Git & GitHub Tutorial

Software-Engineering

by

Daniel Schädler and Jean-Pierre Hotz

22. Oktober 2018

Course: TINF17B4

Lecturer: K. Berkling, Ph.D.

Disclaimer

Since the usage of the git commandline and ui-clients for git work in basically the same way this (written) tutorial only shows you how to use Git and GitHub from the commandline. Other UIs will be shown during the presentation, though.

On the topic of commandlines: Every command / file shown in this tutorial is enclosed in a box with a black outline. In case a command is shown, any user input will be denoted by a preceding dollar sign (as it's done by Git for Windows).

1 Creation of a local repository

There are two different ways to create a local repository, which is used to track your own changes in the exisiting code.

The first option you have is to create an „empty“ repository. In case this is done in a folder, which contains files, these files are not being tracked yet. This can be done using the following command.

```
$ git init
```

The second option (which is probably used more often in teamwork), is to clone from an already existing remote repository, like an repository on GitHub or, an ordinary Git repository, you or a team member has created on a simple file server.

```
$ git clone https://github.com/<User-Name>/<Repository-Name>.git
```

2 Use your Github user in Git

To use your GitHub user in Git (and thus also link your local commits to your Github account) you can change the user name and e-mail-adress as follows:

```
$ git config --global user.name "<your Github-Username>"
$ git config --global user.email <your Github-E-Mail>
```

In case you need verification Git will ask you for valid credentials.

3 Make Git ignore certain files / folders

To make Git ignore certain files and folders you'll have to create an `.gitignore`-file. Every line in this file will represent a pattern to excluded files, also called a rule. In case you want files with a certain pattern to be only excluded in the root-directory you'll have to precede the pattern with a forward-slash `'/'`. To exclude folders you'll have to follow the pattern by a forward-slash `'/'`. To make a pattern match anything you can simply type an asterisk `'*'`. To create an exception to a general rule, you can simply give a more specific rule (for the files you want to **include**), and precede that rule with a bang `'!'`.

The `.gitignore`-file is always useful to exclude IDE-specific files (e.g. `.iml`-files in IntelliJ) or compiled binaries. An example `.gitignore`-file for a personal Java-project is shown below:

```
*.iml
target/
.idea/*
!.idea/copyright/
lib/
```

4 Make versions of files

Since Git and Github is all about file versioning, we can make Git capture single versions of files with commits. Before creating commits though, we'll have to add changes that are to be included in the version we want to capture. This process is called 'staging changes' and can be done either by staging all changes, or by only staging certain files.

```
> git add --all
> git add <files>
```

To then create a commit with the staged changes you can use following command:

```
> git commit -m "<Commit-message (can be multi-line)>"
```