Software Engineering

CS305, Autumn 2020

Week 2

Last Week…

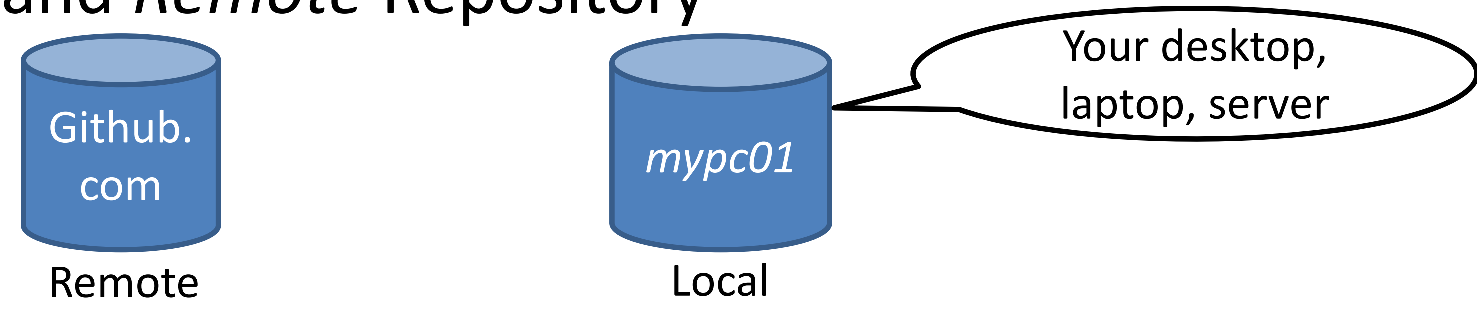
- Software Engineering Overview
  - What is it? Why needed? How to manage complexity?
  - Different software process models
  - How to choose a model? factors to consider
  - Tools for developer productivity

This class…

- Tools overview – Git

Git

- Version Control System
  - – Manage versions of your code – access to different versions when needed
  - – Lets you collaborate
- ‘Repository’ – term used to represent storage
  - – *Local* and *Remote* Repository



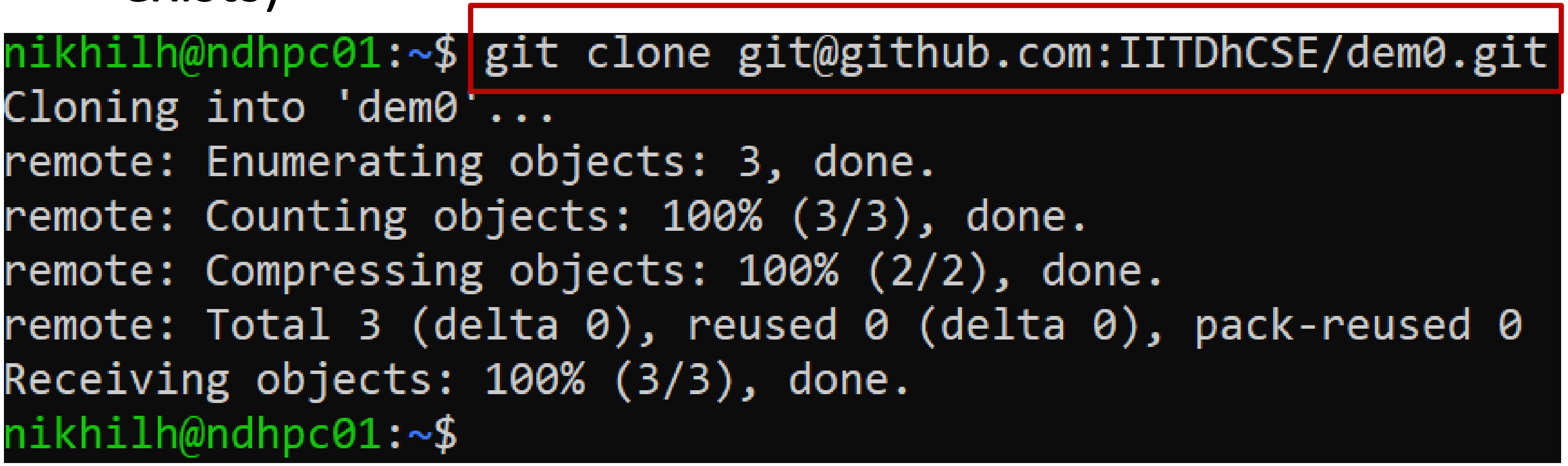
Git – Creating Repositories

- Two ways:
  1. ‘Clone’ / Download an existing repository from GitHub
  2. Create local repository first and then make it available on GitHub

`git clone` for creating local working copy

– ‘Clone’ / Download an existing repository from GitHub – get your own copy of source code

- `git clone` (when a remote repository on GitHub.com exists)

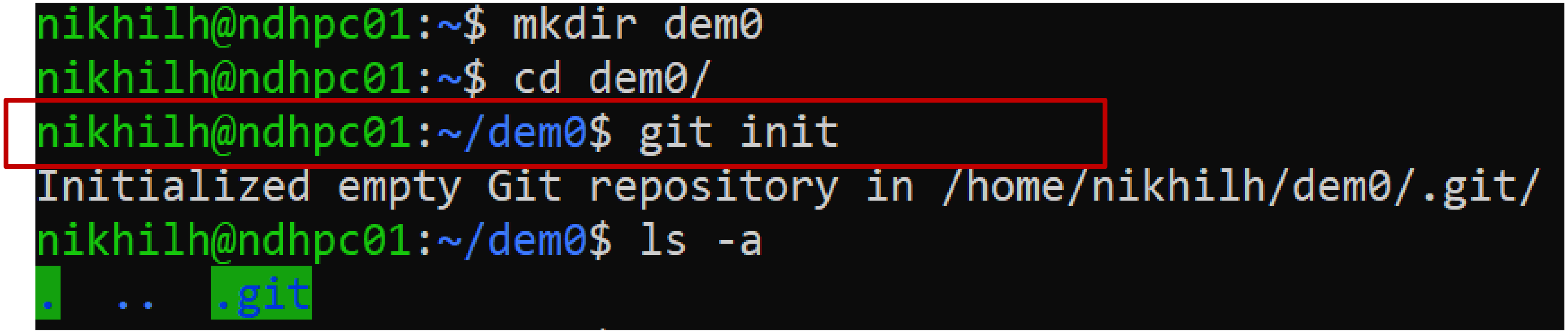


```
nikhilh@ndhpc01:~$ git clone git@github.com:IITDhCSE/dem0.git
Cloning into 'dem0'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (3/3), done.
nikhilh@ndhpc01:~$
```

`Git init` for initializing local repository

– Create local repository first and then make it available on GitHub

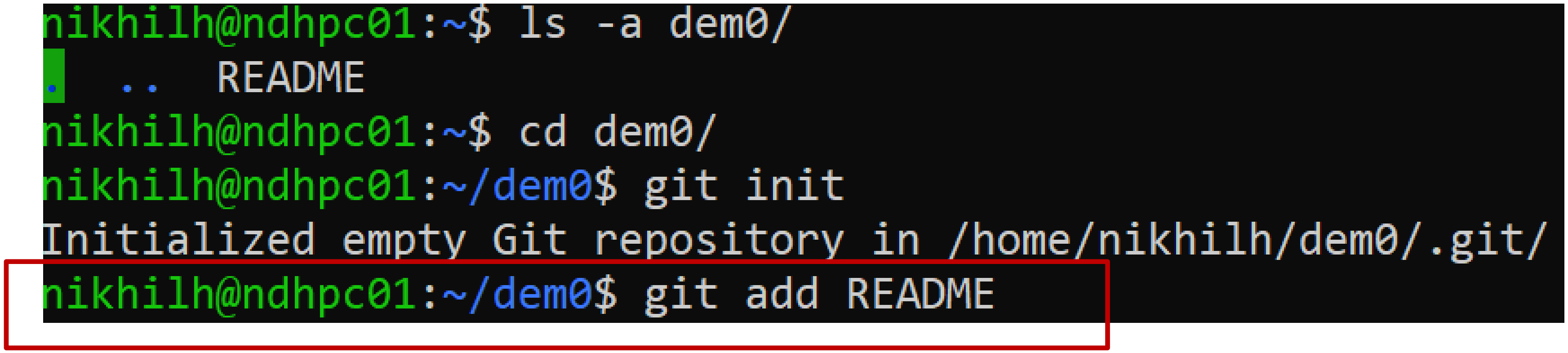
1. `git init` - converts a directory to Git local repo



```
nikhilh@ndhpc01:~$ mkdir dem0
nikhilh@ndhpc01:~$ cd dem0/
nikhilh@ndhpc01:~/dem0$ git init
Initialized empty Git repository in /home/nikhilh/dem0/.git/
nikhilh@ndhpc01:~/dem0$ ls -a
.  ..  .git
```

`git` add for staging files

2. `git add` – ‘stage’ a file i.e. prepare for saving the file on local repository

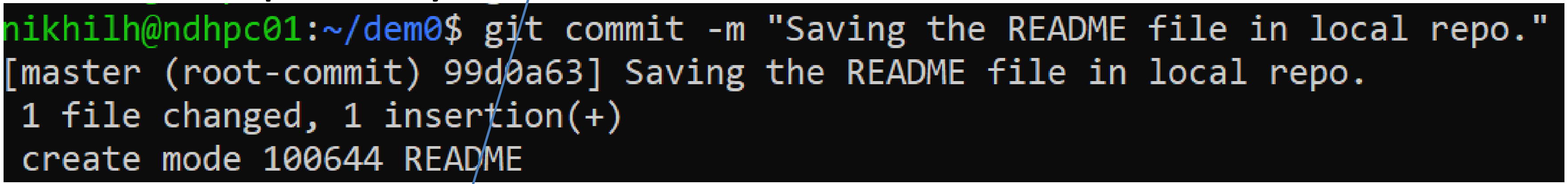


```
nikhilh@ndhpc01:~$ ls -a dem0/
.  ..  README
nikhilh@ndhpc01:~$ cd dem0/
nikhilh@ndhpc01:~/dem0$ git init
Initialized empty Git repository in /home/nikhilh/dem0/.git/
nikhilh@ndhpc01:~/dem0$ git add README
```

Note that creating a file, say, README2 in dem0 directory does not *automatically* make it part of the local repository

`git commit` for saving changes in local repository

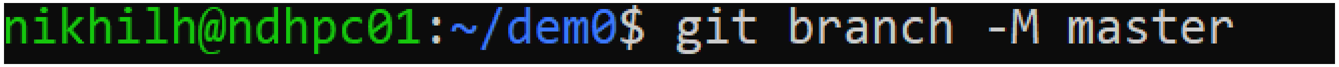
3. `git commit` – ‘commit’ changes i.e. save all the changes (adding a new file in this example) in the local repository



```
nikhilh@ndhpc01:~/dem0$ git commit -m "Saving the README file in local repo."
[master (root-commit) 99d0a63] Saving the README file in local repo.
 1 file changed, 1 insertion(+)
 create mode 100644 README
```

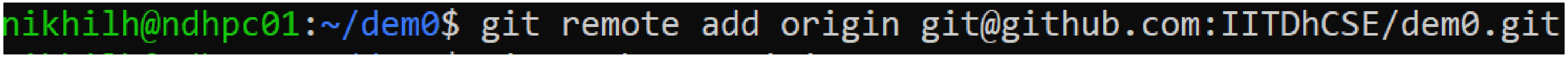
How to save changes done when you must overwrite an existing file?

4. `git branch -M master` – rename the current as ‘`master`’ (-M for force rename even if a branch by that name already exists)



```
nikhilh@ndhpc01:~/dem0$ git branch -M master
```

5. `git remote add origin git@github.com:IITDhCSE/dem0.git` – prepare the local repository to be managed as a tracked repository



```
nikhilh@ndhpc01:~/dem0$ git remote add origin git@github.com:IITDhCSE/dem0.git
```

command to manage remote repo.

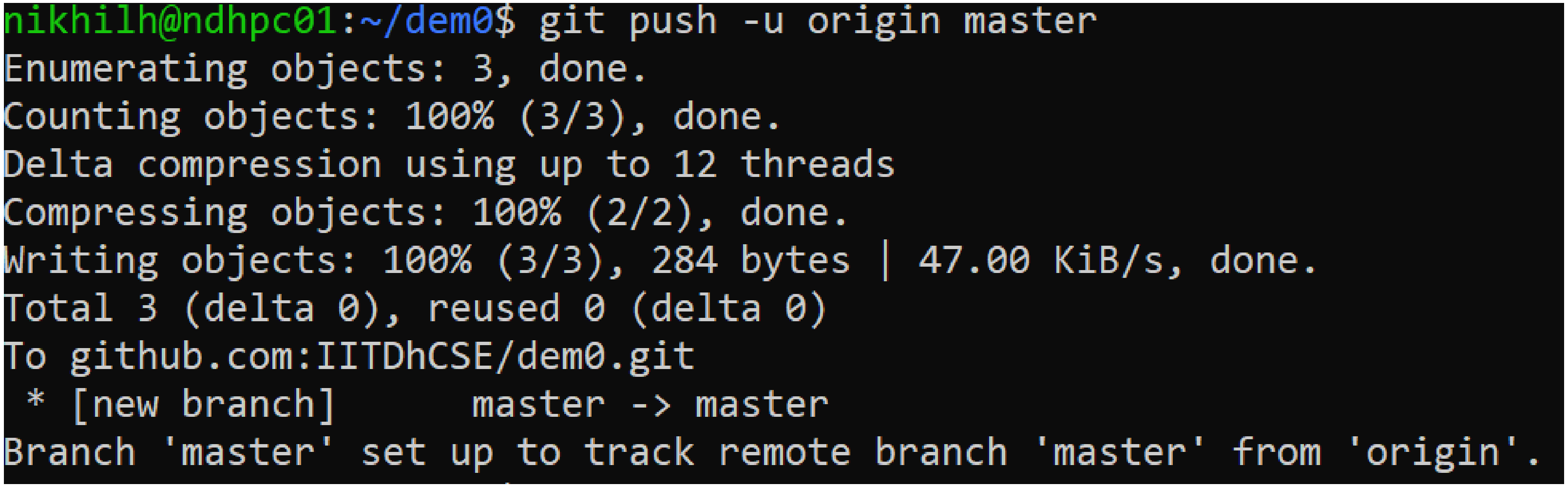
associates a name ‘origin’ with the remote repo’s URL

The URL of the repository on GitHub.com.

- This URL can be that of any other user’s or server’s address.
- uses SSH protocol
  - HTTP protocol is an alternative. Looks like: https://github.com/IITDhCSE/dem0.git

git push for saving changes in remote repo

6. `git push –u origin master` – ‘push’ or save all the changes done to the ‘master’ branch in local repo to remote repo. *(necessary for guarding against deletes to local repository)*



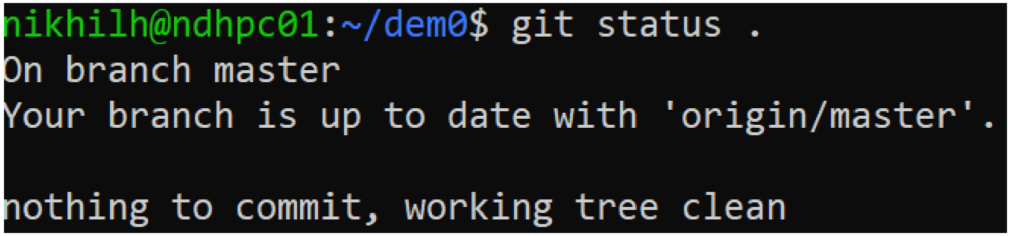
```
nikhilh@ndhpc01:~/dem0$ git push -u origin master
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 12 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 284 bytes | 47.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To github.com:IITDhCSE/dem0.git
 * [new branch]      master -> master
Branch 'master' set up to track remote branch 'master' from 'origin'.
```

syntax: `git push <remotename> <branchname>`

what does the –u option do?

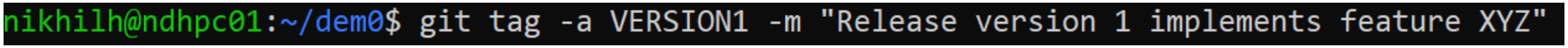
Git – Releasing Code

- Tagging
  - Check for unsaved changes in local repository.



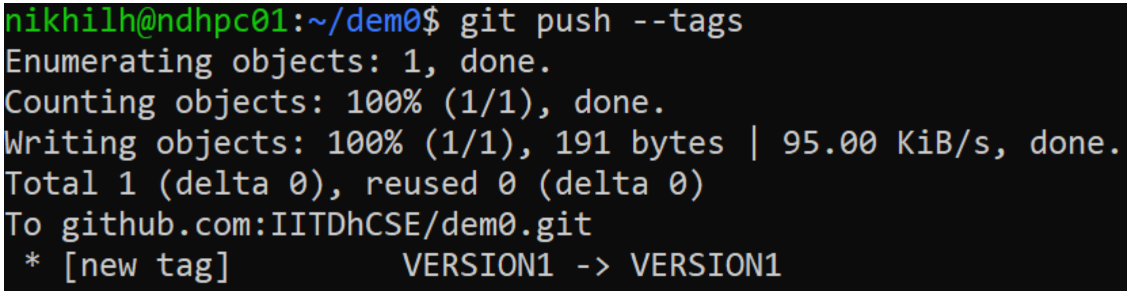
```
nikhilh@ndhpc01:~/dem0$ git status .
On branch master
Your branch is up to date with 'origin/master'.

nothing to commit, working tree clean
```

  - Create a tag and associate a comment with that tag



```
nikhilh@ndhpc01:~/dem0$ git tag -a VERSION1 -m "Release version 1 implements feature XYZ"
```

  - Save tags in remote repository



```
nikhilh@ndhpc01:~/dem0$ git push --tags
Enumerating objects: 1, done.
Counting objects: 100% (1/1), done.
Writing objects: 100% (1/1), 191 bytes | 95.00 KiB/s, done.
Total 1 (delta 0), reused 0 (delta 0)
To github.com:IITDhCSE/dem0.git
 * [new tag]         VERSION1 -> VERSION1
```

Git – Recap..

- Please read https://git-scm.com/book/en/v2 for details

```
1.  git clone (creating a local working copy)
2.  git add (staging the modified local copy)
3.  git commit (saving local working copy)
4.  git push (saving to remote repository)
5.  git tag (Naming the release with a label)
6.  git push --tags (saving the label to remote)
```

Demo