1. linux上使用git
2. 下载 yum install git
3. 本地配置

git config --global user.name 'liulx'

git config --global user.email ['123456@qq.com'](mailto:'123456@qq.com')

1. Linux创建ssh秘钥

ssh-keygen #三次回车

通过cat 查看~/.ssh/id\_rsa.pub，复制文件中的所有内容，添加到Github账户信息Account Settings->SSH Key

ssh -T [git@github.com](mailto:git@github.com) #验证SSH秘钥是否通过

1. 把github上的项目clone到本地

git clone git://gitee.com/liulx22/jingyingtest.git

1. 修改本地项目提交到github上

cd jingying

mkdir testdir & cd testdir

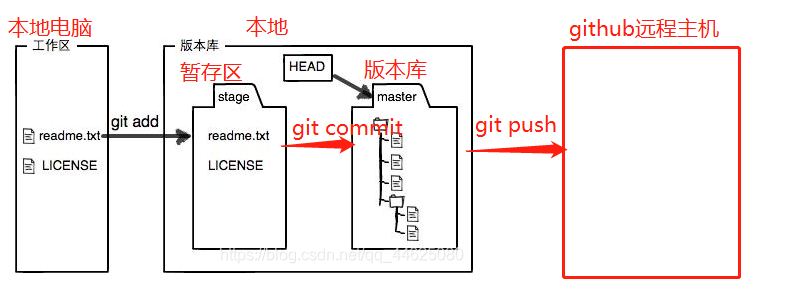
touch index.html

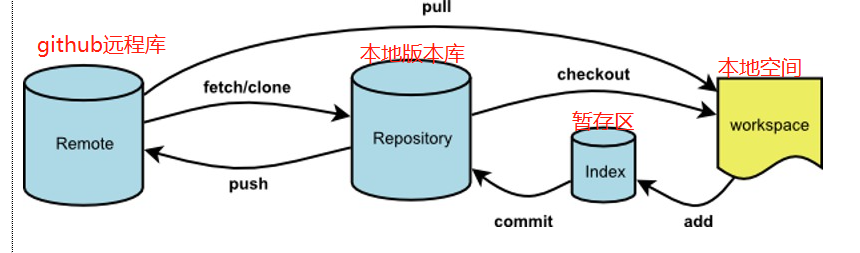
git add . #添加文件到本地仓库

git commit -m "first commit" #提交到本地库并备注，此时变更仍在本地。

git push #将本地master分支的文件提交到Github版本库中，此时才更新了本地变更到github服务上。

1. Git命令的区别





要更新你的本地仓库至最新改动，git pull

6.新建项目提交到gitee上

mkdir testdir & cd testdir

touch README.md

git init #初始化一个本地库,创建一个.git的文件

git add README.md #添加文件到本地仓库

git commit -m "first commit" #提交到本地库并备注，此时变更仍在本地。

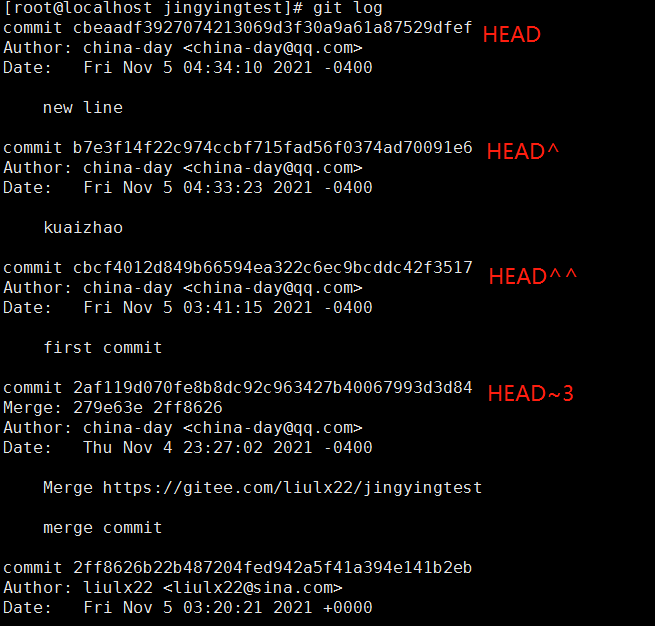
git remote add origin https://gitee.com/liulx22/jingyingtest.git

#添加远程服务器

git push -u origin master #由于远程库是空的，我们第一次推送master分支时，加上了-u参数，Git把本地的master分支内容推送的远程新的master分支，还会把本地的master分支和远程的master分支关联起来，在以后的推送或者拉取时就可以简化命令，git push，此时才更新了本地变更到github服务上。

1. 版本回退

git log查看本地版本库的历史记录,用HEAD表示当前版本.



[root@localhost jingyingtest]# git reset --hard HEAD^

HEAD is now at b7e3f14 kuaizhao

[root@localhost jingyingtest]# git push -f #强制同步到远程

1. 分支

分支在实际中有什么用呢？假设你准备开发一个新功能，但是需要两周才能完成，第一周你写了50%的代码，如果立刻提交，由于代码还没写完，不完整的代码库会导致别人不能干活了。如果等代码全部写完再一次提交，又存在丢失每天进度的巨大风险。现在有了分支，就不用怕了。你创建了一个属于你自己的分支，别人看不到，还继续在原来的分支上正常工作，而你在自己的分支上干活，想提交就提交，直到开发完毕后，再一次性合并到原来的分支上，这样，既安全，又不影响别人工作。

1）一开始的时候，master分支是一条线，Git用master指向最新的提交，再用HEAD指向master，就能确定当前分支，以及当前分支的提交点：



每次提交，master分支都会向前移动一步，这样，随着你不断提交，master分支的线也越来越长。

2）当我们创建新的分支，例如dev时，Git新建了一个指针叫dev，指向master相同的提交，再把HEAD指向dev，就表示当前分支在dev上：



实战：首先，我们创建dev分支，然后切换到dev分支：

[root@localhost jingyingtest]# git checkout -b dev

Switched to a new branch 'dev'

[root@localhost jingyingtest]# git branch #查看当前分支

\* dev

master

你看，Git创建一个分支很快，因为除了增加一个dev指针，改改HEAD的指向，工作区的文件都没有任何变化！

3）不过，从现在开始，对工作区的修改和提交就是针对dev分支了，比如新提交一次后，dev指针往前移动一步，而master指针不变：



实战：dev分支上提交代码

[root@localhost jingyingtest]# vi index.html

[root@localhost jingyingtest]# git add .

[root@localhost jingyingtest]# git commit -m "dev branch commit 01"

[dev a6889e1] dev branch commit 01

1 file changed, 1 insertion(+)

[root@localhost jingyingtest]# vi index.html

[root@localhost jingyingtest]# git add .

[root@localhost jingyingtest]# git commit -m "dev branch commit02"

[dev d21eec0] dev branch commit02

1 file changed, 1 insertion(+)

4）假如我们在dev上的工作完成了，就可以把dev合并到master上。Git怎么合并呢？最简单的方法，就是直接把master指向dev的当前提交，就完成了合并：



所以Git合并分支也很快！就改改指针，工作区内容也不变！

实战：

[root@localhost jingyingtest]# git checkout master #切换到master分支

Switched to branch 'master'

[root@localhost jingyingtest]# git merge dev #把dev合并到master上

Updating b7e3f14..d21eec0

5）合并完分支后，甚至可以删除dev分支。删除dev分支就是把dev指针给删掉，删掉后，我们就剩下了一条master分支：



真是太神奇了，你看得出来有些提交是通过分支完成的吗？

实战：

[root@localhost jingyingtest]# git branch -d dev

Deleted branch dev (was d21eec0).

[root@localhost jingyingtest]# git branch

\* master

[root@localhost jingyingtest]#

1. 解决每次push需要输入github用户名密码的问题

1）查看克隆的方式是https的方式

git remote -v

origin  https://github.com/username/repository.git (fetch)

origin  https://github.com/username/repository.git (push)

2）删除并重新设置为git的方式

git remote rm origin（删除远程库）

git remote add origin git@github.com:username/repository.git

1. 重新查看方式

git remote -v

origin  git@github.com:username/repository.git (fetch)

origin  git@github.com:username/repository.git (push)

1. push一下

git push -u origin master

注意：

第二步解除了本地和远程的绑定关系，并不是物理上删除了远程库。远程库本身并没有任何改动。要真正删除远程库，需要登录到GitHub，在后台页面找到删除按钮再删除。

要关联一个远程库，使用命令

git remote add origin git@server-name:path/repo-name.git；关联一个远程库时必须给远程库指定一个名字，origin是默认习惯命名；关联后，使用命令git push -u origin master第一次推送master分支的所有内容；

此后，每次本地提交后，只要有必要，就可以使用命令git push origin master推送最新修改；

1. Git status查看本地版本库的状态

Git log查看本地版本库的历史记录