DevTech Training

Short Course - Day 1

Training Staff

- WORGIE V. FLORES

  ETRACS Developer

In This Training

In This Training

- ETRACS Environment Setups
- ETRACS Deployment Setups
- Working Environment Setup
- Git

In This Training

- Virtualization
- Virtual Machine vs Docker
- Docker
- iReport Designer

ETRACS Environment Setups

Standalone

- Windows, Mac & Linux
- MySQL / MSSQL
- Java
- Git

Docker Deployment

- Windows, Mac & Linux
- MySQL / MSSQL
- Docker Engine
- Git

- Optional Add-ons
  - Hypervisor

ETRACS Deployment Setups

	Standalone	Docker
Main		✓
Province	✓	✓
Municipality	✓	✓
City	✓	✓
Remote Barangay, Hospital, Market, Terminals, etc…	✓	

Working Environment Setup

Working Environment Setup

Batch-1

- **WSL 2**
- **Docker Desktop**
- **Ubuntu** (18 or 20) from the Microsoft Store
- **Database Engine** (MySQL / MSSQL)
- **Java** 1.8

Working Environment Setup

Batch-1

WSL 2

- Windows Subsystem For Linux (**WSL**) is a tool provided by Microsoft to run Linux natively on Windows
- Essentially providing a full Linux shell that can interact with your Windows file system
- WSL 2, is a new version that powers the architecture to run ELF64 Linux binaries on Windows, and increase the file system performance, as well as adding full system call compatibility

Working Environment Setup

Batch-1

Docker Desktop

- An easy-to-install application for your Mac or Windows environment that enables you to build and share containerized applications and micro-services

- Includes **Docker Engine**, **Docker CLI** client, **Docker Compose**, **Docker Content Trust**, and **Credential Helper**

Working Environment Setup

Batch-2

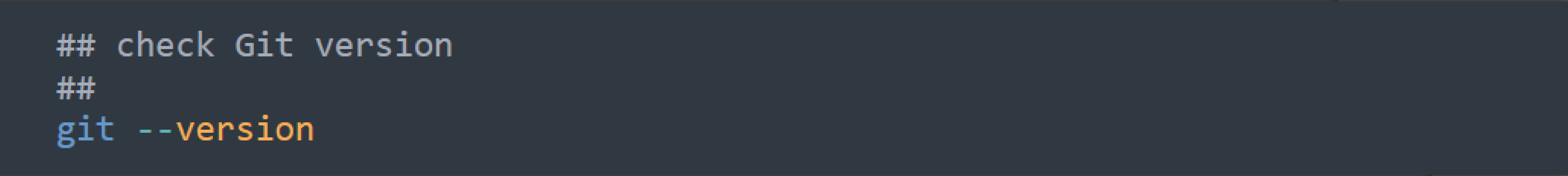
- Dual-Booting Feature
- Linux Mint 20.3 Cinnamon
- Git
- Docker Engine
- Java 1.8

Check Setup Status

1. Boot to **Linux Mint** operating system and login

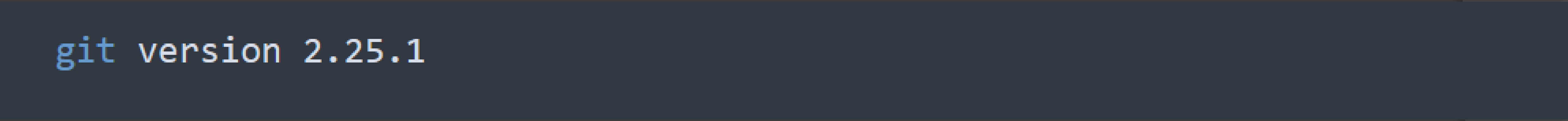
2. Press **CTRL** **+** **ALT** **+** **T** to open a new window terminal or shell

Command



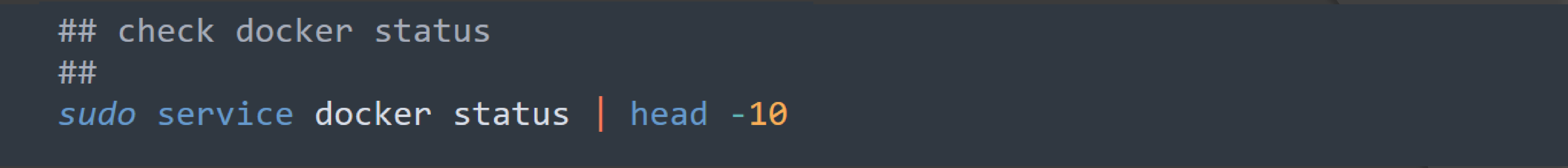
```
## check Git version
##
git --version
```

Output



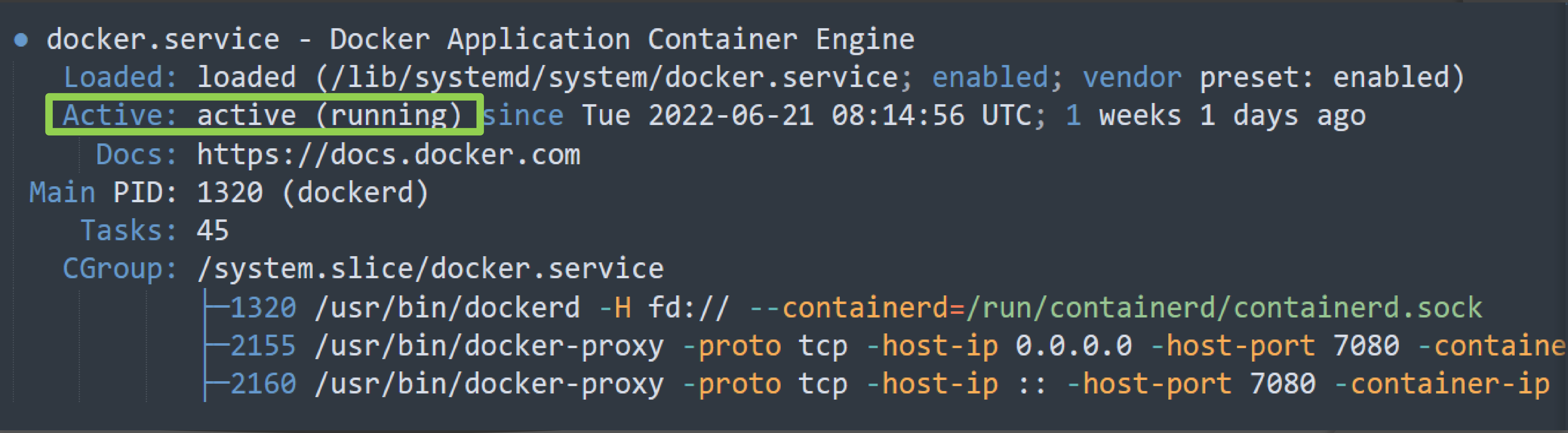
```
git version 2.25.1
```

Command



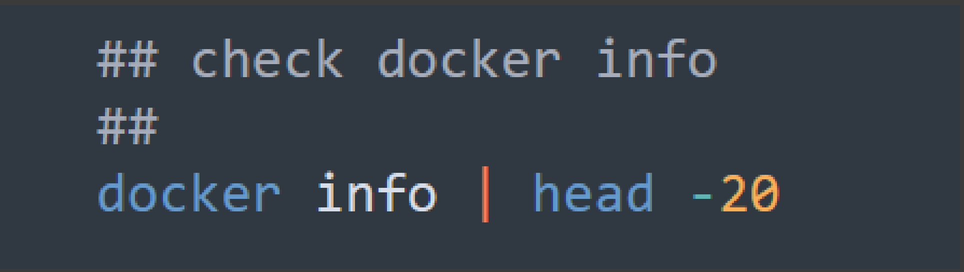
```
## check docker status
##
sudo service docker status | head -10
```

Output



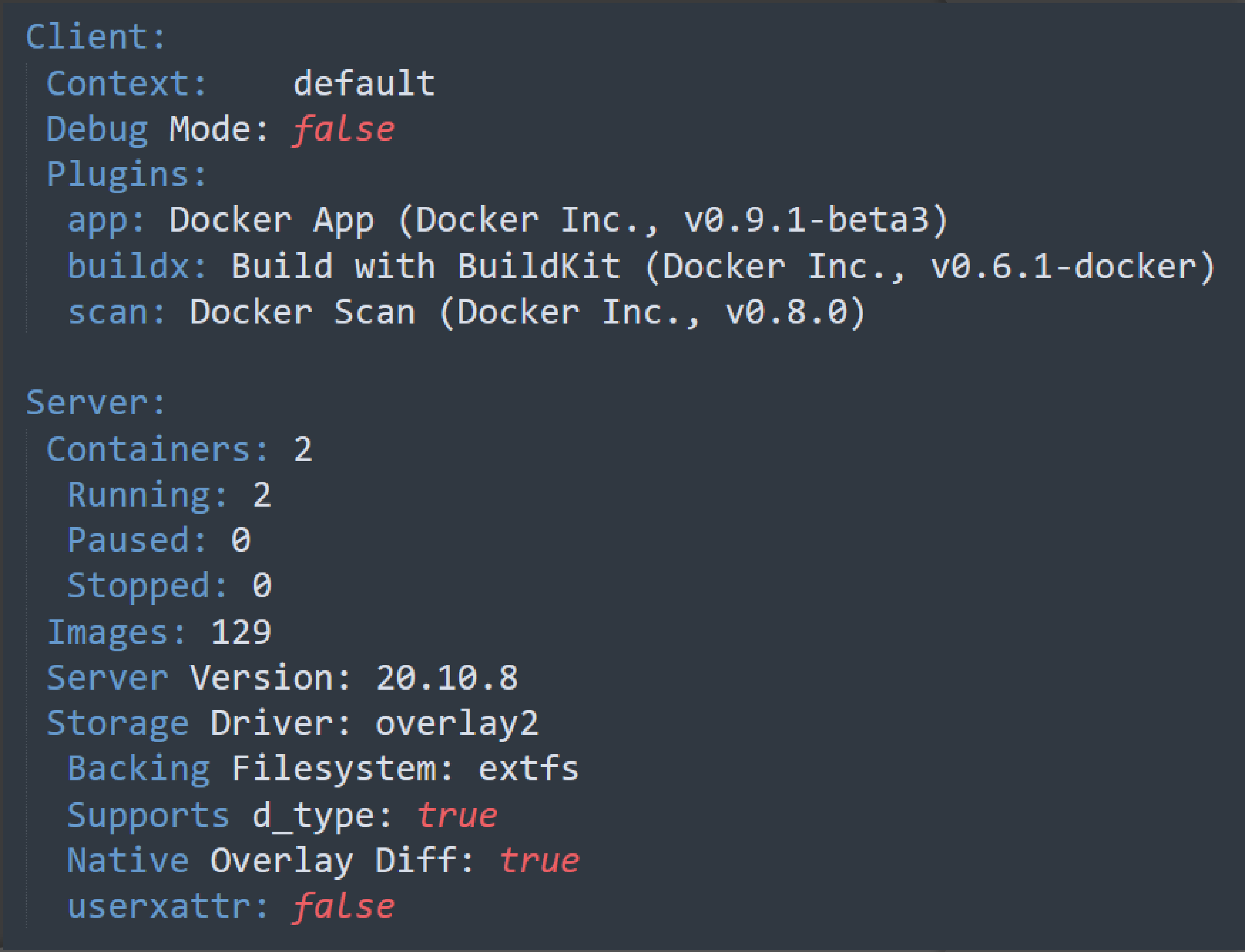
```
● docker.service - Docker Application Container Engine
   Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)
   Active: active (running) since Tue 2022-06-21 08:14:56 UTC; 1 weeks 1 days ago
     Docs: https://docs.docker.com
 Main PID: 1320 (dockerd)
    Tasks: 45
   CGroup: /system.slice/docker.service
           ├─1320 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock
           ├─2155 /usr/bin/docker-proxy -proto tcp -host-ip 0.0.0.0 -host-port 7080 -containe
           ├─2160 /usr/bin/docker-proxy -proto tcp -host-ip :: -host-port 7080 -container-ip
```

Command



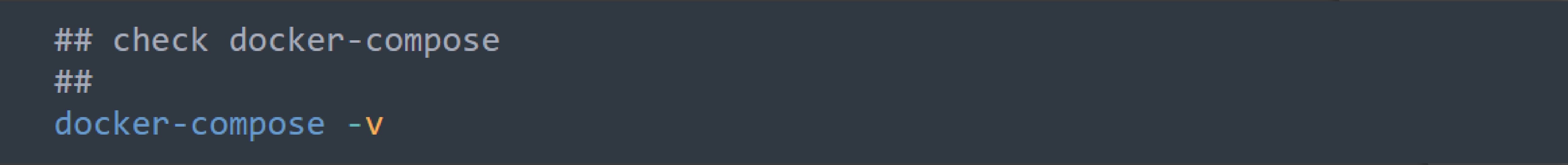
```
## check docker info
##
docker info | head -20
```

Output



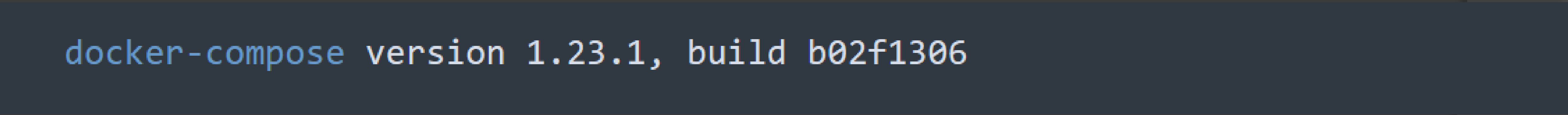
```
Client:
 Context:    default
 Debug Mode: false
 Plugins:
  app: Docker App (Docker Inc., v0.9.1-beta3)
  buildx: Build with BuildKit (Docker Inc., v0.6.1-docker)
  scan: Docker Scan (Docker Inc., v0.8.0)

Server:
 Containers: 2
  Running: 2
  Paused: 0
  Stopped: 0
 Images: 129
 Server Version: 20.10.8
 Storage Driver: overlay2
  Backing Filesystem: extfs
  Supports d_type: true
  Native Overlay Diff: true
  userxattr: false
```

Command



```
## check docker-compose
##
docker-compose -v
```

Output



```
docker-compose version 1.23.1, build b02f1306
```

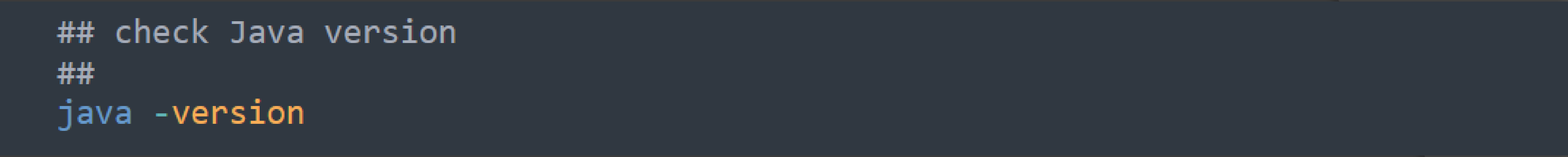
Command

```
## check docker images
##
docker images
```

Output

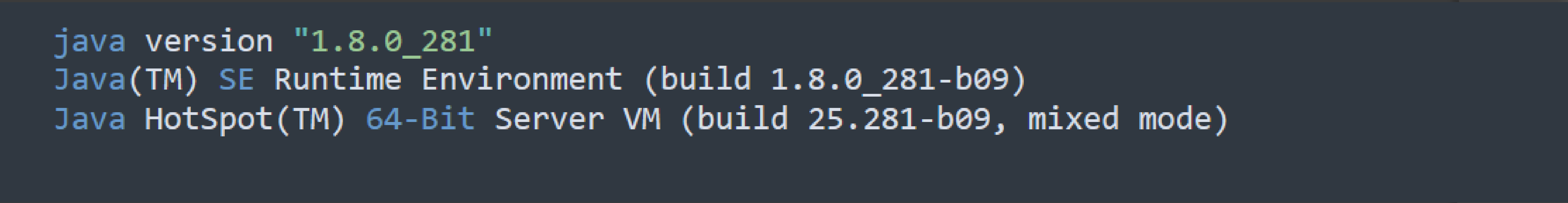
```
REPOSITORY                              TAG            IMAGE ID       CREATED         SIZE
nginx                                   latest         0e901e68141f   4 weeks ago     142MB
portainer/portainer-ce                  latest         7b6e59279c38   6 weeks ago     275MB
ramesesinc/etracs-server-municipality   2.5.04.05.01   4c265f59309d   7 weeks ago     177MB
ramesesinc/etracs-server-city           2.5.04.05.01   2ca8d816771f   7 weeks ago     177MB
ramesesinc/mail-server                  1.01           6ec78652c153   5 months ago    191MB
ramesesinc/gdx-client                   1.04.03        efba9f4ea0e0   5 months ago    174MB
ramesesinc/gdx-proxy-server             v004           ac272443b193   5 months ago    133MB
ramesesinc/local-epayment-server        2.5.01.02.06   81be8a28a405   11 months ago   189MB
ramesesinc/etracs-web                   2.5.02.01      db18909cb4b5   12 months ago   177MB
ramesesinc/node-download-server         0.0.3          68d1fcf48059   17 months ago   93.2MB
ramesesinc/queue-server                 2.5.02.01      f914c7ebe04f   17 months ago   215MB
mysql                                   5.7.31         42cdba9f1b08   20 months ago   448MB
ramesesinc/notification-server          1.0            0327153a182b   2 years ago     83.3MB
```

Command



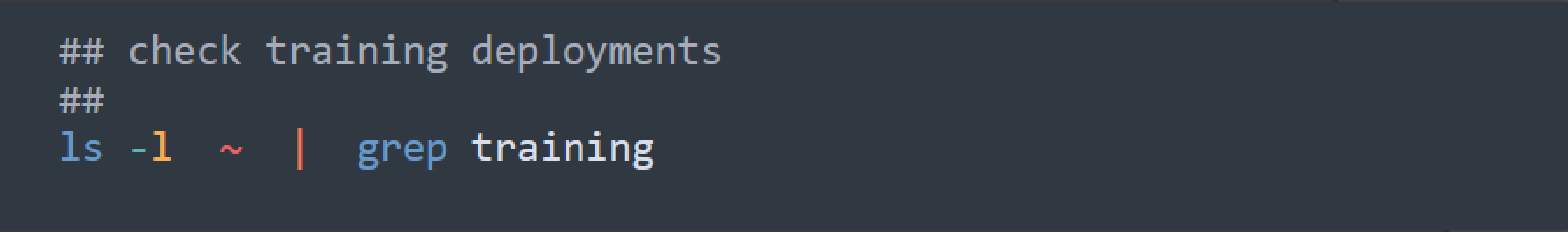
```
## check Java version
##
java -version
```

Output



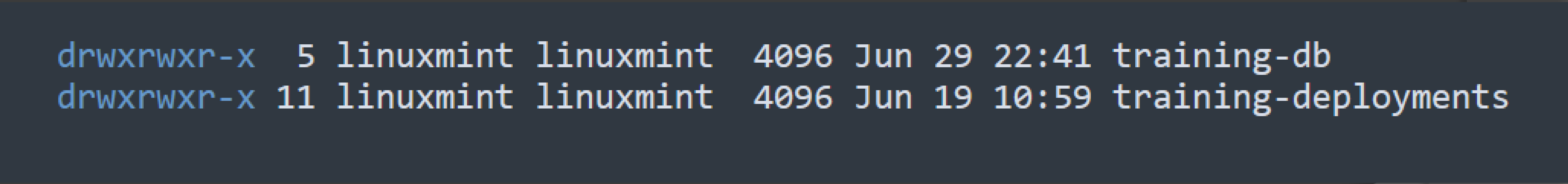
```
java version "1.8.0_281"
Java(TM) SE Runtime Environment (build 1.8.0_281-b09)
Java HotSpot(TM) 64-Bit Server VM (build 25.281-b09, mixed mode)
```

Command



```
## check training deployments
##
ls -l  ~  |  grep training
```

Output



```
drwxrwxr-x  5 linuxmint linuxmint  4096 Jun 29 22:41 training-db
drwxrwxr-x 11 linuxmint linuxmint  4096 Jun 19 10:59 training-deployments
```

Git

(Play Video 01)

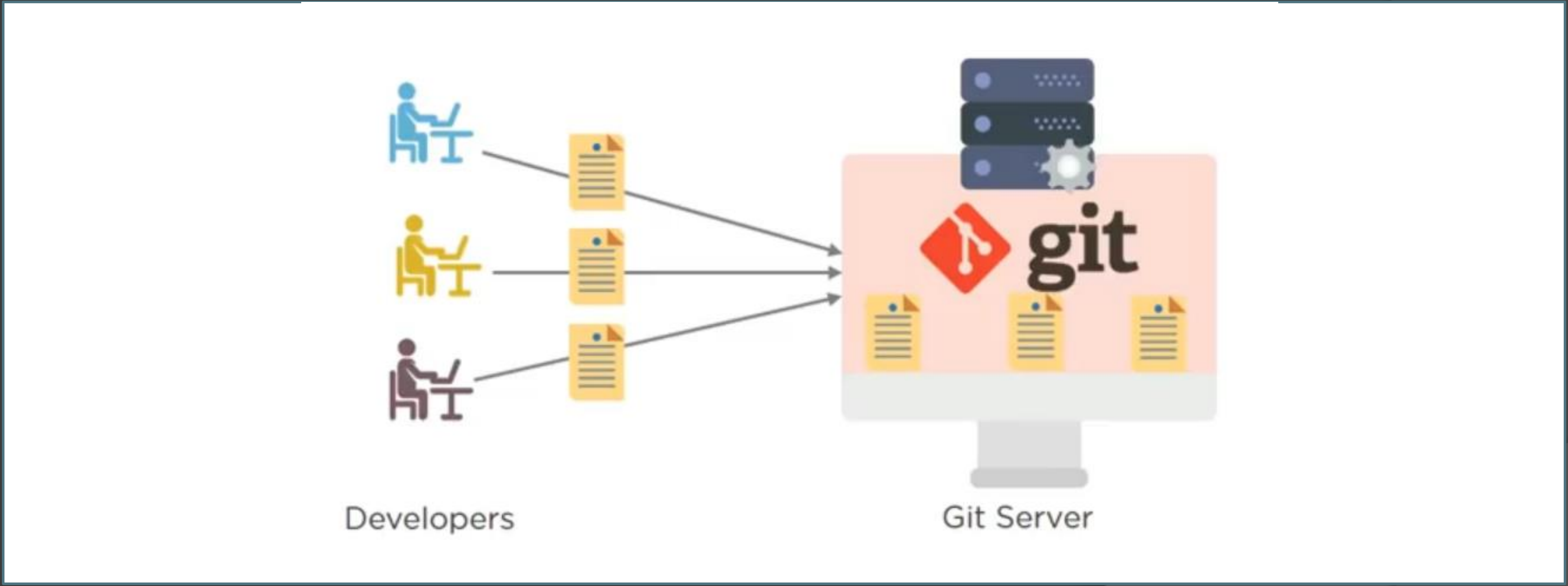
About Git

- Introduction
- Features
- Workflow
- Branching
- Commands
- Demo

What is Git ?

- Git is a distributed version control tool
- It is a popular version control system
- It is used for:
  - Tracking code changes
  - Tracking who made changes
  - Coding collaboration
  - Maintaining historical and current versions of source code
- It allows multiple developers to work together
- Supports non-linear development because of its thousands of parallel branches

What is Git ?

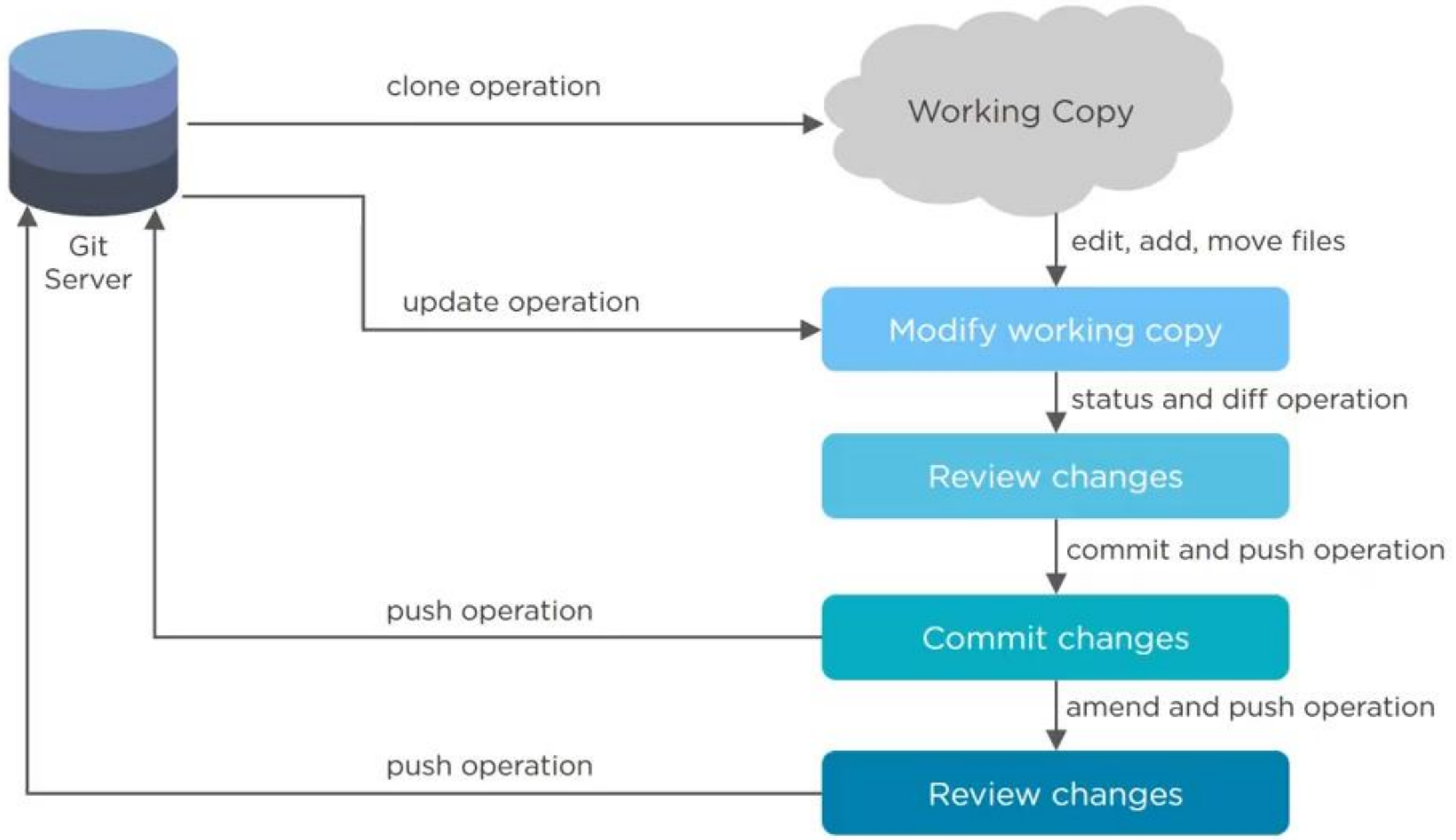


Allows multiple developers to work together

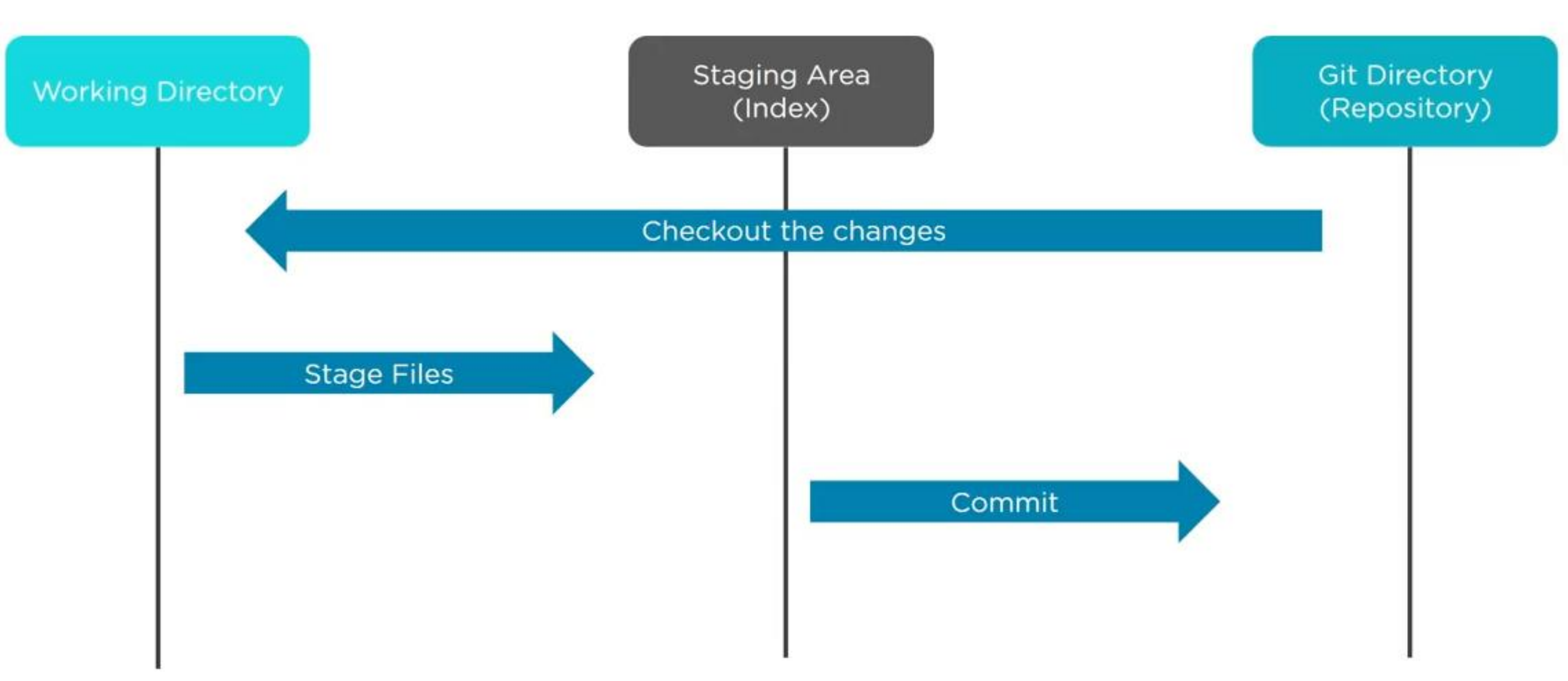
Features of Git

- Free and Open Source
- Tracks History
- Supports Non-Linear Development
- Creates Backup
- Scalable
- Supports Collaboration
- Branching
- Distributed Development

Git Workflow



Git Workflow - 3 States

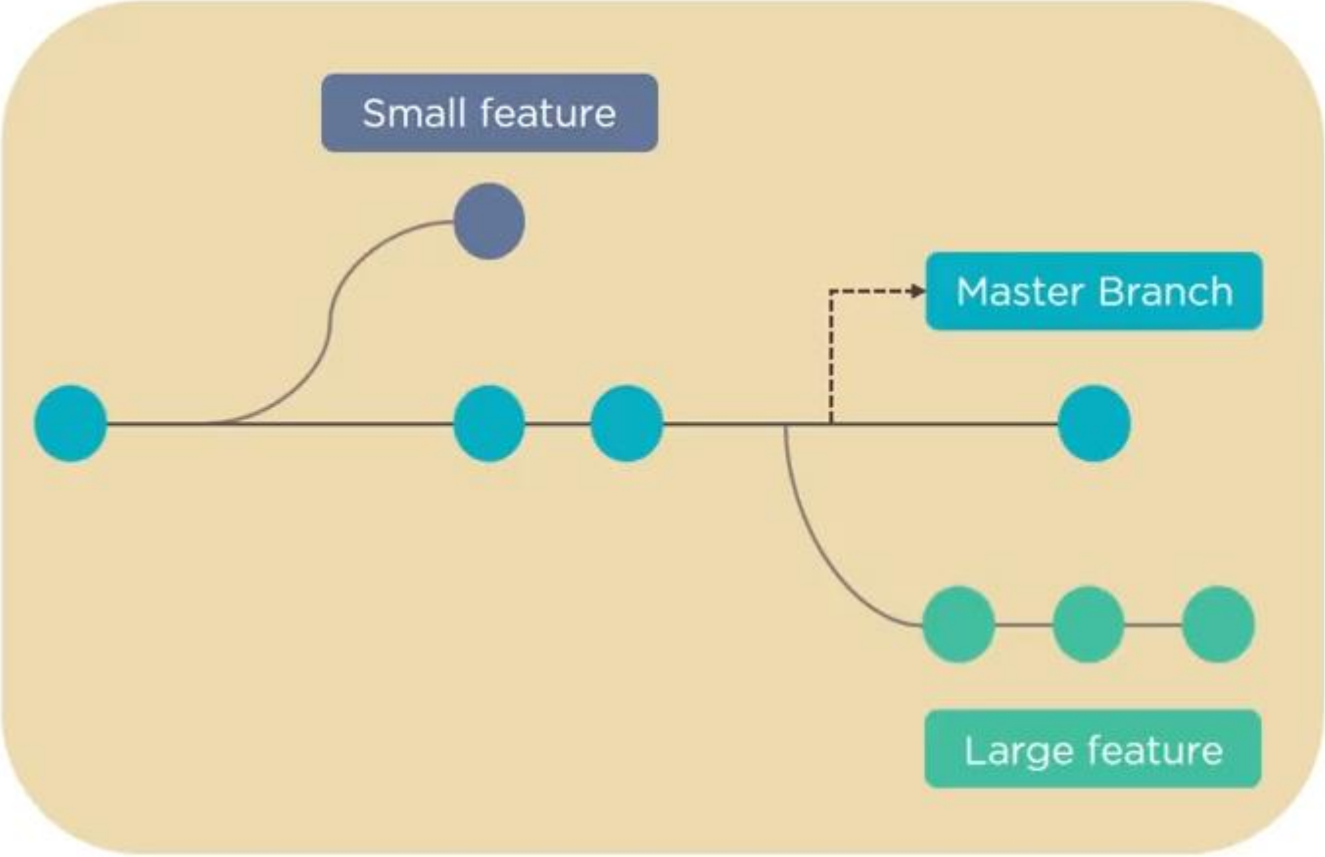


Branch in Git

- It is used to keep your changes until they are ready

- You can do your work on a branch while the main branch (master) remains stable.   After you are done with your work, you can merge it to the main branch

Branch in Git



- The diagram shows there is a master branch
- There are 2 more branches, Small feature and Large feature working separately

Basic Commands in Git

Git Commands

git config

- A convenience function that is used to set **Git** configuration values on a global or local project level
- These configuration levels correspond to the **.gitconfig** text files

Git Commands

```
git  init
```

- Create a new **Git** repository or initialize a new empty repository
- Creates a **.git** subdirectory in the current working directory, which contains all of the necessary Git metadata for the new repository

Git Commands

git clone

- Used to target an existing repository and creates a clone, or copy of the target repository

Git Commands

git status

- Gives all the necessary information about the current branch.
- Displays the state of the working directory and the staging area
- It lets you see which changes have been staged, which haven’t, and which files aren’t being tracked by **Git**

Git Commands

git add

- Adds a change in the working directory to the staging area

Git Commands

git commit

- The most-used command of Git. Once we reach a certain point in development, we want to save our changes (maybe after a specific task or issue).
- Git commit is like setting a checkpoint in the development process which you can go back to later if needed.
- We also need to write a short message to explain what we have developed or changed in the source code.

Git Commands

git remote

- Manage set of tracked repositories

Git Commands

git push

- Uploads your commits to the remote repository.

Git Commands

git pull

- Used to get updates from the remote repository

Git Commands

git fetch

- Download objects and refs from another repository

Git Commands

git branch

- Used to create, list, rename, and delete branches

Git Commands

git checkout

- Used mostly for switching from one branch to another
- Restore working tree files

Git Commands

git status

- Show the working tree status

Git Commands

git diff

- Show changes between commits, commit and working tree, etc…

Git Commands

git log

- Show commit logs

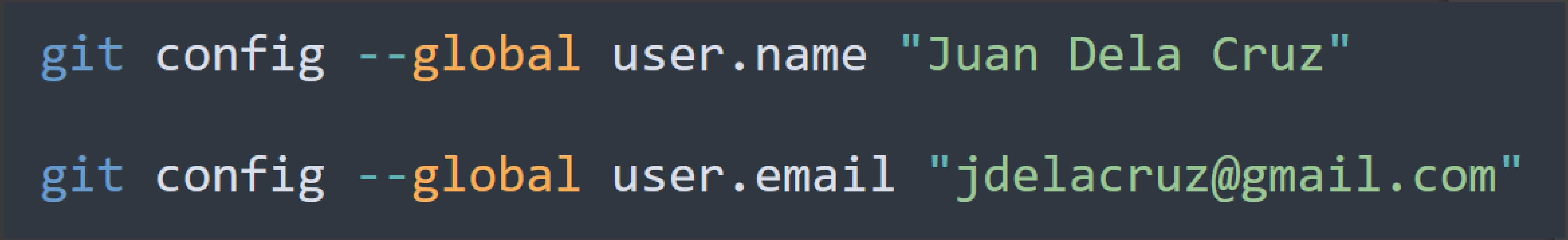
Git Commands

```
git  --help
```

- Shows the help information

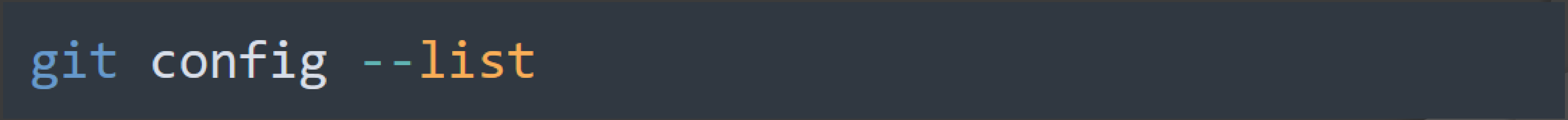
Demo on Git

Configure Git for the first time



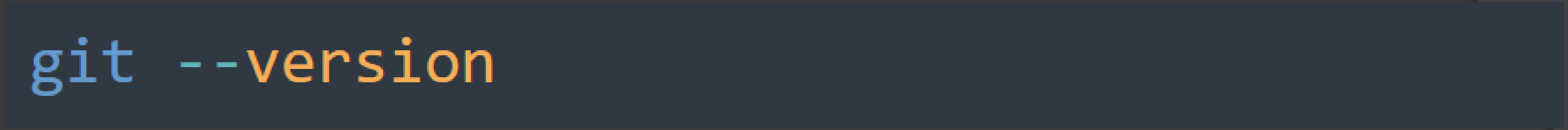
```
git config --global user.name "Juan Dela Cruz"

git config --global user.email "jdelacruz@gmail.com"
```

Display the configuration settings



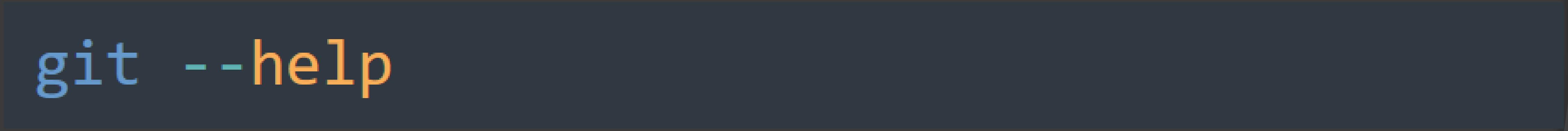
```
git config --list
```

To check the version



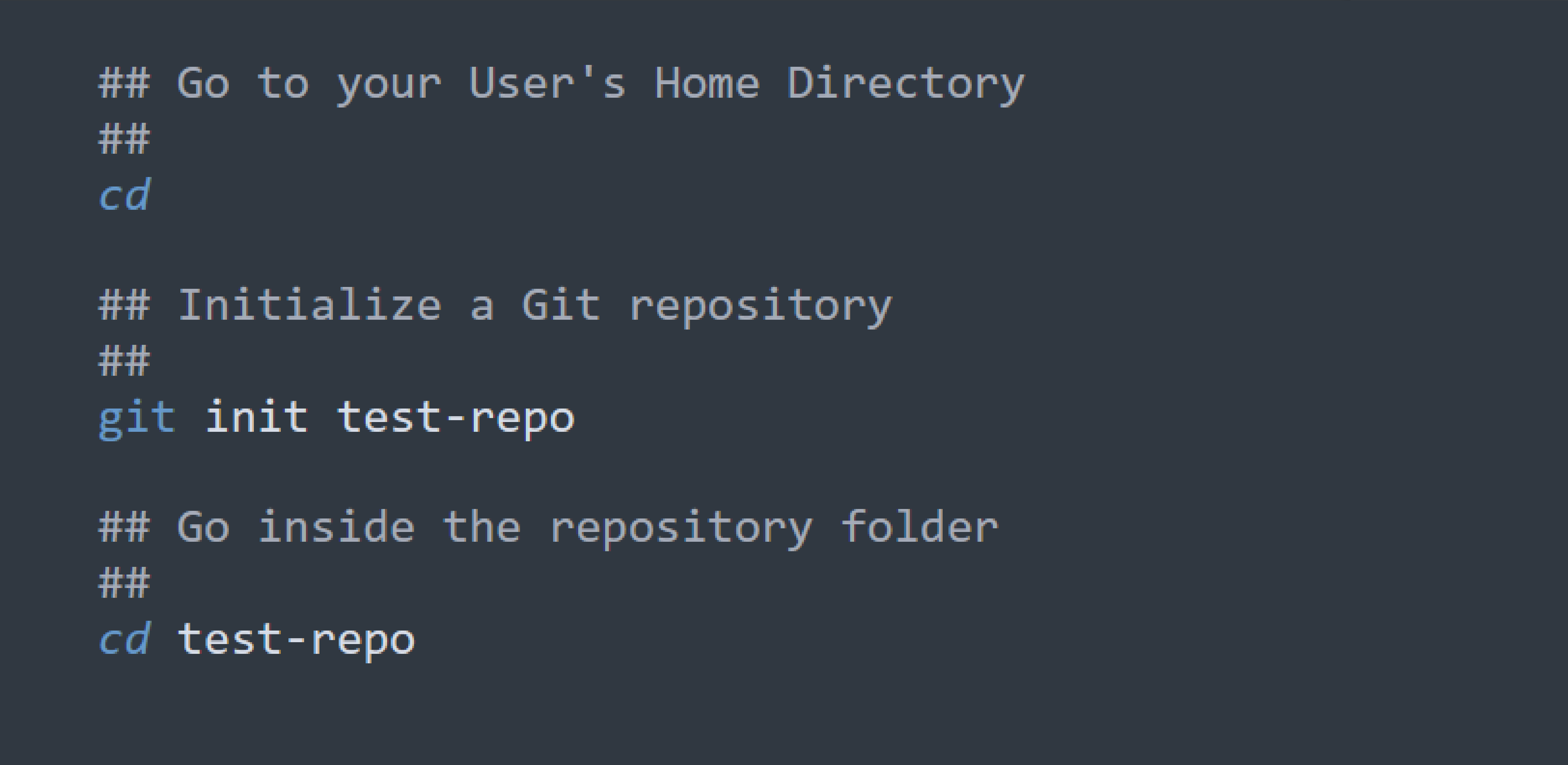
```
git --version
```

To check the help information



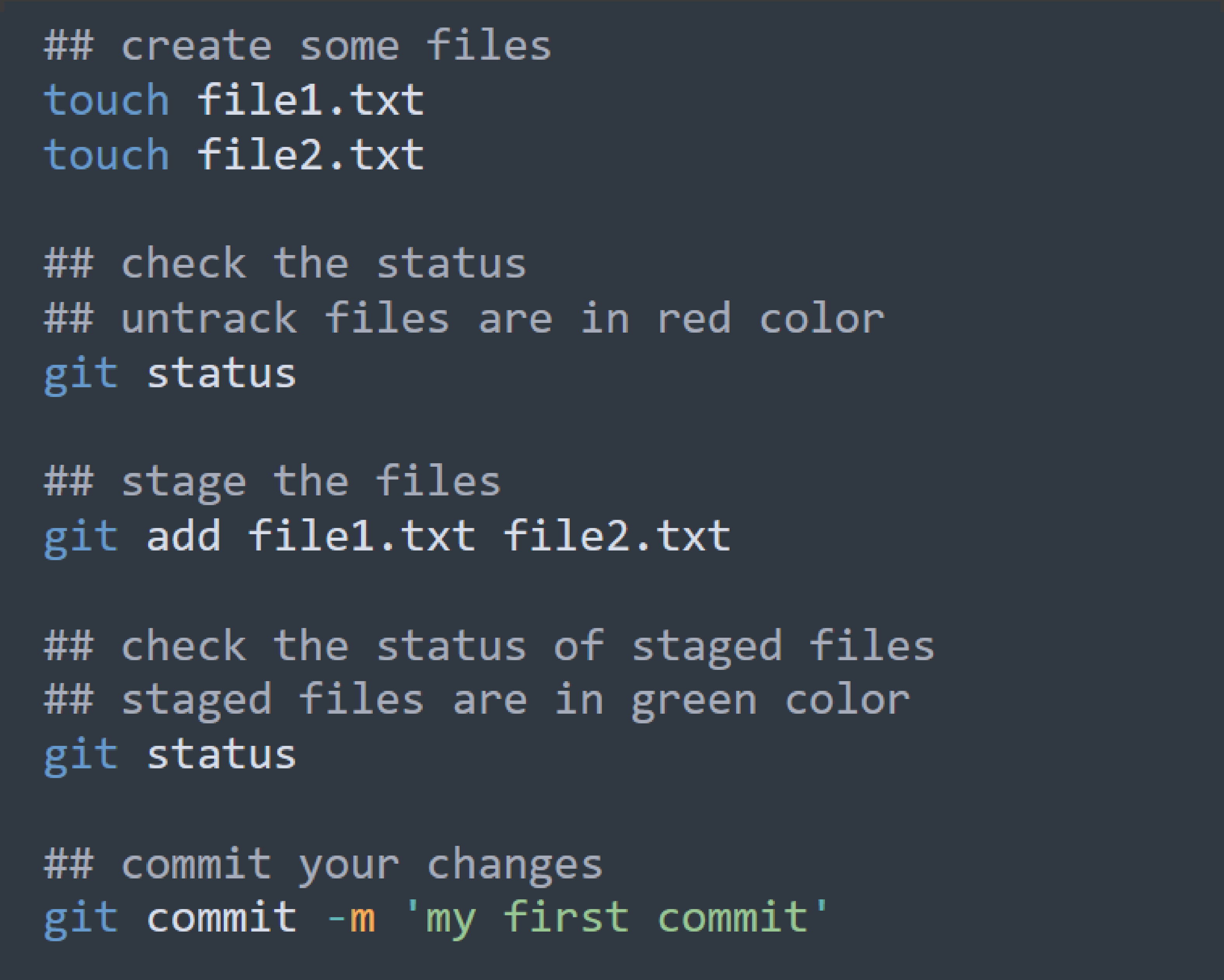
```
git --help
```

Create a new local repository



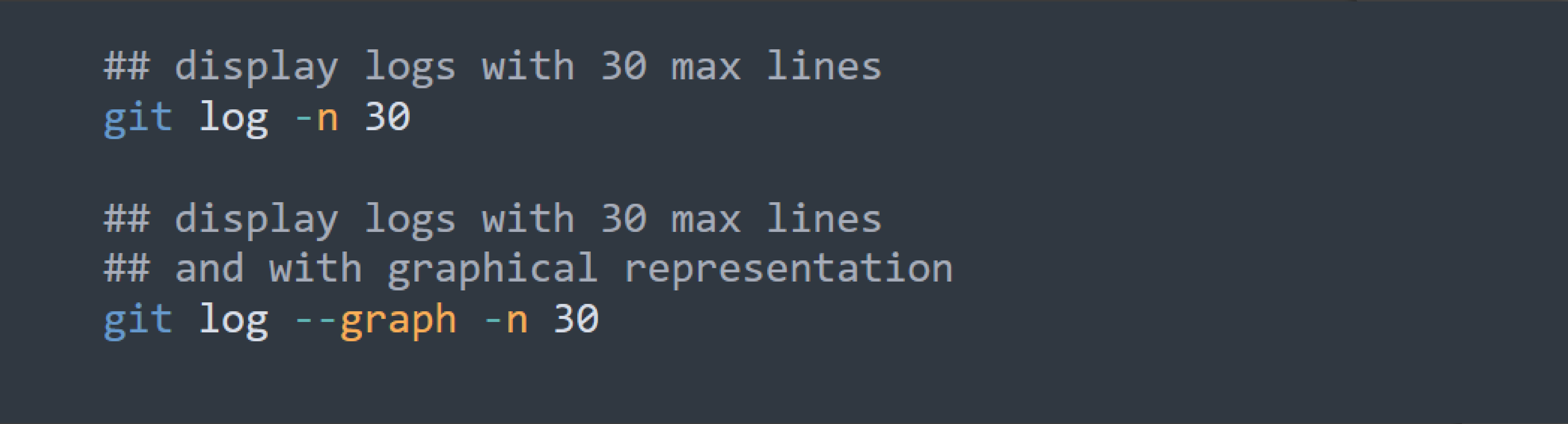
```
## Go to your User's Home Directory
##
cd

## Initialize a Git repository
##
git init test-repo

## Go inside the repository folder
##
cd test-repo
```

Adding files to the repository



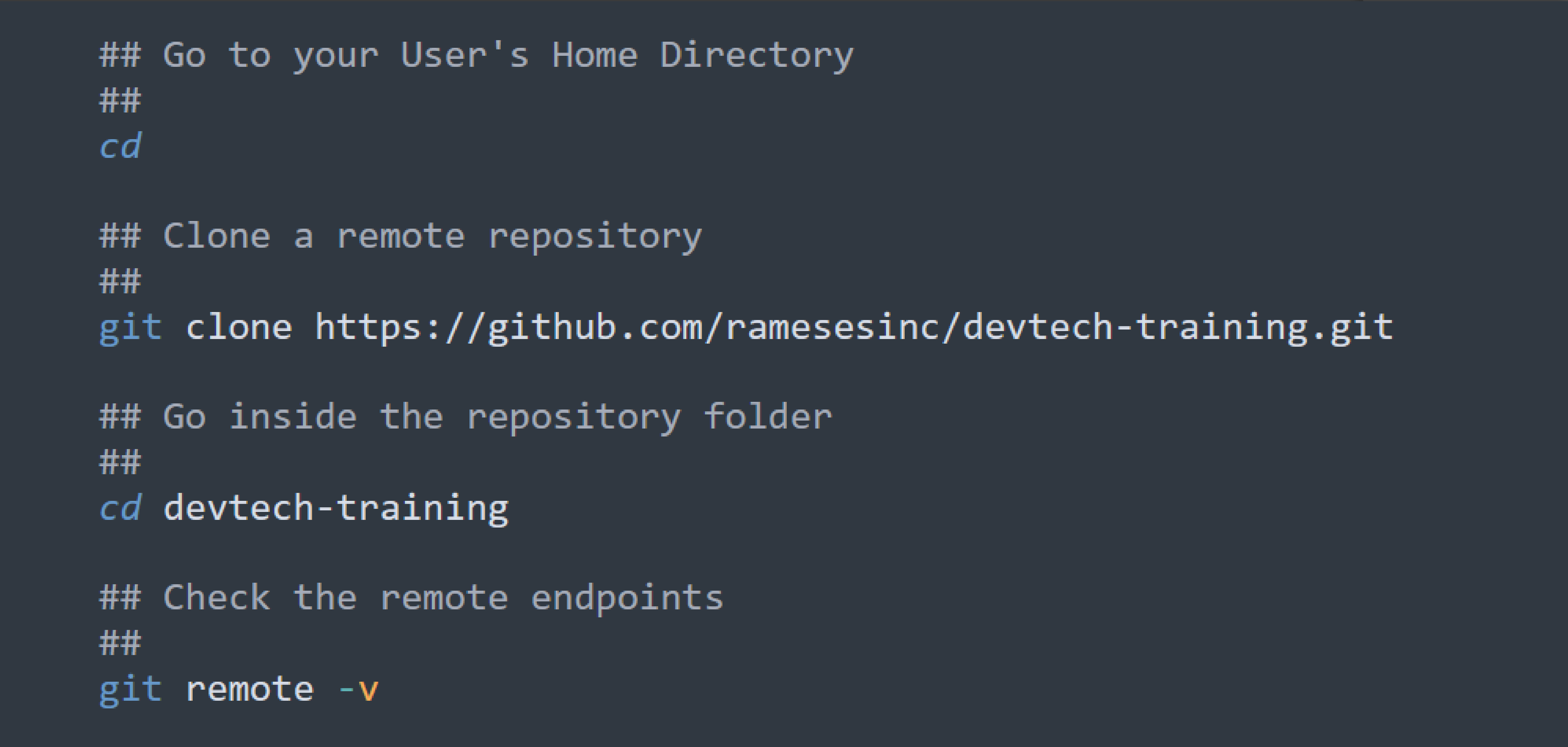
```
## create some files
touch file1.txt
touch file2.txt

## check the status
## untrack files are in red color
git status

## stage the files
git add file1.txt file2.txt

## check the status of staged files
## staged files are in green color
git status

## commit your changes
git commit -m 'my first commit'
```

Check repository logs



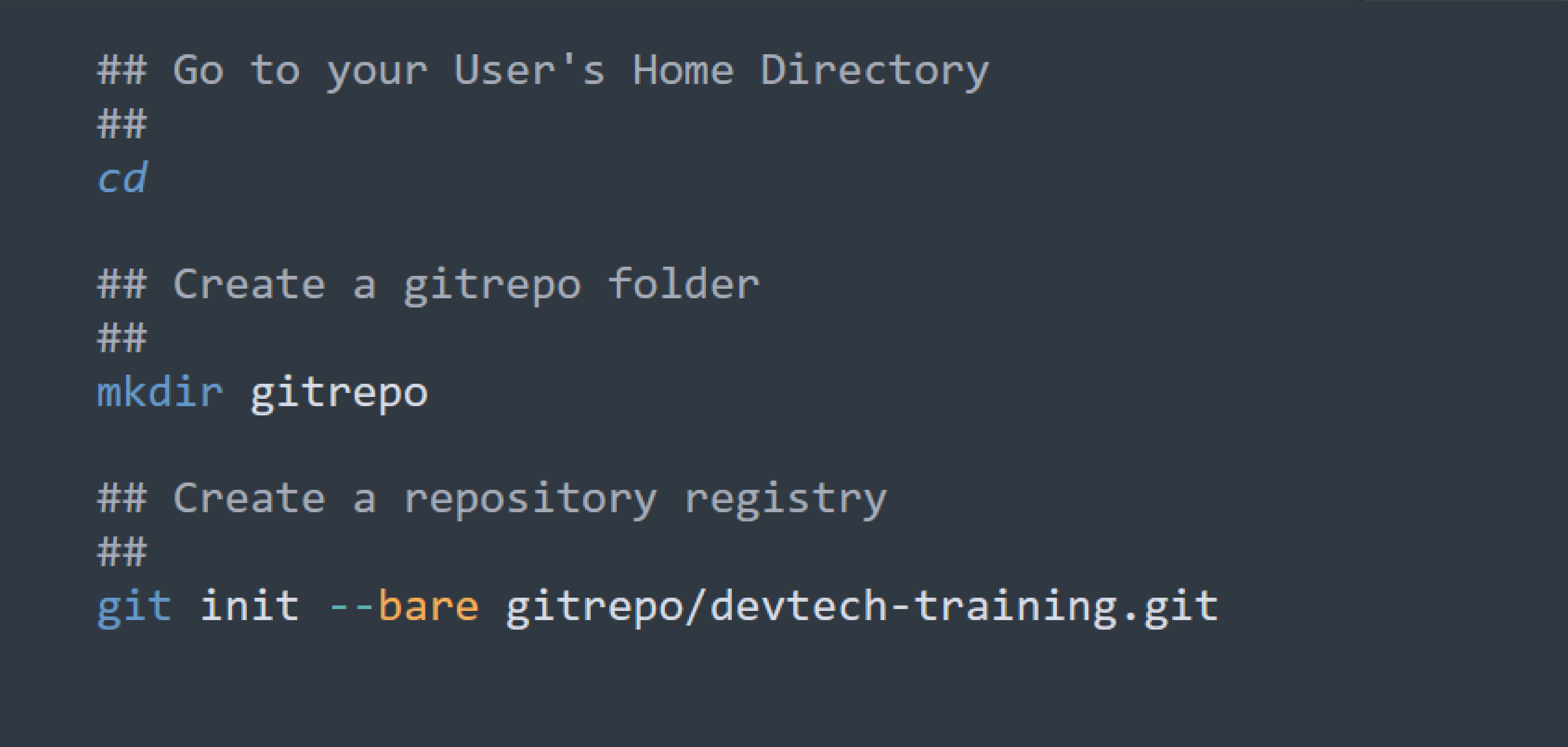
```
## display logs with 30 max lines
git log -n 30

## display logs with 30 max lines
## and with graphical representation
git log --graph -n 30
```

Cloning repository from GitHub



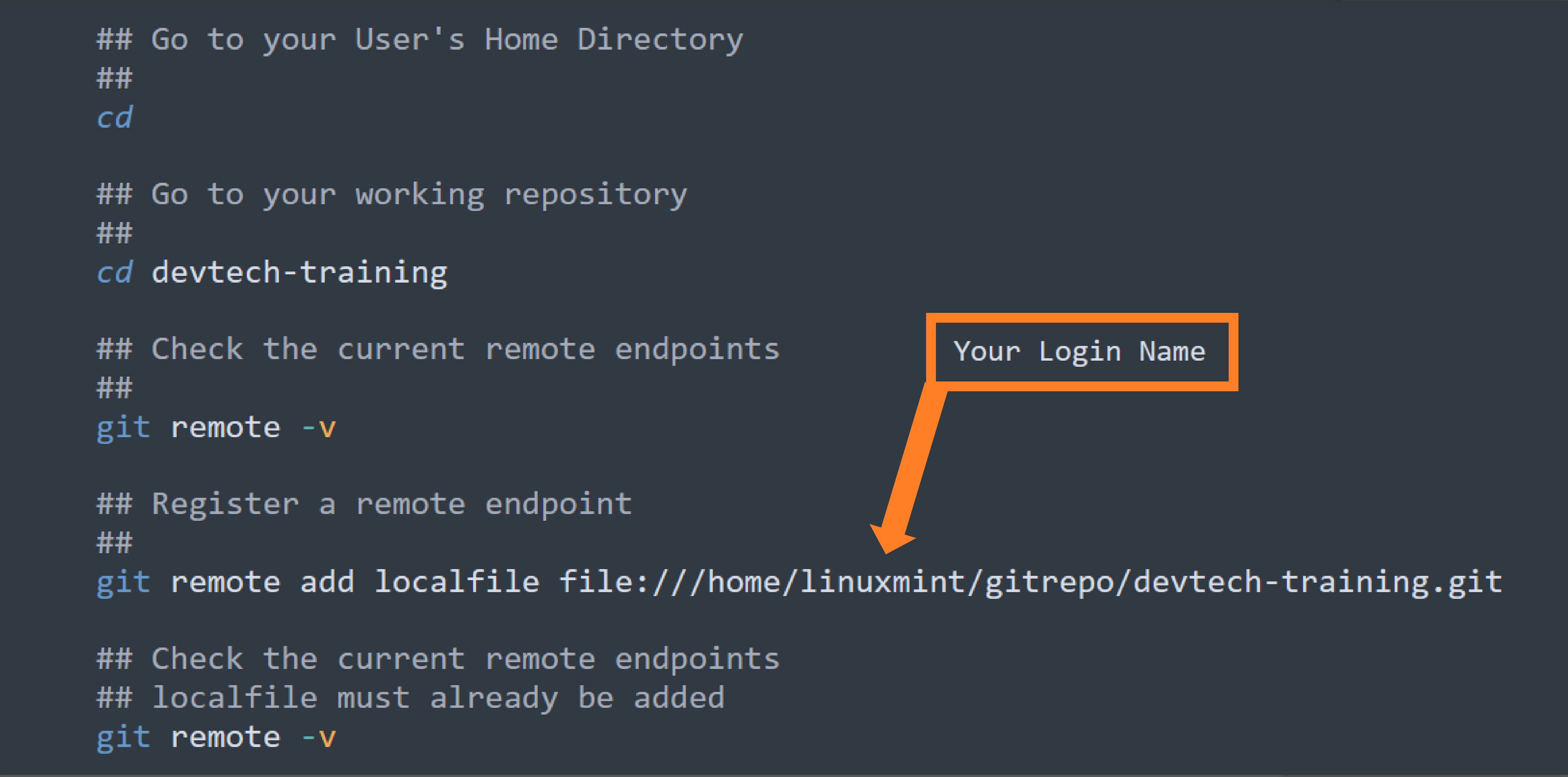
```
## Go to your User's Home Directory
##
cd

## Clone a remote repository
##
git clone https://github.com/ramesesinc/devtech-training.git

## Go inside the repository folder
##
cd devtech-training

## Check the remote endpoints
##
git remote -v
```

Create a local repository registry



```
## Go to your User's Home Directory
##
cd

## Create a gitrepo folder
##
mkdir gitrepo

## Create a repository registry
##
git init --bare gitrepo/devtech-training.git
```

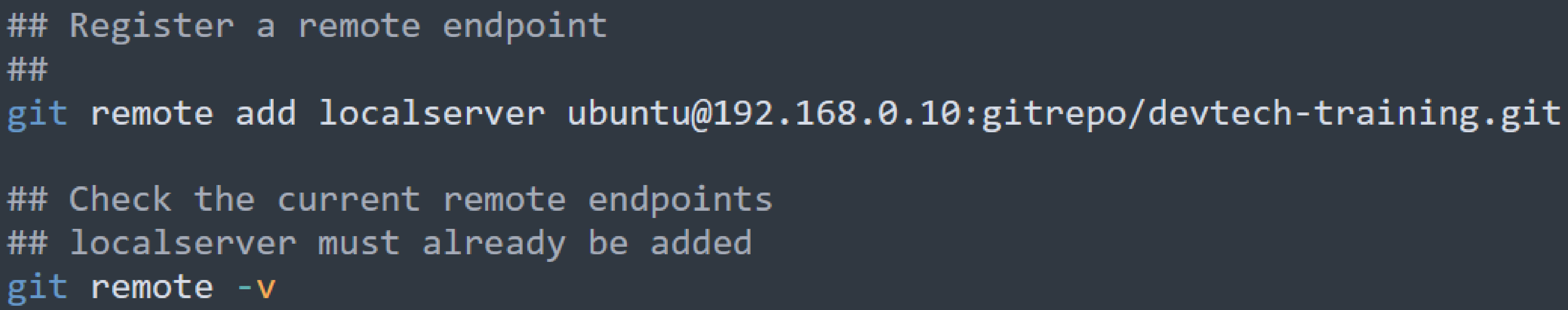
Mount a local repository registry from file



```
## Go to your User's Home Directory
##
cd

## Go to your working repository
##
cd devtech-training

## Check the current remote endpoints
##
git remote -v

## Register a remote endpoint
##
git remote add localfile file:///home/linuxmint/gitrepo/devtech-training.git

## Check the current remote endpoints
## localfile must already be added
git remote -v
```

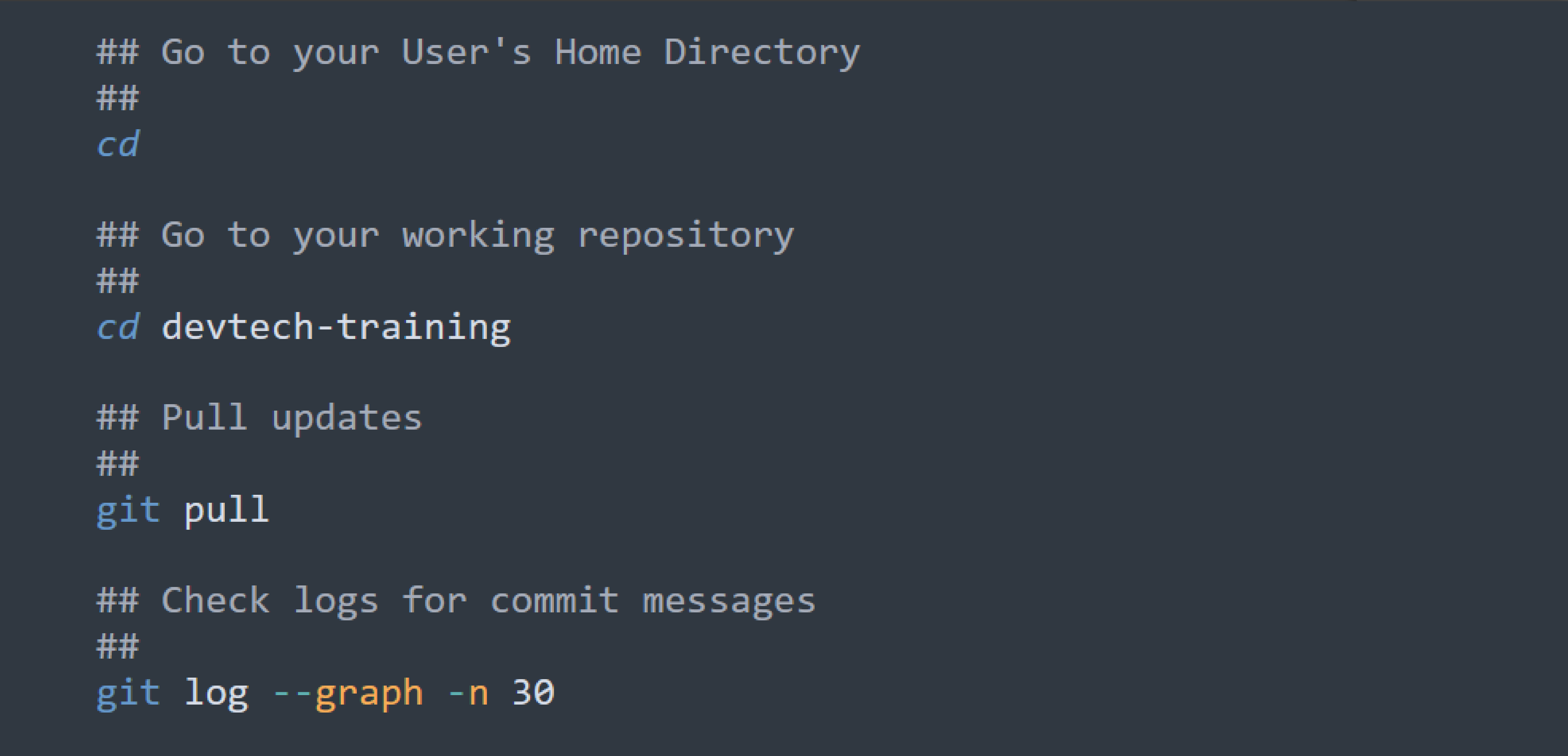
Your Login Name

Mount a local repository registry from your local server



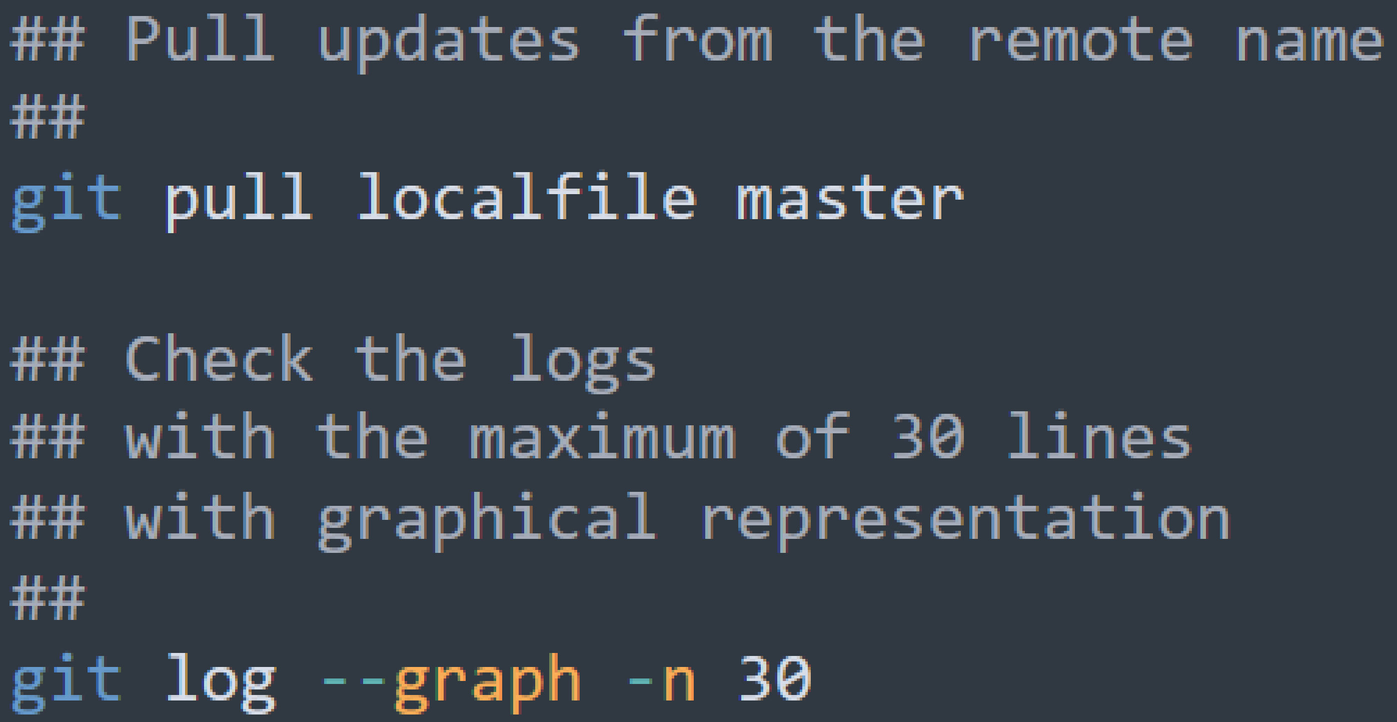
```
## Register a remote endpoint
##
git remote add localserver ubuntu@192.168.0.10:gitrepo/devtech-training.git

## Check the current remote endpoints
## localserver must already be added
git remote -v
```

Pull updates from remote repository



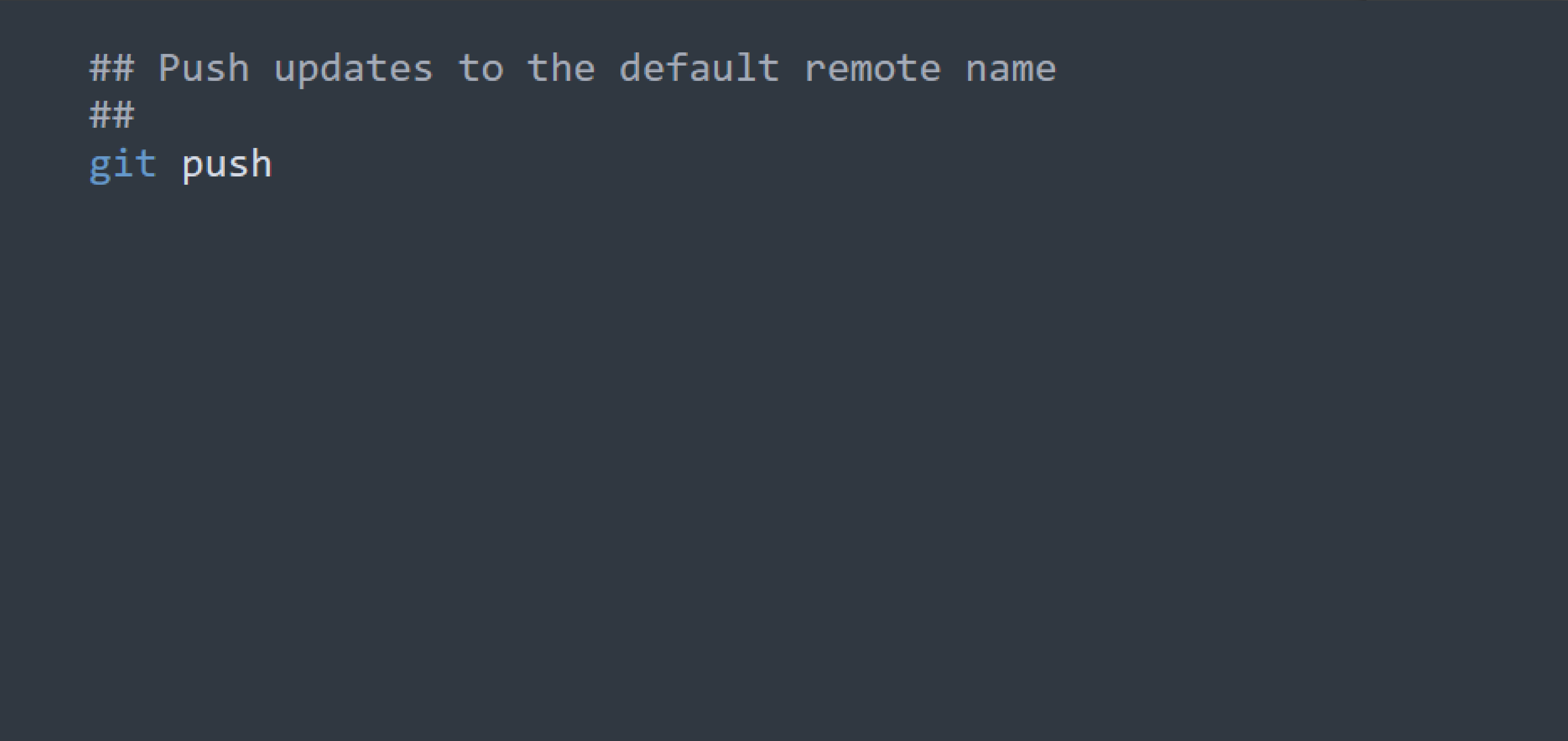
```
## Go to your User's Home Directory
##
cd

## Go to your working repository
##
cd devtech-training

## Pull updates
##
git pull

## Check logs for commit messages
##
git log --graph -n 30
```

Pull updates from other remote repository



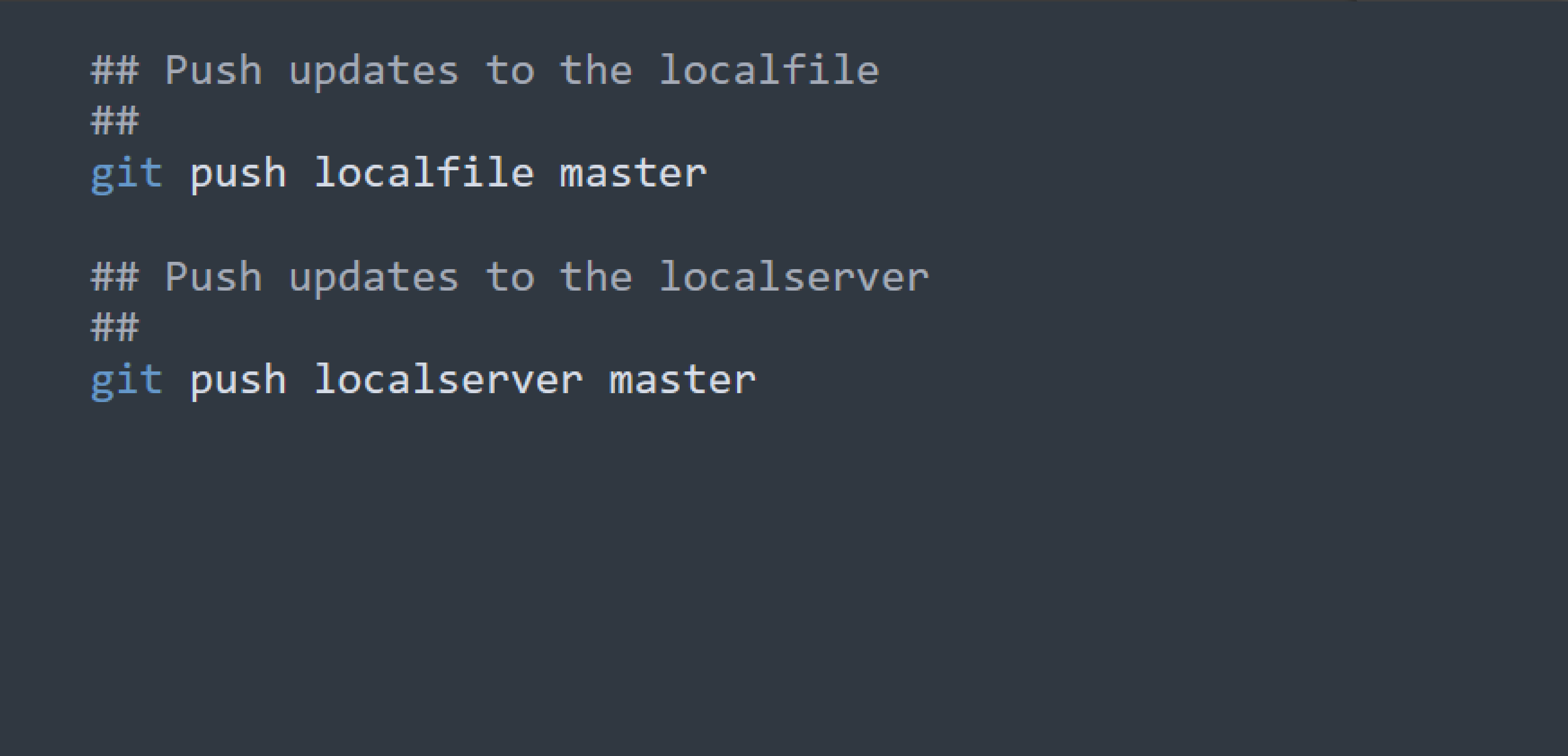
```
## Pull updates from the remote name
##
git pull localfile master

## Check the logs
## with the maximum of 30 lines
## with graphical representation
##
git log --graph -n 30
```

Push updates to remote repository



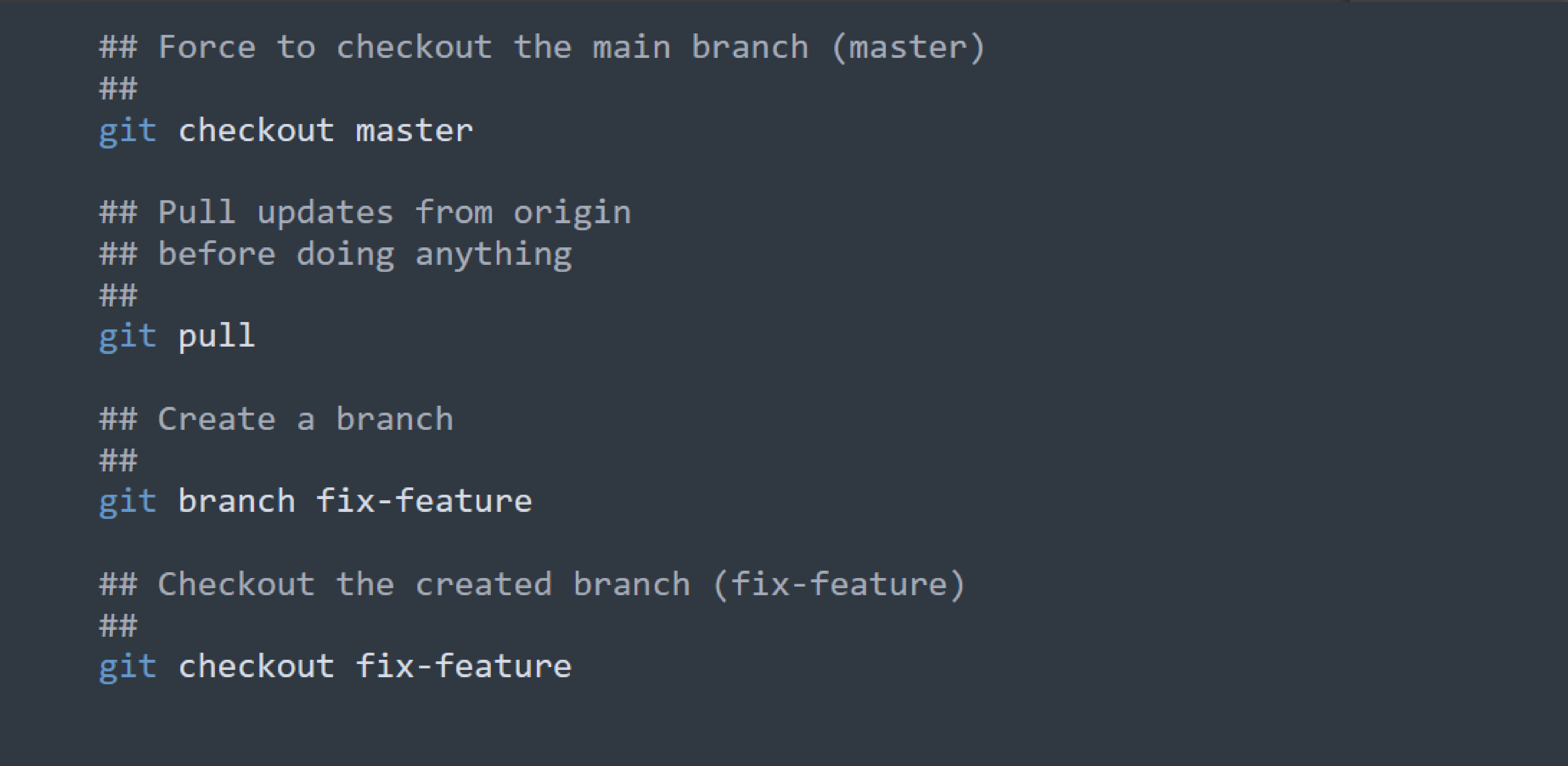
```
## Push updates to the default remote name
##
git push
```

Push updates to other remote repository



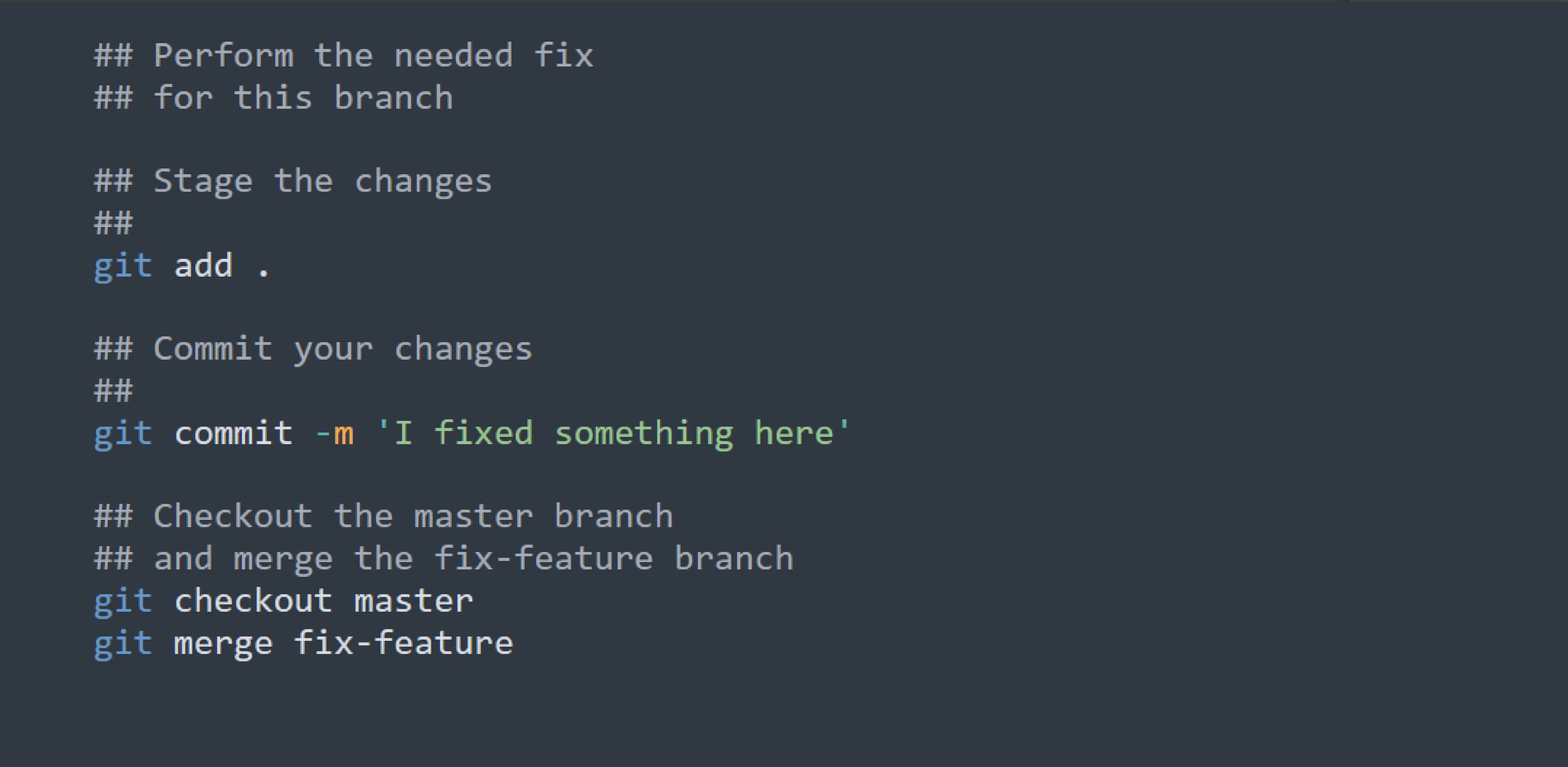
```
## Push updates to the localfile
##
git push localfile master

## Push updates to the localserver
##
git push localserver master
```

Create a branch to fix isolated bug



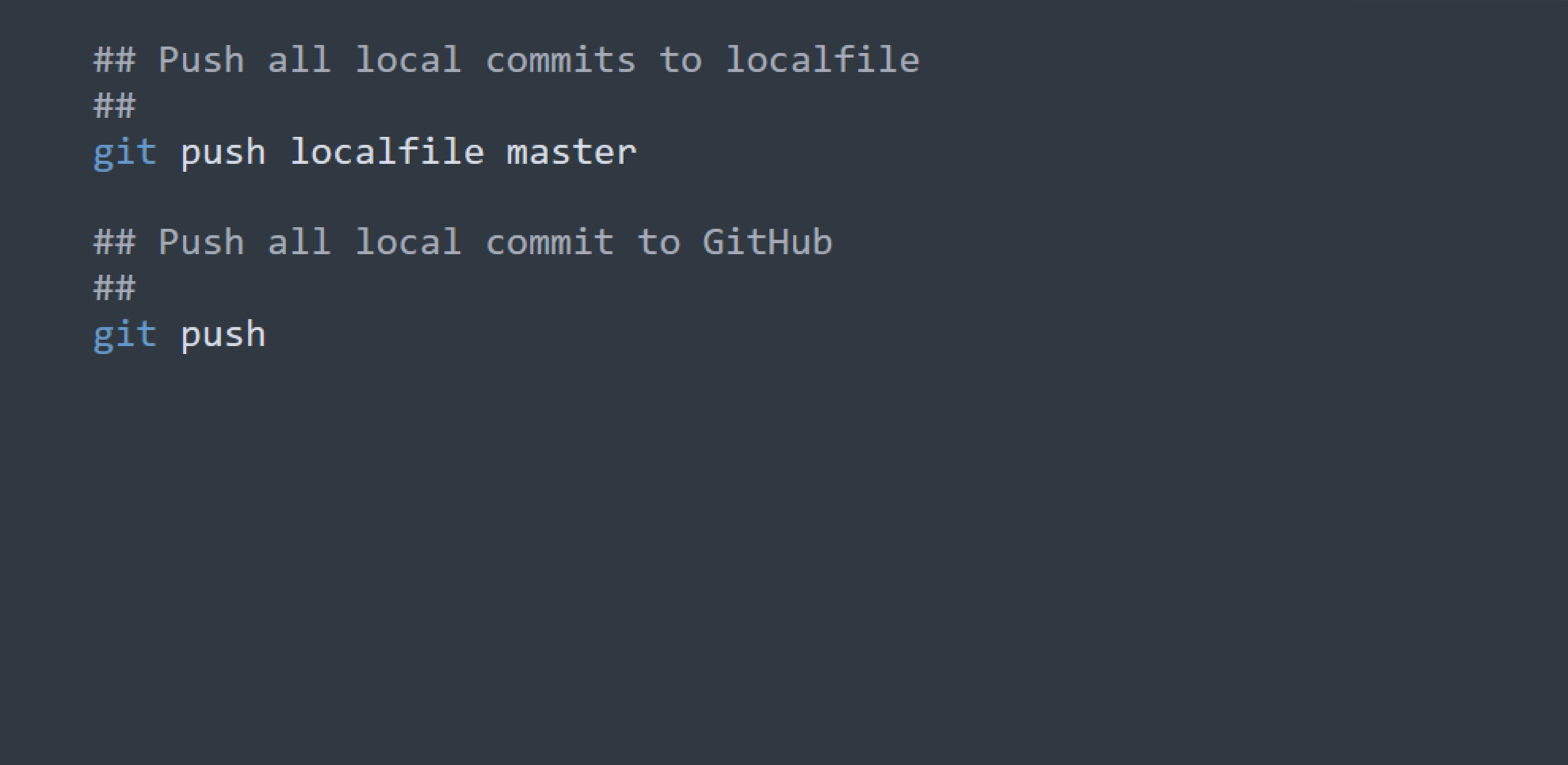
```
## Force to checkout the main branch (master)
##
git checkout master

## Pull updates from origin
## before doing anything
##
git pull

## Create a branch
##
git branch fix-feature

## Checkout the created branch (fix-feature)
##
git checkout fix-feature
```

Create a branch to fix isolated bug



```
## Perform the needed fix
## for this branch

## Stage the changes
##
git add .

## Commit your changes
##
git commit -m 'I fixed something here'

## Checkout the master branch
## and merge the fix-feature branch
git checkout master
git merge fix-feature
```

Create a branch to fix isolated bug



```
## Push all local commits to localfile
##
git push localfile master

## Push all local commit to GitHub
##
git push
```

Documentations

- **Git**

  `https://git-scm.com/doc`

- **GitHub**

  `https://github.com`

Up Next

DevTech Training - Part 2

- Virtualization
- Virtual Machine vs Docker
- Docker
- iReport Designer
- Report Editing and Management