**COURSE CODE: DJ19ITL602**   **DATE:**

**COURSE NAME: Software Engineering Laboratory**  **CLASS: T.Y.BTech**

**EXPERIMENT NO. 7**

**CO/LO** Analyze real world problem using software engineering principles.

**AIM / OBJECTIVE**: To Perform Version Control on any project using any Version control tool (GIT).

**THEORY**:

Git is a free and open-source distributed version control system designed to handle everything from small to very large projects with speed and efficiency.

Git is easy to learn and has a tiny foot print with lightning-fast performance.

Git is a version control system for tracking changes in computer files and coordinating work on those files among multiple people. It is primarily used for source code management in software development, but it can be used to keep track of changes in any set of files.

It outclasses SCM tools like Subversion, CVS, Perforce and ClearCase with features like cheap local branching, convenient staging areas, and multiple workflows.

Some of the basic operations in Git are:

1.Initialize

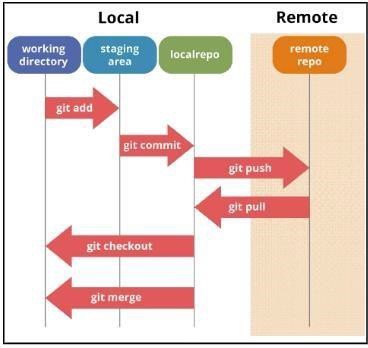
2.Add

3.Commit

4.Pull

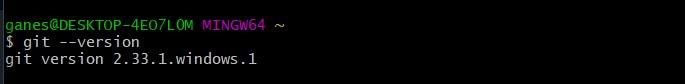
5.Push

* Some advanced Git operations are:
  1. Branching
  2. Merging
  + The following diagram depict the all supported operations in GIT

Shape

# **COMMANDS:**

* + 1. $git –version

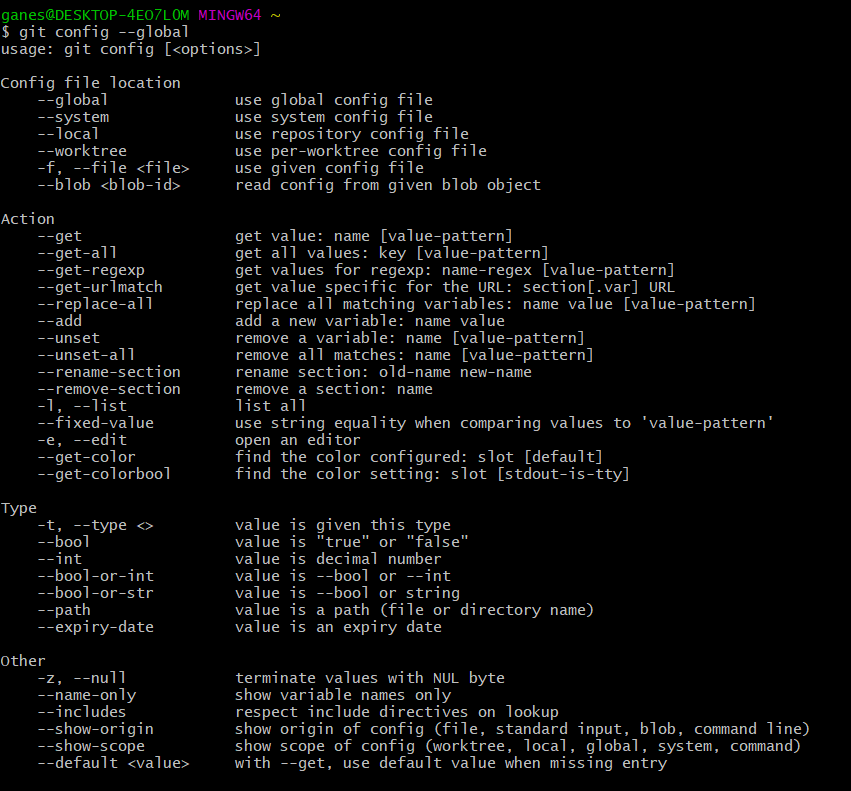


* + 1. $git config

The git config command is a convenience function that is used to set Git configuration values on a global or local project level.

$git config –global

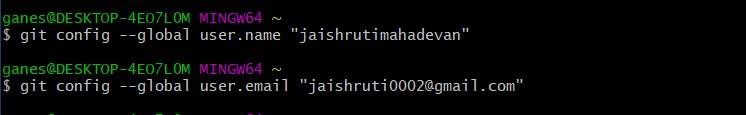
Global level configuration is user-specific, meaning it is applied to an operating system user. Global configuration values are stored in a file that is located in a user's home directory.



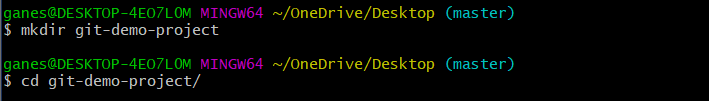
* + 1. git config --global --list



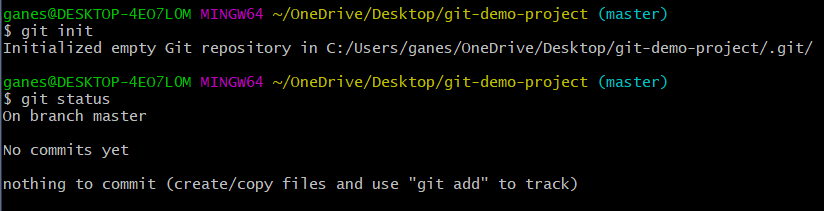
* + 1. $git config --global user.name “<user\_name>”

$git config –global user.email “<user\_email>”

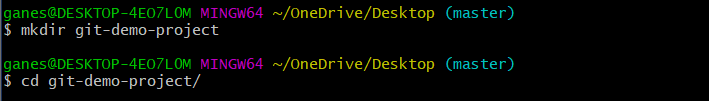
* + 1. mkdir git-demo-project

cd git-demo-project

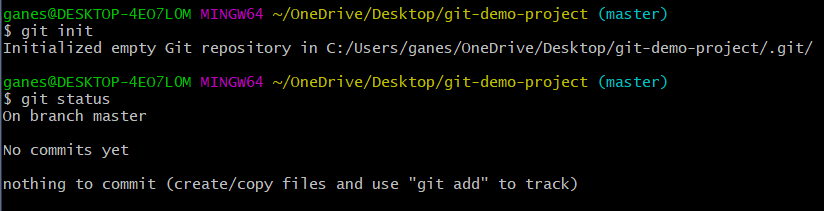
* + 1. git init git status



1. mkdir git-demo-project

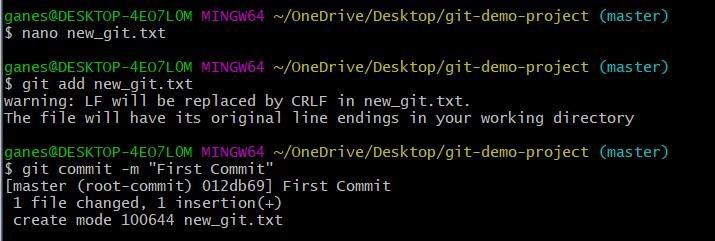
cd git-demo-project

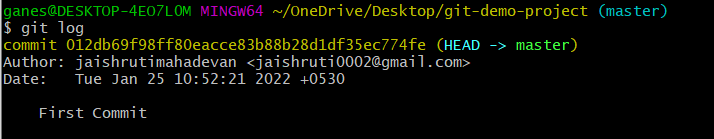
* + 1. git init git status

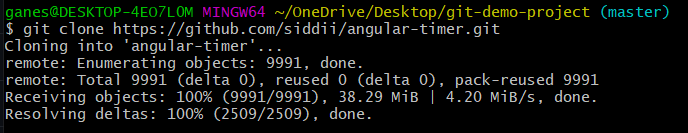


* + 1. git add new\_git.txt

git commit -m “First Commit”

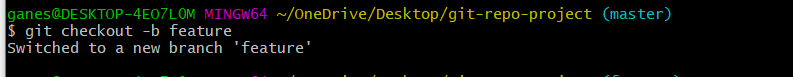
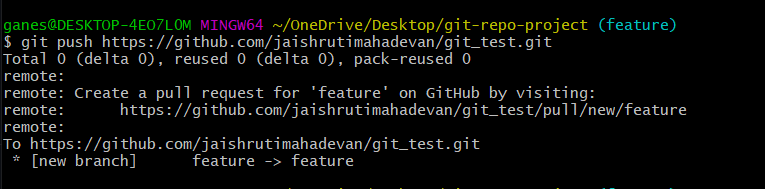
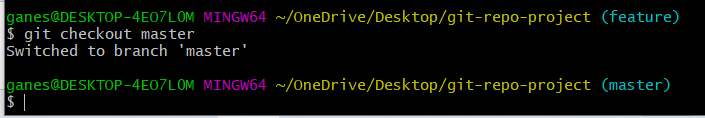


* + 1. git log
    2. git clone <https://github.com/siddii/angular-timer.git>



* + 1. git push origin master



* + 1. git checkout -b feature
    2. git commit -m “New Commit”
    3. git push <https://github.com/jaishrutimahadevan/git_test.git>
    4. git checkout master

**OUTPUT:**

1. Screenshots of every step

QUESTION:

1. What are the advantages of using GIT?
2. What are the benefits of using Version Control System?

**REFERENCE**

www.geeksforgeeks.com

**Steps to Perform Version Control using Git:**

**1. Initialize a Git Repository**

cd your-project-folder  
git init  
  
This creates a .git/ folder in your project, which tracks all version history.

**2. Add Files to Git**

git add .  
or  
git add index.html app.js  
  
This stages all files or specific files to be tracked.

**3. Commit Your Changes**

git commit -m "Initial commit: added base project files"  
  
Commits are snapshots of your project.

**4. Connect to a Remote Repository (e.g., GitHub)**

git remote add origin https://github.com/your-username/your-repo.git  
  
Replaces the URL with your actual GitHub repo URL.

**5. Push to GitHub**

git push -u origin master  
or (if main branch)  
git push -u origin main

**6. Track Changes Over Time**

git add changed\_file.py  
git commit -m "Updated the visualization logic"  
git push

**7. Create and Switch to a New Branch**

git checkout -b feature-new-graph  
  
Later merge into main:  
git checkout main  
git merge feature-new-graph

**8. View Git History**

git log

**Bonus: Useful Git Commands Summary**

|  |  |
| --- | --- |
| Task | Command |
| Clone a repo | git clone <repo\_url> |
| View status | git status |
| View diff before commit | git diff |
| Discard changes | git checkout -- filename |
| Pull latest changes | git pull |
| Revert to a previous commit | git revert <commit\_hash> |