Severin Reiz MPE, March 2019

Git course - Tutorial

Branching and merging

Now, we will step through some easy branching and merging in the command line to get familiar with the respective commands. If you feel like it, you can simultaneously open the repository in GitKraken and after each step try to predict how the visualized graph will look.

a) Create a new local repository.

```
$ mkdir repo
$ cd repo
$ git init
```

b) Create a file and check what git says about that with `git status`.

```
$ touch f0
$ git status
```

c) Add the file to the index and run `git status` again.

```
$ git add f0
$ git status
```

d) Now commit the index and check again what git wants to tell you.

```
$ git commit -m 'added f0'
$ git status
```

e) Create a new branch and, in the new branch, create a new file (with a different name than the last one). Commit your changes.

```
$ git checkout -b new_branch
$ touch f1
$ git add f1
$ git commit -m 'added f1 to new branch'
```

f) Switch back to the master branch and create yet another file with a new name. Commit that as well.

```
$ git checkout master
$ touch f2
$ git add f2
$ git commit -m 'added f2 to master branch'
```

g) Take a look at the graph with `git log --graph --oneline --all`.

```
$ git log --graph --oneline --all
```

h) Now merge the new branch into the master branch.

```
$ git merge new_branch
```

i) Print out the graph again and look at the files in the folder. Did that work out the way you thought?

```
$ git log --graph --oneline --all
```

Resolving a merge conflict

For this exercise we will reuse the local repository we just created.

a) Switch to the master branch if you are not on it and create a new file. Write something into it and commit the changes.

```
$ git checkout master
$ echo 'This is a file' >> f3
$ git add f3
$ git commit -m 'yet another file'
```

b) Switch to the other branch and create a file with the same name. Write something else into it. Commit that as well.

```
$ git checkout new_branch
$ echo 'same name. different conent.' >> f3
$ git add f3
$ git commit -m 'this surely will cause no problems'
```

c) Now try to merge the master branch into the current branch. This should fail.

```
$ git merge master
```

d) Take a look at the file causing the conflict. You should find the unresolved differences marked in the way explained in the lecture.

```
$ less f3
```

e) Resolving this little conflict directly in the command line is easy. So, you can give it a try or take a look at how the situation looks like in GitKraken. It has an integrated merge tool that works similar to most other tools around and can come in handy when having to resolve larger conflicts. After you resolved the conflict, you can commit the changes.

```
$ vim f3
<delete part of the conflicting content>
$ git add f3
$ git commit -m 'resolved merge conflict'
```

f) Suppose you are not happy with the result of you merge. Just reset the branch to its previous state with `git reset HEAD --hard`.

```
$ git reset HEAD~ --hard
```

Website

The website with slides and tutorials can be found at
`https://www5.in.tum.de/wiki/index.php/Python_/_Git_/_Bash_Course`.