**What is there in .git?**

Remove the file fruits.c using the command **rm fruits.c**
We do not have any files in this dir. Now we will check the output of few git commands in this empty dir.

git status will give the status of the files if they are being tracked by git or not.

```
fms@git_practice:~/dev_fruits$
fms@git_practice:~/dev_fruits$ git status
On branch master
No commits yet
nothing to commit (create/copy files and use "git add" to track)
fms@git_practice:~/dev_fruits$
```

git branch will give the list of local branches

```
fms@git_practice:~/dev_fruits$ git branch
fms@git_practice:~/dev_fruits$
```

git log will give the list of commits made on the current branch

```
fms@git_practice:~/dev_fruits$ git log
fatal: your current branch 'master' does not have any commits yet
fms@git_practice:~/dev_fruits$
```

git diff will give the difference between the local workspace and the staging area

```
fms@git_practice:~/dev_fruits$ git diff
fms@git_practice:~/dev_fruits$
```

git show will give the details of the latest/specified commit

```
fms@git_practice:~/dev_fruits$ git show
fatal: your current branch 'master' does not have any commits yet
fms@git_practice:~/dev_fruits$
```

We will learn more about them.  For time being, we need to understand how to issue the git commands and what parameters to pass.

Now let us focus on creating revisions using git.

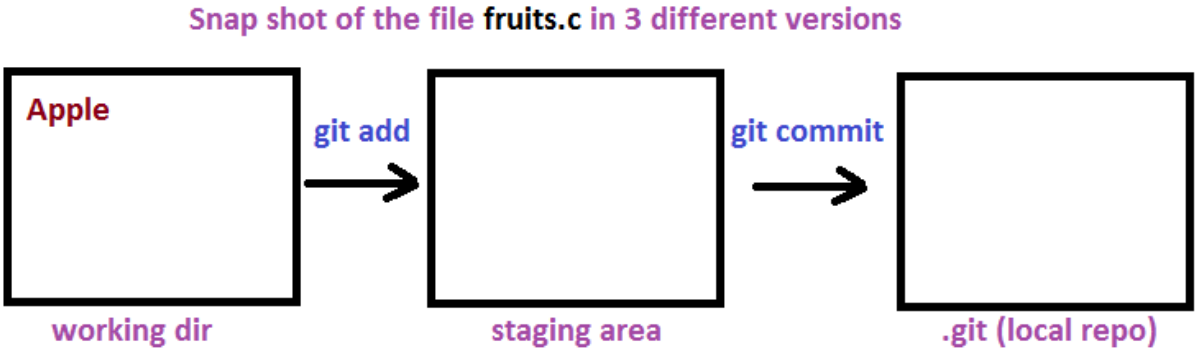
A developer (dev_fruits) has created a file fruits.c with one line ‘Apple’.

```
fms@git_practice:~/dev_fruits$ vi fruits.c
fms@git_practice:~/dev_fruits$
fms@git_practice:~/dev_fruits$ cat fruits.c
Apple
fms@git_practice:~/dev_fruits$
```

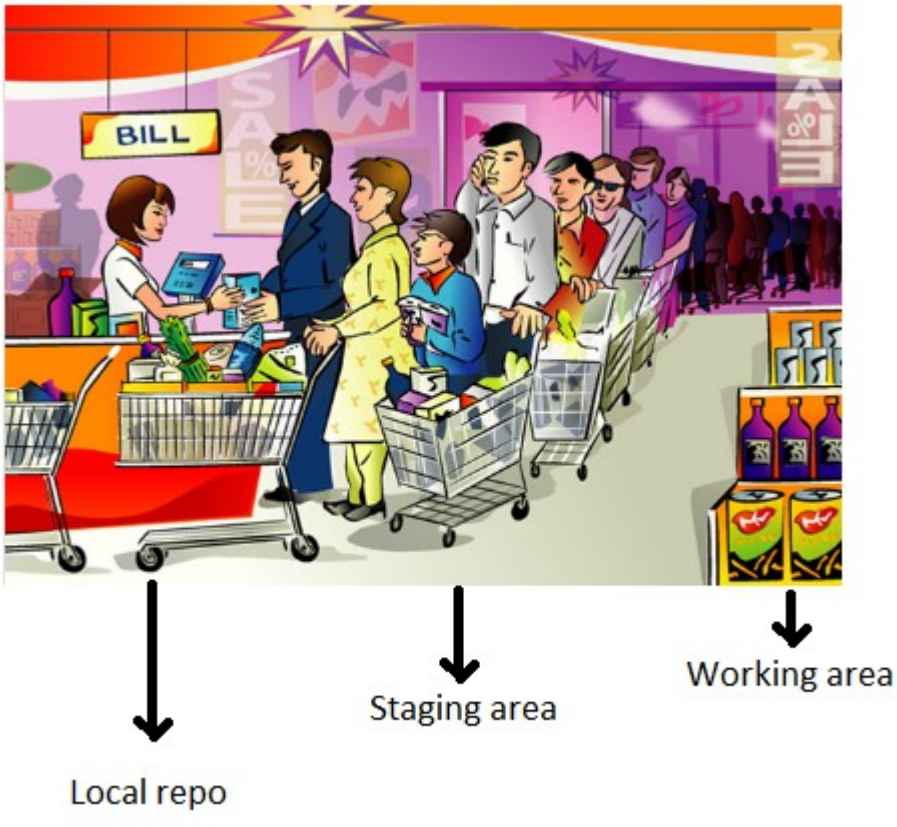
Since git is initialized in this directory, git starts to track this file.

```
fms@git_practice:~/dev_fruits$ git status
On branch master
No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    fruits.c

nothing added to commit but untracked files present (use "git add" to track)
fms@git_practice:~/dev_fruits$
```

git status says that the file fruits.c is under ‘Untracked’ files.
So any changes made to this file will not get tracked until we add the file to git.




Let us add this file to staging area using git add.
Staging area is nothing but index file in .git




1. In working area we have all our files/directories (items in the racks).
2. Whichever file we want to track can be placed in staging area (cart).

   If we do not want to track any of those files in staging area or if we would like modify the file which is already in the staging area, we can always reset it and put it back in the working area. This is something like selecting and deselecting the items in the cart before going to the bill counter.

3. Local repo is like bill counter. Once we do a commit, it is like paying the bill. All the commits will be in .git and the history or order of these commits cannot be changed later.

With this understanding, let us go ahead and add the file to staging area.


fms@git_practice:~/dev_fruits$ **git add fruits.c**
fms@git_practice:~/dev_fruits$

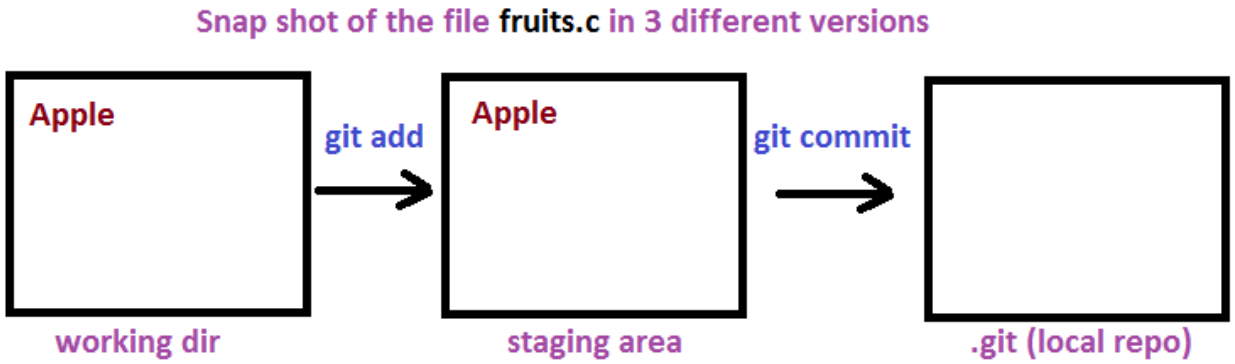
fms@git_practice:~/dev_fruits$ **git status**
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

    new file:   fruits.c

fms@git_practice:~/dev_fruits$




Now we will move the file to local repo by using git commit.
Git commit expects a commit message. First line of the commit should be in one line explaining why the commit is being made (what bug you are fixing). It can be followed by a paragraph to explain in detail.

fms@git_practice:~/dev_fruits$ **git commit -m "created a file fruits.c"**

*** Please tell me who you are.

Run

  git config --global user.email "you@example.com"
  git config --global user.name "Your Name"

to set your account's default identity.
Omit --global to set the identity only in this repository.

fatal: unable to auto-detect email address (got 'fms@git_practice.(none)')
fms@git_practice:~/dev_fruits$

git is unable to identify who is committing the changes. So configure your name and email using below 2 commands before doing your first commit.

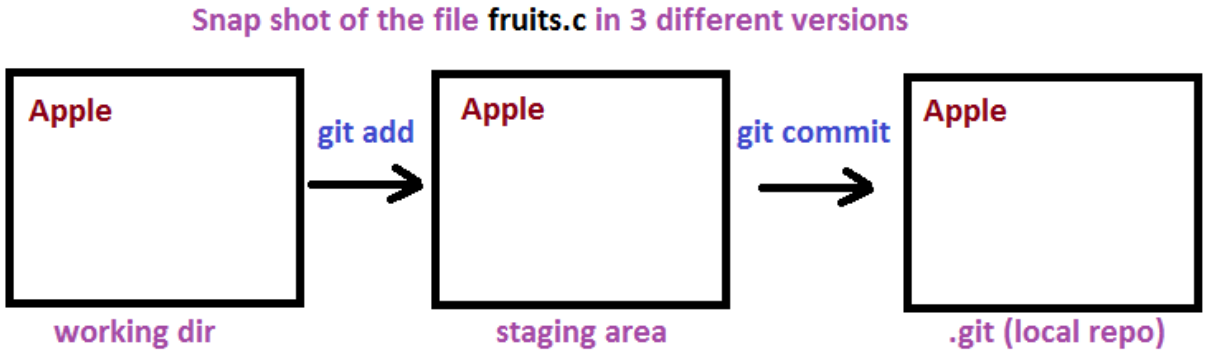
fms@git_practice:~/dev_fruits$ **git config --global user.email "dev_fruits@FRUITS.com"**
fms@git_practice:~/dev_fruits$

fms@git_practice:~/dev_fruits$ **git config --global user.name "Dev-Fruits"**
fms@git_practice:~/dev_fruits$

Now try to commit the changes as below with a commit text.

fms@git_practice:~/dev_fruits$ **git commit -m "created a file fruits.c"**
[master (root-commit) 36c5a24] created a file fruits.c
 1 file changed, 1 insertion(+)
 create mode 100644 fruits.c
fms@git_practice:~/dev_fruits$

This is how it looks.




Ok. Check git log for the commit we made just now.

```
fms@git_practice:~/dev_fruits$ git log
commit 36c5a24d35f6103fb391139008161f938bcefbaf (HEAD -> master)
Author: Dev-Fruits <dev_fruits@FRUITS.com>
Date:   Mon Nov 5 17:33:09 2018 +0530

    created a file fruits.c
fms@git_practice:~/dev_fruits$
```

commit number is too long right. It is called sha number which is unique for every commit. It makes use of SHA1 algorithm to generate this unique number.

It also shows the author of this commit along with the time stamp. With this we can identify who changed what.

Since the file that is modified/created is committed, git status should be clean.

```
fms@git_practice:~/dev_fruits$ git status
On branch master
nothing to commit, working tree clean
fms@git_practice:~/dev_fruits$
```

Now after making this commit, we will see if .git is tracking these revisions.

```
fms@git_practice:~/dev_fruits$ cd .git
fms@git_practice:~/dev_fruits/.git$

fms@git_practice:~/dev_fruits/.git$ ls -al
total 52
```

```
drwxr-xr-x 8 fms fms 4096 Nov  5 17:33 .
drwxr-xr-x 3 fms fms 4096 Nov  5 17:30 ..
drwxr-xr-x 2 fms fms 4096 Nov  5 17:29 branches
-rw-r--r-- 1 fms fms   24 Nov  5 17:33 COMMIT_EDITMSG
-rw-r--r-- 1 fms fms   92 Nov  5 17:29 config
-rw-r--r-- 1 fms fms   73 Nov  5 17:29 description
-rw-r--r-- 1 fms fms   23 Nov  5 17:29 HEAD
drwxr-xr-x 2 fms fms 4096 Nov  5 17:29 hooks
-rw-r--r-- 1 fms fms  137 Nov  5 17:31 index
drwxr-xr-x 2 fms fms 4096 Nov  5 17:29 info
drwxr-xr-x 3 fms fms 4096 Nov  5 17:33 logs
drwxr-xr-x 7 fms fms 4096 Nov  5 17:33 objects
drwxr-xr-x 4 fms fms 4096 Nov  5 17:29 refs
fms@git_practice:~/dev_fruits/.git$

/* commit message is displayed here */
fms@git_practice:~/dev_fruits/.git$ cat COMMIT_EDITMSG
created a file fruits.c

/* config details */
fms@git_practice:~/dev_fruits/.git$
fms@git_practice:~/dev_fruits/.git$ cat config
[core]
    repositoryformatversion = 0
    filemode = true
    bare = false
    logallrefupdates = true
fms@git_practice:~/dev_fruits/.git$

/* description of the project */
fms@git_practice:~/dev_fruits/.git$ cat description
Unnamed repository; edit this file 'description' to name the repository.
fms@git_practice:~/dev_fruits/.git$

/* current branch info */
fms@git_practice:~/dev_fruits/.git$ cat HEAD
ref: refs/heads/master
fms@git_practice:~/dev_fruits/.git$

/* index that stores the last commit */
fms@git_practice:~/dev_fruits/.git$ cat index
DIRC [ 0     \j [ 0     \j   B       ń    0   < 5 w] fruits.cTREE 1 0
&     y GHQ )] ] *_ m   x  Zq  ;  ý@*
```

fms@git_practice:~/dev_fruits/.git$

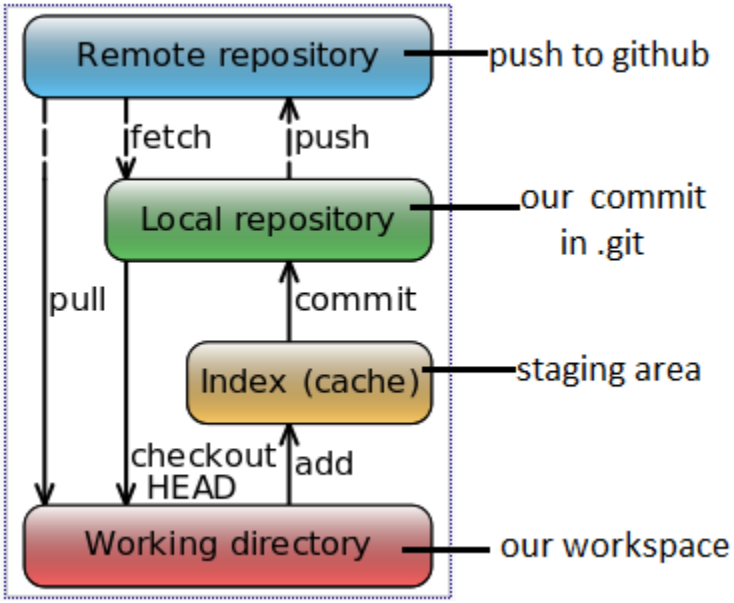
/* list of branches and the info related to each branch */
fms@git_practice:~/dev_fruits/.git$ **cd branches/**
fms@git_practice:~/dev_fruits/.git/branches$
fms@git_practice:~/dev_fruits/.git/branches$ **ls**
fms@git_practice:~/dev_fruits/.git/branches$
/* log info */
fms@git_practice:~/dev_fruits/.git/branches$ **cd ../logs**
fms@git_practice:~/dev_fruits/.git/logs$
fms@git_practice:~/dev_fruits/.git/logs$ **ls**
HEAD  refs
fms@git_practice:~/dev_fruits/.git/logs$ **cat HEAD**
0000000000000000000000000000000000000000 36c5a24d35f6103fb391139008161f938bcefbaf Dev-Fruits <dev_fruits@FRUITS.com> 1541419389 +0530    commit (initial): created a file fruits.c
fms@git_practice:~/dev_fruits/.git/logs$

Notice that this number stored in HEAD is same as the commit number we got in git log

fms@git_practice:~/dev_fruits$ **git log**
commit 36c5a24d35f6103fb391139008161f938bcefbaf (HEAD -> master)
Author: Dev-Fruits <dev_fruits@FRUITS.com>
Date:   Mon Nov 5 17:33:09 2018 +0530

    created a file fruits.c
fms@git_practice:~/dev_fruits$

So with this evidence, we are now sure that git tracks every git operation we perform on the files in this directory (where we initialized git). Entire history of the files is stored in .git.  We can now push our first commit to github.


Remote repository
push to github
fetch
push
Local repository
our commit
in .git
pull
commit
Index (cache)
staging area
checkout
HEAD
add
Working directory
our workspace