## Git

更多详细操作

https://www.liaoxuefeng.com/wiki/0013739516305929606dd18361248578c67b8067c8c017b000

### 1.安装和配置

<https://git-scm.com/downloads>

<https://pan.baidu.com/s/1kU5OCOB#list/path=%2Fpub%2Fgit>

通过以上网址下载windows版本的git

然后运行安装包，默认安装

配置用户,使用以下命令

git config –global user.name “名字”

Git config –global user.email “email”

### 2.创建git库

创建一个空文件夹 mkdir gitRepository

切换到创建的文件夹里 gitRepository

初始化git库 git init

### 3.添加文件/或修改文件

切换至gitRepository目录

然后使用命令 git add gitTest.txt gitTest.txt为要添加或者已经修改的文件

最后要提交 git commit –m “备注内容”

### 4.查看提交日志

Git log 可以添加参数--pretty=oneline 简化输出信息

### 5.还原版本

git reset –hard HEAD^ 还原到上一个版本

git reset –hard HEAD^^ 还原到上上一个版本

git reset –hard HEAD~100 还原到一百个上一个版本

git reset –hard commitID 可以用于恢复到过去的版本或者未来的版本

git commit --amend 修改上次提交的备注

### 6.查看操作历史

git reflog 可以查看每次操作之后的版本号，用于恢复

### 7.查看工作区的状态

git status 可以查看哪些文件是否提交

### 8.理解工作区、暂存区、分支

**工作区**：文件夹中的我们看见的文件

**暂存区**：修改之后，需要先保存到暂存区

**分支**: 暂存区的修改，才能保存到分支

Add 相当于将本次修改的操作 保存到暂存区

Commit命令 相当于将暂存区的修改操作 保存到当前分支上

### 9.撤销修改

git checkout -- readme.txt 撤销回最后一次add或者commit操作之后

命令中的 **–** 很重要，如果没有则变成了分支的命令了

git reset head 文件名 将改文件保存到暂存区的操作撤销回工作区

### 10.删除

当在工作空间删除了文件之后，

可以使用git rm 文件名 真正从版本库删除

也可以使用 git checkout –- 文件名 恢复到工作空间

### 11.关联远程仓库 并更新代码

1）ssh-keygen -t id\_rsa 使用此命令生成sshkey，名字一定要是id\_rsa并且将生成的文件拷贝到用户目录下的.ssh下

2）打开生成的文件（xxx.pub那个），复制全文，添加的github账户上sshkey列表

3）git remote add origin git@github.com:Zhaojing033033/Document.git

使用此命令关联远程仓库，其中赵zhaojing033033为自己的账户

Document为自己的仓库名

远程仓库的默认名为origin

4）git pull origin master使用此命令抓取远程仓库

5）git push -u origin master 将本机的修改推送至远程仓库

之后提交之后，就不需要-u这个参数

6）将远程仓库更新  
git fetch origin master //从远程的origin仓库的master分支下载代码到本地的origin master

git log -p master.. origin/master//比较本地的仓库和远程参考的区别

git merge origin/master//把远程下载下来的代码合并到本地仓库，远程的和本地的合并

### 12.克隆远程仓库

git clone [git@github.com:michaelliao/gitskills.git](mailto:git@github.com:michaelliao/gitskills.git)

michaelliao为克隆的账户

gitskills为克隆的仓库

### 13创建与合并分支

git checkout -b dev 创建dev分支，参数-b是切换至dev分支

以上命令同

git branch dev 创建dev分支

git checkout dev切换到dev分支

git branch 列出分支信息，当前分支会前面有\* 标识

git merge dev合并dev分支到当前分支

默认采用fast-forward策略，会将master指向dev分支；

但是添加参数—no-ff 会禁用fast-forward策略，此时会形成一次新的提交，所以还需要添加参数-m “备注内容”

git branch -d dev 删除dev分支

15.bug分支

当编码尚未完成的时候，接到紧急任务，需要先结局紧急任务。然而当前的工作空间尚未完成，不能提交。这是时候就需要bug分支了

命令 git stash 将当前空间的修改储存，此时用git status看当前空间是空的（没有被git管理的文件除外）。

然后创建bug分支，修改，提交之后

使用git stash apply /git stash drop/git stash pop 来恢复之前存储的工作空间。不过第一个命令，恢复之后，不会删除存储的空间

### 14.打标签

切换至需要打标签的分支

git tag <标签名> 创建标签 默认创建标签是创建在最后一次提交上

git tag <标签名> <commitID> 在指定提交号上打标签

git tag <标签名> <commitID> -m “说明” 创建带说明的标签

git tag 查看所有标签

可以使用git log --pretty=oneline --abbrev-commit查看到提交的commitID

git show <标签名> 查看带备注以及其他信息

15、git分支合并

|  |
| --- |
| #创建分支  git branch dev  #切换分支  git checkout dev  #将dev的结果合并到master  #先切到master  git checkout dev  #合并  git merge dev |