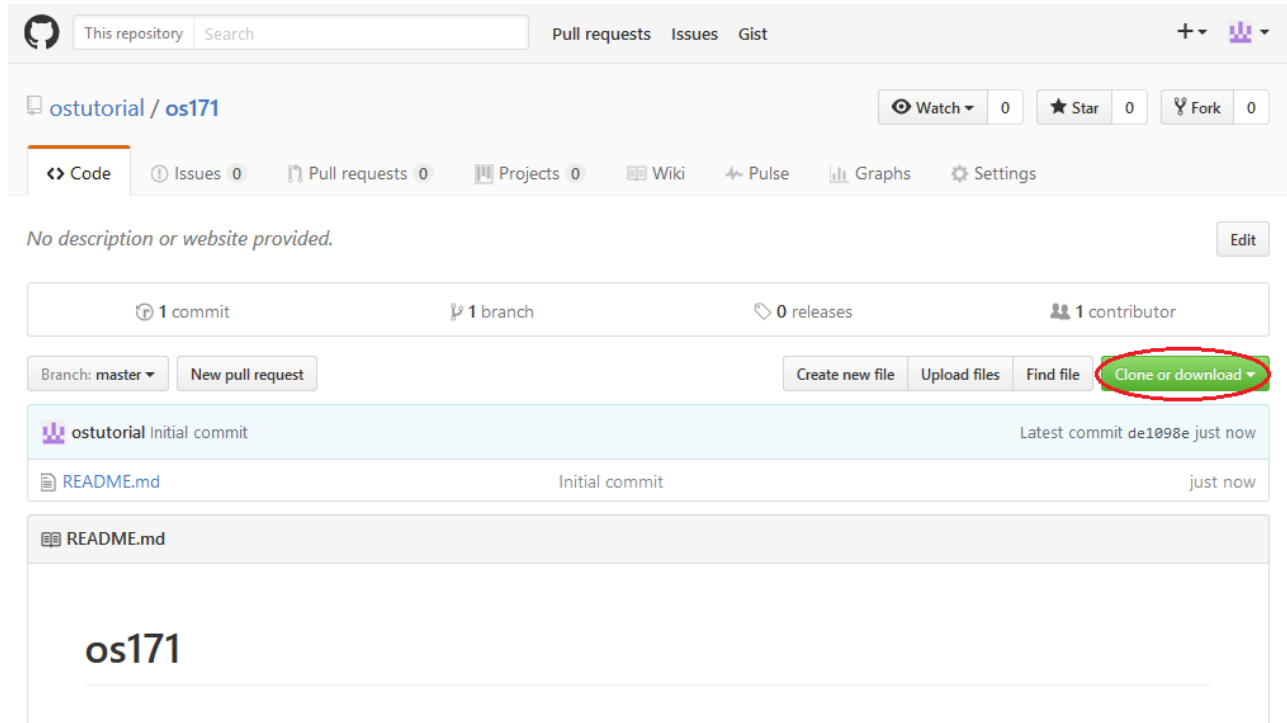


## Week 01

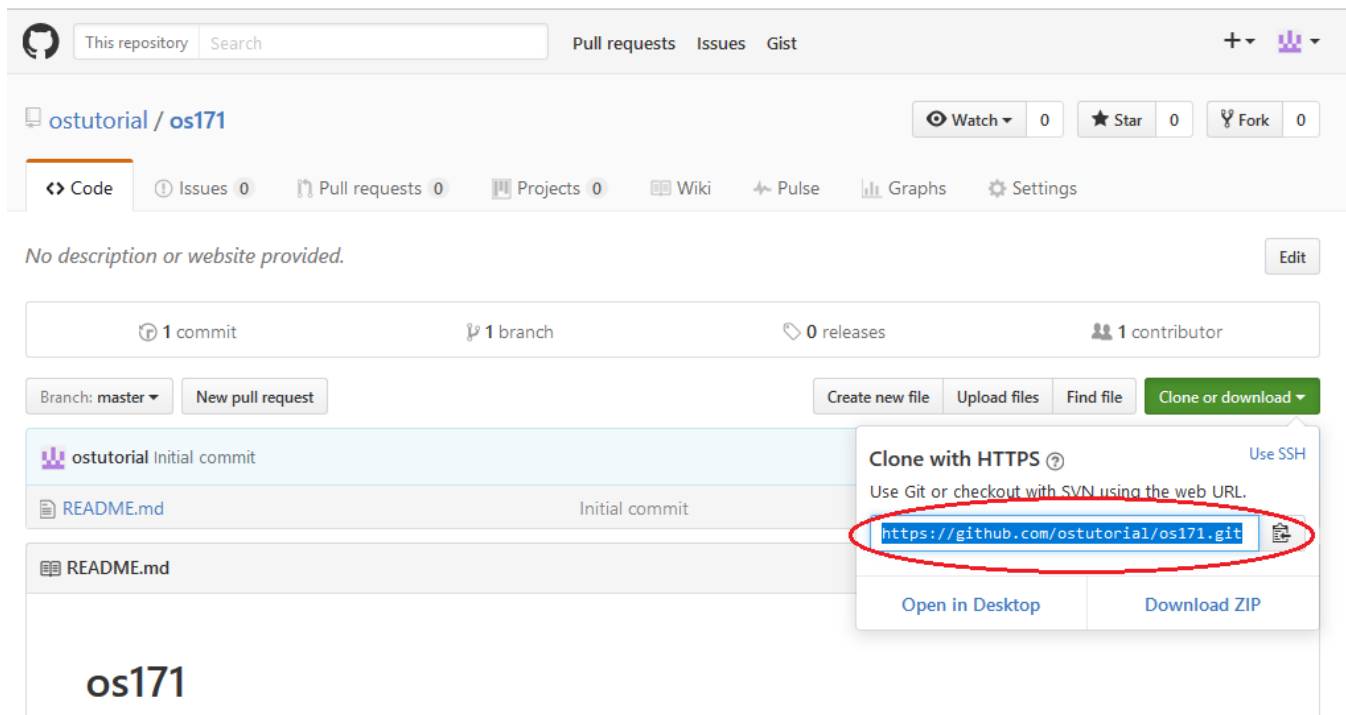
**Deadline: Saturday, 18<sup>th</sup> February 2017 at 15:00**

### Clone Git Repository

1. Open your github repository in browser, then click on **Clone or Download** button.



2. Copy your repository link that appeared



- Then you can clone your repository by using command “*git clone your\_repository\_link*” on your terminal

```
intan.dwi41@badak:~$ git clone https://github.com/ostutorial/os171.git
Cloning into 'os171'...
remote: Counting objects: 3, done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
Checking connectivity... done.
intan.dwi41@badak:~$
```

- Congratulation, your repository has been succesfully cloned. Check the cloned repository by using command “*ls -al*”, make sure your cloned repository is on the list.

```
intan.dwi41@badak:~$ ls -al
total 172
drwx--x--x 20 intan.dwi41 StafCS  4096 Jan 18 16:08 .
drwxr-xr-x  7 root          root   4096 Sep  3 09:12 ..
-rw-r--r--  1 intan.dwi41 StafCS  1609 Sep  1 17:36 01-09-16
-rw-r--r--  1 intan.dwi41 StafCS  1950 Sep  4 18:31 04-09-16
-rw-----  1 intan.dwi41 StafCS 27950 Jan 17 19:05 .bash history
drwxr-xr-x  3 intan.dwi41 StafCS  4096 Jan 18 16:08 os171
```

## Intro to Git

**Git** is a version control system that is used for software development and other version control tasks. As a distributed revision control system it is aimed at speed, data integrity, and support for distributed, non-linear workflows (Wikipedia)

For more information about how **Git** works, take a look on <<https://git-scm.com/book/en/v1>>

5. Change your current directory to `os171` using command `"cd <path>"`

```
$ cd os171
```

6. Create 11 new folders named `"weekXX"` where `XX` is number from 00-10, 4 folders named `"log"`, `"key"`, `"extra"`, `"sandbox"`. To create new folder, use command `"mkdir <folder_name1> <folder_name2> ... "`

```
$ mkdir week00 week01 week02 ... log key extra sandbox
```

7. verify if the folders created using command `"ls -al"`. This command does listing files and folders in current directory

```
$ ls -al
```

8. change your current directory to `week00`

```
$ cd week00
```

9. create an empty file with the name `"dummy"`. To create a file from nothingness, use command `"touch <filename>"`

```
$ touch dummy
```

10. go back to previous directory

```
$ cd ..
```

11. do step 8 and 9 on each of the following directory: `week01`, `week02`, `week03`, `week04`, `week05`, `week06`, `week07`, `week08`, `week09`, `week10`, `log`, `key`, `extra`

12. push all newly created folders to github and then check if it pushed successfully. You can see how to do that on `03_TutorialPushandPullRepository.pdf` file

13. change your current directory to `sandbox`

```
$ cd sandbox
```

\*linux is case-sensitive. "sandbox" is different with "Sandbox" or "SandBox".

Do you not believe it? Try executing command "ls" and "LS" :)

**\*\*please make sure your folder/file name is as expected, because you will get **penalty for each wrong name.****

14. create an empty file (the name is up to you).

Example:

```
$ touch rotibakarenak
```

15. push that file to github and check if it pushed successfully.

16. change current directory to week00

```
$ cd ../week00
```

\*".." means "go to parent directory". So in this case, we go from our current directory (sandbox) to it's parent directory (os171) and then go to week00

17. Remove the "dummy" file that you have made before at this file by using this command

"rm -f <filename>" like this:

```
$ rm -f dummy
```

18. Write short description of task from week 0 (last week) using your favorite text editor (e.g vi, vim, nano) and named it "report.txt"

*short guide using text editor (vi and nano)*

- \$ vi report.txt # for geek person
  - By executing the above command you will enter vi. By default the first time you enter vi you will be in command mode. This mode is used to access commands in vi, not to write, so that when you try to type (try typing letter g), it's not printed in the text area because vi translate your typing into commands that can be accessed by typing letter g. This command mode can be accessed anytime by pushing **Esc** (Escape) button on the left side of your keyboard. Now go ahead and press **Esc**.

- To be able to write we will have to enter insert mode. There are few ways to get into insert mode. For now let's try typing letter **i**. And you can see on the bottom of `vi` that you are in insert mode

```
-- INSERT --                                0,1      All
```

- Now everything that you type will be written in the text area. Write your name with format `"#your_name"`
  - Your writing will be input as a text. Now get back into command mode (by clicking **Esc** button), then save and quit from your work by typing `:wq` and click **Enter**.
- `$ nano report.txt`                      # for normal person
    - nano is super-friendly text editor. I bet you can use it at the first glance.
    - You just need to know some shortcut
      - `ctrl + o`              save
      - `ctrl + k`              cut (one line)
      - `ctrl + u`              paste
      - `ctrl + x`              exit

19. push that file to github and check if it pushed successfully.

## Intro to CLI

20. change current directory to `os171`

21. prepare a file to record your work progress named `"lab01.txt"`

```
$ touch lab01.txt
```

22. record the contents of current directory to `lab01.txt` using command

```
"ls -al > lab01.txt".
```

\* The `">"` operator means "redirect the output of command on left (`ls -al`) to the file on the right (`lab01.txt`)". Beside that operator, there is `">>"` operator too. It is similar to the other one but "used when you want the output appended to the bottom of the file on the right file's content".

23. move lab01.txt to folder named week01. Use command "mv <source>

<destination>" to do that

```
$ mv lab01.txt week01
```

24. change your current directory to week01

```
$ cd week01
```

25. verify your current directory using command "pwd" and check is there is lab01.txt

```
$ pwd
```

```
/home/XXX/os171/week01
```

```
$ ls -al
```

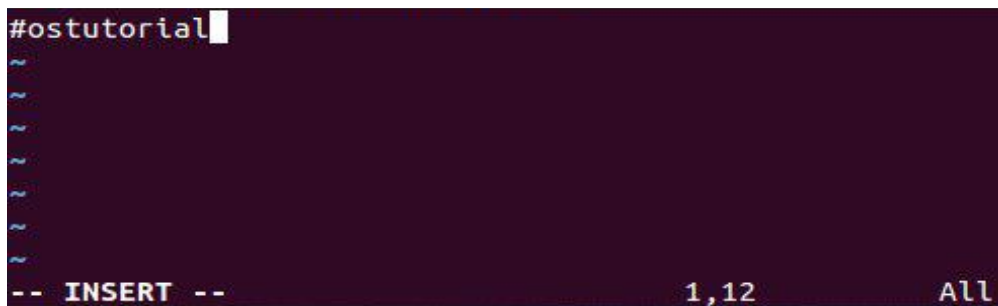
26. record your current directory to lab01.txt using command "pwd >> lab01.txt". Note it's using ">>" operator. What the different? The ">>" operator means "redirect the output of command in left (pwd) and **append it** to file in right (lab01.txt)"

27. create file "myExpectation.txt" with content:

```
#your_github_account
```

Example:

```
$ vi myExpectation.txt
```



28. push that file to github and check if it pushed successfully.

29. edit myExpectation.txt from browser. Add the content with your expectation in this class (OS)



30. commit the changes, and don't forget to give meaningful commit message.
31. update your local repository on badak server. To do so, open your badak account terminal again and doing the **pull** repository tutorial on 03\_TutorialPushandPullRepository.pdf file
32. change your current directory to week01
33. Remove the “dummy” file that you have made before at this file by using this command  
“rm -f <filename>” like this:  

```
$ rm -f dummy
```
34. Write short description of task from week 1 (this week) using your favorite text editor (e.g vi, vim, nano) and named it “report.txt” (just like for week 0)
35. push that file to github and check if it pushed successfully

## Intro to scripting

36. well, it's just a demo to show you how to make a simple script in linux
37. change current directory to week01
38. create an empty file (name is up to you, WARNING: do not use symbols like (‘), (“), (’), (;), (>), (<), (&), (|))  

```
$ vi what-time-script.sh # in linux, script usually marked by
```

```
suffix .sh
```

39. paste the following code and save it

```
#!/bin/bash
DATE=`date +%d\ %b\ %Y\ at\ %H:%M:%S`
echo "Today is $DATE"
exit 0
```

40. change the permission so we can execute it

```
$ chmod +x what-time-script.sh
```

41. execute it

```
$ ./what-time-script.sh >> lab01.txt
Today is 08 Feb 2017 at 08:32:09
```

42. you can see the output of the script at script-output.txt by using this command:

```
$ cat lab01.txt
```

the output will be at the last line, similar to this (depends on the time set on your computer when you execute the script)

```
Today is 08 Feb 2017 at 08:32:09
```

43. now you already tried scripting. Commit and push the changes.

44. congratulation, you just complete the week 1 task. See you again on week 2 task.

***P.S. : Uncle Google is your friend, jusk ask him if you have question or, ask on this forum [link](#)***



Dont forget to check your files/folders. After this lab, your current `os171` folder should looks like:

```
os171
  key
    dummy
  log
    *log.txt
  sandbox
    <some_random_name>
  week00
    report.txt
  week01
    lab01.txt
    report.txt
    myExpectation.txt
    what-time-script.sh
  week02
    dummy
  week03
    dummy
  week04
    dummy
  week05
    dummy
  week06
    dummy
  week07
    dummy
  week08
```

dummy

week09

dummy

week10

dummy

xtra

dummy

PS: \*Follow the instruction on 04\_LOG-additionaltaskforeveryweek.pdf