Checking out commits in a Git

mkdir checkrepo

cd checkrepo

git init

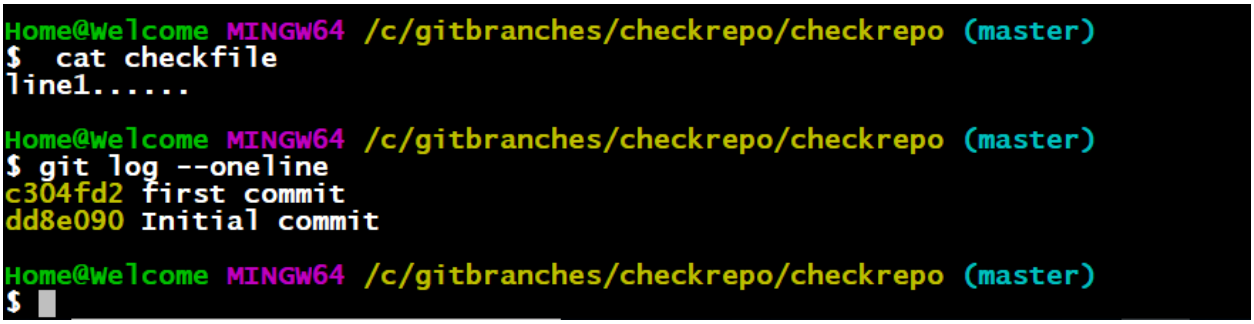
vi checkfile

---------- Insert line1…..

git add .

git commit –m “first commit”

git log –oneline



```
Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$ git log --oneline
1b48b0a second commit
c304fd2 first commit
dd8e090 Initial commit

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$ vi checkfile

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$ git add .
warning: LF will be replaced by CRLF in checkfile.
The file will have its original line endings in your working directory.

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$ git commit -m "third commit"
[master f9e2fcf] third commit
 1 file changed, 1 insertion(+), 1 deletion(-)

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$ git log --oneline
f9e2fcf third commit
1b48b0a second commit
c304fd2 first commit
dd8e090 Initial commit

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$ cat checkfile
line1......
line2......
experiment on commit

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$
```

```
Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$ git log --oneline
47989ac fifth commit
5d73d2e fourth commit
f9e2fcf third commit
1b48b0a second commit
c304fd2 first commit
dd8e090 Initial commit

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$ cat checkfile
line1......
line2......
experiment on commit
line4......
line5......

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$
```

We have done 5 commits.

Here I can say the final commit means fifth comment called as HEAD of our repository.

Now I want to retain changes till second commit

git checkout 1b48b0a (above example commit id of second commit message)

```
Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$ git log --oneline
47989ac fifth commit
5d73d2e fourth commit
f9e2fcf third commit
1b48b0a second commit
c304fd2 first commit
dd8e090 Initial commit

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$ cat checkfile
line1......
line2......
experiment on commit
line4......
line5......

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$ git checkout 1b48b0a
Note: checking out '1b48b0a'.

You are in 'detached HEAD' state. You can look around, make experimental
changes and commit them, and you can discard any commits you make in this
state without impacting any branches by performing another checkout.

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

  git checkout -b <new-branch-name>

HEAD is now at 1b48b0a... second commit

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo ((1b48b0a...))
$
```

Now it is in detached HEAD state it is not belongs to master branch or not with any other local branch. That's why it is giving some valuable message for above command. It is says that read above message.

```
Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$ git checkout 1b48b0a
Note: checking out '1b48b0a'.

You are in 'detached HEAD' state. You can look around, make experimental
changes and commit them, and you can discard any commits you make in this
state without impacting any branches by performing another checkout.

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

  git checkout -b <new-branch-name>

HEAD is now at 1b48b0a... second commit

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo ((1b48b0a...))
$
```

Now am going back to my master branch given below

```
Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo ((1b48b0a...))
$ git log --oneline
1b48b0a second commit
c304fd2 first commit
dd8e090 Initial commit

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo ((1b48b0a...))
$ git checkout master
Previous HEAD position was 1b48b0a... second commit
Switched to branch 'master'
Your branch is ahead of 'origin/master' by 5 commits.
  (use "git push" to publish your local commits)

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$ git log --oneline
47989ac fifth commit
5d73d2e fourth commit
f9e2fcf third commit
1b48b0a second commit
c304fd2 first commit
dd8e090 Initial commit

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$
```

Again we are get back to our detached state like previous.

```
Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$ git checkout 1b48b0a
Note: checking out '1b48b0a'.

You are in 'detached HEAD' state. You can look around, make experimental
changes and commit them, and you can discard any commits you make in this
state without impacting any branches by performing another checkout.

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

  git checkout -b <new-branch-name>

HEAD is now at 1b48b0a... second commit

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo ((1b48b0a...))
$
```

In this state of my project we can compile file we can edit files

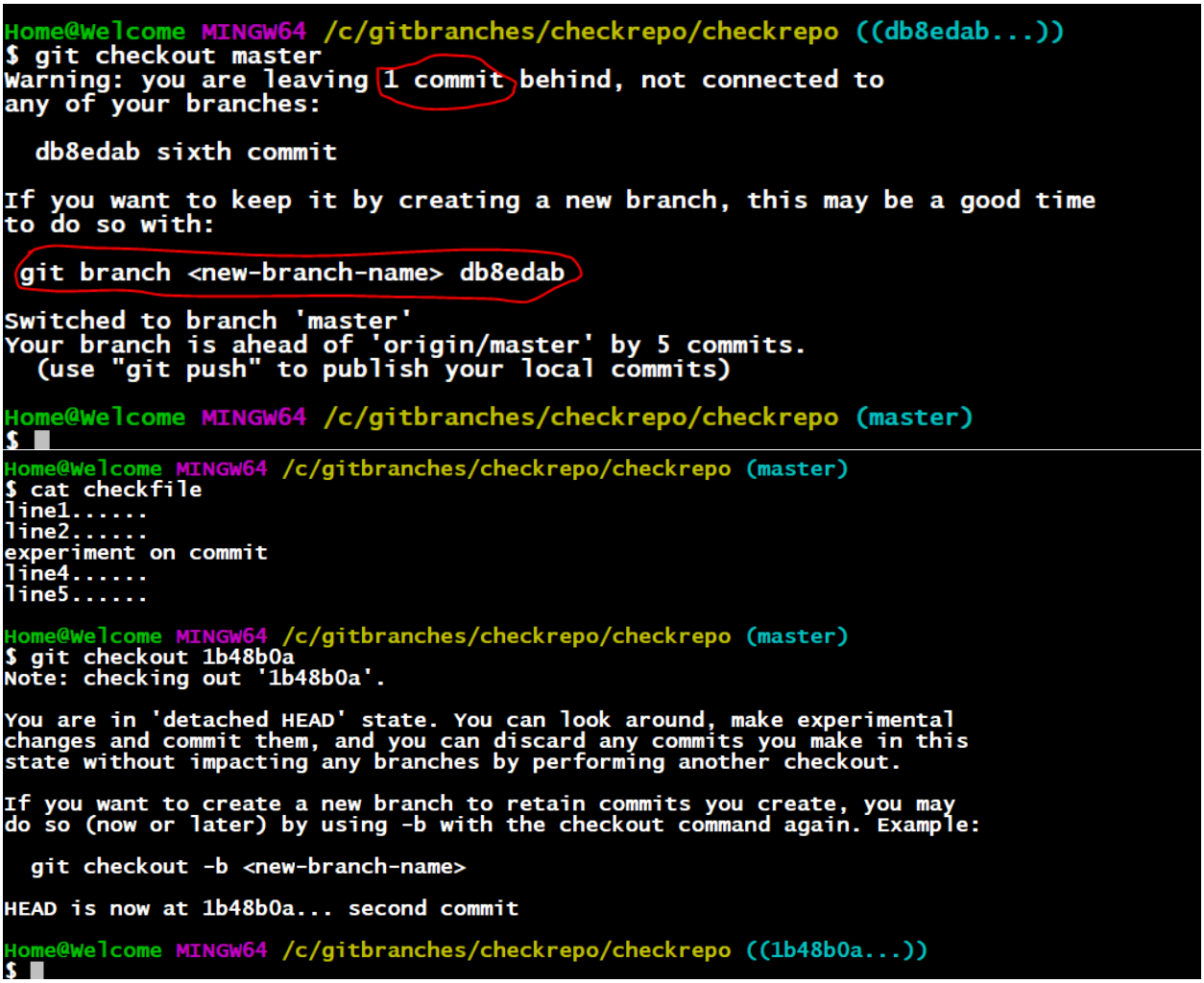
```
Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo ((1b48b0a...))
$ git log --oneline
1b48b0a second commit
c304fd2 first commit
dd8e090 Initial commit

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo ((1b48b0a...))
$

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo ((1b48b0a...))
$ cat checkfile
line1......
line2......


Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo ((1b48b0a...))
$
```

But we were having 5 lines but we came to second commit so we are having only 2 lines of data. And we are going to do one more commit here in this state of project.

```
Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo ((1b48b0a...))
$ cat checkfile
line1......
line2......


Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo ((1b48b0a...))
$ vi checkfile

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo ((1b48b0a...))
$ git commit -am "sixth commit"
[detached HEAD db8edab] sixth commit
 1 file changed, 1 insertion(+), 1 deletion(-)

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo ((db8edab...))
$ cat checkfile
line1......
line2......
line6......

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo ((db8edab...))
$
```

```
Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo ((db8edab...))
$ git log --oneline
db8edab sixth commit
1b48b0a second commit
c304fd2 first commit
dd8e090 Initial commit

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo ((db8edab...))
$
```

I can see here now clearly sixth commit but it is not permanent if we go back to our HEAD master this commit will go off.


Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo ((db8edab...))
$ git checkout master
Warning: you are leaving 1 commit behind, not connected to
any of your branches:
db8edab sixth commit
If you want to keep it by creating a new branch, this may be a good time
to do so with:
git branch <new-branch-name> db8edab
Switched to branch 'master'
Your branch is ahead of 'origin/master' by 5 commits.
(use "git push" to publish your local commits)
Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$
Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$ cat checkfile
line1......
line2......
experiment on commit
line4......
line5......
Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$ git checkout 1b48b0a
Note: checking out '1b48b0a'.
You are in 'detached HEAD' state. You can look around, make experimental
changes and commit them, and you can discard any commits you make in this
state without impacting any branches by performing another checkout.
If you want to create a new branch to retain commits you create, you may
do so (now or later) by using -b with the checkout command again. Example:
git checkout -b <new-branch-name>
HEAD is now at 1b48b0a... second commit
Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo ((1b48b0a...))
$

```
Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$ git checkout 1b48b0a
Note: checking out '1b48b0a'.

You are in 'detached HEAD' state. You can look around, make experimental
changes and commit them, and you can discard any commits you make in this
state without impacting any branches by performing another checkout.

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

  git checkout -b <new-branch-name>

HEAD is now at 1b48b0a... second commit

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo ((1b48b0a...))
$ git log --oneline
1b48b0a second commit
c304fd2 first commit
dd8e090 Initial commit

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo ((1b48b0a...))
$ cat checkfile
line1......
line2......


Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo ((1b48b0a...))
$
```

Now we are unable to see my kline6 code with sixth commit.

Again we are going to add line6 and commiting now

```
Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo ((1b48b0a...))
$ vi checkfile

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo ((1b48b0a...))
$ git commit -am "sixthcommit"
[detached HEAD 2417994] sixthcommit
 1 file changed, 1 insertion(+), 1 deletion(-)

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo ((2417994...))
$ git log --oneline
2417994 sixthcommit
1b48b0a second commit
c304fd2 first commit
dd8e090 Initial commit

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo ((2417994...))
$ cat checkfile
line1......
line2......
line6......

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo ((2417994...))
$
```

If we want to have this change we have to create a new branch now.

```
Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo ((2417994...))
$ git checkout -b newbranch
Switched to a new branch 'newbranch'

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (newbranch)
$ git log --oneline
2417994 sixthcommit
1b48b0a second commit
c304fd2 first commit
dd8e090 Initial commit

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (newbranch)
$ cat checkfile
line1......
line2......
line6......

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (newbranch)
$
```

Now we are going back to master branch to test

```
Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (newbranch)
$ git checkout master
Switched to branch 'master'
Your branch is ahead of 'origin/master' by 5 commits.
  (use "git push" to publish your local commits)

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$ cat checkfile
line1......
line2......
experiment on commit
line4......
line5......

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$ git log -oneline
fatal: unrecognized argument: -oneline

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$ git log --oneline
47989ac fifth commit
5d73d2e fourth commit
f9e2fcf third commit
1b48b0a second commit
c304fd2 first commit
dd8e090 Initial commit

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$
```

Now going back to new branch to verify for sicth file

```
Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$ git checkout newbranch
Switched to branch 'newbranch'

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (newbranch)
$ git log --oneline
2417994 sixthcommit
1b48b0a second commit
c304fd2 first commit
dd8e090 Initial commit

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (newbranch)
$ cat checkfile
line1......
line2......
line6......

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (newbranch)
$
```

This is very good we have created new branch when we want to some parallel development.

```
Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$ git log --oneline
47989ac fifth commit
5d73d2e fourth commit
f9e2fcf third commit
1b48b0a second commit
c304fd2 first commit
dd8e090 Initial commit

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$ cat checkfile
line1......
line2......
experiment on commit
line4......
line5......

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$ git checkout f9e2fcf checkfile

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$ git status
On branch master
Your branch is ahead of 'origin/master' by 5 commits.
  (use "git push" to publish your local commits)
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   checkfile

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
```

Earlier when we used checkout for project state of commits project status not changed but when checkout for specific file commits project state also can be changed.

If you want to go back to latest data of a file you can cal back using below command

```
Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$ git checkout HEAD checkfile

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$ git status
On branch master
Your branch is ahead of 'origin/master' by 5 commits.
  (use "git push" to publish your local commits)
nothing to commit, working tree clean

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$
```

Am just going to back to specific file of commit

```
Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$ git checkout f9e2fcf checkfile

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$ git status
On branch master
Your branch is ahead of 'origin/master' by 5 commits.
  (use "git push" to publish your local commits)
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   checkfile


Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$ git commit -m "sixth commit checkfile"
[master 0d6f2d3] sixth commit checkfile
 1 file changed, 2 deletions(-)

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$ git log --oneline
0d6f2d3 sixth commit checkfile
47989ac fifth commit
5d73d2e fourth commit
f9e2fcf third commit
1b48b0a second commit
c304fd2 first commit
dd8e090 Initial commit

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$
```

```
Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$ cat checkfile
line1......
line2......
experiment on commit

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$
```

Now we can check the data with that specific commit only later commits data has been deleted.

I have just added one more line into and again I want to go back previous HEAD state use below commands

```
Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$ vi checkfile

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$ git add .

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$ git commit -m "git revert to previous change"
[master d6e9015] git revert to previous change
 1 file changed, 1 insertion(+)

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$ git log --oneline
d6e9015 git revert to previous change
0d6f2d3 sixth commit checkfile
47989ac fifth commit
5d73d2e fourth commit
f9e2fcf third commit
1b48b0a second commit
c304fd2 first commit
dd8e090 Initial commit
```

```
Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$ git revert HEAD
[master b99e96b] Revert "git revert to previous change"
 1 file changed, 1 deletion(-)

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$ git log --oneline
b99e96b Revert "git revert to previous change"
d6e9015 git revert to previous change
0d6f2d3 sixth commit checkfile
47989ac fifth commit
5d73d2e fourth commit
f9e2fcf third commit
1b48b0a second commit
c304fd2 first commit
dd8e090 Initial commit

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$ cat checkfile
line1......
line2......
experiment on commit

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$
```

What ever we just added line4 that again removed using HEAD revert command.

Reset :

Reset we can use when we add any file to staging area there we can revert to modified state like below

```
Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$ vi checkfile

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$ git add .

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$ git status
On branch master
Your branch is ahead of 'origin/master' by 9 commits.
  (use "git push" to publish your local commits)
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   checkfile


Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$ git reset checkfile
Unstaged changes after reset:
M       checkfile

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$ git status
On branch master
Your branch is ahead of 'origin/master' by 9 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 checkout -- <file>..." to discard changes in working directory)

        modified:   checkfile

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

Home@Welcome MINGW64 /c/gitbranches/checkrepo/checkrepo (master)
$
```

Now we just add 2 files like checkfile then do the same thing that time while doing reset no need to give any file name just we have to give below command

git reset

```
HP-PC@HP git-reset-demo-2 (master) $ git reset
Unstaged changes after reset:
M       goldies-time-table
M       jimmys-time-table
HP-PC@HP git-reset-demo-2 (master) $ git status
On branch master
Your branch is up-to-date with 'origin/master'.
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working director
y)

        modified:   goldies-time-table
        modified:   jimmys-time-table

no changes added to commit (use "git add" and/or "git commit -a")
HP-PC@HP git-reset-demo-2 (master) $
```

Now we have done just staging area to modifying state but now using reset we can do moving from staging area plus removing modifications also.

git reset –hard

git log –oneline

we can do same way for commit ids also why means in our project we may have lots of files so in this example if we want to practice we need to have more than one file in staging area.

git reset <commitid>

git status

then add only one file

git add <filename> from above files

we can do for hard also

git reset –hard <commit id>.

==========