop)

$ git merge main -m "Merging Main to Develop"

Auto-merging sampletestfile1.txt

CONFLICT (content): Merge conflict in sampletestfile1.txt

Automatic merge failed; fix conflicts and then commit the result.

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (develop|MERGING)

$ vi sampletestfile1.txt

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (develop|MERGING)

$ git status

On branch develop

You have unmerged paths.

(fix conflicts and run "git commit")

(use "git merge --abort" to abort the merge)

Changes to be committed:

modified: sampletestfile2.txt

Unmerged paths:

(use "git add <file>..." to mark resolution)

both modified: sampletestfile1.txt

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (develop|MERGING)

$ git add .

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (develop|MERGING)

$ git commit

hint: Waiting for your editor to close the file... 0 [sig] bash 2790! sigpacket::process: Suppressing signal 18 to win32 process (pid 13556)

17666048 [sig] bash 2790! sigpacket::process: Suppressing signal 18 to win32 process (pid 13556)

19650652 [sig] bash 2790! sigpacket::process: Suppressing signal 18 to win32 process (pid 13556)

20209843 [sig] bash 2790! sigpacket::process: Suppressing signal 18 to win32 process (pid 13556)

[develop 25c6922] Merg Main to Develop

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (develop)

$ git status

On branch develop

nothing to commit, working tree clean

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (develop)

$ git log --oneline --graph

\* 25c6922 (HEAD -> develop) Merg Main to Develop

|\

| \* b616214 (origin/main, origin/HEAD, main) Git comparision commit 1

| \* 24edc13 Revert "Git revert commit 2"

| \* c06164f Git revert commit 2

| \* eada059 Git revert commit 1

| \* baef1bc Commit Reset 1

\* | ca2b21e Sample merge on develop

|/

\* 258a9df (origin/develop) Merge pull request #1 from spjadhav78/main

|\

| \* 3c410d6 commit sample file on main

\* | 8222f7b Commit sample file on develop

|/

\* 65aab9f Commit Message ssh test connection

\* 81f1441 It will add file as well will Commit Message 2 with -am

\* 7406ae2 Create testfile.txt

\* 5bbb62c Initial commit

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (develop)

$ git checkout main

Switched to branch 'main'

Your branch is up to date with 'origin/main'.

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ ll

total 4

-rw-r--r-- 1 spjad 197609 64 Mar 11 23:12 README.md

-rw-r--r-- 1 spjad 197609 96 Apr 3 00:03 sampletestfile1.txt

-rw-r--r-- 1 spjad 197609 72 Apr 2 23:54 sampletestfile2.txt

-rw-r--r-- 1 spjad 197609 0 Mar 12 22:35 sampletestfile\_main1.txt

-rw-r--r-- 1 spjad 197609 131 Mar 12 20:05 testfile.txt

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ cat sampletestfile1.txt

This is sample file 1

Testing reset commit

Testing revert commit

This is Comparison lecture

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git log --oneline -graph

fatal: unrecognized argument: -graph

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git log --oneline --graph

\* b616214 (HEAD -> main, origin/main, origin/HEAD) Git comparision commit 1

\* 24edc13 Revert "Git revert commit 2"

\* c06164f Git revert commit 2

\* eada059 Git revert commit 1

\* baef1bc Commit Reset 1

\* 258a9df (origin/develop) Merge pull request #1 from spjadhav78/main

|\

| \* 3c410d6 commit sample file on main

\* | 8222f7b Commit sample file on develop

|/

\* 65aab9f Commit Message ssh test connection

\* 81f1441 It will add file as well will Commit Message 2 with -am

\* 7406ae2 Create testfile.txt

\* 5bbb62c Initial commit

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git merge develop -m "Merging Develop to Main"

Updating b616214..25c6922

Fast-forward (no commit created; -m option ignored)

sampletestfile1.txt | 3 +++

1 file changed, 3 insertions(+)

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git add .

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git commit

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

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git log --oneline --graph

\* 25c6922 (HEAD -> main, develop) Merg Main to Develop

|\

| \* b616214 (origin/main, origin/HEAD) Git comparision commit 1

| \* 24edc13 Revert "Git revert commit 2"

| \* c06164f Git revert commit 2

| \* eada059 Git revert commit 1

| \* baef1bc Commit Reset 1

\* | ca2b21e Sample merge on develop

|/

\* 258a9df (origin/develop) Merge pull request #1 from spjadhav78/main

|\

| \* 3c410d6 commit sample file on main

\* | 8222f7b Commit sample file on develop

|/

\* 65aab9f Commit Message ssh test connection

\* 81f1441 It will add file as well will Commit Message 2 with -am

\* 7406ae2 Create testfile.txt

\* 5bbb62c Initial commit

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ cat sampletestfile1.txt

This is sample file 1

This is modified for merge branch in git practice.

Testing reset commit

Testing revert commit

This is Comparison lecture

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$

**Git Stash:**

Git stash temporarily saves changes you have made to your working copy so you can work on something else, and then come back and re-apply them later on. Stashing is handy if you need to quickly switch context and work on something else, but you are mid-way through a code change and aren't quite ready to commit.

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ ll

total 4

-rw-r--r-- 1 spjad 197609 64 Mar 11 23:12 README.md

-rw-r--r-- 1 spjad 197609 152 Apr 3 00:04 sampletestfile1.txt

-rw-r--r-- 1 spjad 197609 72 Apr 2 23:54 sampletestfile2.txt

-rw-r--r-- 1 spjad 197609 0 Mar 12 22:35 sampletestfile\_main1.txt

-rw-r--r-- 1 spjad 197609 131 Mar 12 20:05 testfile.txt

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ echo "This is Stash lecture, Change in File 2" >> sampletestfile2.txt

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ echo "This is Stash lecture, Change in File 1" >> sampletestfile1.txt

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git status

On branch main

Your branch is ahead of 'origin/main' by 2 commits.

(use "git push" to publish your local commits)

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: sampletestfile1.txt

modified: sampletestfile2.txt

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

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git stash

warning: LF will be replaced by CRLF in sampletestfile1.txt.

The file will have its original line endings in your working directory

warning: LF will be replaced by CRLF in sampletestfile2.txt.

The file will have its original line endings in your working directory

Saved working directory and index state WIP on main: 25c6922 Merg Main to Develop

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git statsh

git: 'statsh' is not a git command. See 'git --help'.

The most similar command is

stash

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git stash

No local changes to save

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ 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

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ cat sampletestfile1.txt

This is sample file 1

This is modified for merge branch in git practice.

Testing reset commit

Testing revert commit

This is Comparison lecture

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ ll

total 4

-rw-r--r-- 1 spjad 197609 64 Mar 11 23:12 README.md

-rw-r--r-- 1 spjad 197609 152 Apr 3 00:16 sampletestfile1.txt

-rw-r--r-- 1 spjad 197609 72 Apr 3 00:16 sampletestfile2.txt

-rw-r--r-- 1 spjad 197609 0 Mar 12 22:35 sampletestfile\_main1.txt

-rw-r--r-- 1 spjad 197609 131 Mar 12 20:05 testfile.txt

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ echo "This is Stash lecture, Change in Main File" >> sampletestfile\_main1.txt

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ ll

total 5

-rw-r--r-- 1 spjad 197609 64 Mar 11 23:12 README.md

-rw-r--r-- 1 spjad 197609 152 Apr 3 00:16 sampletestfile1.txt

-rw-r--r-- 1 spjad 197609 72 Apr 3 00:16 sampletestfile2.txt

-rw-r--r-- 1 spjad 197609 43 Apr 3 00:19 sampletestfile\_main1.txt

-rw-r--r-- 1 spjad 197609 131 Mar 12 20:05 testfile.txt

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git status

On branch main

Your branch is ahead of 'origin/main' by 2 commits.

(use "git push" to publish your local commits)

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: sampletestfile\_main1.txt

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

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git commit

On branch main

Your branch is ahead of 'origin/main' by 2 commits.

(use "git push" to publish your local commits)

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: sampletestfile\_main1.txt

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

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git commit -am "Urgent Fix Change"

warning: LF will be replaced by CRLF in sampletestfile\_main1.txt.

The file will have its original line endings in your working directory

[main 77caae1] Urgent Fix Change

1 file changed, 1 insertion(+)

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git push

Enumerating objects: 15, done.

Counting objects: 100% (14/14), done.

Delta compression using up to 8 threads

Compressing objects: 100% (8/8), done.

Writing objects: 100% (9/9), 947 bytes | 473.00 KiB/s, done.

Total 9 (delta 4), reused 0 (delta 0), pack-reused 0

remote: Resolving deltas: 100% (4/4), completed with 2 local objects.

To https://github.com/spjadhav78/git\_training\_v3.git

b616214..77caae1 main -> main

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git status

On branch main

Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git stash list

stash@{0}: WIP on main: 25c6922 Merg Main to Develop

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ ll

total 5

-rw-r--r-- 1 spjad 197609 64 Mar 11 23:12 README.md

-rw-r--r-- 1 spjad 197609 152 Apr 3 00:16 sampletestfile1.txt

-rw-r--r-- 1 spjad 197609 72 Apr 3 00:16 sampletestfile2.txt

-rw-r--r-- 1 spjad 197609 43 Apr 3 00:19 sampletestfile\_main1.txt

-rw-r--r-- 1 spjad 197609 131 Mar 12 20:05 testfile.txt

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ cat sampletestfile1.txt

This is sample file 1

This is modified for merge branch in git practice.

Testing reset commit

Testing revert commit

This is Comparison lecture

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git stash list

stash@{0}: WIP on main: 25c6922 Merg Main to Develop

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git stash apply

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: sampletestfile1.txt

modified: sampletestfile2.txt

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

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ cat sampletestfile1.txt

This is sample file 1

This is modified for merge branch in git practice.

Testing reset commit

Testing revert commit

This is Comparison lecture

This is Stash lecture, Change in File 1

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git commit -am "Stash Changes"

[main a6c5188] Stash Changes

2 files changed, 2 insertions(+)

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git push

Enumerating objects: 7, done.

Counting objects: 100% (7/7), done.

Delta compression using up to 8 threads

Compressing objects: 100% (4/4), done.

Writing objects: 100% (4/4), 447 bytes | 447.00 KiB/s, done.

Total 4 (delta 2), reused 0 (delta 0), pack-reused 0

remote: Resolving deltas: 100% (2/2), completed with 2 local objects.

To https://github.com/spjadhav78/git\_training\_v3.git

77caae1..a6c5188 main -> main

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git status

On branch main

Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git stash list

stash@{0}: WIP on main: 25c6922 Merg Main to Develop

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git stash drop

Dropped refs/stash@{0} (9eab41456b3ea860e249606bbd556087281fc5eb)

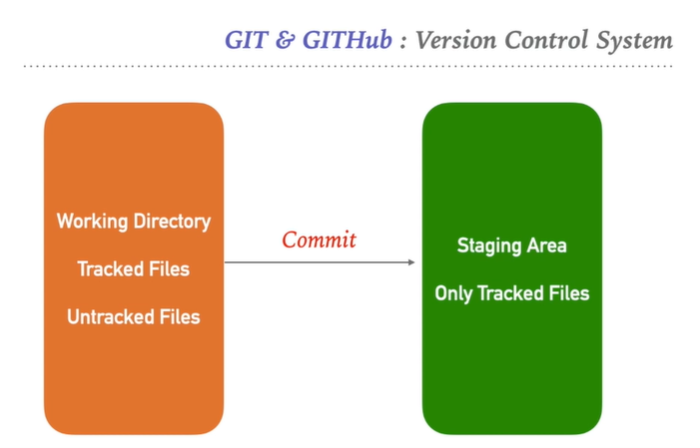
spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git stash list

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$

**Stash Untracked Files:**



Stash Untracked Files:

git stash pop will stash the untracked files as will it wil move the file from stash area where git stash apply can't do same function.

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ echo "This is Stash lecture, Change for untracked Files in File1" >> sampletestfile\_main1.txt

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git status

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: sampletestfile\_main1.txt

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

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ echo "This is Stash lecture, Change for untracked Files in File1" >> sampletestfile\_main3.txt

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ ll

total 6

-rw-r--r-- 1 spjad 197609 64 Mar 11 23:12 README.md

-rw-r--r-- 1 spjad 197609 193 Apr 3 00:23 sampletestfile1.txt

-rw-r--r-- 1 spjad 197609 113 Apr 3 00:23 sampletestfile2.txt

-rw-r--r-- 1 spjad 197609 102 Apr 3 00:38 sampletestfile\_main1.txt

-rw-r--r-- 1 spjad 197609 59 Apr 3 00:38 sampletestfile\_main3.txt

-rw-r--r-- 1 spjad 197609 131 Mar 12 20:05 testfile.txt

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git status

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: sampletestfile\_main1.txt

Untracked files:

(use "git add <file>..." to include in what will be committed)

sampletestfile\_main3.txt

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

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git stash

warning: LF will be replaced by CRLF in sampletestfile\_main1.txt.

The file will have its original line endings in your working directory

Saved working directory and index state WIP on main: a6c5188 Stash Changes

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git status

On branch main

Your branch is up to date with 'origin/main'.

Untracked files:

(use "git add <file>..." to include in what will be committed)

sampletestfile\_main3.txt

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

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git stash -a

warning: LF will be replaced by CRLF in sampletestfile\_main3.txt.

The file will have its original line endings in your working directory

Saved working directory and index state WIP on main: a6c5188 Stash Changes

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git status

On branch main

Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git stash list

stash@{0}: WIP on main: a6c5188 Stash Changes

stash@{1}: WIP on main: a6c5188 Stash Changes

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git stash pop

Already up to date!

On branch main

Your branch is up to date with 'origin/main'.

Untracked files:

(use "git add <file>..." to include in what will be committed)

sampletestfile\_main3.txt

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

Dropped refs/stash@{0} (7f0d58b1a8cf5681d9040dd7bca418dcbfd56044)

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git stash list

stash@{0}: WIP on main: a6c5188 Stash Changes

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ 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: sampletestfile\_main1.txt

Untracked files:

(use "git add <file>..." to include in what will be committed)

sampletestfile\_main3.txt

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

Dropped refs/stash@{0} (28134bd8bb05380520406b21da4cd59297cf79fb)

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git stash list

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$

Manage Multiple Stash:

>Stash with Message:

git statsh save "message text"

>Find Changes Done in Specific Stash:

git stash show stash@{index}

>Apply Specific Stash:

git stash apply stash@{index}

$ git status

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: sampletestfile\_main1.txt

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

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ echo "This is Stash lecture, Change for untracked Files in File1" >> sampletestfile\_main3.txt

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ ll

total 6

-rw-r--r-- 1 spjad 197609 64 Mar 11 23:12 README.md

-rw-r--r-- 1 spjad 197609 193 Apr 3 00:23 sampletestfile1.txt

-rw-r--r-- 1 spjad 197609 113 Apr 3 00:23 sampletestfile2.txt

-rw-r--r-- 1 spjad 197609 102 Apr 3 00:38 sampletestfile\_main1.txt

-rw-r--r-- 1 spjad 197609 59 Apr 3 00:38 sampletestfile\_main3.txt

-rw-r--r-- 1 spjad 197609 131 Mar 12 20:05 testfile.txt

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git status

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: sampletestfile\_main1.txt

Untracked files:

(use "git add <file>..." to include in what will be committed)

sampletestfile\_main3.txt

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

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git stash

warning: LF will be replaced by CRLF in sampletestfile\_main1.txt.

The file will have its original line endings in your working directory

Saved working directory and index state WIP on main: a6c5188 Stash Changes

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git status

On branch main

Your branch is up to date with 'origin/main'.

Untracked files:

(use "git add <file>..." to include in what will be committed)

sampletestfile\_main3.txt

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

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git stash -a

warning: LF will be replaced by CRLF in sampletestfile\_main3.txt.

The file will have its original line endings in your working directory

Saved working directory and index state WIP on main: a6c5188 Stash Changes

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git status

On branch main

Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git stash list

stash@{0}: WIP on main: a6c5188 Stash Changes

stash@{1}: WIP on main: a6c5188 Stash Changes

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git stash pop

Already up to date!

On branch main

Your branch is up to date with 'origin/main'.

Untracked files:

(use "git add <file>..." to include in what will be committed)

sampletestfile\_main3.txt

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

Dropped refs/stash@{0} (7f0d58b1a8cf5681d9040dd7bca418dcbfd56044)

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git stash list

stash@{0}: WIP on main: a6c5188 Stash Changes

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ 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: sampletestfile\_main1.txt

Untracked files:

(use "git add <file>..." to include in what will be committed)

sampletestfile\_main3.txt

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

Dropped refs/stash@{0} (28134bd8bb05380520406b21da4cd59297cf79fb)

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git stash list

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git status

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: sampletestfile\_main1.txt

Untracked files:

(use "git add <file>..." to include in what will be committed)

sampletestfile\_main3.txt

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

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git stash save "Change in sampletestfile\_main1.txt"

Saved working directory and index state On main: Change in sampletestfile\_main1.txt

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git stash list

stash@{0}: On main: Change in sampletestfile\_main1.txt

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git status

On branch main

Your branch is up to date with 'origin/main'.

Untracked files:

(use "git add <file>..." to include in what will be committed)

sampletestfile\_main3.txt

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

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git stash 0a save "Change in sampletestfile\_main3.txt"

fatal: unknown subcommand: 0a

usage: git stash list [<options>]

or: git stash show [<options>] [<stash>]

or: git stash drop [-q|--quiet] [<stash>]

or: git stash ( pop | apply ) [--index] [-q|--quiet] [<stash>]

or: git stash branch <branchname> [<stash>]

or: git stash clear

or: git stash [push [-p|--patch] [-k|--[no-]keep-index] [-q|--quiet]

[-u|--include-untracked] [-a|--all] [-m|--message <message>]

[--pathspec-from-file=<file> [--pathspec-file-nul]]

[--] [<pathspec>...]]

or: git stash save [-p|--patch] [-k|--[no-]keep-index] [-q|--quiet]

[-u|--include-untracked] [-a|--all] [<message>]

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git stash -a save "Change in sampletestfile\_main3.txt"

fatal: subcommand wasn't specified; 'push' can't be assumed due to unexpected token 'save'

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git stash -a

Saved working directory and index state WIP on main: a6c5188 Stash Changes

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git stash list

stash@{0}: WIP on main: a6c5188 Stash Changes

stash@{1}: On main: Change in sampletestfile\_main1.txt

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git stash show stash@{0}

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git stash show stash@{1}

sampletestfile\_main1.txt | 1 +

1 file changed, 1 insertion(+)

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git stash pop stash@{1}

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: sampletestfile\_main1.txt

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

Dropped stash@{1} (17877ff391feff4e1508df9e70561f10516eddeb)

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$ git stash list

stash@{0}: WIP on main: a6c5188 Stash Changes

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$

Git stash show will only work for the tracked files not for untracked file.

**All Git Commands:**

$ history

1 git --version

2 git help

3 git help pull

4 git help -pull

5 git config --global user

6 git config --global user.name"Sanjay Jadhav"

7 git config --global user.name "Sanjay Jadhav"

8 git config --global user.email "spjadhav78@gmail.com"

9 git config --list

10 history

11 pwd

12 ll

13 ll

14 cd gittraining\_v1/

15 pwd

16 git init

17 ll

18 ls -lrt

19 ls -a

20 cd .git

21 ls -a

22 cd ..

23 pwd

24 history

25 ll

26 git commit

27 git status

28 git status

29 git add first\_file.txt

30 git status

31 git add .

32 git status

33 git commit -m "First File Commit"

34 git status

35 git status

36 git diff

37 git commit -m "First File Commit edited"

38 git add .

39 git commit -m "First File Commit edited"

40 git status

41 git status

42 touch my\_secondfile.txt

43 ll

44 vi first\_file.txt

45 git status

46 git add .

47 git status

48 git commit -m "My another commit for second file"

49 git log

50 git log --author="Sanjay Jadhav"

51 history

52 clear

53 pwd

54 cd ..

55 pwd

56 clear

57 git clone -b main https://github.com/spjadhav78/git\_training\_v3.git

58 ll

59 cd git\_training\_v3

60 ll

61 ls -a

62 git status

63 git pull

64 ll

65 vi testfile.txt

66 git status

67 git commit -m "Commit Message 2"

68 git commit -am "It will add file as well will Commit Message 2 with -am"

69 git push

70 exit

71 git status

72 git --version

73 pwd

74 LL

75 ll

76 cd git\_training\_v3

77 git status

78 ll

79 history

80 ssh-keygen -t ed25519 -C "spjadhav78@gmail.com"

81 ll

82 ssh-keygen -t ed25519 -C "spjadhav78@gmail.com"

83 ssh-keygen -t ed25519 -C "spjadhav78@gmail.com"

84 ll

85 rm -a sshkeyfile.pub

86 rm -f sshkeyfile.pub

87 ll

88 rm -f sshkeyfile

89 ll

90 eval ssh-agent -s

91 ssh-add ~/.ssh/sshkey

92 clip < ~/.ssh/sshkey.pub

93 ssh-add ~/.ssh/sshkey.pub

94 exec ssh-agent bash

95 eval ssh-agent -s

96 ssh-add ~/.ssh/sshkey.pub

97 pwd

98 ;;

99 pwd

100 ll

101 ll

102 ls

103 ssh-add ~/.ssh/id\_rsa

104 ssh-add ~/.ssh/

105 eval `ssh-agent -s`

106 pwd

107 cd /

108 pwd

109 ls -lrt

110 cd bin

111 ls -lrt

112 cd ..

113 cd /c/Users/spjad/git\_training\_v3

114 pwd

115 ls -lrt

116 ssh-keygen -t rsa -b 4096 -C "spjadhav78@gmail.com"

117 ps aux | grep ssh

118 kill 819

119 kill 718

120 kill 747

121 kill 761

122 ps aux | grep ssh

123 eval `ssh-agent -s`

124 ssh-add ~/.ssh/id\_rsa

125 ssh-add ~/.ssh/sshkey1

126 ssh-add ~/.ssh/sshkey1

127 clip < ~/.ssh/sshkey1.pub

128 ssh -T git@github.com

129 git push

130 pwd

131 git status

132 ll

133 ls -lrt

134 ls -lrt

135 git status

136 vi testfile.txt

137 git status

138 git commit -m "Commit Message ssh test connection"

139 git commit -am "Commit Message ssh test connection"

140 git push

141 history

142 git branch

143 ll

144 git branch develop

145 git branch

146 git switch develop

147 git branch

148 ll

149 ls

150 touch sampletestfile1.txt

151 touch sampletestfile2.txt

152 echo "This is sample file 1" >> sampletestfile1.txt

153 echo "This is sample file 2" >> sampletestfile2.txt

154 ls

155 git status

156 git add .

157 git status

158 git commit -am "Commit sample file on develop"

159 ls

160 git push

161 git push origin develop

162 git switch main

163 ls

164 touch sampletestfile\_main1.txt

165 ls

166 git commit -am "commit sample file on main"

167 git add .

168 git commit -am "commit sample file on main"

169 git push

170 ls

171 git swith develop

172 git switch develop

173 ls

174 history

175 git pull origin develop

176 ls

177 history

178 exit

179 git config --list

180 cd gittraining\_v1/

181 ll

182 git branch

183 ll

184 git status

185 exit

186 git --list

187 git list

188 git config --list

189 git clone -b main https://github.com/spjadhav78/git\_training\_v3.git

190 ll

191 cd git\_training\_v3

192 ll

193 git status

194 git switch main

195 ll

196 cls

197 clear

198 history

199 ll

200 git branch

201 git switch develop

202 ll

203 clear

204 ll

205 git switch main

206 ll

207 git pull

208 git merge develop

209 ll

210 git add.

211 git add .

212 git status

213 git commit

214 git push

215 history

216 ll

217 git log --online

218 git log --oneline

219 vi sampletestfile1.txt

220 git commit -am "Commit Reset 1"

221 vi sampletestfile2.txt

222 git commit -am "Commit Reset 2"

223 git log --oneline

224 git reset --soft

225 git reset --soft baef1bc

226 git log --oneline

227 cat sampletestfile2.txt

228 git status

229 git log --oneline

230 git reset --hard baef1bc

231 cat sampletestfile2.txt

232 cat sampletestfile1.txt

233 git log --oneline

234 cat sampletestfile2.txt

235 cat sampletestfile1.txt

236 git log --oneline

237 cat sampletestfile1.txt

238 cat sampletestfile2.txt

239 vi sampletestfile2.txt

240 cat sampletestfile2.txt

241 git commit -am

242 git commit -am "Git revert commit 1"

243 vi sampletestfile1.txt

244 git commit -am "Git revert commit 2"

245 git push

246 git log --oneline

247 git revert HEAD

248 git push

249 git log --oneline

250 cat sampletestfile1.txt

251 history

252 it status

253 git status

254 ll

255 vi sampletestfile1.txt

256 echo "This is Comparison lecture" >> sampletestfile1.txt

257 echo "This is Comparison lecture" >> sampletestfile2.txt

258 git status

259 git diff sampletestfile1.txt

260 git diff sampletestfile2.txt

261 git diff

262 vi sampletestfile1.txt

263 git status

264 git add .

265 git status

266 git commit -am "Git comparision commit 1"

267 git status

268 git push

269 git status

270 git diff sampletestfile1.txt

271 git status

272 git log --oneline

273 hostory

274 history

275 git diff sampletestfile1.txt

276 git diff sampletestfile2.txt

277 git commit -am "Sample Commit"

278 git log --oneline

279 git diff

280 git diff b616214 24edc13

281 cat sampletestfile1.txt

282 cat sampletestfile2.txt

283 history

284 git branch -a

285 git branch hotfix1

286 git branch -a

287 git status

288 git branch -a

289 ll

290 git switch hotfix1

291 git branch -a

292 ll

293 git branch -m hotfix1 hot\_fix1

294 git branch -a

295 git branch -d hot\_fix1

296 git checkout main

297 git branch -d hot\_fix1

298 git branch -a

299 git branch -a

300 git switch develop

301 git branch -a

302 vi sampletestfile1.txt

303 cat sampletestfile1.txt

304 git status

305 git diff

306 git commit -am "Sample merge on develop"

307 git checkout main

308 ll

309 cat sampletestfile1.txt

310 git log --oneline --graph

311 git checkout develop

312 git log -oneline --graph

313 git log --oneline --graph

314 ll

315 cat sampletestfile2.txt

316 git checkout main

317 ll

318 cat sampletestfile2.txt

319 git checkout develop

320 git merge main -m "Merging Main to Develop"

321 vi sampletestfile1.txt

322 git status

323 git add .

324 git commit

325 git status

326 git log --oneline --graph

327 git checkout main

328 ll

329 cat sampletestfile1.txt

330 git log --oneline -graph

331 git log --oneline --graph

332 git merge develop -m "Merging Develop to Main"

333 git add .

334 git commit

335 git log --oneline --graph

336 cat sampletestfile1.txt

337 git status

338 git status

339 git pull

340 ll

341 echo "This is Stash lecture, Change in File 2" >> sampletestfile2.txt

342 echo "This is Stash lecture, Change in File 1" >> sampletestfile1.txt

343 git status

344 git stash

345 git statsh

346 git stash

347 git status

348 cat sampletestfile1.txt

349 ll

350 echo "This is Stash lecture, Change in Main File" >> sampletestfile\_main1.txt

351 ll

352 git status

353 git commit

354 git commit -am "Urgent Fix Change"

355 git push

356 git status

357 git stash list

358 ll

359 cat sampletestfile1.txt

360 git stash list

361 git stash apply

362 cat sampletestfile1.txt

363 git commit -am "Stash Changes"

364 git push

365 git status

366 git stash list

367 git stash drop

368 git stash list

369 echo "This is Stash lecture, Change for untracked Files in File1" >> sampletestfile\_main1.txt

370 git status

371 echo "This is Stash lecture, Change for untracked Files in File1" >> sampletestfile\_main3.txt

372 ll

373 git status

374 git stash

375 git status

376 git stash -a

377 git status

378 git stash list

379 git stash pop

380 git stash list

381 git stash pop

382 git stash list

383 git status

384 git stash save "Change in sampletestfile\_main1.txt"

385 git stash list

386 git status

387 git stash 0a save "Change in sampletestfile\_main3.txt"

388 git stash -a save "Change in sampletestfile\_main3.txt"

389 git stash -a

390 git stash list

391 git stash show stash@{0}

392 git stash show stash@{1}

393 git stash pop stash@{1}

394 git stash list

395 history

spjad@LAPTOP-TL0QD8L2 MINGW64 ~/git\_training\_v3 (main)

$