

# 1 gitlabUsage guide

## 1.1 Create new Project:

- go to website, create new Project, follow instructions, to the mkdirs in your project directory
- Create SSHkey, to project
- add remote add origin
- push -u origin master (the master branch is now the origin branch, all next pushes are going to this branch)

## 1.2 define sshkeys

1. ssh-keygen -t rsa -C "\$your\_email" to create a ssh key in linux cat ~/.ssh/id\_rsa.pub show the ssh key
2. github -> settings -> deploy keys -> Add deploy key

if we have different git repositories we can make a config file in the ~/.ssh/ folder

```
Host krios HostName krios.tbi.univie.ac.at IdentityFile ~/.ssh/id_rsa_krios
Host github.com HostName github.com IdentityFile ~/.ssh/id_rsa_github
```

## 1.3 important commands

### 1.3.1 git status

show the status, what is already commit, what is new what is modified

### 1.3.2 git pull

update your local folder

### **1.3.3 git diff filename1**

show the difference between the current file and the file in the repository

### **1.3.4 git rm**

removes a file from the git repository and from the file system, changes are recognized now by commit and push

### **1.3.5 git clone SSH address project**

copies an entire gitlab project to the current folder

### **1.3.6 git checkout file1**

drop every change made with file1, and restore the information in the repository

## **1.4 change a file and add it to the repository**


- `git pull`
- `git status`
- `git diff filename1`
- `git add filename1`
- `git commit -m "your comment"`
- `git status`
- `git push` or `git push -u origin branchname1` if the commits should go to another branch

## **1.5 managing branches**

- `git branch`  
show all branches, and the current branch
- `git checkout branchname1`  
switch to branch branchname1

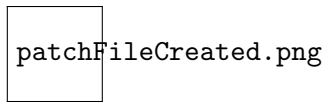
- `git checkout -b newBranchName`  
create new branch newBranchName
- `git push -u origin newBranchName`  
now commits will be pushed to this branch if `git push` was called
- `git branch -d branchName1`  
delete branch
- `git branch -m newname`  
rename current branch to newname
- `git rebase branch1`  
take the current branch and set it as head to branch1 Caution: merges can occur

## 1.6 Get back in the git History, not working



gitLog.png


1. `git log` (show all commits)
2. `git format-patch Hashkey` (of commitNode where we want to go back)



3. `less patchfile` ( show the patchfile changes)



4. `git reset --hard` hashkey (of commitNode where we want to go back)



reseted.png

5. `patch -p1 < patchfile`

(undo saved changes, p1 if hirarchy is a/filename and b/filename, else 2 or 3)