Version Control Systems
(Mercurial)

by Nikola Ilkov Bozhinov
11A grade in TUES
Sofia,Bulgaria

1.What is a Version Control System?

Version Control Systems (or as they are more commonly known **VCS systems**) are mainly used in team projects when it's best if more than one person works on the assignment. For this purpose the source code is uploaded to an internet storage known as „repository“ and each member of the team can „pull“ it from there and make changes. When the developer is done he can „push“ the code back to the repository and the team manager can combine all the new additions with the old source code in order to get the new product. The whole system which is responsible for the operations we make on the project in the repository is called VCS system.

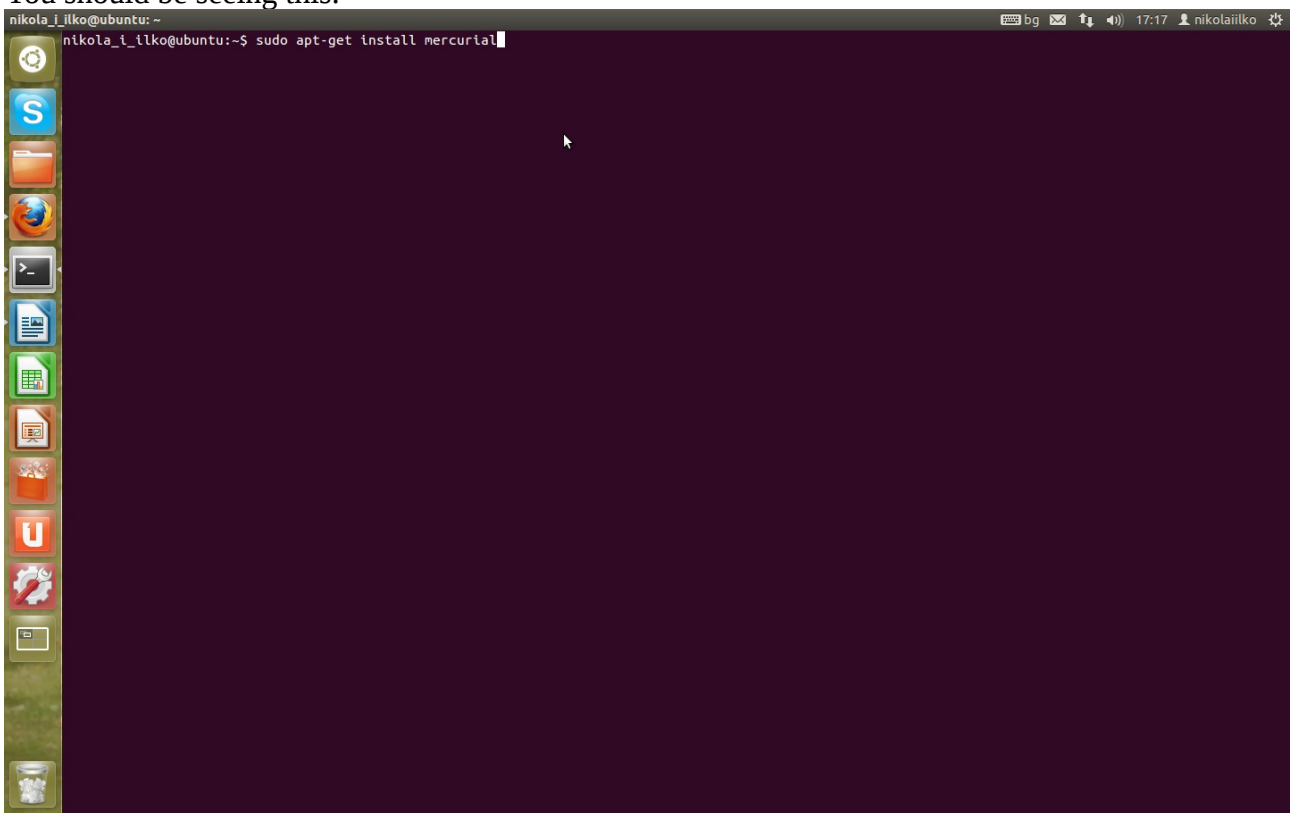
In this tutorial I will be using the VCS Mercurial.

2.How to install Mercurial on Ubuntu?

Step 1. Open your terminal (Bash)

Step 2. Copy this line: `sudo apt-get install mercurial`
and right click beside the empty line in your terminal (or you can just rewrite it).

You should be seeing this.



Step 3 Press Enter and when you are asked for your password type it (this will work only if you have the rights to install programs).

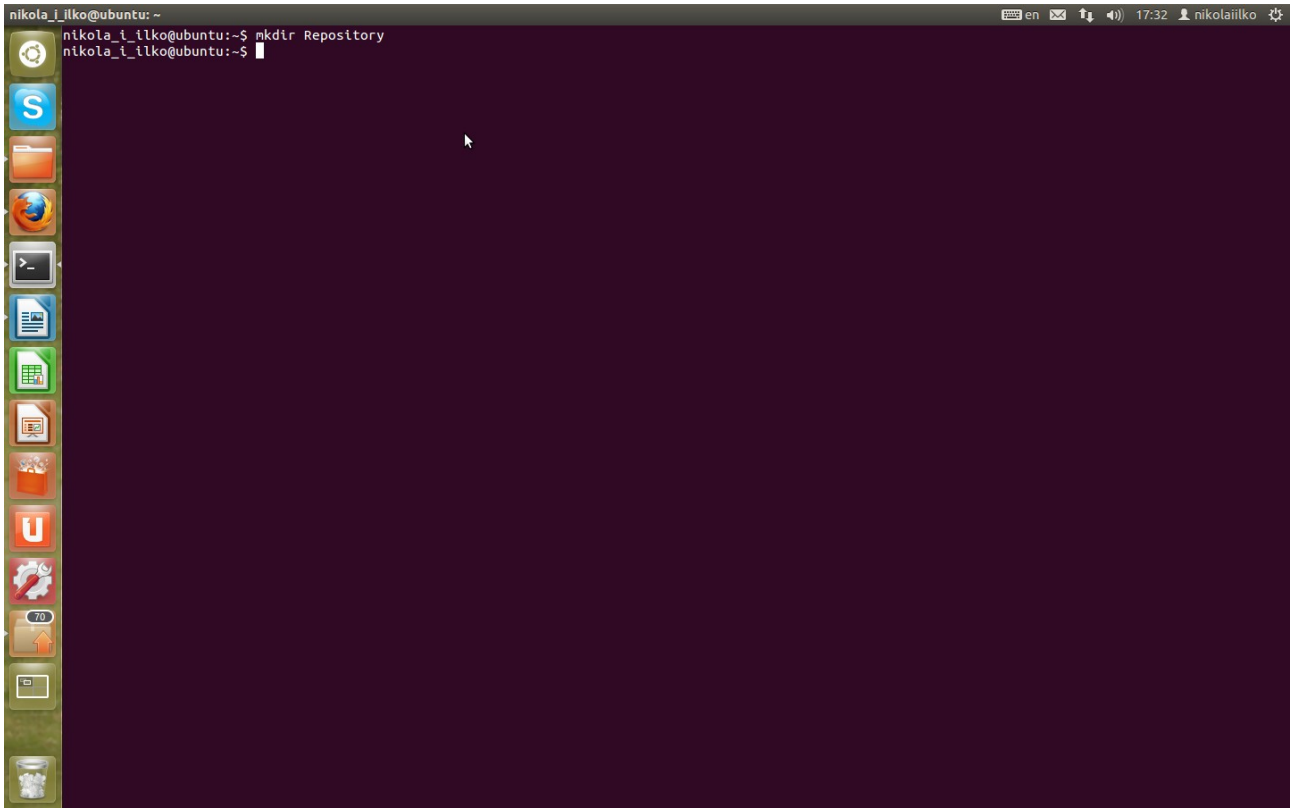
Now if everything is ok you should have Mercurial installed on your Ubuntu.

3.Initialise a repository and add files to it.

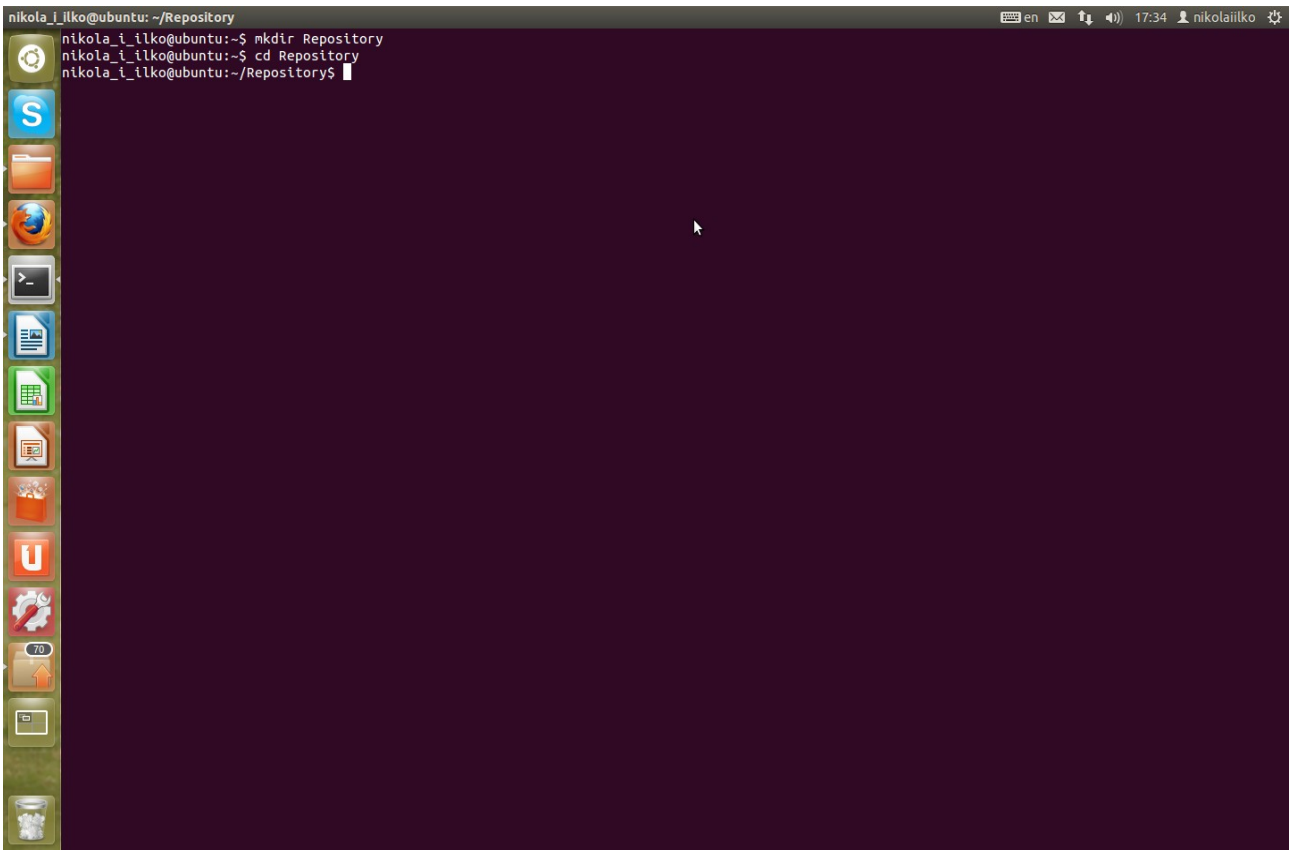
But of what good will Ubuntu be to us if we don't make a new project? Let us make one together and upload it to the internet (You must have an online repository, which can be easily made via registration in www.bitbucket.org)

Step 1. Make a new folder with the command mkdir <write_here_the_name_of_the_folder> and this will create it.

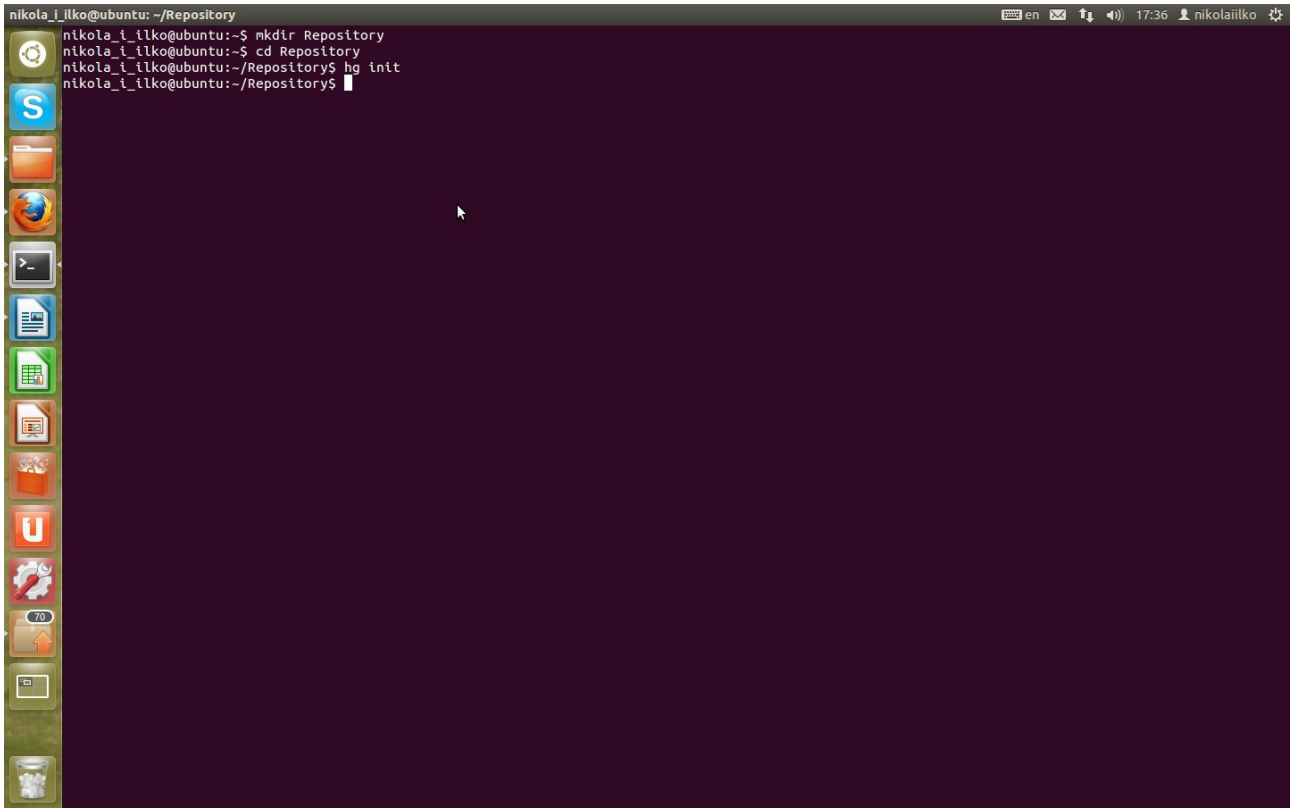
Example:



Step 2. Change your directory by using the command:
cd <name_of_directory_you_wish_to_go_in>

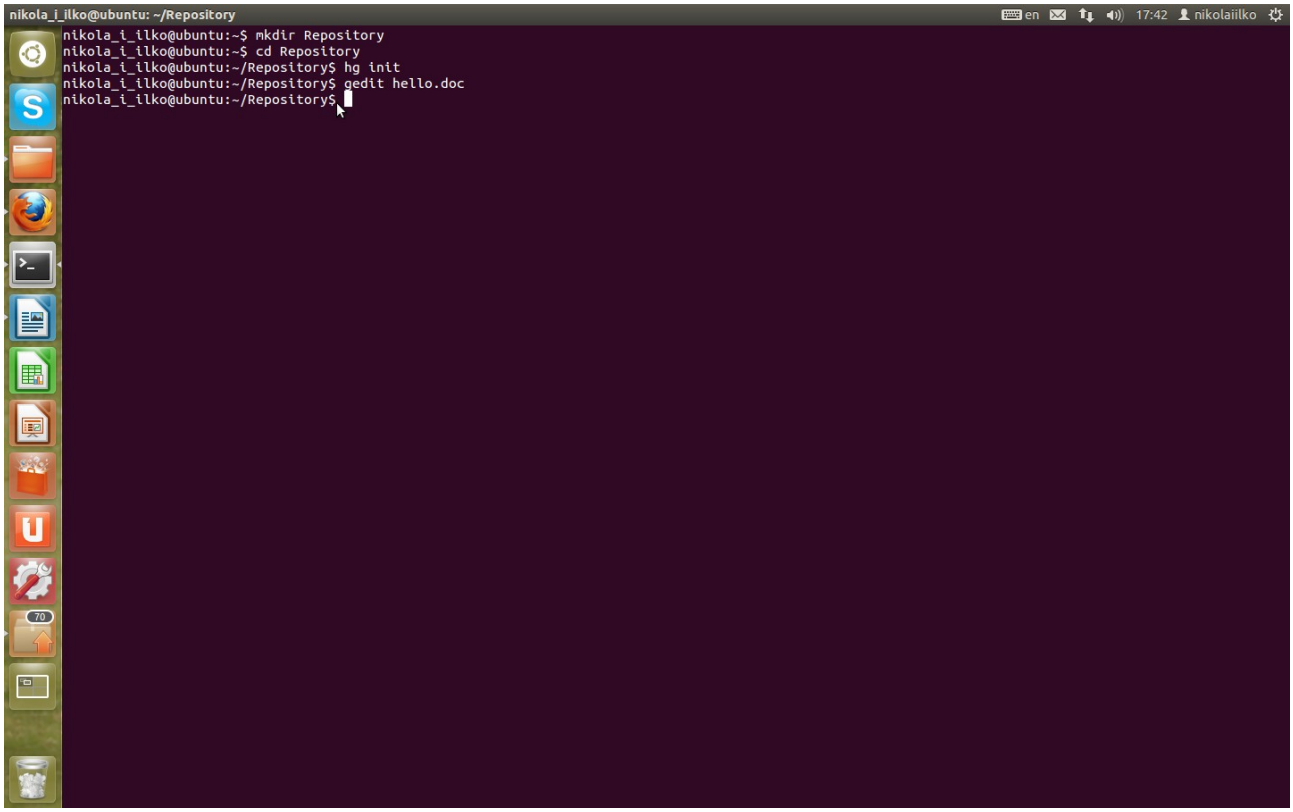


Step 3. To initialise(create) a new project type the command:
hg init



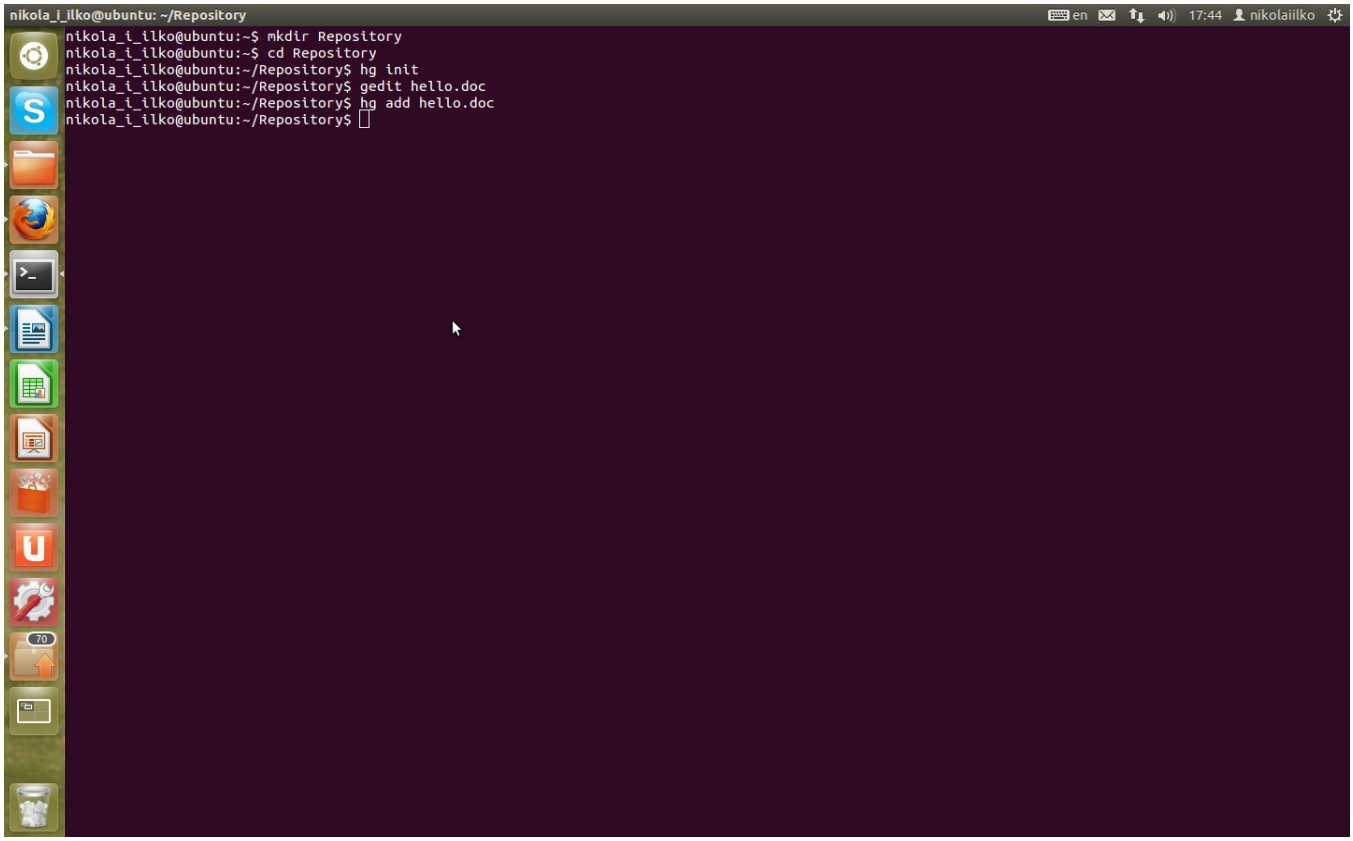
Congradulations!!!You made your first repository.

Step 4. Create a file inside your current directory. This can be done in several ways.
-type: gedit <filename> in the terminal and make a new file via gedit
-type: nano <filename> in the terminal and make a new file via nano
-go to your folder mannually and create a new file



Write whatever you want inside it.For mine I will use gedit and name it hello.doc
I wrote „Hello World!!!“ inside and closed it after saving.

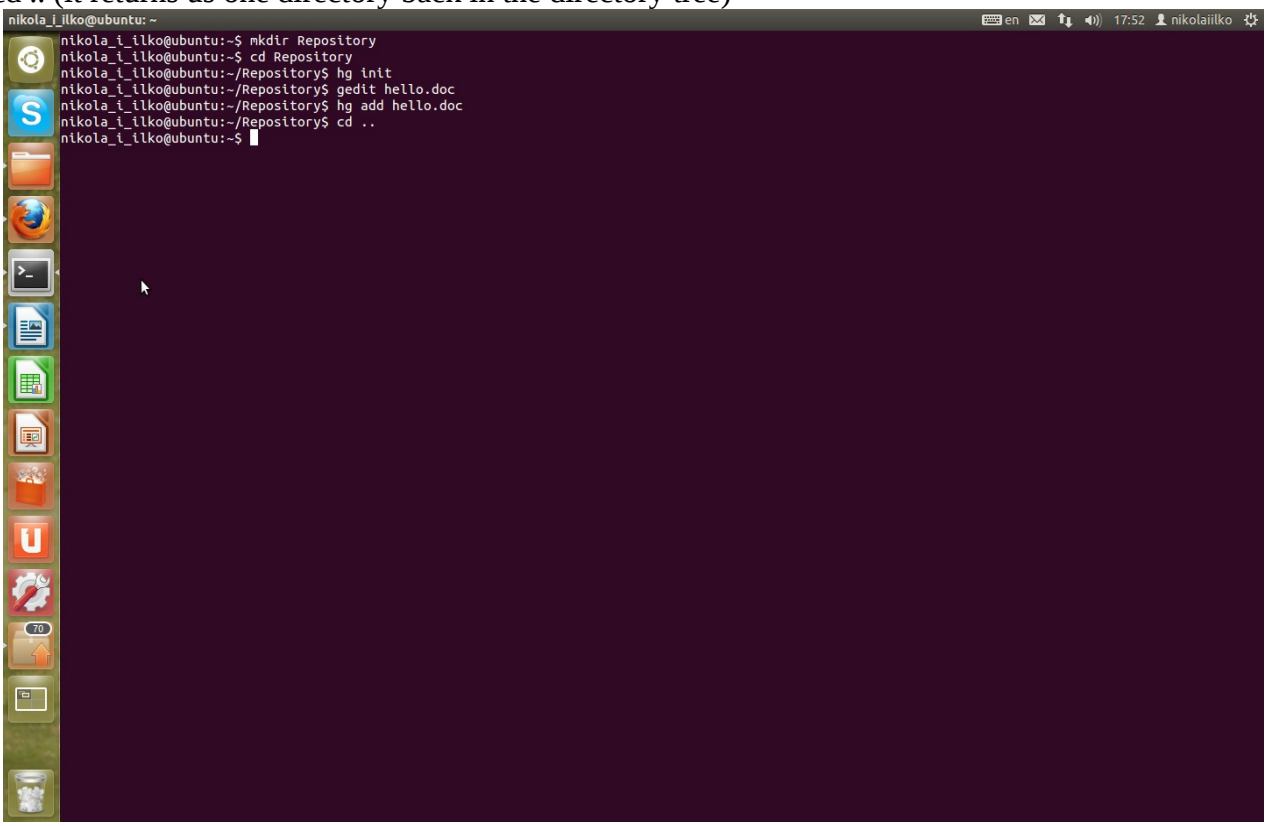
Step 5. Lets add this file to the repository so it can be tracked. Use the command:
hg add <filename>



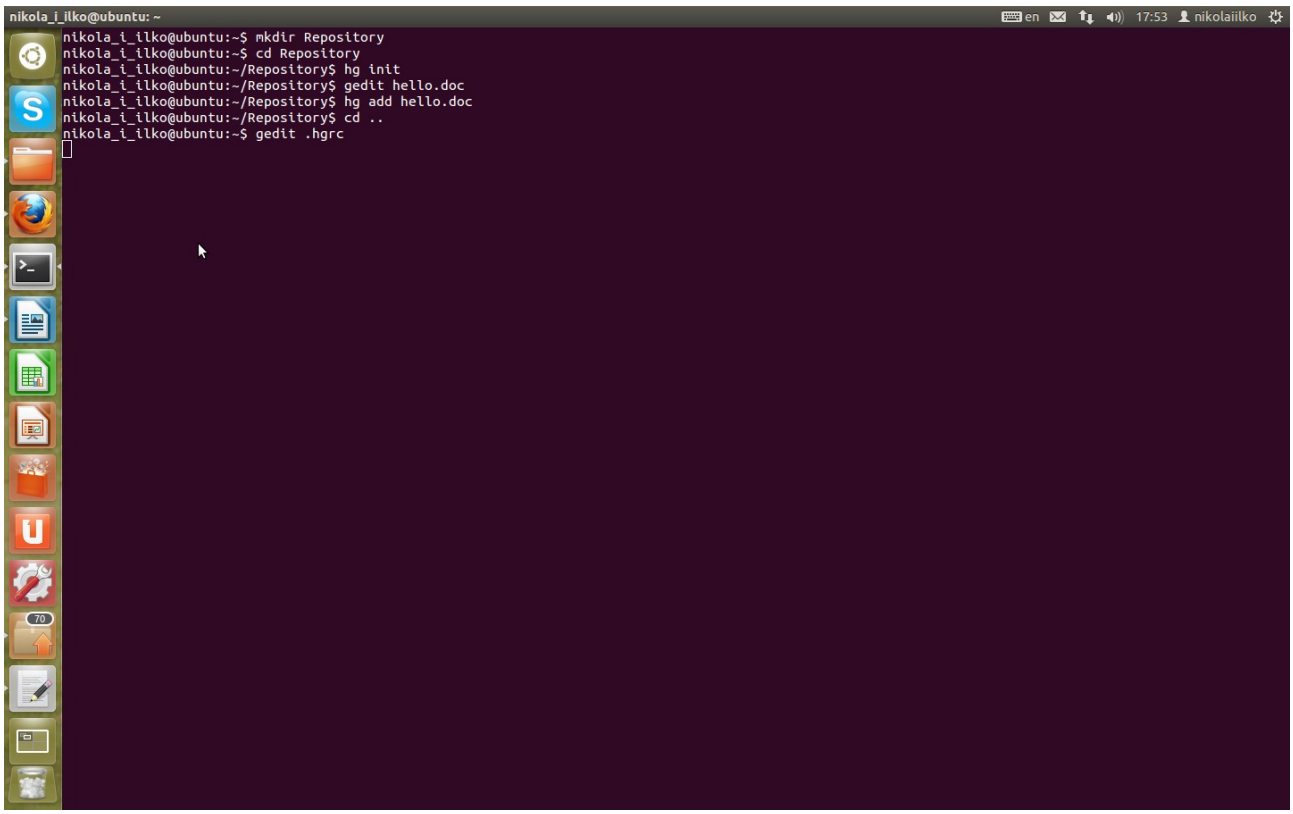
It can be used for folders too and will add all the files inside of it.

Step 6. Everything is good we have added a file for tracking but in fact mercurial still doesn't really know we have made changes to our repository. In order to yell this to the VCS we need to use the command :
hg commit -m „This is the message we can later read about the current changes to the repository“

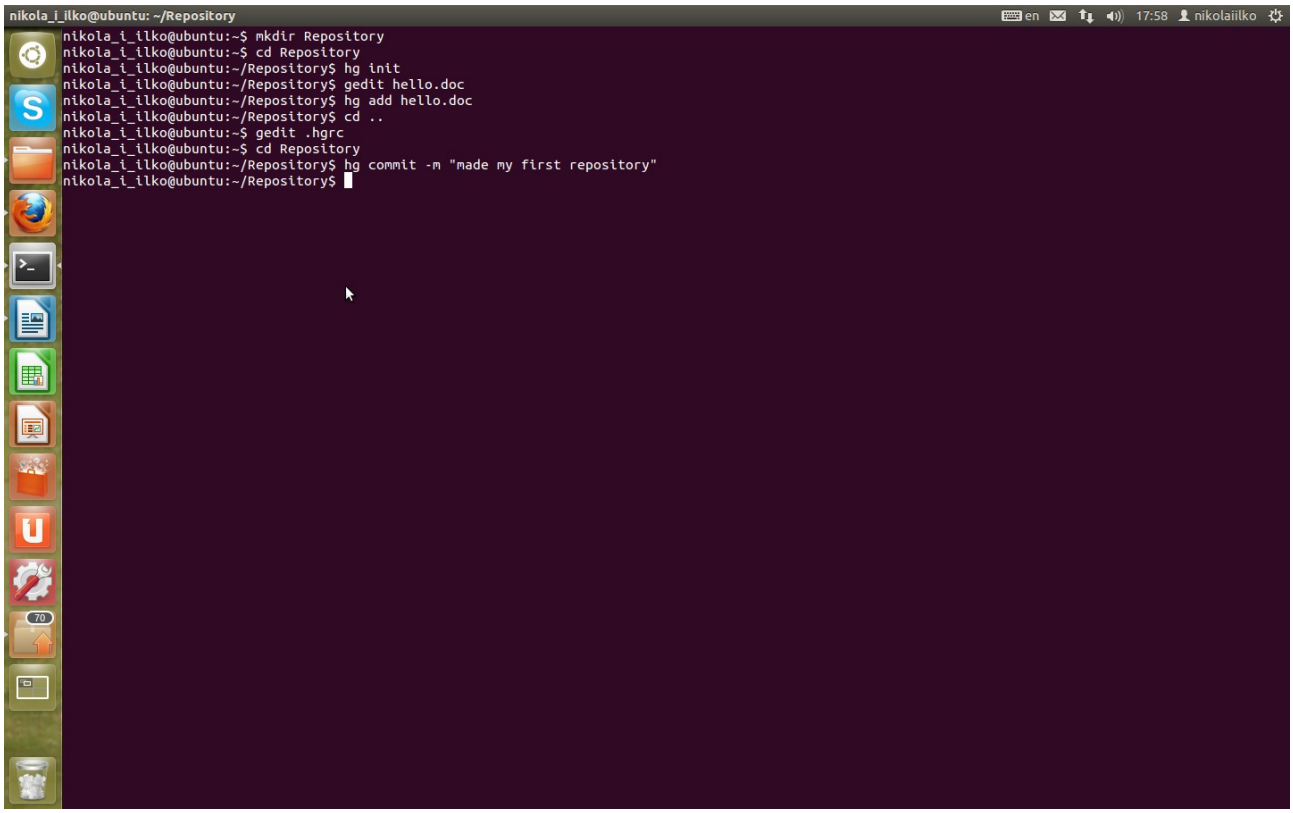
But first we need to do something else-identify ourselves.
this can either done by returning back to the main Directory with the command:
cd .. (it returns us one directory back in the directory tree)



Now write:
gedit .hgrc



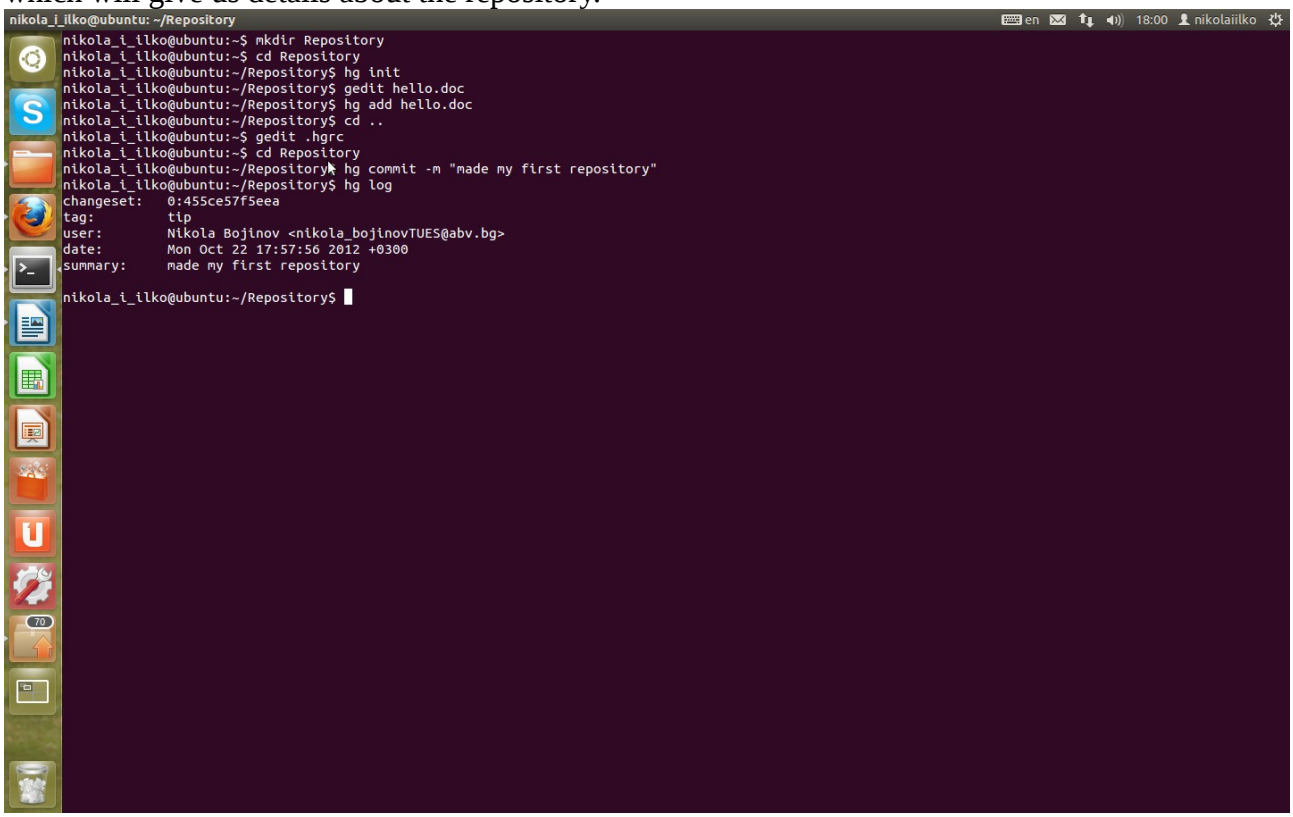
and write this:
[ui]
username= your_name <your_email>
Example:
[ui]
username = Nikola Bojinov <nikola_bojinovTUES@abv.bg> (use the same e-mail as the one you used to create the bitbucket account)
and save it.



Now that we have identified ourselves go back to the repository directory with the cd command and use the commit command described earlier.

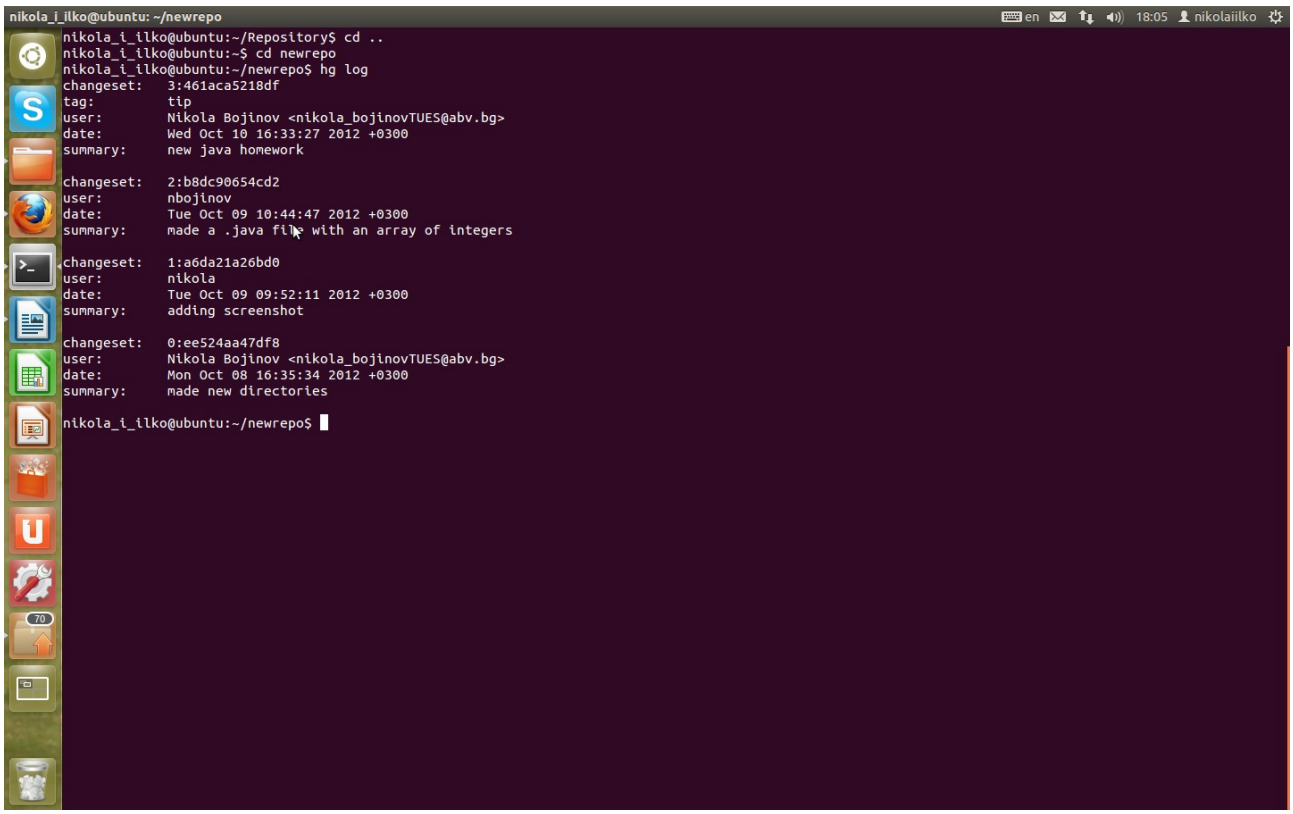
If everything is ok you shouldn't have received an error... If not please go back and check if you have done everything properly.

To check out new repository we will use the command:
hg log
which will give us details about the repository.



Briefly:
changeset contains 2 numbers the first shows the order of our update compared to the ones before it. Because we haven't had any changes it is now 0 but if we had another it would have grown with one.

For example this is one of my other repositories:



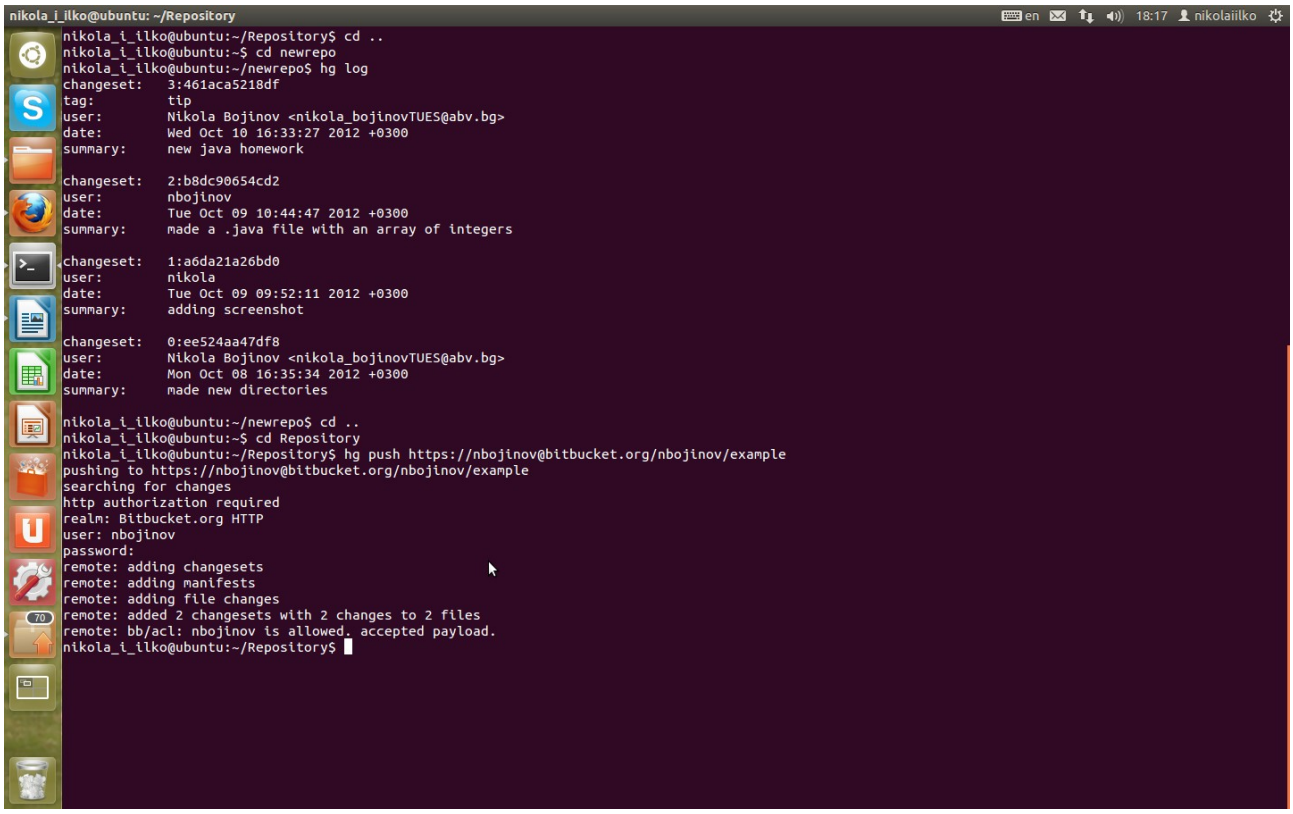
As you can see the newest one has the number 3 and all others with one less...

The second part of the number is a unique sequence which is used for accessing each changeset but we won't talk about this now.

The line
tag tag
shows us the changeset we are using now.
I think all other are pretty obvious.

4.Upload the content to an online repository.

We can now upload our new content to the repository you have in bitbucket. Mine is at
`https://nbojinov@bitbucket.org/nbojinov/example`
and the command is:
hg push (while you are in the folder of the repository)
so the whole line should look like this
`hg push https://nbojinov@bitbucket.org/nbojinov/example`



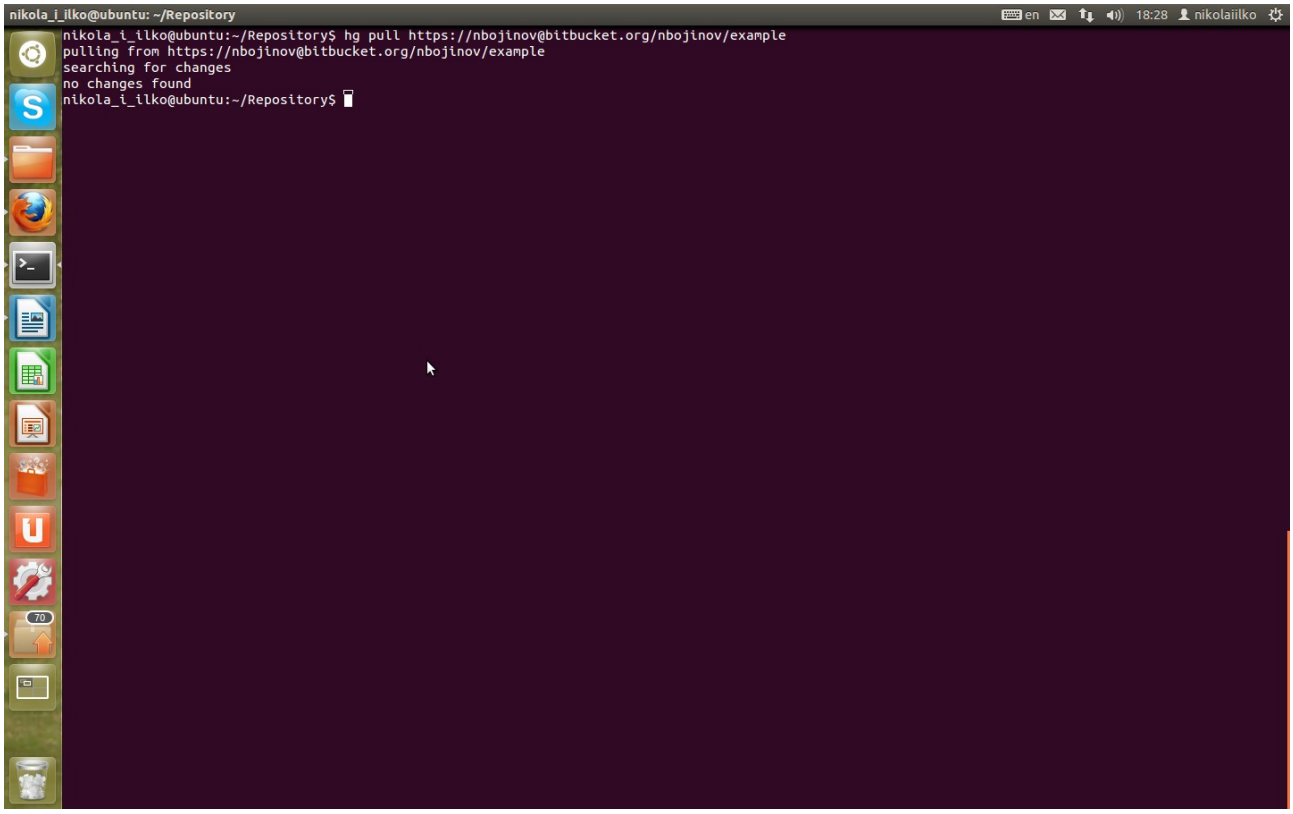
When you are asked for password write the one you used for your bitbucker account (don't worry if you can't see the keys you have pressed it's supposed to be like that :))

5.Pull and Update.

When working with many people on the same project there will definetly be changes to the project while you aren't working on it.

In order to get these changes use the commands:
hg pull <site>(to pull the changes from the repository)
hg update (to get these new changes working)

if there are no new changes you will get this message:



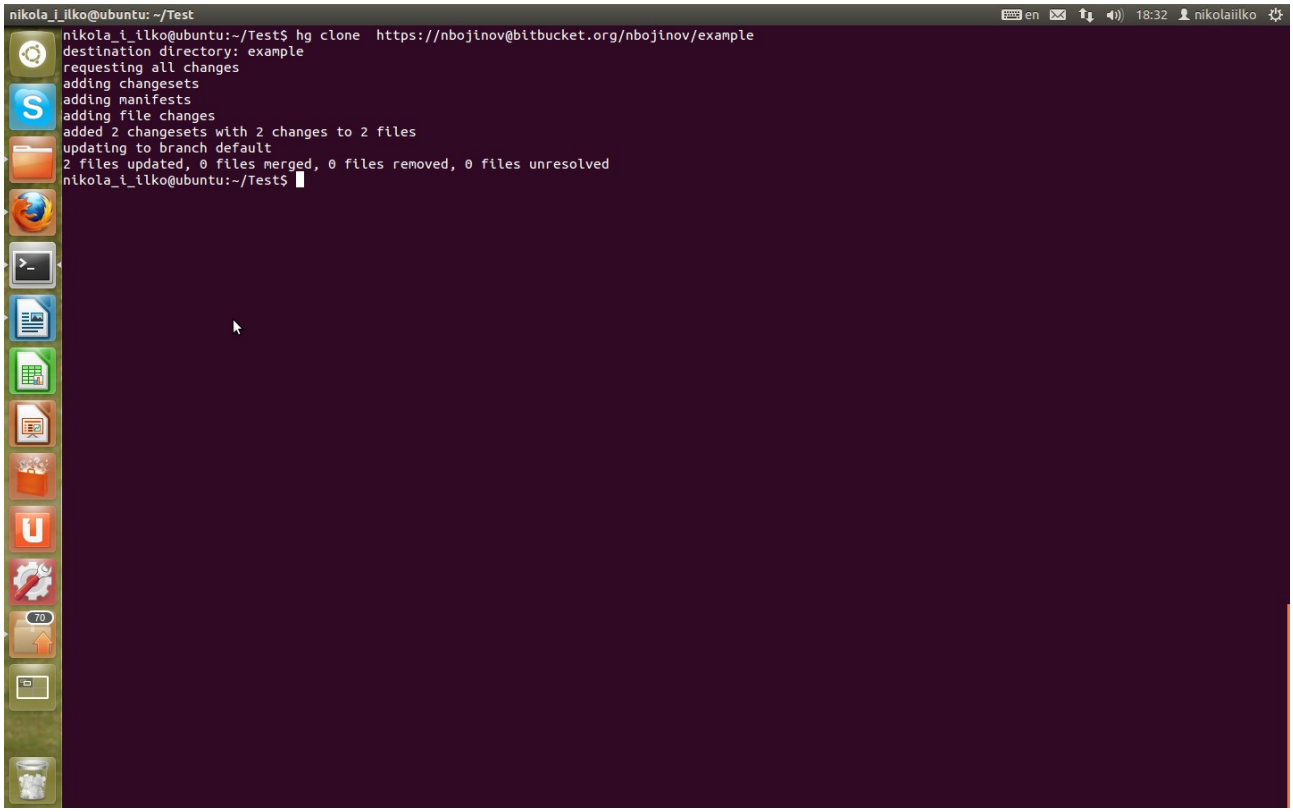
6.Clone.

The clone command is used in a combination with HG INIT when you don't have a copy of an repository and want to download it.

Step 1.Initialise a new repository in a new folder as shown above.

Step 2. Use the command:
hg clone <site>

example:



If the repository isn't private you should be able to download it easily nad have a working copy of it.

I hope that this tutorial will be useful to you. Feel free to copy information and use it as long as you mention the source.