

SWEN90016

Software Processes and Management

Tutorial 07 – Configuration Management, Version Control

- Rahul Sharma <sharma1@student.unimelb.edu.au>

Today's aim

Become familiar with

GIT

and octocat



So what is Git?

- Version Control System
- Made by Linus Torvalds! (developer of the Linux kernel)



Git Advantages

- Distributed (everyone has their own code repository local to them!)
- Open Source (everyone likes open source code :))



General Git Workflow

1. git init

Initialize local repository called **master**

2. git add filename

Track changes made to the contents of this file

3. git commit -m "message"

Save this file in local repository

4. git push

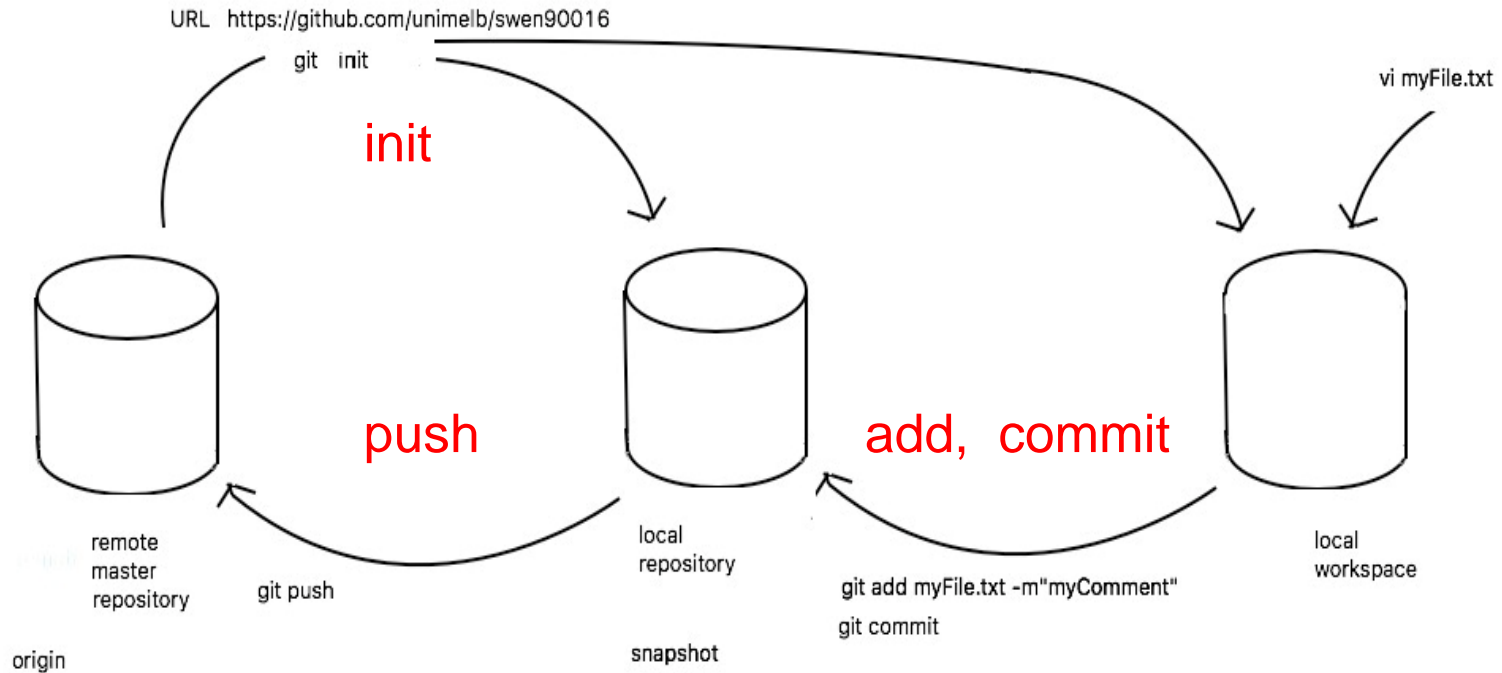
Save this collection

- to remote repository called **origin**

- from local repository called **master**



Git Round Trip



git init

What does it do: Initializes an empty Git repository locally

Its empty!



git clone

its not empty!

What does it do:

1. Initializes an empty local repository
2. fetches code from the remote repository into your local repository

git pull remote_repository_URL

What does it do: fetches code from the remote repository into your local repository



Git Concept: Clone

What does it do:

1. Initializes an empty Git repository locally, and
2. fetches code from the remote Git repository into your local repository

similar to svn checkout but you get the whole copy, a lot of history, and can take a while

Example command:

```
git clone https://github.com/tiow/swen90016_workshop05
```



Git Concept: Status

What does it do: Shows you the status of your local repository

- shows files to be added, modified and untracked files as well.

Example command:

git status

```
# On branch master
# Changes to be committed:
# (use "git reset HEAD <file>..." to unstage)
#
#modified:   hello.py
#
# Changes not staged for commit:
# (use "git add <file>..." to update what will be committed
# (use "git checkout -- <file>..." to discard changes in wo
#
#modified:   main.py
#
# Untracked files:
# (use "git add <file>..." to include in what will be commi
#
#hello.pyc
```

Git Concept: **Add / Remove**

- Adds or removes a file from your workspace to your local repository
- Files in local repository are called “staged”
- Staged files have their contents monitored

Example command:

git add filename OR **git rm** filename



Git Concept: **Commit**

What does it do: This saves your changes to the local repository.

Example command:

```
git commit -m "some message!"
```

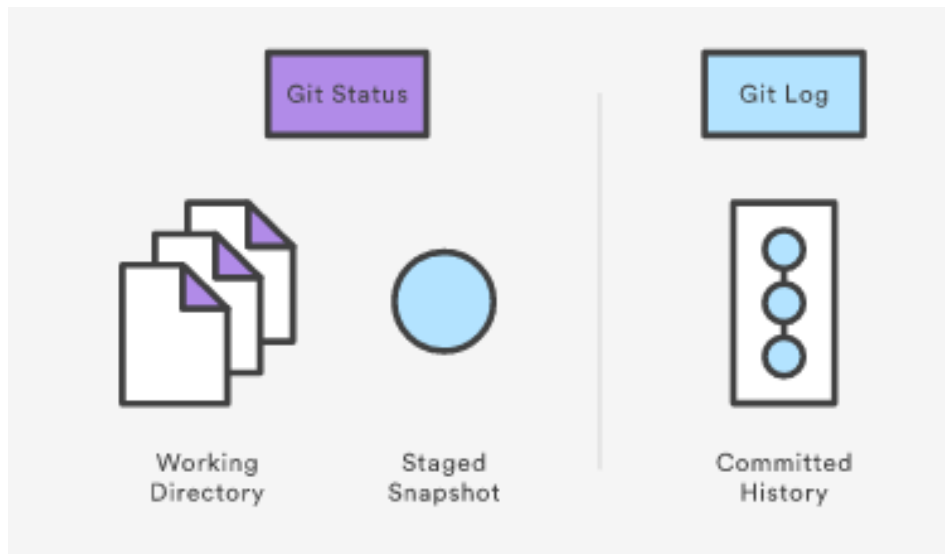


Git Concept: Log

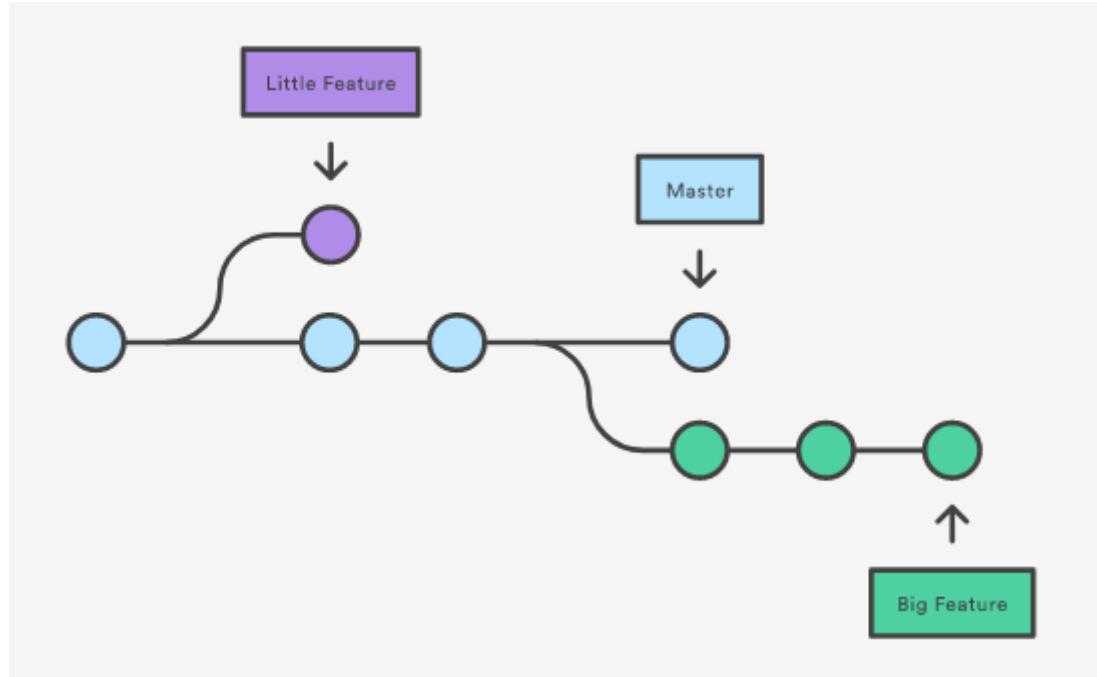
What does it do: Shows the history of commits into the system.

Example command:

git log



Git Concept: **Branching**



Exercise: learn git online

Access git tutorial online from: <https://try.github.io>

Now type git commands:

git init

git status



git add octocat.txt

This will stage all the files in the current directory. Type:

git status

This will show the staging area.

git commit -m "my first commit"

git log

This will show the current logs of all your commit.

```
-bash-4.1$ ll
total 1
-rw-r--r--. 1 rahuls2 mstaff 20 Sep  9 22:16 testOne.txt
-bash-4.1$ git init
Initialized empty Git repository in /silo-q04/users/r/rahuls2/testDir/.git/
-bash-4.1$ git add .
-bash-4.1$ git status
# On branch master
#
# Initial commit
#
# Changes to be committed:
#   (use "git rm --cached <file>..." to unstage)
#
#       new file:   testOne.txt
#
-bash-4.1$
```


Find the new, empty GitHub remote repository, with the following URL

git remote add origin <https://github.com/try-git/try-git.git>

and

Push our local *repo* to the remote GitHub server repository

git push -u origin master



Access git on the UniMelb server

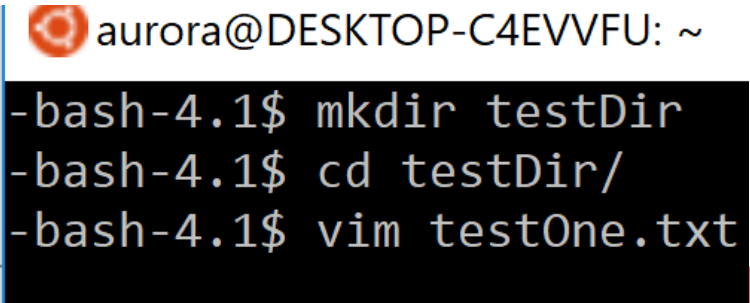
- Use *any SSH client* (PuTTY/ MobaXterm/ Terminal/ Bitvise) to ssh into Uni server
- Login: `ssh <your LMS Username>@swenshare.cis.unimelb.edu.au`
- Once you have access, try the following commands:


mkdir testDir

cd testDir

- Create a file:

vim testOne.txt

A terminal window with a black background and white text. The prompt is 'aurora@DESKTOP-C4EVVUFU: ~'. The commands entered are 'mkdir testDir', 'cd testDir/', and 'vim testOne.txt'. The window has a blue border on the left and a pink geometric pattern on the right.

 aurora@DESKTOP-C4EVVUFU: ~

```
-bash-4.1$ mkdir testDir  
-bash-4.1$ cd testDir/  
-bash-4.1$ vim testOne.txt
```

Type some content into the vim editor using insertion mode (press 'i') and after you are done, press Esc. Then type **:wq** and hit enter.

Now type:

II (LL CAPITALISED)

This will show you the contents of the directory where you should be able to see testOne.txt present.

Now type:

git init



git add .

This will stage all the files in the current directory. Type:

git status

This will show the status of the files.

git commit -m "My First Commit"

git log

This will show the current logs of all your commit.

```
-bash-4.1$ ll
total 1
-rw-r--r--. 1 rahuls2 mstaff 20 Sep  9 22:16 testOne.txt
-bash-4.1$ git init
Initialized empty Git repository in /silo-q04/users/r/rahuls2/testDir/.git/
-bash-4.1$ git add .
-bash-4.1$ git status
# On branch master
#
# Initial commit
#
# Changes to be committed:
#   (use "git rm --cached <file>..." to unstage)
#
#       new file:   testOne.txt
#
-bash-4.1$
```

vim testOne.txt

Add some text at the end of the file (in insertion mode 'i') then press Esc and type **:wq** and hit Enter.

git status

This will show that the file has not been staged. Meaning you cannot switch to a different commit right now else your changes will be lost. (use **-f** to force it).

git add .

git commit -m "My Second Commit"



git log

View all your commits and save the hash of the commits to notepad or somewhere.

Type: **git checkout <hash of first commit>**

vim testOne.txt

Modify the text (in insertion mode 'i') then press Esc.

Type **:wq** and press Enter.

```
-bash-4.1$ git log
commit f2c469258c584307a69309ebef1bf6e1523d781e
Author: Rahul SHARMA <rahuls2@4000v-cis-digitalis-1.eng.unimelb.edu.au>
Date: Sat Sep 9 22:22:09 2017 +1000

    My Second Commit

commit b104c8167a95951158a5af923ba293182aab683d
Author: Rahul SHARMA <rahuls2@4000v-cis-digitalis-1.eng.unimelb.edu.au>
Date: Sat Sep 9 22:19:32 2017 +1000

    My First Commit
-bash-4.1$
```

git add .

git status

git commit -m “My Parallel Commit”

git log

Where is My second commit ?

Type:

git checkout <second commit hash>

git log

```
-bash-4.1$ git log
commit f6e803fa1592df510a24a0b4d015b08bb35a04c2
Author: Rahul SHARMA <rahuls2@4000v-cis-digitalis-l.eng.unimelb.edu.au>
Date: Sat Sep 9 22:25:35 2017 +1000
```

My Parallel Commit

```
commit b104c8167a95951158a5af923ba293182aab683d
Author: Rahul SHARMA <rahuls2@4000v-cis-digitalis-l.eng.unimelb.edu.au>
Date: Sat Sep 9 22:19:32 2017 +1000
```

My First Commit

```
-bash-4.1$ git checkout f2c469258c584307a69309ebef1bf6e1523d781e
Warning: you are leaving 1 commit behind, not connected to
any of your branches:
```

f6e803f My Parallel Commit

If you want to keep them by creating a new branch, this may be a good time to do so with:

```
git branch new_branch_name f6e803fa1592df510a24a0b4d015b08bb35a04c2
```

HEAD is now at f2c4692... My Second Commit

```
-bash-4.1$ git log
commit f2c469258c584307a69309ebef1bf6e1523d781e
Author: Rahul SHARMA <rahuls2@4000v-cis-digitalis-l.eng.unimelb.edu.au>
Date: Sat Sep 9 22:22:09 2017 +1000
```

My Second Commit

```
commit b104c8167a95951158a5af923ba293182aab683d
Author: Rahul SHARMA <rahuls2@4000v-cis-digitalis-l.eng.unimelb.edu.au>
Date: Sat Sep 9 22:19:32 2017 +1000
```

My First Commit

To create a branch, type:

git checkout -b <name of your branch>

git show-branch --all

vim testOne.txt

git add .

git commit -m "Mario's first commit"

git show-branch.

```
-bash-4.1$ git checkout -b mario
Switched to a new branch 'mario'
-bash-4.1$ git show-branch --all
* [mario] My Second Commit
! [master] My Second Commit
--
*+ [mario] My Second Commit
-bash-4.1$ vim testOne.txt
-bash-4.1$ git add .
-bash-4.1$ git commit -m "Mario's first commit"
[mario e14ef1a] Mario's first commit
Committer: Rahul SHARMA <rahuls2@4000v-cis-digitalis-1.eng.unimelb.edu.au>
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:

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

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

    git commit --amend --reset-author

1 file changed, 1 insertion(+), 1 deletion(-)
-bash-4.1$ git show-branch
* [mario] Mario's first commit
! [master] My Second Commit
--
* [mario] Mario's first commit
*+ [master] My Second Commit
-bash-4.1$
```


Usually we pass our code through multiple stages like test environments and so on using tools or defined processes before committing our code into the master branch which hosts our production level code. (for Ex - <https://cucumber.io/>)

Here, we assume that everything is alright and we can commit to the master branch directly. Type:

git checkout master

git merge Mario

git log

```
-bash-4.1$ git checkout master
Switched to branch 'master'
-bash-4.1$ git merge mario
Updating f2c4692..e14ef1a
Fast-forward
 testOne.txt | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)
-bash-4.1$ git log
commit e14ef1a0c68aa271d8505d0a0f2d705b488ab7c8
Author: Rahul SHARMA <rahuls2@4000v-cis-digitalis-l.eng.unimelb.edu.au>
Date:   Sat Sep 9 22:29:34 2017 +1000

    Mario's first commit


commit f2c469258c584307a69309ebef1bf6e1523d781e
Author: Rahul SHARMA <rahuls2@4000v-cis-digitalis-l.eng.unimelb.edu.au>
Date:   Sat Sep 9 22:22:09 2017 +1000

    My Second Commit

commit b104c8167a95951158a5af923ba293182aab683d
Author: Rahul SHARMA <rahuls2@4000v-cis-digitalis-l.eng.unimelb.edu.au>
Date:   Sat Sep 9 22:19:32 2017 +1000

    My First Commit
-bash-4.1$
```

Resources

- Check out: <https://education.github.com/pack> (provides student pack info)
 - When using git on cloud, use a tool: <https://www.sourcetreeapp.com/>
 - Look at FOSS for every use case you encounter because they are usually the cheapest (free) and best resource (open source community – well supported) out there for most projects. Most proprietary tools use FOSS underneath.
 - Look at IDE's and text editors. Most modern text editors have in-built support for git with direct integration with github and bitbucket.
- 

Thank you

References:

<https://www.atlassian.com/git/tutorials>

<https://git-scm.com/>

<https://try.github.io/>

http://thehackernews.com/2015/03/github-hit-by-massive-ddos-attack-from_27.html

