After creating .git repository inside git directory(/working tree):

See working directory contents using (ls -la) command.

I have added any small abc.c file in working tree using file explorer.

|  |
| --- |
| HP@DESKTOP-FOKS471 MINGW64 /i/COLLEGE/CODING/Git/learnGit (master)  $ ls -la  total 5  drwxr-xr-x 1 HP 197121 0 Sep 27 15:42 ./  drwxr-xr-x 1 HP 197121 0 Sep 27 15:42 ../  drwxr-xr-x 1 HP 197121 0 Sep 27 07:30 .git/  -rw-r--r-- 1 HP 197121 56 May 19 13:01 abc.c |

Staging area(INDEX) is a git managed file that acts like a virtual Index of all the changes that are made and that are going to go in our next commit. Adding files to commit means actually taking real snapshots (backup) of the files rather than just indexing.

See status of our staging area that what file is added to it and what not.

|  |
| --- |
| HP@DESKTOP-FOKS471 MINGW64 /i/COLLEGE/CODING/Git/learnGit (master)  $ git status  On branch master  No commits yet  Untracked files:  (use "git add <file>..." to include in what will be committed)  abc.c  nothing added to commit but untracked files present (use "git add" to track) |

We see that abc.c file is not yet tracked .All those files not added to staging area and not committed are not tracked. Use (git add) command to add file to staging area. Then use (git commit) command to start tracking of the file structure.

Every commit needs an instruction message to be written by us or unless commit won’t occur.

|  |
| --- |
| HP@DESKTOP-FOKS471 MINGW64 /i/COLLEGE/CODING/Git/learnGit (master)  $ git add abc.c  HP@DESKTOP-FOKS471 MINGW64 /i/COLLEGE/CODING/Git/learnGit (master)  $ git commit |

While initial git commit command, it will open a separate VIM text editor. To start and move further. Press I button to enable editing then just enter command any text message like (Initial commit) and commit will be made. To exit press esc key then type command ( :wq ) and press enter. It will show output like below table… then we can check the status of commit using (git status)

|  |
| --- |
| HP@DESKTOP-FOKS471 MINGW64 /i/COLLEGE/CODING/Git/learnGit (master)  $ git commit  [master (root-commit) 7955e00] Initial Commit  1 file changed, 5 insertions(+)  create mode 100644 abc.c  HP@DESKTOP-FOKS471 MINGW64 /i/COLLEGE/CODING/Git/learnGit (master)  $ git status  On branch master  nothing to commit, working tree clean |

To better understand git functioning of git we add few more files in working tree by creating blank new files using (touch) command,

|  |
| --- |
| HP@DESKTOP-FOKS471 MINGW64 /i/COLLEGE/CODING/Git/learnGit (master)  $ touch about.c  HP@DESKTOP-FOKS471 MINGW64 /i/COLLEGE/CODING/Git/learnGit (master)  $ touch contact.c  HP@DESKTOP-FOKS471 MINGW64 /i/COLLEGE/CODING/Git/learnGit (master)  $ touch monuments.c |

Then we add all these new files to staging area all in one go using (–A) parameter

|  |
| --- |
| HP@DESKTOP-FOKS471 MINGW64 /i/COLLEGE/CODING/Git/learnGit (master)  $ git add -A  HP@DESKTOP-FOKS471 MINGW64 /i/COLLEGE/CODING/Git/learnGit (master)  $ git status  On branch master  Changes to be committed:  (use "git restore --staged <file>..." to unstage)  new file: about.c  new file: contact.c  new file: monuments.c |

Now if we add something or mare changes in any of our staged files and then check status of directory then it will look like =

|  |
| --- |
| HP@DESKTOP-FOKS471 MINGW64 /i/COLLEGE/CODING/Git/learnGit (master)  $ git status  On branch master  Changes to be committed:  (use "git restore --staged <file>..." to unstage)  new file: about.c  new file: contact.c  new file: monuments.c  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: contact.c |

So it less us the modified file. To continue tracking of all files back together, we add the modified file back to the staging area using (git add filename) or simply use (-A) parameter with (git add) to add all files to staging area in one go…

|  |
| --- |
| HP@DESKTOP-FOKS471 MINGW64 /i/COLLEGE/CODING/Git/learnGit (master)  $ git add -A  HP@DESKTOP-FOKS471 MINGW64 /i/COLLEGE/CODING/Git/learnGit (master)  $ git status  On branch master  Changes to be committed:  (use "git restore --staged <file>..." to unstage)  new file: about.c  new file: contact.c  new file: monuments.c |

Then we can commit all together using (commit –m “ ”) shortcut. Here we used (-m){dash m flag} as shortcut to prevent opening editor.

|  |
| --- |
| HP@DESKTOP-FOKS471 MINGW64 /i/COLLEGE/CODING/Git/learnGit (master)  $ git commit -m "Added more c files"  [master 9d05ba0] Added more c files  3 files changed, 5 insertions(+)  create mode 100644 about.c  create mode 100644 contact.c  create mode 100644 monuments.c |

If we want to edit a file, we can open it from git bash directly into our favourite editor like atom editor in following way=

|  |
| --- |
| HP@DESKTOP-FOKS471 MINGW64 /i/COLLEGE/CODING/Git/learnGit (master)  $ atom abc.c |

We see that once again, we have no changes to commit.

|  |
| --- |
| HP@DESKTOP-FOKS471 MINGW64 /i/COLLEGE/CODING/Git/learnGit (master)  $ git status  On branch master  nothing to commit, working tree clean |

Because the change we made has gone through the full cycle of modified, staged and committed. So to sum up, we work on modified files in our working tree. When they're ready, we staged these files by adding them to the staging area.

Finally, we commit the changes sitting in our staging area, which takes a snapshot of those files and stores them in the database that lives in the Git directory.

Inshort = We make changes to our files, stage them with git add, and commit them with git commit.