The files which managed by git have three status

* + Modified: indicate this file have been modified(red color word)
  + Staged: indicate this file have been added to stage
  + Committed : indicate this file have been committed to repository

Three trees

Working tree Stage tree Repository tree

Working 
Directo 
stage 
sitory 

HAED point the latest version

* + config the user name and email address:
    - git config --global user.name "USERNAME"
    - git config --global user.email "EMAIL ADDRESS"
  + Create a new repository
    - git init
  + Add a file to stage
    - git add FILENAME
  + Commit the files which in the stage to the Repository
    - git commit -m "comment"

:\Git>git commit -m "add a readme file" 
[master (root—commit) 53c6917] add a readme file 
1 file changed, 1 insertion (+) 
create mode 100644 readme. txt 

* + Quire the status
    - git status

: status 
n branch master 
othing to commit, working tree clean 

Branch master is the default branch

Nothing to commit means stage tree is empty

Working tree clean means files and data that in the workspace folder no updated

: status 
n branch master 
ntracked files: 
(use "git add <file>. 
to include in what will be committed) 
othing added to commit but untracked files present (use "git add" 
to track) 

Untracked files means this file is new added and have not been add to stage.

: add license. txt 
: status 
n branch master 
hanges to be committed: 
(use "git restore —staged < file).. " 
now filo: 
I iconso. txt 
to unstage) 
commit —m "add a license file" 
[master 809c0e2] add a license file 
1 file changed, 1 insertion (+) 
create mode 100644 license. txt 

D: status 
On branch master 
nothing to commit, working tree clean 

* + If you modify the license.txt file in workspace, the status will change.

status 
branch master 
ges not staged for commit: 
(use "git add " 
(use "git restore <file>. 
no changes added to commit 
to update what will be 
to discard changes 
(use "zit add" and/or 
commi t ted) 
in working directory) 
zit commit -a") 

If use command git restore <file>, it will using the file which store in stage to cover the modified file.

D: status 
On branch master 
ges not staged for commit: 
(use "git add " 
to update what will be committed) 
(use "git restore <file>. 
to discard changes in working directory) 
no changes added to commit (use "git add" and/or "git commit —a") 
D: restore license. txt 
D: status 
On branch master 
nothing to commit, working tree clean 

* + If we add modified file(version1) to stage then we modified this file(version2) again what will happen?

status 
n branch master 
anges 
(use 
anges 
(use 
(use 
to be committed: 
git 
not 
"git 
"git 
--staged 
restore 
staged for commit: 
to unstage) 
to update what will be committed) 
add 
restore " 
to discard changes in working directory) 

The green words part indicate the version1 file which store in stage

The red words part indicate the version2 file which store in workspace

We can use git commit -m "XXXX" command to upload the version1 file

We can also use git add [FILENAME] to add version2 file to stage and then use commit command to upload the version2 file.

:\Git>git add license. txt 
: \Git>git commit -m "add new version license 
[master 79a2d00] add new version license 
I file changed, 4 insertions(+), 
1 deletion 

* + If you want to undo the operation that track the file you just add, you can use git restore --staged <file>
    - git reset --staged <file>

fatal: you 
.\Git>git 
--staged 
restore 
must specify path (s) to restore 
restore --staged license. txt 

: status 
n branch master 
ntracked files: 
(use "git add < file). " 
to include in what will be committed) 
nothing added to commit but untracked files present (use "git add" 
to track) 

* 1. Using git log command to quire the history committed record.
     + git log

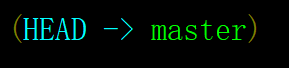
log 
omit 79a2d00f0417867cb738048999cc6d6d2bde2b04 
(HEAD 
master) 
u thor: 
ate: 
EvenYi com) 
Wed Nov 13 11:32:32 2019 -0500 
add new version license 
omit 809cOe2bac9097f5f5271dc87468dd98c8d7f751 
11 : 
ate: 
EvenYi com) 
Wed Nov 13 2019 -0500 
add a license file 
omit 53c6917d995ef800a97b352db7ce02a7cdc9638a 
u thor: 
ate: 
EvenYi <yiyangmagi@gmail. com) 
wed Nov 13 2019 -0500 
add a readme file 

We have three record means we have committed three times.

This commit code is generate(SHA) by git and this code is the id for each commit version. From the id we can find the unique version

C:\Users\Sphinx\AppData\Local\Packages\Microsoft.Office.OneNote_8wekyb3d8bbwe\TempState\msohtmlclip\clip_image014.png

This is HEAD pointer to indicate the latest version



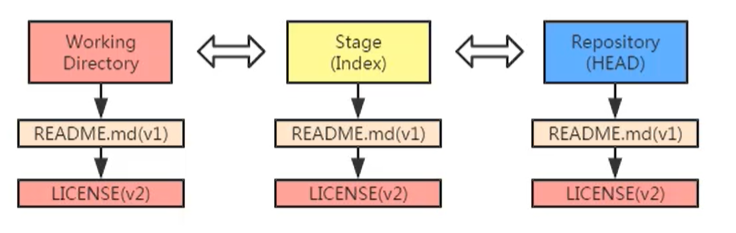
* 1. Reset command

Working 
Directory 
Stage 
(Index) 
omml 
Re posit OW 

Current HEAD point.

Repesitory ( 
72564eb„, 
ucENsE'v•.' 
bd2ab20„ 

* + Every rectangle in repository is a version Snapshot

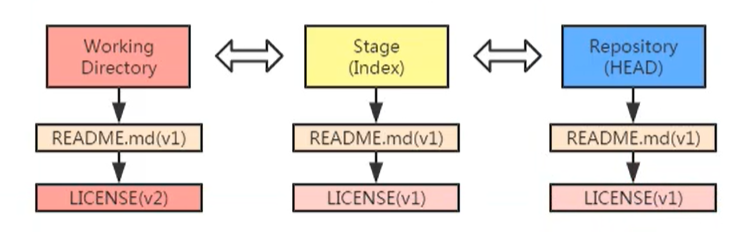


* + git reset HEAD~ or git reset HEAD~1
  + HEAD + ~ means the pervious version of the HEAD version
  + 1 indicate the one ~ git reset HEAD~3 means the pervious 3 version
  + So if I run the command "git reset HEAD~" the reset version is the second (license(v1)) one as follow show.
  + We can use command " git reset ID" instead use HEAD to reset the version

Like" git reset 809c0" == "get reset HEAD~" we just need the id first 5 letter.

log 
commit 79a2d00f0417867cb738048999cc6d6d2bde2b04 
uthor: EvenYi com) 
Wed Nov 13 2019 -0500 
ate: 
add new version license 
commit 809c0e2bac9097f5f5271dc87468dd98c8d7f751 
11thor: EvenYi l. com) 
Wed Nov 13 2019 -0500 
te: 
add a license file 
comfit 53c6917d9950f800a97b352db7ce02a7cdc9638a 
11thor: EvenYi l. com) 
wed Nov 13 2019 -0500 
te: 
add a readme file 
reset HEAD v 
nstaged changes after reset: 
license. txt 
(HEAD 
master) 

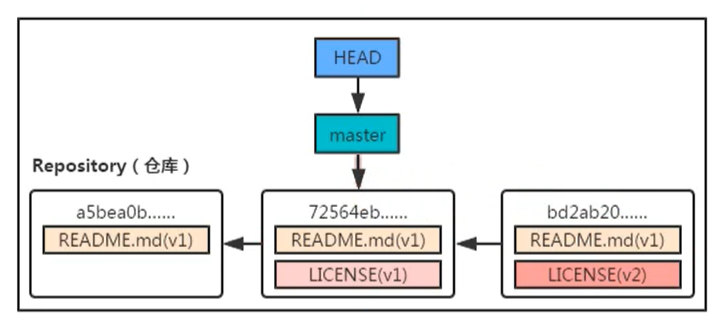
And the second version(license(v1)) is reset to stage tree



Since the license file in working directory is version2 is the latest version, the status is as follow show

status 
n branch master 
hanges not staged for commit: 
(use "git add " 
to update what will be tted) 
(use "git restore .. " 
to discard changes in working directory) 

After run the command "git reset HEAD~" , the HEAD Pointer as follow show. The HEAD Point is point to v1



D: log 
:ommit 809c0e2bac9097f5f5271dc87468dd98c8d7f751 
EvenYi <yiyanglnagigglnail. com> 
Date: 
Wed Nov 13 2019 -0500 
add a license file 
tommit 53c6917d995ef800a97b352db7ce02a7cdc9638a 
EvenYi <yiyanglnagiggmail. com> 
Date: 
Wed Nov 13 2019 -0500 
add a readme file 
(HEAD 
master) 

* + git reset --mixed HEAD~ (the optional parameter --mixed is default)

This command will move the HEAD pointer to pervious version according to the number of ~

Then reset the HEAD pointer pointing version snapshot to stage tree.

* + git reset --soft HEAD~

This command will move the HEAD pointer to pervious version according to the number of ~

This command do not change the context fo stage tree just move the HEAD pointer

* + git reset --hard HEAD~

This command will move the HEAD pointer to pervious version according to the number of ~

Then reset the HEAD pointer pointing version snapshot to stage tree.

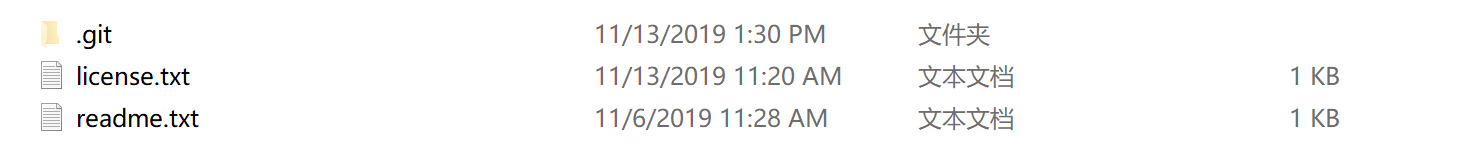
Then reset the version snapshot to working tree

This command not only reset from repository tree to stage tree but also reset from stage tree to working tree.

Eg:

Befor I run git reset --hard HEAD~ the working tree as follow show

D: log 
:ommit 809c0e2bac9097f5f5271dc87468dd98c8d7f751 
EvenYi <yiyanglnagigglnail. com> 
Date: 
Wed Nov 13 2019 -0500 
add a license file 
tommit 53c6917d995ef800a97b352db7ce02a7cdc9638a 
EvenYi <yiyanglnagiggmail. com> 
Date: 
Wed Nov 13 2019 -0500 
add a readme file 
(HEAD 
master) 



After I run git reset --hard HEAD~

D: reset --hard HEADW 
is now at 53c6917 add a readme file 
D: log 
53c6917d9950f800a97b352db7ce02a7cdc9638a (HEAD 
Author: EvenYi l. com) 
wed Nov 13 2019 -0500 
Date: 
add a readme file 
master) 

status 
n branch master 
othing to commit, working tree clean 

_git 
readme.txt 
11/13/2019 1:57 PM 
11/6/2019 1 AM 

asbe 匕 : 
LICENSECv1' 
7255 b - 
LICENSE(V 

D i recto 
REAOMEmi(v1J 
stag. 
REAOME.md(v1) 
READMEn-Z(v11 

* + Reset special file

git reset VERSIONSNAPSHOT FILENAME/PATH

* + Reset to the version snapshot which after that HEAD pointer pointing version. We need to know the version id

git reset VERSIONID

Eg: reset to the latest version snapshot

Before run command

D: log 
commit 53c6917d995ef800a97b352db7ce02a7cdc9638a (HEAD 
master) 
All thor: 
Date: 
EvenYi com) 
Wed Nov 13 2019 -0500 
add a readme file 

After run command

D: reset 79a2d 
Unstaged changes after reset: 
license. txt 

D: log 
commit 79a2d00f0417867cb738048999cc6d6d2bde2b04 
(HEAD 
master) 
Author : 
Date: 
EvenYi <yiyangmagiggmail. com> 
Wed Nov 13 11:32:32 2019 -0500 
add new version license 
commit 809c0e2bac9097f5f5271dc87468dd98c8d7f751 
Author : 
Date: 
EvenYi <yiyangmagiggmail. com> 
Wed Nov 13 2019 -0500 
add a license file 
commit 53c6917d995ef800a97b352db7ce02a7cdc9638a 
Author : 
Date: 
EvenYi <yiyangmagiggmail. com> 
Wed Nov 13 2019 -0500 
add a readme file 

status 
n branch master 
anges not staged for commit: 
(use "git add/ rm " 
to update what wi Il 
(use "git restore (file). " 
to discard changes 
(use "git add" and/or 
o changes added to commit 
be committed) 
in working directory) 
"git commit —a") 

.git 
readme.txt 
11/13/2019 2:09 рм 
11/6/2019 11:28 АМ 

Use git restore <file> command to restore the license.txt to working tree.

Or initially we just use "git reset --hard 79a2d " to reset both stage tree and working.

status 
n branch master 
hanges not staged for commit: 
(use "git add/rm 
to update what will be committed) 
(use "git restore " 
to discard changes in working directory) 
o changes added to commit (use "git add" and/or "git commit —a") 
restore license- txt 

.git 
license.txt 
readme. txt 
11/13/2019 2:12 PM 
11/13/2019 2:12 PM 
11/6/2019 11:28 AM 

* + Command "git reflog"
    - If you do not remenber pervious version snapshot id you can use "git reflog" to quire all the change

reflog 
79a2dOO 
53c6917 
809coe2 
79a2dOO 
809coe2 
(HEAD -> master) 
HEADO{O): reset: moving to 79a2d 
moving to HEAD v 
reset: 
moving to 
reset: 
(HEAD -> master) 
: commit: add new version license 
commit: add a license file 
(initial): add a readme file 
conuli t 

{n} is the history record of the HEAD change

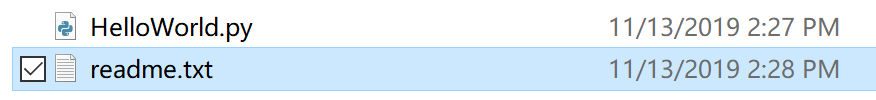
* 1. Compare the different version

Environmental preparation:

* + Create a new repository under "Git2" folder

:\Git2>git init 
Initialized empty Git repository in D:/Git2/. git/ 

* + Create a python HelloWorld.py file and a readme.txt file



C:\Users\Sphinx\AppData\Local\Packages\Microsoft.Office.OneNote_8wekyb3d8bbwe\TempState\msohtmlclip\clip_image040.png

C:\Users\Sphinx\AppData\Local\Packages\Microsoft.Office.OneNote_8wekyb3d8bbwe\TempState\msohtmlclip\clip_image041.png

* + Add and commit two files

: add HelloWor1d. py readme. txt 
: commit —m "add files HelloWr01d. py and readme. txt" 
[master (root-commit) 239cadc] add files HelloWrold. py and readme. txt 
2 files changed, 2 insertions (+) 
create mode 100644 HelloWor1d. py 
create mode 100644 readme. txt 

* + Modify the HelloWorld.py and the readme.txt

Git2> + Helloworld.py 
for i in range(5): 
print( ) 

This is a Hello world programm 
print five l"He110 world" 

* + Compare the version between working tree and stage tree

Command "git diff"

: diff 
iff --git a/HelloWorld. py b/HelloWorld. py 
index 9575d7f.. (1970628 100644 
- a/ HelloWorld. PY 
++ b/He110Wor1d. py 
\ No newline at end of file 
fur i 111 range (5) • 
print ("Hel I "Wnrld") 
-git a/readme. txt b/readme. txt 
iff 
index dadbfle.. 4ed8ffd 100644 
a/readme. txt 
++ b/readme. txt 
-1 +1,2 
No newline at end of file 
print five "Hello world" 
\ No newline at end of file 

"diff --git a/HelloWorld.py b/HelloWorld.py": means compare the HelloWorld.py files between working tree and stage tree

"index 9575d7f..d970628 100644": 9575d7f d970628 is the id of the files, 100644 is the file type and permission

"--- a/HelloWorld.py": "---" means this is a old file "a/" means this file store in stage tree

"+++ b/HelloWorld.py": "+++" means this is a new file "b/" means this file store in working tree

"@@ -1 +1,2 @@": "-" means old file "+" means new file, "+1,2" means new file first row and show 2 row, Since new file include the old file context so it just "-1" do not have the part ",2" behind the "-1"

* + Compare two history version snapshot

Command "git diff ID1 ID2"

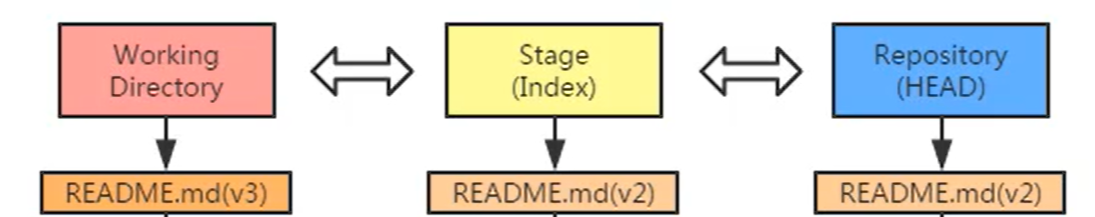
D: commit —anl "add Hello World program and readme file" 
[master 4e2clld] add Hello World program and readme file 
2 files changed, 5 insertions (+) , 2 deletions (-) 

D: log 
commit 4e2c11d6e67af5f990f8158c643b621f36c38f4f (HEAD 
Author: EvenYi <yiyangmagi@gmail. com) 
Date: 
Wed Nov 13 2019 -0500 
add Hello World program and readme file 
commit 239cadcf5fd,12916h2f58e3afOda33f95a6a0129 
Author: EvenYi com> 
Date: 
Wed Nov 13 2019 -0500 
add files HelloWrold. py and readme. txt 
—> master) 

D: diff 4e2c1 239ca 
diff —git a/HelloWorld. py b/HelloWorld. py 
index 72195f2.. 9575d7f 100644 
a/HelloWorld. PY 
b/He110Wor1d. py 
'print 
\ NO newline at end Of file 
diff —git a/ readme. txt b/ readme. txt 
index 4ed8ffd.. dadbfle 100644 
a/readme. txt 
b,'readme. txt 
" -1,2 
\ No newline at end of 
file 
prograru:l 
\ Nn newl ine at end 
file 

* + Compare current working tree version with repository tree version snapshot
    - Modify readme.txt

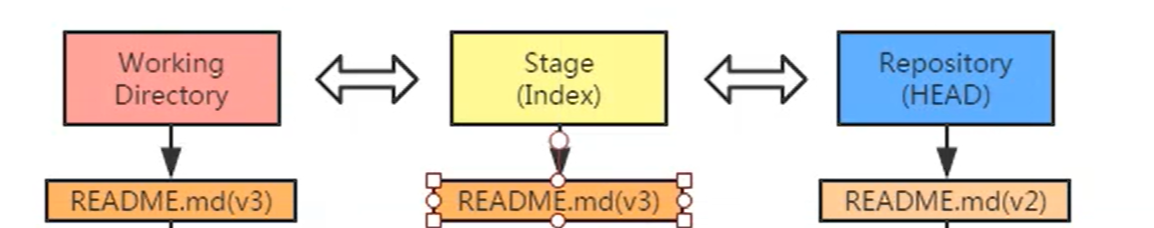
This is a Hello world programm 
print five "Hello world" 
This is V 



* + Command "git diff ID"
  + If just want to compare working tree version with repository latest version snapshot we can use "git diff HEAD"

D: diff 4e2c 
git a/HelloWorld. py b/HelloWorld. py 
diff 
index 72195f2.. 31c9be3 100644 
a/HelloWorld. PY 
b/He110Wor1d. py 
print ("HelloWor1d") 
for i in range (5) . 
print ("HelloWorld") 
\ No newline at end of file 
git a/readme. txt b/readme. txt 
diff 
index 4ed8ffd.. 605f4d8 100644 
a/readme. txt 
b/readme. txt 
This is a Hello world programm 
\ No newline 
•print five 
\ No newline 
at end of file 
"Hello world" 
at end of file 

* + Compare stage tree version with repository snapshot
    - Add v3 readme.txt to stage tree



* + Command "git diff --cached [id]"

D: diff --cached 
diff --git a/ readme. txt b/readme. txt 
index 4ed8ffd.. 605f4d8 100644 
- a/readme. txt 
++4 b/readlne. txt 
This is a Hello world progranun 
\ No newline 
+print five 
+This is vg 
\ No newline 
at end Of file 
"110110 world" 
at end of file 

