Laboratory Work No. 1

Luchianenco Filip FAF-121

Command Line Interface; CLI Editors; Setting Server Environment; Version Control Systems

Mandatory Tasks:

- Connect to a remote server via SSH
- Initialize a repository on server
- Create a file in repository folder, write in your name, save it and commit it

Tasks With Points:

- Connect to server using public key (1 pt)
- Create 2 more branches with at least one unique committed file per branch (1 pt)
- Set a branch to track a remote origin on which you are able to push (ex. github, bitbucket or a custom server) (1 pt)
- Reset a branch to previous commit, reset a branch to some specific commit (1 pt)
- Restore a reset branch back to its previous state (1 pt)
- ~~GIT cherry-pick, rebase~~ (1 pt)
- Create a VCS hook (1 pt)
- Install a code-highlighter plugin in your CLI text editor (1 pt)
- Create a VCS alias (1 pt)
- Master any CLI editor (ex. VIM). Learn 10 commands to prove your mastery (1 pt)
- Create your own server (ex. virtual machine) (2 pt)
- ~~Create a VCS merge conflict and solve it~~ (1 pt)

Note: ~~Deleted Tasks Were not done~~

Report

Connect to a remote server via SSH

I have connected to fafstudent@95.65.10.xx to ssh server *using the provided password from Facebook* by typing the following command in Terminal:

`ssh fafstudent@95.65.10.xx`

Initialize a repository on server & Create a file in repository folder, write in your name, save it and commit it

The repository has been initiated as follows:

```
ssh fafstudent@95.65.10.xx
mkdir Luchianenco_Filip_FAF-121
cd Luchianenco_Filip_FAF-121/
echo "LuchFilip added some text to README.md  from Terminal" >> README.md
mkdir IDE
cd IDE/
vim LuchFilip.txt
#pressed i to enter insert mode and wrote some text*
#pressed ESC and typed
wq #write & quit
cd ../..
git init
git status
git add #add necessary files to git
git commit -m 'initial commit'
```

Connect to server using public key

In order to use a public key we need to created the key on local machine, insert the content of the public key `.pub` to authorized_keys file which is located in `.ssh/` folder. Then `chmod 600` all keys on local and remote machine, so that other users cannot access them. Here are the commands:

```
cd .ssh/
ssh-keygen # name of file: luchfilip; also password may be written
# two files are created luchfilip and luchfilip.pub
cat luchfilip.pub | copy #copy will copy the output to clipboard
chmod 600 luchfilip* # chmod both files
ssh fafstudent@95.65.10.**
cd .ssh/
echo "<paste key here>" >> authorized_keys
chmod 600 authorized_keys
exit #close session
#test if everything works fine
ssh fafstudent@95.65.10.**
#I am asked to write the password which I wrote when the key was created
```

Create 2 more branches with at least one unique committed file per branch

```
#on remote machine
git branch develop
#push branch on github if necessary
git push origin develop
#switch to new branch
git checkout develop
#make sure we are in the new branch
git branch
  #output
*develop
 master
echo "content of the new created file in the new branch" >> luchfilip_develop.txt
git add luchfilip_develop.txt
git commit -m 'new branch created'
git branch release
```

Set a branch to track a remote origin on which you are able to push

```
  #on remote machine
  git branch --track newFeature origin/master
  #output
  Branch newFeature set up to track local branch master.
  #check result
  git branch -a
  develop
* master
  newFeature
  release
```

Reset a branch to previous commit, reset a branch to some specific commit

Reset to previews commit `git reset --soft 'HEAD'`

Reset to some specific commit

`git log` *output commit log to see which one we want to reset and copy the first ~6 characters*

`git reset --hard f014ed` *in my case the first 6 where these ones*

Restore a reset branch back to its previous state

```
git fsck --lost-found
git reflog
```

Create a VCS hook

There are two types of VCS hooks:

- Client Side : for client operations such as committing and merging
- Server Side :for Git server operations such as receiving pushed commits

  By default git creates sample hook files in `.git/` folder. I used the `pre-commit.sample` file by renaming it to `pre-commit`. So every time I will commit this script will be ran and it should exit with non-zero status.

  Here is the edited hook file which it will check for non-ascii filenames:

```
#!/bin/sh

#the next line allows us to config the value of allownonascii from Terminal
allownonascii=$(git config hooks.allownonascii)

# Redirect output to stderr.
exec 1>&2

if [ "$allownonascii" != "true" ] &&
    test $(git diff --cached --name-only --diff-filter=A -z $against |
      LC_ALL=C tr -d '[ -~]\0' | wc -c) != 0
then
    echo "Error: Attempt to add a non-ascii file name."
  echo
    exit 1
fi

# If there are whitespace errors, print the offending file names and fail.
exec git diff-index --check --cached $against --
```

Install a C++ code-highlighter in vim

```
  git clone https://github.com/octol/vim-cpp-enhanced-highlight.git
  cd vim-cpp-enhanced-highlight
  #copy folder and content to .vim/ directory
  cp after/ -r ~/.vim/
```

Create a VCS alias

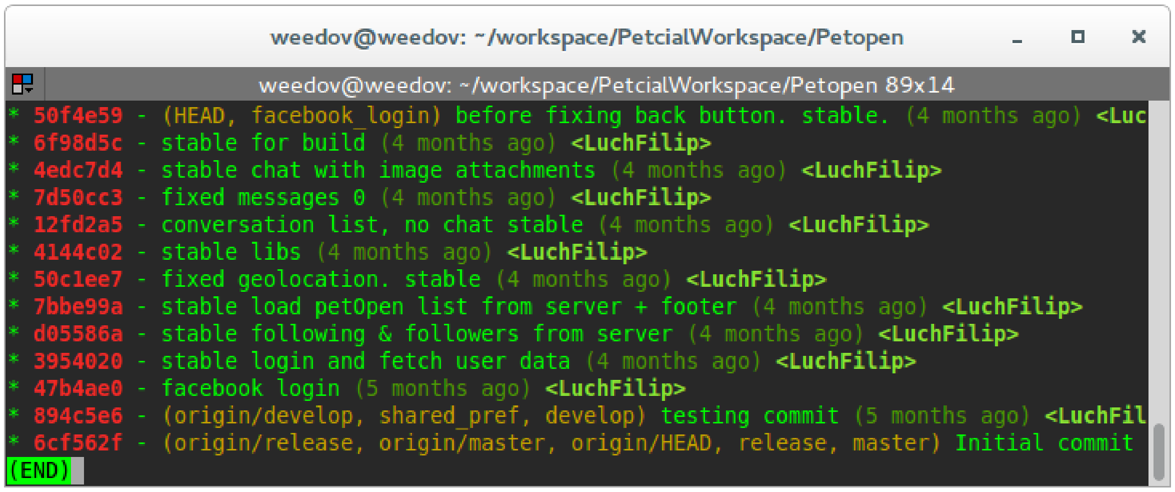
In order to create a VCS alias we need to edit the `~/.gitconfig` file

`vim ~/.gitconfig`

Add all necessary aliases under `alias`

```
  [alias]
  st = status
  co = checkout
  df = diff
  lg = log --graph --pretty=format:'%C(bold red)%h%Creset
      -%C(yellow)%d%Creset %s %Cgreen(%cr) %C(bold green)<%an>%Creset' --abbrev-commit --date=relative
```

Now in order to get a nice looking log of my commits I can use `git lg` and the output will be as follows:


weedov@weedov: ~/workspace/PetcialWorkspace/Petopen
weedov@weedov: ~/workspace/PetcialWorkspace/Petopen 89x14

```
* 50f4e59 - (HEAD, facebook_login) before fixing back button. stable. (4 months ago) <Luc
* 6f98d5c - stable for build (4 months ago) <LuchFilip>
* 4edc7d4 - stable chat with image attachments (4 months ago) <LuchFilip>
* 7d50cc3 - fixed messages 0 (4 months ago) <LuchFilip>
* 12fd2a5 - conversation list, no chat stable (4 months ago) <LuchFilip>
* 4144c02 - stable libs (4 months ago) <LuchFilip>
* 50c1ee7 - fixed geolocation. stable (4 months ago) <LuchFilip>
* 7bbe99a - stable load petOpen list from server + footer (4 months ago) <LuchFilip>
* d05586a - stable following & followers from server (4 months ago) <LuchFilip>
* 3954020 - stable login and fetch user data (4 months ago) <LuchFilip>
* 47b4ae0 - facebook login (5 months ago) <LuchFilip>
* 894c5e6 - (origin/develop, shared_pref, develop) testing commit (5 months ago) <LuchFil
* 6cf562f - (origin/release, origin/master, origin/HEAD, release, master) Initial commit
(END)
```

Learn 10 vim commands

Commands I have used the most were:

- `i`
- `esc`
- `dd` delete current line
- `gg` move cursor to first line
- `G` move cursor to last line
- `dG` delete all lines
- `:wq` write and quit
- `h` `j` `k` `l` move through file

Commands I found useful now:

- `:w filename` write a copy of the file you are editing as filename
- ` `.` jump to last modified location

Create your own server

I have Installed a linux distro on my virtualbox on my linux machine. Shortly, we need to install openshh-server if not installed already, start the server and then port forward ports from VirtuaBox settings.

I also wrote a post in my blog on how to set up ssh server on virtualBox step by step.

Link: http://luchfilip.wordpress.com/2014/02/25/setup-virtualbox-ssh-linux-server/

Conclusion

This Laboratory Work seemed to be extremelly simple for me at the beginning, but when I started to follow all tasks, I understood a lot of little important details which every programmer has to know by default. I found some difficulties in formatting the README.md file and outputting it to a nice looking pdf.

Shortly, it was fun to follow the give tasks, and I regret for not starting to do this Laboratory work earlier, so that I could complete 100% of the tasks.

References

- Git Branch Management
- SSH public key authentication
- Copy file from local machine to SSH Server
- Markdown examples
- Git Remote Branches
- Git Remote Tracking Branches
- Revert Previews Git Commit
- Git Hooks
- vim C++ highlighter
- Copy directory with content
- Git Aliases
- Markdown to PDF easy way