# Weile git学习

<https://github.com/Lil-Wei/>

<https://git-scm.com/docs> （官方文档）

#### 学习：

>mkdir gitlearning

>cd gitlearning

>git init (建立了一个master分支)

>touch testgit

>git add testgit

>git commit -m “this is a test”

#### 如何放到远程github仓库

先在github创建一个空仓库:gitlearning

把当前分支push到远程仓库下的master分支

>git push [git@github.com:Lil-Wei/gitlearning.git](mailto:git@github.com:Lil-Wei/gitlearning.git) master （如没有master分支，则创建master分支）

[git@github.com:Lil-Wei/gitlearning.git](mailto:git@github.com:Lil-Wei/gitlearning.git) 太他妈难记了，有没有缩写，有！

>git remote add **originA** [git@github.com:Lil-Wei/gitlearning.git](mailto:git@github.com:Lil-Wei/gitlearning.git)

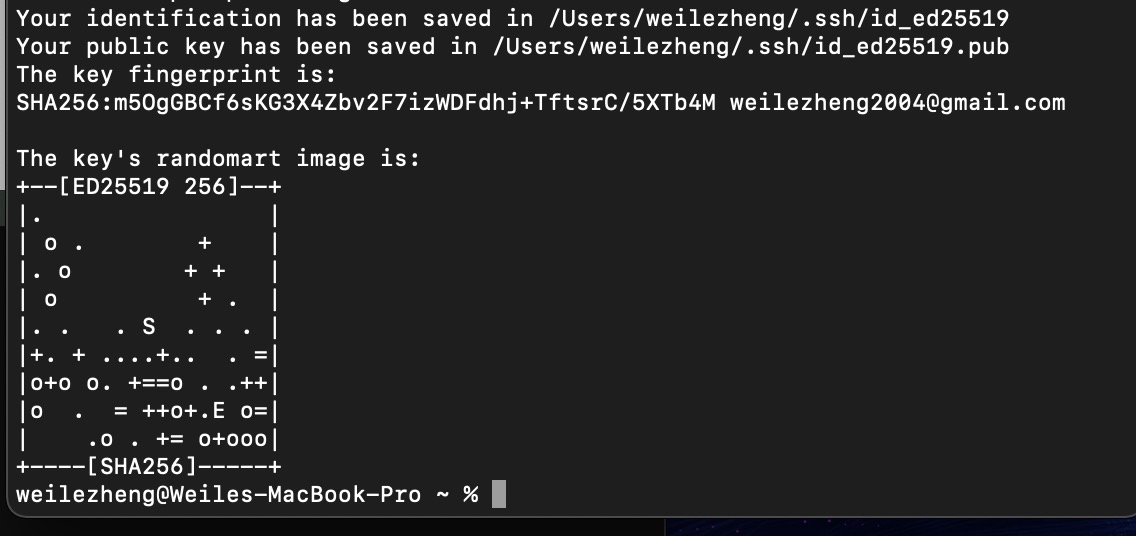
那么就可以简写成：git push **originA** master

#### github提示deny的解决办法，

* 在电脑上生成SSH密钥

$ssh-keygen -t ed25519 -C ["](mailto:\"your_email@example.com\")*[your\_email@example.com](mailto:\"your_email@example.com\")*["](mailto:\"your_email@example.com\") （改变邮件地址：登录github的email）

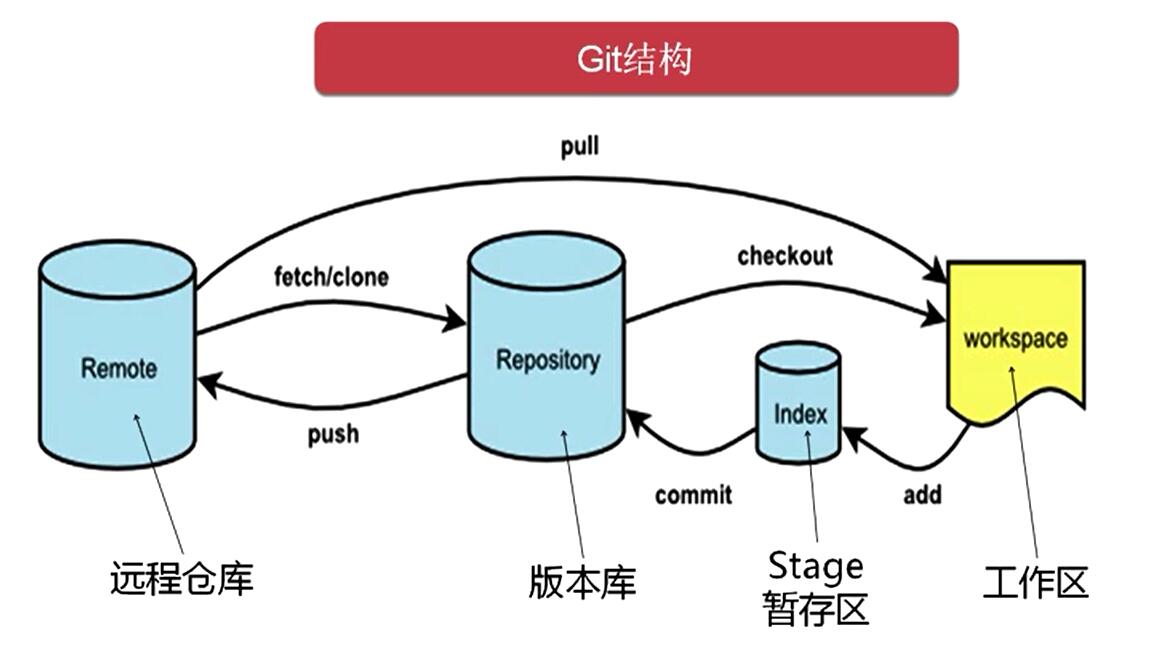
几个回车：



* 把公钥配置在github上面

把/Users/weilezheng/.ssh/id\_ed25519.pub 文件内容保存到github里面设置SSH里面。

#### Git结构



#### git基本命令

* 本地仓库相关

git status

git branch

git add

git commit -m “”

* 远程代码仓库相关

建立**远程仓库链接的别名**

>git remote add <别名> <远程仓库地址>

如:>

>git remote add origin [git@github.com:curbst/weatherapp.git](mailto:git@github.com:curbst/weatherapp.git)

>git remote remove origin （删除）

>git remote -v （查看远程仓库链接名）

#### git分支branch

* **在本地仓库中创建分支:**

>git branch feature1 ( feature1是分支名)

>git branch 查看分支及当前分支

* **切换分支**

***>git checkout feature1*** ( feature1是分支名)

（以上**创建分支**并**切换分支**的命令合并在一起: git checkout -b feature1）

* **在远程仓库(github)中创建分支**

>git push ***[git@github.com:curbst/weatherapp.git](mailto:git@github.com:curbst/weatherapp.git) feature1***

>git push origin feature1

* **删除本地仓库分支**

>git branch -d feature1

* **删除远程仓库分支**

>git push origin -d feature1

切换分支：

在任意分支中创建的文件，只要未将这个文件git commit，那么无论切换为哪个分支，这个文件都会存在于任何分支中，在任何分支中都是untracked未被跟踪状态！

但是一旦（git commit）之后，这个文件只能在执行git commit的分支中看见，其他分支中看不见

一个文件在任意分支被git add 后，在某一分支"分支1"中被commit提交，那么这个文件就单独属于"分支1"了，意思也就是：我们再切换到不是“分支1”的任何其他分支(如:“分支2”,"分支3"等)中，我们都不会再可以找到这个文件的踪影。这个文件已经单独属于"分支1"了。

**结论**：一个文件在"分支1"中被commit提交，那么这个文件已经单独属于"分支1"了，即便这个文件被如何修改或处于任何状态，这个文件只与"分支1"有关系，其他任何分支都找不到"committed.txt"的任何蛛丝马迹。

#### Git 里面的 origin 到底代表啥意思?

<https://www.zhihu.com/question/27712995>

#### git log 查看提交历史

参考:https://stackoverflow.com/questions/1057564/pretty-git-branch-graphs

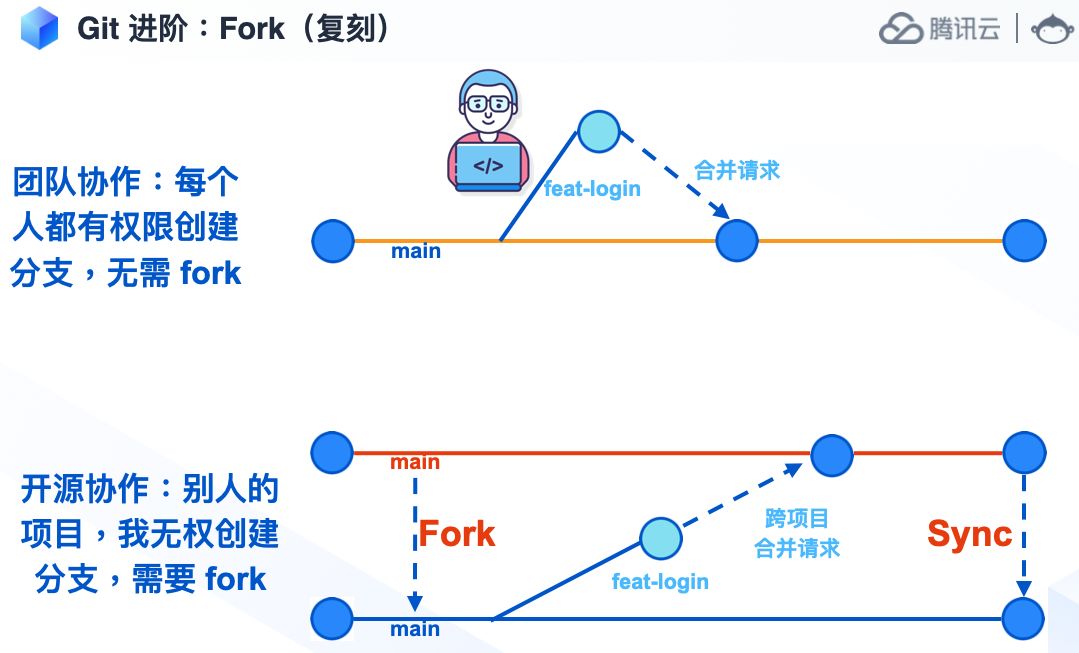
配置一个别名:

git config --global alias.adog "log --all --decorate --oneline --graph"

使用方法

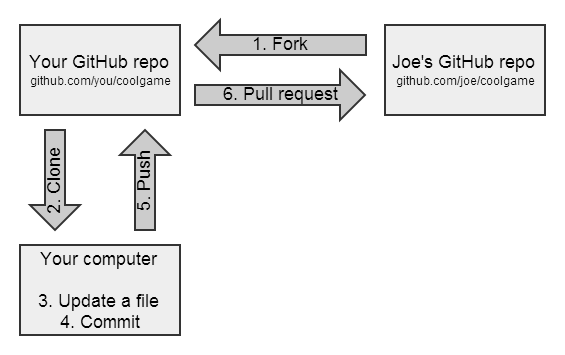
>git adog

#### Fork简单的说明：



GitHub中Fork 是 ***服务端的代码仓库克隆***（即 新克隆出来的代码仓库在远程服务端），包含了原来的仓库（即upstream repository，上游仓库）所有内容，如分支、Tag、提交。代码托管服务（如Github、BitBucket）提供了方便的完成Fork操作的功能（在仓库页面点一下Fork按钮）。

这样有了一个你***自己的*** 可以自由提交的远程仓库，然后可以通过的 **Pull Request** 把你的提交***贡献回*** 原仓库。而对于原仓库Owner来说，鼓励别人Fork他的仓库，通过Pull Request 能给他的仓库***做贡献***，也是提升了原仓库的***知名度***。

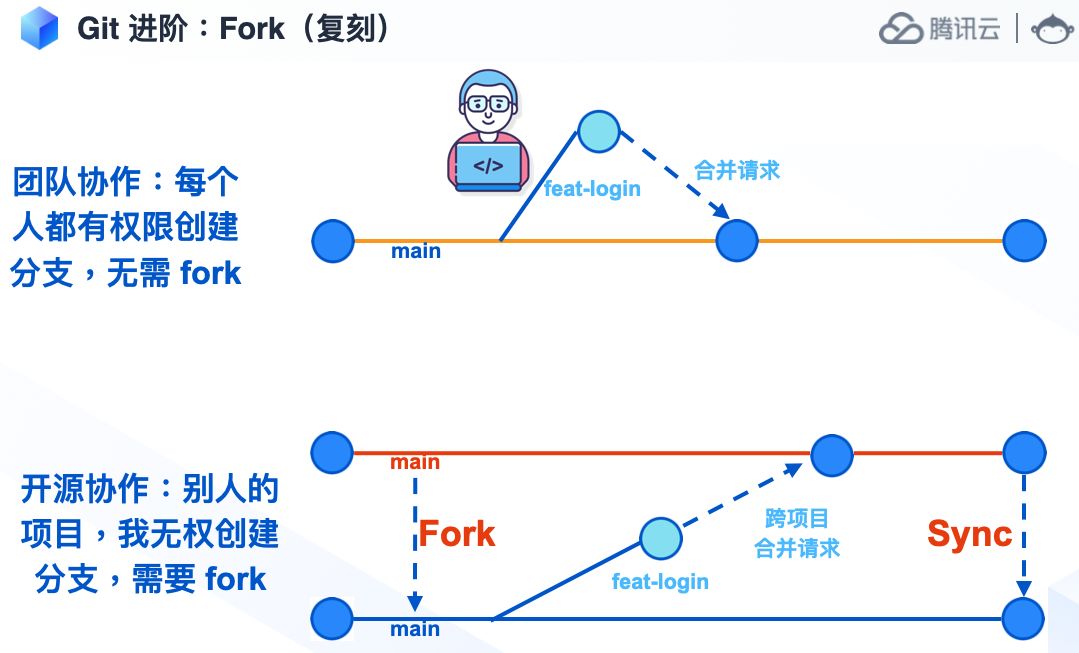


Fork 即「复刻」——把别人的项目复制为自己的，可随意修改。官方文档：[复刻仓库 - GitHub 帮助](https://link.zhihu.com/?target=https://help.github.com/cn/github/getting-started-with-github/fork-a-repo" \t "_blank)

Fork 千万不要用于私有项目团队协作，它仅适用于开源协作。

### 作业:

**git merge 和git rebase命令的区别**

Git Fork

GitHub Flow：面向团队成员协作，禁止提交主干，每个人有权创建临时分支，发起合并请求（pull request）。官方文档：[Understanding the GitHub flow](https://link.zhihu.com/?target=https://guides.github.com/introduction/flow/" \t "_blank)

Fork：面向团队外部协作者，无权创建临时分支，所以不得不复刻，发起跨项目合并请求。

Fork 不属于 GitHub Flow，也不属于任何一种 Git Workflow。

#### Git 里面的 origin 到底代表啥意思?

<https://www.zhihu.com/question/27712995>