

## GIT

1. It is used for [Source code Management](#) in Software development
2. It is a decentralized system
3. It can be used to keep track of changes in the files of a project
4. It allow you to do [nonlinear Development](#)

### Git Architecture

There are two types of environment

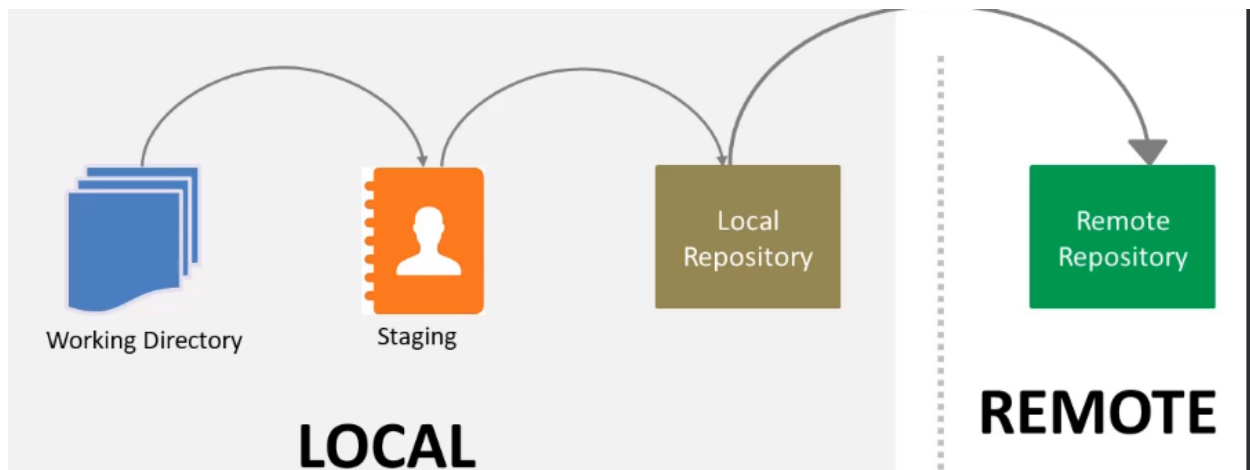
Local -> only [you](#) can access the code

Remote -> It is a [Shared repository](#) you can [push](#) your code form [Local-> Remote](#)

Working with your team

Sink your local repository to [Pull](#)

Ex



### Commands

Git init -> to initialize a repository on local

Git clone-> to clone a remote repo on local

## Creation of remote repository

Initialize your files as Git repository

\$ -> cmd-> GitBash

\$ Pwd.

\$ cd/ -> Location of the folder

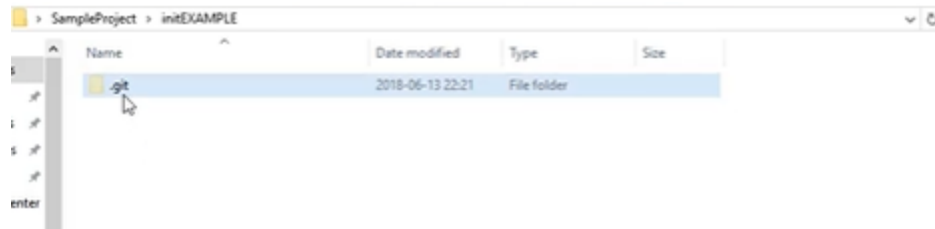
**\$git init**

- After this commands there is an hidden folder is created and inside that folder there is many sub folders in it ->(configuration files )
- It is actually your local git setup
- Local, working copy, Staging Environment, Local repository -> they are not different stored in different folders they are stored in these configuration files only
- You cannot see (configuration files ) as a physical files
- Using GitBash to access it

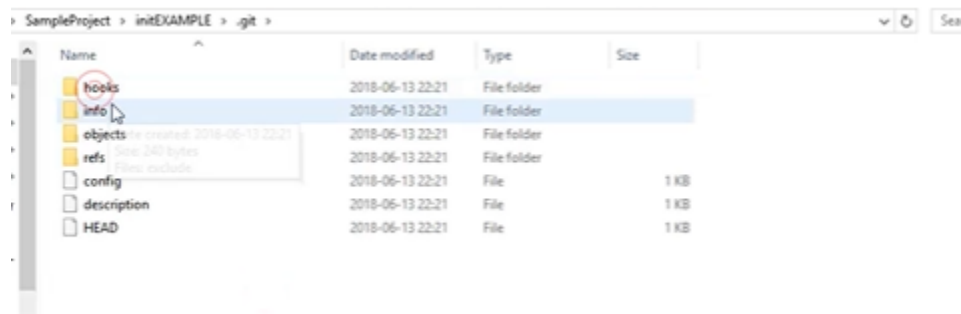
Ex code to create remote repository

```
CKGS5099@EQ-EQ6285240 MINGW64 ~  
$ pwd  
/c/Users/CKGS5099  
  
CKGS5099@EQ-EQ6285240 MINGW64 ~  
$ cd Desktop/SampleProject/initEXAMPLE/  
  
CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/SampleProject/initEXAMPLE  
$ git init  
Initialized empty Git repository in C:/Users/CKGS5099/Desktop/SampleProject/initEXAMPLE/.git/  
  
CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/SampleProject/initEXAMPLE (master)
```

**Created folder**



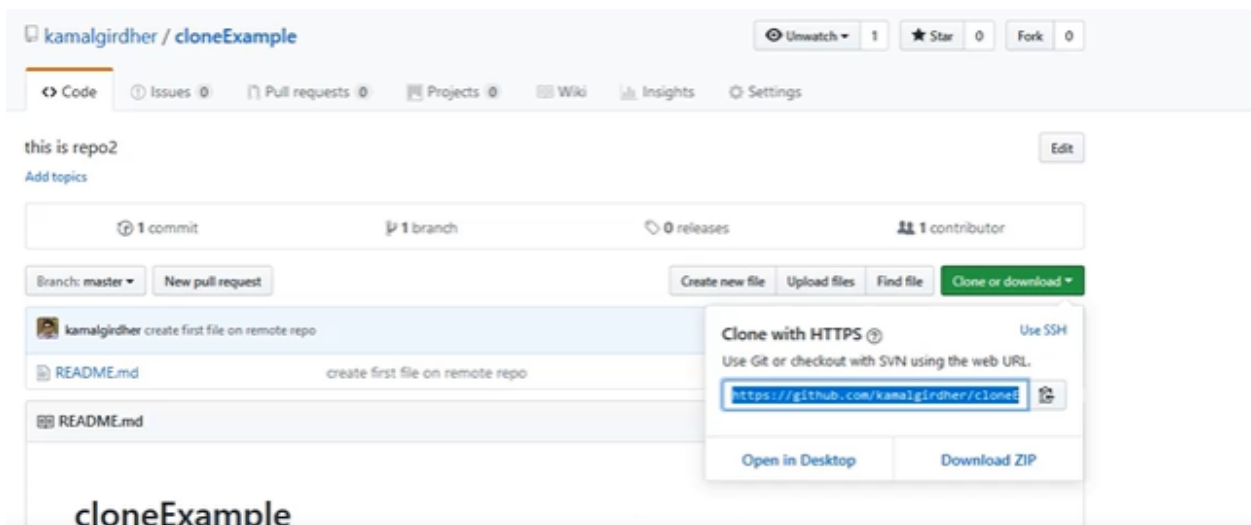
## Sub folders



## Git clone

Go to remote repository-> website

Click the Button-> clone or download



And go to Git Bash and type the following commands below

Locate the file on local using `$cd/ file name`

And use `$git clone https-> link in the website` command to clone the repository

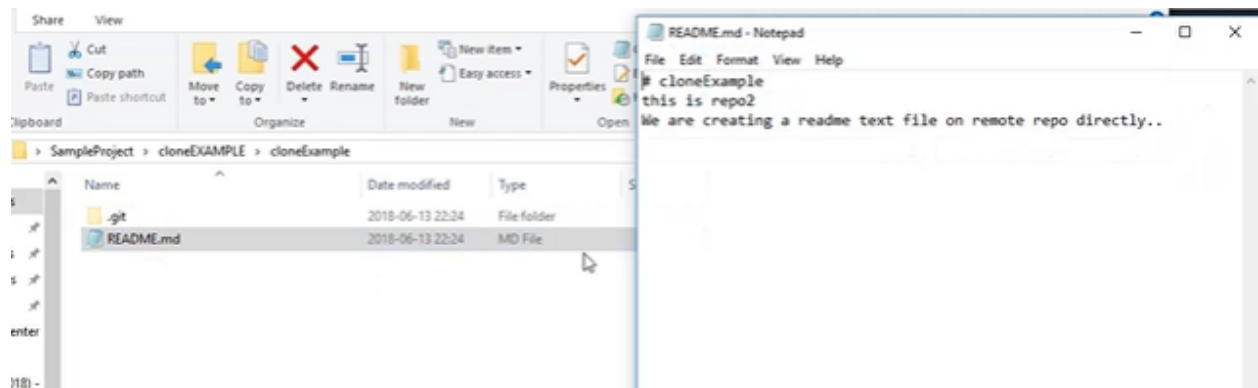
```
CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/SampleProject/initEXAMPLE (master)
$ cd ..

CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/SampleProject
$ cd cloneEXAMPLE/

CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/SampleProject/cloneEXAMPLE
$ git clone https://github.com/kamalgirdher/cloneExample.git
Cloning into 'cloneExample'...
remote: Counting objects: 3, done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
```

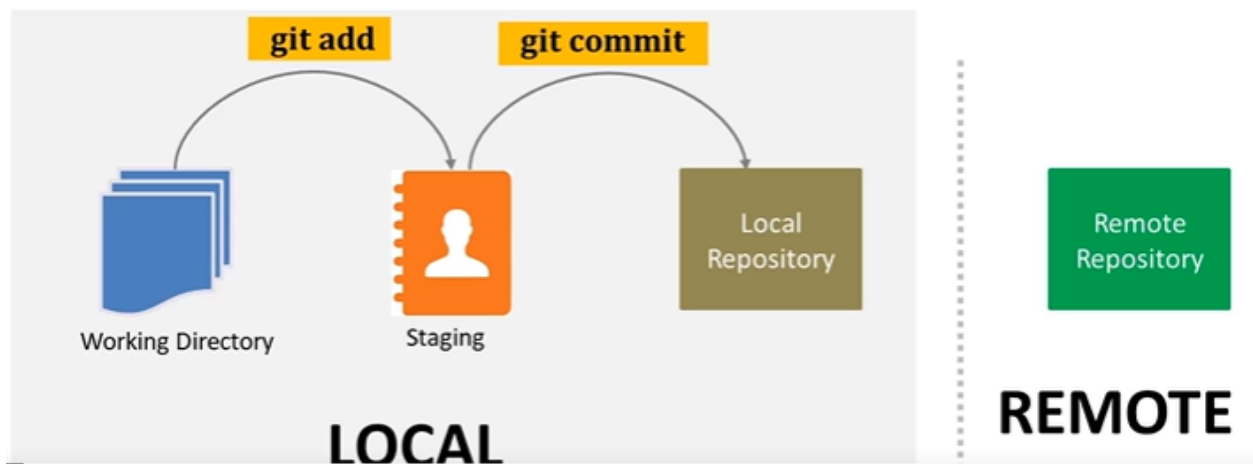
When you can see inside to your local file

You saw



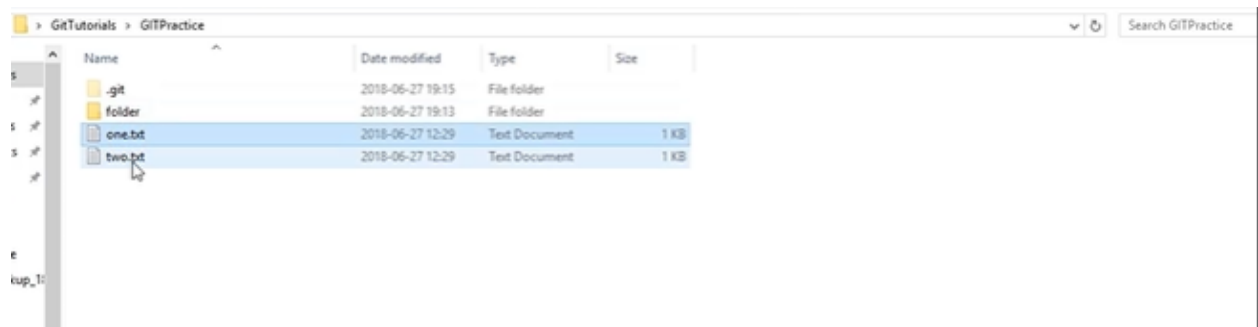
## Git Add

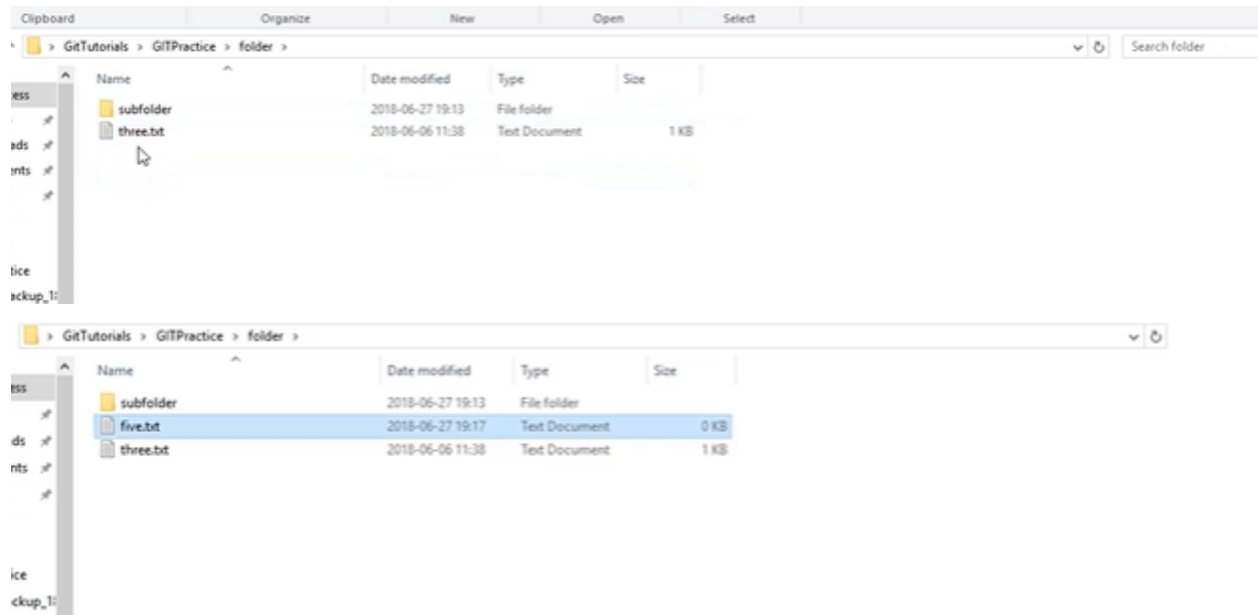
In Local



We can change the changes, deletion and new updates in [working directory](#) -> [local](#) using [git add](#)

Changes in the Local Directory





These are the insert , update and delete occurred in the local directory

\$ git status

It is used to find the current status of the [local directory](#)

Ex

```
CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/GitTutorials/GITPractice (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   one.txt
        deleted:    two.txt

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

        folder/five.txt

no changes added to commit (use "git add" and/or "git commit -a")
CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/GitTutorials/GITPractice (master)
$
```

\$git add -A

Then all the changes are updated in the [Staging](#)

Ex

```

CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/GitTutorials/GITPractice (master)
$ git add -A

CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/GitTutorials/GITPractice (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        new file:   folder/five.txt
        modified:   folder/subfolder/four.txt
        modified:   one.txt
        deleted:    two.txt

CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/GitTutorials/GITPractice (master)
$

```

After updated /modified/deleted files are shown in the **green color**  
**\$ git reset**

```

CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/GitTutorials/GITPractice (master)
$ git add --all

CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/GitTutorials/GITPractice (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        new file:   folder/five.txt
        modified:   folder/subfolder/four.txt
        modified:   one.txt
        deleted:    two.txt

CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/GitTutorials/GITPractice (master)
$

```

Use this command to reset the previously added files and folders return to the **staging -> working directory**

```
git add -A ✓
```

```
git add --all ✓
```

```
git add .
```

```
git add folder/*.txt
```

```
git add *
```

These are the add commands

**\$git commit** – Staging ->Local repository

```
git commit -m "put your message here"
```

EX:



```
MINGW64 ~/c/Users/CKG/Desktop/GitTutorials/GITPractice
CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/GitTutorials/GITPractice (master)
$ git commit -m "added 1 file, modified two file with these changes, deleted one file"
[master 65c3524] added 1 file, modified two file with these changes, deleted one file
Committer: GIRDHER Kamal IMT/OLS <kamal.girdher@orange.com>
Your name and email address were configured automatically based on your username and hostname. Please check that they are accurate. You can suppress this message by setting them explicitly. Run the following command and follow the instructions in your editor to edit your configuration file:

    git config --global --edit

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

4 files changed, 7 insertions(+), 4 deletions(-)
create mode 100644 folder/five.txt
delete mode 100644 two.txt
CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/GitTutorials/GITPractice (master)
```

Getting \$ git status

```
CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/GitTutorials/GITPractice (master)
$ git status
On branch master
Your branch is ahead of 'origin/master' by 1 commit.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean
CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/GitTutorials/GITPractice (master)
$
```

How to reset after commit

Use \$ git reset HEAD~

EX:

```

CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/GitTutorials/GITPractice (master)
$ git status
On branch master
Your branch is ahead of 'origin/master' by 1 commit.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean

CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/GitTutorials/GITPractice (master)
$ git reset HEAD~
Unstaged changes after reset:
M   folder/subfolder/four.txt
M   one.txt
D   two.txt

CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/GitTutorials/GITPractice (master)
$ git status

```

Use this command to reset and roll back all the changes after commit

**\$git Diff -> working directory and index**



Enter \$git diff and the file name

```
MINGW64: c:/Users/CKGS5099/Desktop/GitTutorials/GITPractice
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 directory)

        modified:   two.txt

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

CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/GitTutorials/GITPractice (master)
$ git diff two.txt
diff --git a/two.txt b/two.txt
index 2811513..e6b48f6 100644
--- a/two.txt
+++ b/two.txt
@@ -1,2 +1,2 @@
-hello
+hello Kamal. How are you??
  kamal
\ No newline at end of file

CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/GitTutorials/GITPractice (master)
```

The - - file name defines the previous text

The ++ file name what are the changes we done now

```
CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/GitTutorials/GITPractice (master)
$ git add two.txt

CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/GitTutorials/GITPractice (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   two.txt

CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/GitTutorials/GITPractice (master)
```

After using git add file name.txt

If we get the status of the file is get modified and updated



When the changes on in staging we can see the changes through `$git diff - - cached`

Changes between [staging and the index](#)

```
CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/GitTutorials/GITPractice (master)
$ git diff --cached two.txt
diff --git a/two.txt b/two.txt
index 2811513..e6b48f6 100644
--- a/two.txt
+++ b/two.txt
@@ -1,2 +1,2 @@
-hello
+hello Kamal. How are you??
  kamal
\ No newline at end of file
```

Using `$git reset - - hard` to reset all the files back to the previous state

```
MINGW64: c:/Users/CKGS5099/Desktop/GitTutorials/GITPractice
CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/GitTutorials/GITPractice (master)
$ git reset --hard
HEAD is now at 1510aec renamed files
CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/GitTutorials/GITPractice (master)
$
```

\$git rm



git rm

```
git rm <filename>
```

```
git rm -f <filename>
```

```
git rm --cached <filename>
```

```
git rm -r <folderpath>
```

This above rm command can be used to **remove the individual files or a collection of files**

#### Git rm - - cached file name

```
MINGW64/c/Users/CKGS5099/Desktop/GitTutorials/GITPractice

CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/GitTutorials/GITPractice (master)
$ git rm --cached two.txt
rm 'two.txt'

CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/GitTutorials/GITPractice (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        deleted:    two.txt

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

        two.txt

CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/GitTutorials/GITPractice (master)
```

It is used to delete the file from the staging and local but not in working directory

In `$ git rm <file name >` is used to delete the file from staging and delete the actual working file

In `$ git rm -f<filename>` f- means forceful deletion of the file

In `git rm -r<folderPath>` used to delete all the files in the folder

## Branch and Merge

Branch represents an independent line of development

EX : if you want to add some future in the existing project ,you can take a snapshot of the main code that we called as master branch

And then we can start the development

In master branch remains as of existing code

In the future branch contains all the changes of the code in the project

Once you verify the changes you will merge the branches

Then the branches will act as extraction of edit,stage and commit

It is very useful when multiple people working on a same project and everyone is making changes to the source files in the project

Then can anytime merge the future with master and get the latest version of the code with latest features

## Clone the git repository and get the current status

```
CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/GitTutorials
$ git clone https://github.com/kamalgirdher/08Jul2018.git
Cloning into '08Jul2018'...
remote: Counting objects: 3, done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.

CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/GitTutorials
$ git status
fatal: not a git repository (or any of the parent directories): .git

CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/GitTutorials
$ cd 08Jul2018/

CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/GitTutorials/08Jul2018 (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

nothing to commit, working tree clean

CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/GitTutorials/08Jul2018 (master)
$ c|
```

## The new file is still on the working copy(directory)

Using **\$git add** the new file to the staging

```
CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/GitTutorials/08Jul2018 (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

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

    file1.txt

nothing added to commit but untracked files present (use "git add" to
CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/GitTutorials/08Jul2018 (master)
$ git add file1.txt

CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/GitTutorials/08Jul2018 (master)
$ |
```



Run **\$git commit** to move it to local repository

```
nothing added to commit but untracked files present (use 'git add' to
track)
CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/GitTutorials/08Jul2018 (master)
$ git add file1.txt
CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/GitTutorials/08Jul2018 (master)
$ git commit -m "first commit"
[master 45e9937] first commit
Committer: GIRDHER Kamal IMT/OLS <kamal.girdher@orange.com>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly. Run the
following command and follow the instructions in your editor to edit
your configuration file:

    git config --global --edit

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

1 file changed, 1 insertion(+)
create mode 100044 file1.txt
CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/GitTutorials/08Jul2018 (master)
$
```

We can use **\$git log** to see the user history comments

```
CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/GitTutorials/08Jul2018 (master)
$ git log
commit 45e99379a06f83e4f635235dd561fd7d0073a919 (HEAD -> master)
Author: GIRDHER Kamal IMT/OLS <kamal.girdher@orange.com>
Date:   Sun Jul 8 09:41:46 2018 +0530

    first commit

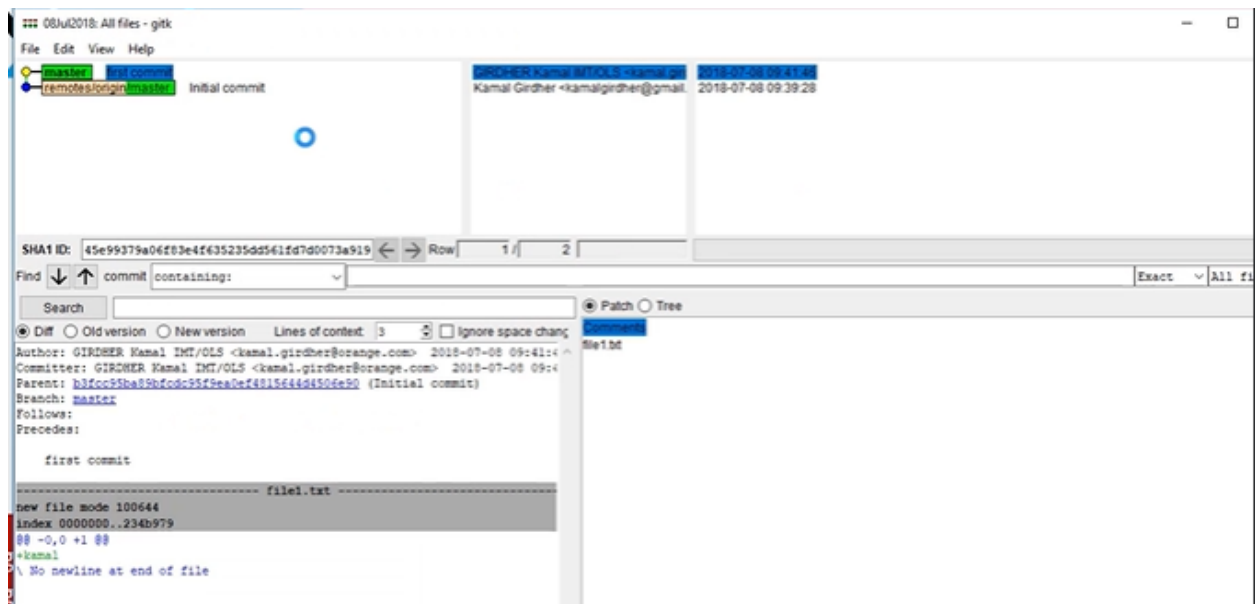
commit b3fcc95ba89bfc95f9ea0ef4815644d4506e90 (origin/master, origin/HEAD)
Author: Kamal Girdher <kamalgirdher@gmail.com>
Date:   Sun Jul 8 09:39:28 2018 +0530

    Initial commit
CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/GitTutorials/08Jul2018 (master)
$
```



We can use the utility called **\$gitk** to see the visual changes

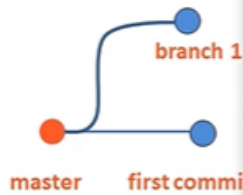
```
Initial commit
CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/GitTutorials/08Jul2018 (master)
$ gitk
```



Creation of a branch



## Branch and Merge



git branch  
git branch [branch-name]  
git checkout  
git merge  
git log

```
MINGW64 ~/Desktop/GitTutorials/08Jul2018 (master)
$ git branch
* master
MINGW64 ~/Desktop/GitTutorials/08Jul2018 (master)
$ git branch branch1
MINGW64 ~/Desktop/GitTutorials/08Jul2018 (master)
$ git branch
* master
  branch1
MINGW64 ~/Desktop/GitTutorials/08Jul2018 (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

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

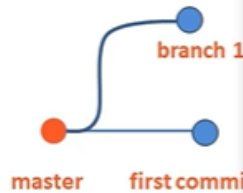
nothing added to commit but untracked files present (use "git add" to track)
MINGW64 ~/Desktop/GitTutorials/08Jul2018 (master)
$
```

Gdemy

Step 2:



## Branch and Merge



git branch  
git branch [branch-name]  
git checkout  
git merge

```
MINGW64 ~/Desktop/GitTutorials/08Jul2018 (master)
$ git add abc.py
MINGW64 ~/Desktop/GitTutorials/08Jul2018 (master)
$ git commit -m "first commit on master"
[master 0b28cbf] first commit on master
Committer: GIRDHER Kamal IMT/OLS <kamal.girdher@orange.com>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly. Run the
following command and follow the instructions in your editor to edit
your configuration file:

    git config --global --edit

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

1 file changed, 3 insertions(+)
create mode 100644 abc.py
MINGW64 ~/Desktop/GitTutorials/08Jul2018 (master)
$
```

Step 3:

Second commit on the master

Add new code to the existing file in the master branch and give commit store in the local repository

**Branch and Merge**

Diagram illustrating the Git workflow:

- master** branch: first commit, second commit
- branch 1** branch: (created from master)

Terminal output:

```
CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/GitTutorials/08Jul2018
$ git branch
branch1
* master

CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/GitTutorials/08Jul2018
$ git status
On branch master
Your branch is ahead of 'origin/master' by 1 commit.
(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 work)

        modified:   README.md

no changes added to commit (use "git add" and/or "git commit")
CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/GitTutorials/08Jul2018
$ git add README.md
$ git commit -m "second commit on master"
```

Commands shown in the video:

- git branch
- git branch [branch-name]
- git checkout
- git merge

2:58 / 4:41

Remember that these changes are in only on the master not in the branch one

Creation of the new branch using `$ git branch2 name/number`

**Branch and Merge**

Diagram illustrating the Git workflow:

- master** branch: first commit, second commit
- branch 1** branch: (created from master)
- branch 2** branch: (created from master)

Terminal output:

```
CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/GitTutorials/08Jul2018
$ git branch
branch1
* master

CKGS5099@EQ-EQ6285240 MINGW64 ~/Desktop/GitTutorials/08Jul2018
$ git branch branch2
```

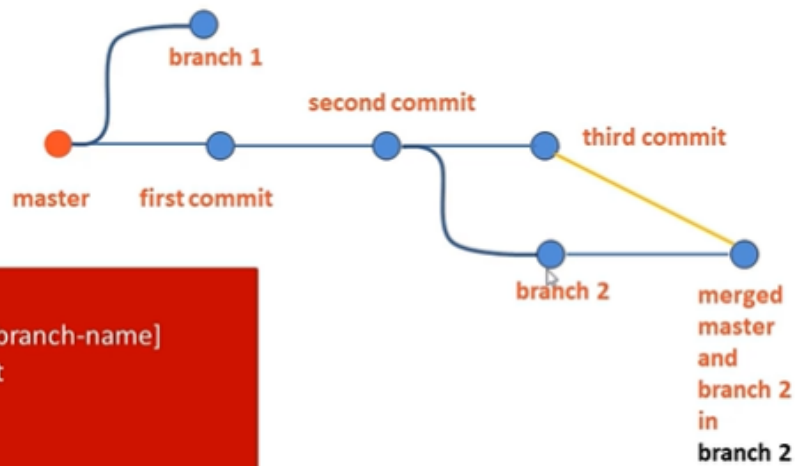
Commands shown in the video:

- git branch
- git branch [branch-name]
- git checkout
- git merge

3:52 / 4:41



## Branch and Merge



git branch  
git branch [branch-name]  
git checkout  
git merge  
git log

GitLab

\$git checkout is used to switch between the branches

Ex: