BASH/SHELL Workshop for Data Science

Session IV

Agenda

- Refresher on file operations
- ls vs find vs tree
- cat vs tac vs rev
- Working with GIT – Locally
- Working with GitHub.com and SSH
- Why is it a good idea to GIT your thesis?
- Basics of snapshot process with GIT

Refresher on file operations: Creating and Removing files

- Let's create some files:
- touch project1.csv project2.txt project3.jpg project4.png project24.txt
- rm *.txt
- rm *2*
- rm *[2,3]*
- In order to remove folders or directories we need to give recursive option
- mkdir deleteproject/dump{1,2,3}/file{1,2}
- rm –r is a powerful command and you may accidentally loose all of your computer data.
- Tip: Use rm –ri #an interactive option where computer asks permission before deleting any file or directory.

- One powerful example: do not ever type : rm -rf /* # Worst mistake any programmer can commit

ls vs find vs tree

- find . –maxdepth 1 #maxdepth defines the level at which find command should stop.
- find / #lists all the files on your computer.
- find . #lists files from the present folder.
- Let’s say we want to find only data files in a folder –
- find . –name ‘*.csv’
- find . –maxdepth 1 –type f # -type d is for directory and –type f is for files.
- find . –name”project1.txt”
- find . –maxdepth 2 –name”*.csv”
- find . –type f –size +100k |wc –l #finds files greater than 100 kilobytes in pwd.

- Tip: incase the computer throws error – we should use sudo before the command

CAT vs TAC vs rev

- USE $cat tac1.txt tact2.txt
- USE $tac tac1.txt tact2.txt
- TAC inverses vertically
- Rev inverses horizontally
- Find . –name “*.txt” | less # piping with less

Introduction to GIT

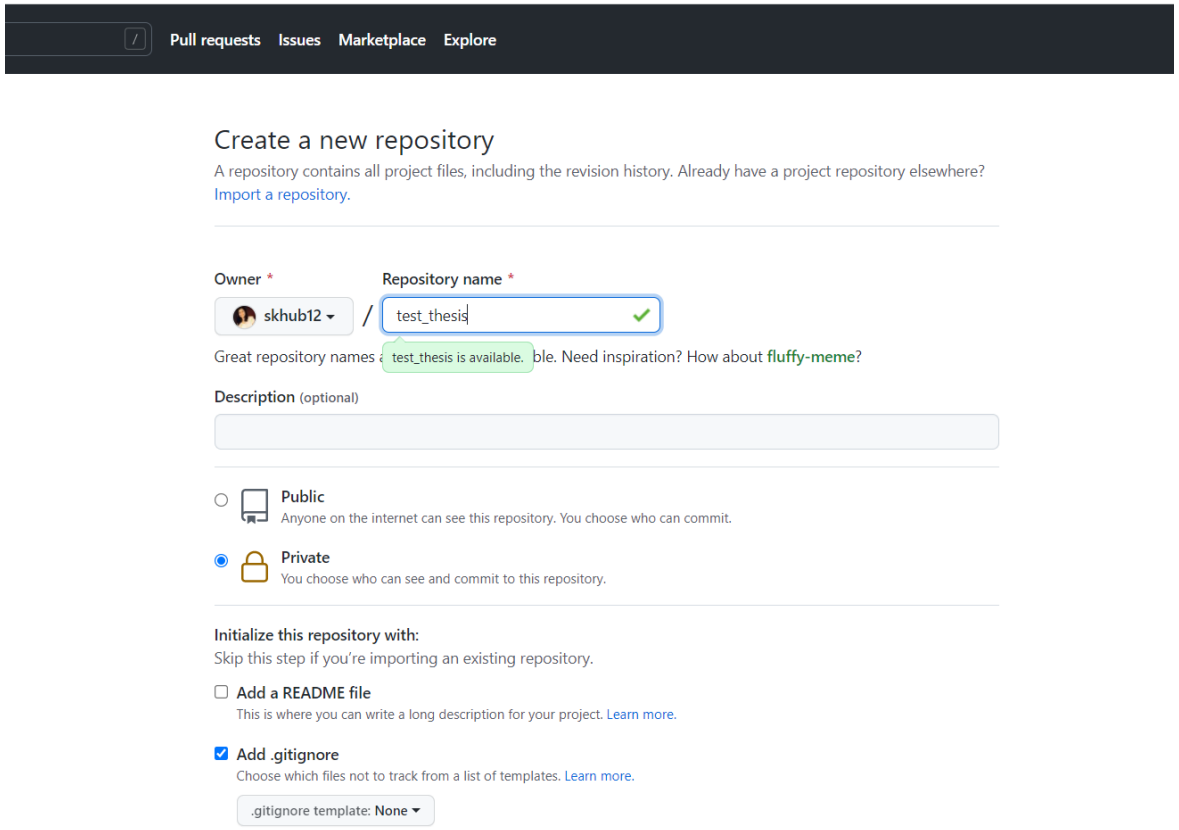
Working with GIT - Locally

- While working with local git repos, it is possible to track inadvertent changes made by yourself.
- Useful to track changes made in data or even a single variable or a word.
    - Setting-up –
    - Let’s create a local testrepo_1 folder.
    - Initialiase git init -
        - git init #it creates an empty git repository or initializes an existing one
        - git add * #adds the changes you have made to the file
        - git commit # It records changes to the repository and commits the changes.
        - git status # It shows the working status – unstracked changes to the files
        - git branch # git-branch is used to create or delete branches to the main repository.
        - git checkout # git-checkout is used to switch branches or restore working tree after the changes are committed.
        - git reset # It resets the current HEAD to the specified (initial) state.

```
sk@DESKTOP-7TPU1IA:~$ mkdir testrepo_1
sk@DESKTOP-7TPU1IA:~$ cd testrepo_1/
sk@DESKTOP-7TPU1IA:~/testrepo_1$ git init
Initialized empty Git repository in /home/sk/testrepo_1/.git/
sk@DESKTOP-7TPU1IA:~/testrepo_1$ touch hello.txt
sk@DESKTOP-7TPU1IA:~/testrepo_1$ git status
On branch master

No commits yet

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

nothing added to commit but untracked files present (use "git add" to track)
sk@DESKTOP-7TPU1IA:~/testrepo_1$ git add hello.txt
sk@DESKTOP-7TPU1IA:~/testrepo_1$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   hello.txt

sk@DESKTOP-7TPU1IA:~/testrepo_1$ git commit
[master (root-commit) 5a444d4] New file added
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 hello.txt
sk@DESKTOP-7TPU1IA:~/testrepo_1$
```

git add README.md"

Working with GIT- remotely with github.com



Advantages

It is possible to work with GitHub's repository hosting service to have a remote repository for a PhD thesis or a paper since it is possible to track word by word changes or manage versions linearly.

It is a good idea to git your thesis as a **private repository**. GitHub private repository provides unlimited storage as long as the file sizes do not exceed 100MB size limit.

- Let's create a repo to manage writing thesis with github.
- Let's create a repository called "test_thesis"

Setting up GIT SSH

Setting you access to github.com from Bash CLI

1. You'll need an account on github.com
    - create a test-repo-on github.com
    - OR, fork an existing repo
2. Next we'll see how to access your personal (private / public) from your BASH client.

- What' we'll need to do is to create a key-pair -- {a private key, and a public key} -- this keypair tightly related to each other.
- For now, we will not enter a passphrase or a filename.
- A public-key will be stored in .pub file – open it with notepad or $cat command and copy the string (a public key).

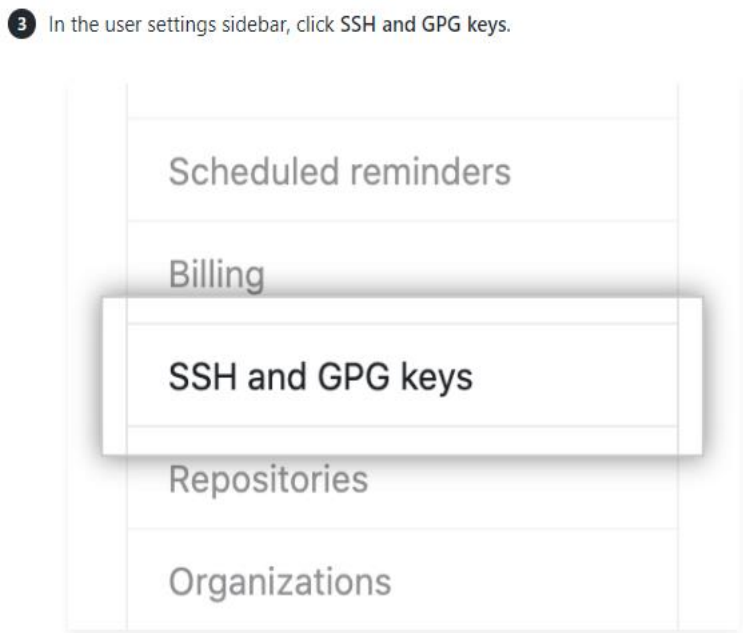
```
bash_prompt ~ $ ssh-keygen -t ed25519 -C "your_email@example.com"
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/your_username/.ssh/id_ed25519):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/your_username/.ssh/id_ed25519
Your public key has been saved in /home/your_username/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:wtFy+kZ6qxlCokBc51UiwG6sPeOveDGZYWXhLVtsSgk your_email@example.com
The key's randomart image is:
+--[ED25519 256]--+
|  .E.+.o..       |
|. ..=o*..        |
| oo o*o=o        |
|.  *..*=         |
|. * =o+ S        |
|.o @   =         |
|. . * o +        |
|  .o . = .       |
| ...o.o..        |
+----[SHA256]-----+
bash_prompt ~ $  # ---> no passphrase is okay to begin with
bash_prompt ~ $  # as long as you can ensure no one can copy your ~/.ssh/ folder
bash_prompt ~ $
```

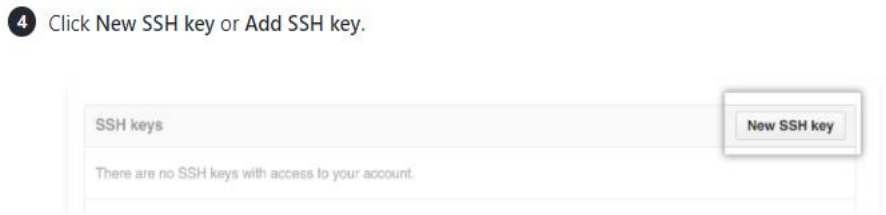
Some notes:

- The private key, you keep private!
- The public key, you can publish in a newspaper if you want!

Adding SSH key to GitHub

1. Cloning your repo, if it is public (on github.com), needs no extra authentication (just like you could always clone a repo from Bash without any special access permissions)
    - However, to be able to make changes to the repo, and ‘git push’ the changes, you’ll need to setup some authentication mechanism.
        - Of course, if this weren’t needed, anybody could make changes to your public github repo!
        - A public-key will be stored in .pub file –add this to your github account – to do this In the upper-right corner of any page, click your profile photo, then click **Settings**. Using settings sidebar, click **New SSH key or Add SSH key**. *



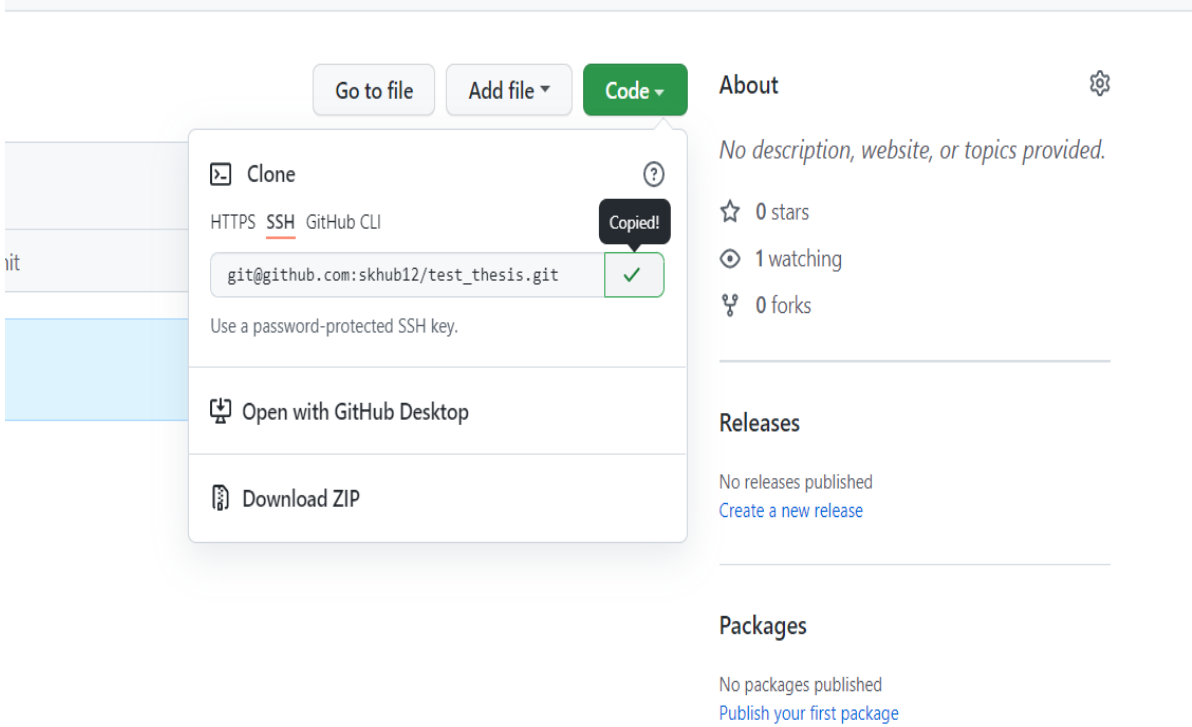


*Reference: https://docs.github.com/en/authentication/connecting-to-github-with-ssh/adding-a-new-ssh-key-to-your-github-account

Cloning a test repo and pushing changes

Clone:

- After adding SSH keys to our account, we created a remote repository on GitHub.com.
- Now, we need to clone it on a local computer to work with it. Get the URL of the remote repository using code button and copy the URL under SSH tab (for tracking changes with SSH).
- Syntax for cloning the repo: $git clone <url to remote repo>




```
sk@DESKTOP-7TPU1IA:~$ git clone git@github.com:skhub12/test_thesis.git
Cloning into 'test_thesis'...
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.
```

Basic workflow for tracking paper/thesis through GitHub

Snapshot of the process:

A typical [workflow for basic snapshotting]() involves following steps -

1.Creating/modifying files of the repository - nano newthesis.tex

2.staging files to git tracking system - git add *

3.committing files to commit history - git commit -m "<my message>“

(in case you write just “git commit” - it will open ‘nano’ in a new window for you to write the commit message.)

```
sk@DESKTOP-7TPU1IA: ~/testre
  GNU nano 4.8                                        /home/sk/testre
New file added


# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
#
# On branch master
#
# Initial commit
#
# Changes to be committed:
#       new file:   hello.txt
#
```

4.pushing the commits to the remote repo - git push

5.go to step 1.

6.Follow this process iteratively to manage version control.

```
sk@DESKTOP-7TPU1IA: ~        sk@DESKTOP-7TPU1IA: ~/test_
sk@DESKTOP-7TPU1IA:~$ cd test_thesis/
sk@DESKTOP-7TPU1IA:~/test_thesis$ nano newthesis.tex
sk@DESKTOP-7TPU1IA:~/test_thesis$ git status
On branch main
Your branch is up to date with 'origin/main'.

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

nothing added to commit but untracked files present (use "git add" to track)
sk@DESKTOP-7TPU1IA:~/test_thesis$ git add *
sk@DESKTOP-7TPU1IA:~/test_thesis$ git commit -m "new version added"
[main a1328b3] new version added
 1 file changed, 1 insertion(+)
 create mode 100644 newthesis.tex
sk@DESKTOP-7TPU1IA:~/test_thesis$ git push
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 290 bytes | 145.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To github.com:skhub12/test_thesis.git
   b10fede..a1328b3  main -> main
```

Detailed explanation of the Basic Snapshotting

- To view hidden files in the folder:
- Use ls –a or ls –al command.
- The hidden folder .git holds all the information about the commit history.
- The hidden file .gitignore file contains instructions to exclude a set of files from version control.
- A preliminary scan of the contents of .gitignore shows what kind of files are omitted.
- Let's say we don't want to track changes to certain files in the folder such as journal_template.pdf.
- We need to modify the .gitignore file to exclude specifically the pdf file, change the .gitignore using nano editor as follows:

```
sk@DESKTOP-7TPU1IA: ~        sk@DESKTOP-7TPU1IA: ~/test_
  GNU nano 4.8                                          .gitignore
#some pdf files to untrack
journal_template.pdf

# Byte-compiled / optimized / DLL files
__pycache__/
*.py[cod]
*$py.class

# C extensions
*.so
```

```
sk@DESKTOP-7TPU1IA:~/test_thesis$ ls -a
.  ..  .git  .gitignore  journal_template.pdf  newthesis.tex
sk@DESKTOP-7TPU1IA:~/test_thesis$ git status
On branch main
Your branch is up to date with 'origin/main'.

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

nothing added to commit but untracked files present (use "git add" to track)
sk@DESKTOP-7TPU1IA:~/test_thesis$ nano .gitignore
sk@DESKTOP-7TPU1IA:~/test_thesis$ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   .gitignore

no changes added to commit (use "git add" and/or "git commit -a")
sk@DESKTOP-7TPU1IA:~/test_thesis$ git add *
The following paths are ignored by one of your .gitignore files:
journal_template.pdf
Use -f if you really want to add them.
```

Now we can see that tracking changes to the journal_template.pdf file has indeed been been ignored by the git.

Advantages over GitHub desktop: to enable the word-diff option in GitHub to see more granular changes to a line.

- While it is easy to create and manage repositories using GitHub desktop, it is not possible to track granular level changes.
- Let's say you have a data file and you accidentally deleted/changed some part or variable name in the file.
- We can use git diff --word-diff to track more granular changes that git diff.
- git diff will highlight the previous version in red and the modifications made in green color.

```
sk@DESKTOP-7TPU1IA: ~/test_

sk@DESKTOP-7TPU1IA:~/test_thesis$ ls
journal_template.pdf  newthesis.tex  testdata.csv
sk@DESKTOP-7TPU1IA:~/test_thesis$ git diff
diff --git a/testdata.csv b/testdata.csv
index 96b7b27..0e98a62 100644
--- a/testdata.csv
+++ b/testdata.csv
@@ -1,5 +1,5 @@
 5,Albania,ALB,AL,2002,31.74,1.22,4.83,20.93,,,Consumption,Consumption,Per capita,Per capita,Household,Person,All,All,All,All,Europe,Southern Europe,Europe
and Central Asia,Non-EU,Non-OECD,Upper middle income,2452,2023,US$2011PPP,Year,,,,6895,3126183,New 2021,Average,13,World Bank,PovcalNet,,,http://iresearch.w
orldbank.org/PovcalNet/povOnDemand.aspx,Yes
-6,Albania,ALB,AL,2005,30.6,1.14,4.66,21.3,,,Consumption,Consumption,Per capita,Per capita,Household,Person,All,All,All,All,Europe,Southern Europe,Europe an
d Central Asia,Non-EU,Non-OECD,Upper middle income,2772,2359,US$2011PPP,Year,,,,8208,3086810,New 2021,Average,13,World Bank,PovcalNet,,,http://iresearch.wor
ldbank.org/PovcalNet/povOnDemand.aspx,Yes
+6,Albanio,ALB,AL,2005,30.6,1.14,4.66,21.3,,,Consumption,Consumption,Per capita,Per capita,Household,Person,All,All,All,All,Europe,Southern Europe,Europe an
d Central Asia,Non-EU,Non-OECD,Upper middle income,2772,2359,US$2011PPP,Year,,,,8208,3086810,New 2021,Average,13,World Bank,PovcalNet,,,http://iresearch.wor
ldbank.org/PovcalNet/povOnDemand.aspx,Yes^M
 7,Albania,ALB,AL,2008,29.98,1.11,4.4,21.94,,,Consumption,Consumption,Per capita,Per capita,Household,Person,All,All,All,All,Europe,Southern Europe,Europe a
nd Central Asia,Non-EU,Non-OECD,Upper middle income,3034,2539,US$2011PPP,Year,,,,10119,3002683,New 2021,Average,13,World Bank,PovcalNet,,,http://iresearch.w
orldbank.org/PovcalNet/povOnDemand.aspx,Yes
 8,Albania,ALB,AL,2012,28.96,1.04,4.27,22.02,,,Consumption,Consumption,Per capita,Per capita,Household,Person,All,All,All,All,Europe,Southern Europe,Europe
and Central Asia,Non-EU,Non-OECD,Upper middle income,2877,2491,US$2011PPP,Year,,,,11462,2914091,New 2021,Average,13,World Bank,PovcalNet,,,http://iresearch.
worldbank.org/PovcalNet/povOnDemand.aspx,Yes
 9,Albania,ALB,AL,2014,34.6,1.38,5.93,18.56,,,Consumption,Consumption,Per capita,Per capita,Household,Person,All,All,All,All,Europe,Southern Europe,Europe a
nd Central Asia,Non-EU,Non-OECD,Upper middle income,3067,2508,US$2011PPP,Year,,,,11828,2896307,New 2021,Average,13,World Bank,PovcalNet,,,http://iresearch.w
orldbank.org/PovcalNet/povOnDemand.aspx,Yes
@@ -34,7 +34,7 @@
 38,Argentina,ARG,AR,1961,47.7,2.66,10.5,14.8,,30.8,Income (net/gross),"Income, net/gross",,,,Person,Urban,Nonagricultural sector,All,All,Americas,South Ame
rica,Latin America and the Caribbean,Non-EU,Non-OECD,Upper middle income,,,,,,,,12889,20817270,,Low,7,World Bank,Jain 1975,Synthetic estimates (UN-ECLA 1970
),,,No
 39,Argentina,ARG,AR,1961,43.8,2.23,7.38,16.6,,32,Income (net/gross),"Income, net/gross",No adjustment,No adjustment,Household,Household,All,All,All,All,Ame
ricas,South America,Latin America and the Caribbean,Non-EU,Non-OECD,Upper middle inc:...skipping...
diff --git a/testdata.csv b/testdata.csv
index 96b7b27..0e98a62 100644
--- a/testdata.csv
+++ b/testdata.csv
@@ -1,5 +1,5 @@
 5,Albania,ALB,AL,2002,31.74,1.22,4.83,20.93,,,Consumption,Consumption,Per capita,Per capita,Household,Person,All,All,All,All,Europe,Southern Europe,Europe
and Central Asia,Non-EU,Non-OECD,Upper middle income,2452,2023,US$2011PPP,Year,,,,6895,3126183,New 2021,Average,13,World Bank,PovcalNet,,,http://iresearch.w
orldbank.org/PovcalNet/povOnDemand.aspx,Yes
```