**SVN、GIT日常看我就够了**

[](http://www.jianshu.com/u/2d60352070ec)

作者 [Fly\_Sunshine\_J](http://www.jianshu.com/u/2d60352070ec) 关注

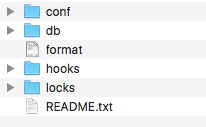
2016.10.31 17:47 字数 1685 阅读 3708评论 9喜欢 110

**配置本地SVN（自己测试使用，咱不能拿公司的SVN服务器做测试吧）**

1. 在本地创建一个仓库，打开终端执行

svnadmin create /Users/vcyber/Desktop/SVNCode/

执行完成桌面多了一个SVNCode文件夹，目录结构如下：

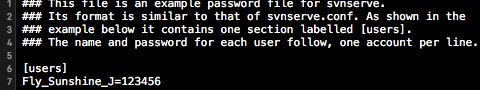


目录结构

1. 配置svn权限  
   2.1 使用Xcode打开/conf/svnserve.conf，将下面项前面的#去掉， command+s保存，退出。
2. anon-access = read
3. auth-access = write
4. password-db = passed

authz-db = author

anon-access代表匿名用户权限，read代表用户只读，none代表禁止访问，write代表可读可写， 这里我就不修改了，毕竟测试这玩嘛。。。  
2.2 使用Xcode打开/conf/passwd/，在[user]下面添加用户名和密码，command+s保存，退出。



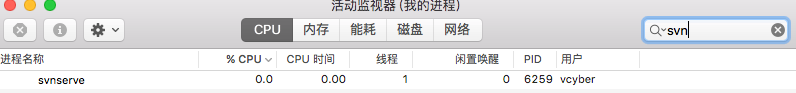
passwd.png

2.3 使用Xcode打开/conf/authz，在[groups]下面添加组名=用户名(可以多个)



authz.png

这里我就一个用户，我就添加一个组，然后享有服务器全部资源，注意添加权限的时候组名前面要加@，用户名不需要。  
2.4 启动svn服务器  
svnserve -d -r /Users/vcyber/Desktop/SVNCode/  
没有任何提示说明启动成功，可以再活动监视器中查看可关闭SVN服务。



活动监视器*\_我的进程*.png

**本地环境配置github(要有github账号)**

1 配置账号信息

git config --global user.name Fly\_Sunshine\_J

git config --global user.email 857402355@qq.com

2 创建本地ssh(用于上传到你对应的github账号)

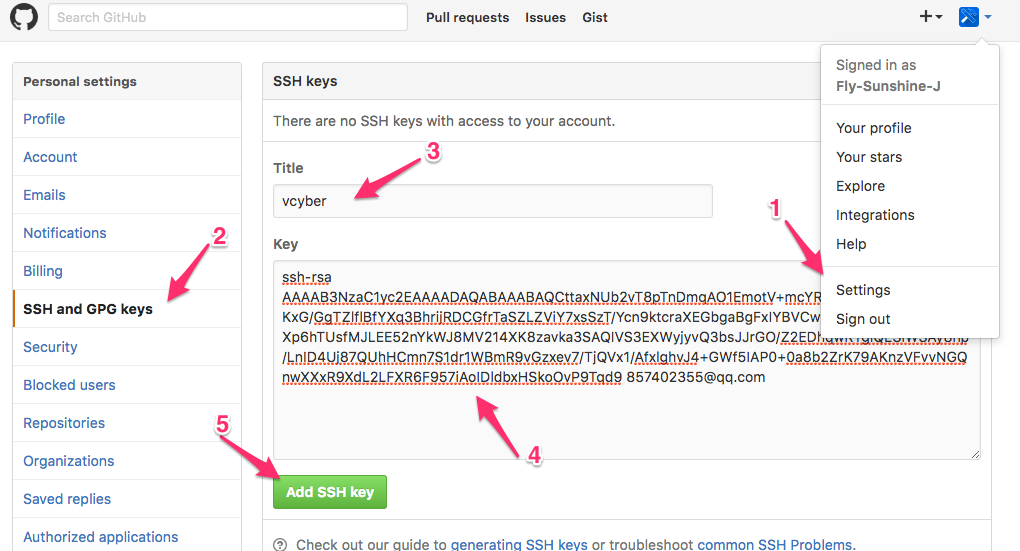
ssh-keygen -t rsa -C "857402355@qq.com"

接下来会让你输入保存路径，你可以选择默认，也可以选择保存到对应位置，最后会让你输入密码， 密码也是可有可无的。



默认保存位置

3 接下来就是找到创建.ssh文件夹，打开id\_rsa.pub，复制内容，登陆github，选择Settings -->SSH and GPG kyes -->New SSH Key -->Add SSH Key



添加SSH

4 验证

ssh -T git@github.com

会出现询问你继续连接，输入yes，然后弹框输入3.2中你设置的密码， 然后出现如下信息说明配置成功

Hi Fly-Sunshine-J! You've successfully authenticated, but GitHub does not provide shell access.

**SVN常用命令**

**import** 导入文件到svn服务器  
示例：

svn import /Users/vcyber/Desktop/Personal/CoreData/RealmTest/ svn://localhost/Desktop/SVNCode/Realm -m "导入项目"

svn import /Users/vcyber/Desktop/Personal/CoreData/RealmTest/ svn://localhost/Desktop/SVNCode/Realm --username=Fly\_Sunshine\_J --password=123456 -m "导入项目"

**checkout(co)** 检出  
示例：

svn checkout svn://localhost/Desktop/SVNCode/Realm /Users/vcyber/Desktop/RealmSVN

svn checkout svn://localhost/Desktop/SVNCode/Realm /Users/vcyber/Desktop/RealmTest --username=Fly\_Sunshine\_J --password=123456

**export** 导出  
示例：

svn export svn://localhost/Desktop/SVNCode/Realm /Users/vcyber/Desktop/RealmExport --username=Fly\_Sunshine\_J --password=123456

svn export svn://localhost/Desktop/SVNCode/Realm /Users/vcyber/Desktop/RealmExport

-r 数字（版本号）

svn export -r 4 svn://localhost/Desktop/SVNCode/Realm /Users/vcyber/Desktop/RealmExport

**delete/remove(rm)** 删除(注意要切换到本地仓库, 别忘了提交，这个操作只删除本地，服务器没有删除)  
示例：

svn delete RealmTest/Realm.framework/

**update** 更新本地仓库  
示例：

svn update 没有文件路径默认全部更新

svn update RealmTest/ViewController.m 更新指定文件

**add** 添加新文件(先把文件添加到work copy中，然后添加，别忘了commit)  
示例：

svn add RealmTest/Realm.framework/

**diff** 比较差异  
示例：

svn diff RealmTest/ViewController.m

比较两个版本的差异

svn diff -r 4:5 RealmTest/ViewController.m

**status** 查看状态  
M：修改，？不存在控制中，你可能需要删除或者添加，C：有冲突，A：新增，K：锁定  
示例：

svn status

//指定文件的状态

svn status RealmTest/ViewController.m

**log** 查看日志  
示例：

svn log

指定文件的日志

svn log RealmTest/ViewController.m

**info** 查看信息  
示例：

svn info

指定文件信息

svn info RealmTest/ViewController.m

**revert** 恢复到上个版本  
注意:它不会恢复被删除的目录。  
示例：

//恢复整个目录

svn revert --recursive

//恢复某一个文件

svn revert RealmTest/ViewController.m

**resolved** 解决冲突  
一般会在你更新的时候有冲突会提示你，然后输入r就可以解决冲突  
示例：

svn resolved RealmTest/ViewController.m

**cp** 创建新的分支  
示例：

svn cp svn://localhost/Desktop/SVNCode/Realm svn://localhost/Desktop/SVNCode/RealmBranch1 -m"创建新的分支"

**remove(rm)** 删除分支  
示例：

svn remove svn://localhost/Desktop/SVNCode/RealmBranch1 -m"删除分支1"

svn rm svn://localhost/Desktop/SVNCode/RealmBranch2 -m"删除分支2"

**merge** 合并分支  
合并分支到另一个分支或者主干  
示例：

主干

svn merge svn://localhost/Desktop/SVNCode/Realm svn://localhost/Desktop/SVNCode/RealmBranch1

分支

svn merge svn://localhost/Desktop/SVNCode/RealmBranch2 svn://localhost/Desktop/SVNCode/RealmBranch1

**Git常用命令**

**init** 新建代码库  
示例：

//在当前目录新建代码库

git init

//在指定位置创建代码库

git init Desktop/GitCode

**git config [--global] user.name/user.email "name/emai"** 配置当前或者全局提交的附带名字或者邮件  
示例：

git config --global user.name "Fly\_SunShine\_J"

git config --global user.email "857402355@qq.com"

//查看设置情况

git config --global user.name

git config --global user.email

**clone** 克隆代码到本地  
示例：

git clone https://github.com/Fly-Sunshine-J/Realm.git Desktop/RealmGit

**commit** 提交暂存区  
示例：

//提交所有

git commit -m"日志"

//提交某一个文件

git commit RealmTest/RealmTest/ViewController.m -m"修改VC"

**status** 查看工作区的文件修改  
示例：

git status

**rm** 删除文件  
示例：

git rm RealmTest/RealmTest/ViewController.m

**add** 添加文件到暂存区  
示例：

//添加指定文件

git add RealmTest/RealmTest/ViewController.m

//添加本地库所有文件

git add .

**mv** 修改某一个文件的名字  
注意，路径要对应，否则会自动移动  
示例：

git mv RealmTest/RealmTest/ViewController.m RealmTest/RealmTest/View.m

**diff** 产看追踪文件的差异  
示例：

//查看追踪文件的差异

git diff

//查看某一文件的差异

git diff RealmTest/RealmTest/ViewController.m

**reset --hard HEAD** 放弃本地全部修改  
示例：

git reset --hard HEAD

**git reset** 回退到某个版本并保存未追踪的改动  
一大串的东西可以通过log来查询  
示例：

git reset 8989920311bacb3f4e3ced7f82ab75ca47c318c7

**git reset --keep**回退到某个版本并保存未提交的改动

git reset --keep 8989920311bacb3f4e3ced7f82ab75ca47c318c7

**checkout HEAD**放弃本地某一文件的修改  
示例：

git checkout HEAD RealmTest/RealmTest/ViewController.m

**revert** 撤销某一次提交  
示例：

git revert 8989920311bacb3f4e3ced7f82ab75ca47c318c7

**checkout** 撤销某次提交的某个文件  
示例：

git checkout 8989920311bacb3f4e3ced7f82ab75ca47c318c7 RealmTest/RealmTest/ViewController.m

**log** 查看提交日志  
示例：

git log

**blame** 查看文件被谁修改  
示例：

git blame RealmTest/RealmTest/ViewController.m

**branch** 分支  
示例：

//查看本地分支

git branch

// 查看远程分支

git branch -r

// 查看远程和本地分支

git branch -a

//创建本地分支，但是不切换

git branch branch1

//新建一个分支并切换

git checkout -b branch2

//新建一个分支，指向指定commit

git branch branch3 8989920311bacb3f4e3ced7f82ab75ca47c318c7

//新建一个分支，与指定的远程分支建立追踪关系

git branch --track branch4 origin/master

//切换到指定分支，并更新工作区

git checkout branch4

//切换到上一个分支

git checkout -

//建立追踪关系，在现有分支与指定的远程分支之间

git branch --set-upstream branch3 origin/master

//合并指定分支到当前分支

git merge branch2

//选择一个commit，合并进当前分支

git cherry-pick 8989920311bacb3f4e3ced7f82ab75ca47c318c7

//删除分支

git branch -d branch1

//删除远程分支

git push origin --delete branch1

git branch -dr branch1

//将当前分支push到指定远程分支

git push origin HEAD:branch1

**tag** 标签

//列出所有tag

git tag

//新建一个tag在当前commit

git tag 1

//新建一个tag在指定commit

git tag 2 8989920311bacb3f4e3ced7f82ab75ca47c318c7

//删除本地tag

git tag -d 1

//查看tag信息

git show 2

//提交所有tag

git push --tags

//删除远程tag

git push origin --delete tag 2

//新建一个分支，指向某个tag

git checkout -b [branch] [tag]

**remote** 远程

// 查看远程库的地址列表

git remote -v

//查看这个远程库的信息

git remote show origin

// 从远程库更新所有的信息到本地，但是不合并

git fetch origin

//从远程库更新所有的信息到本地，但是不合并并清理已删除的远程分支

git fetch -p origin

//从远程库更新数据并立即合并数据

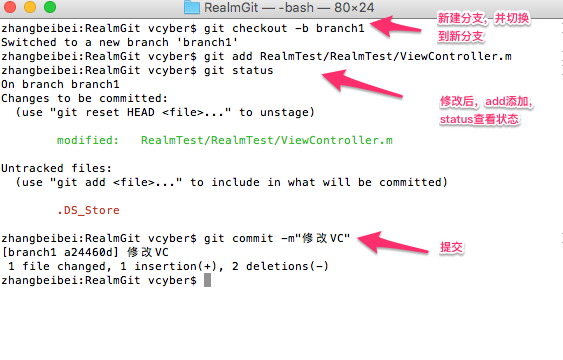
git pull origin branch1

//将本地数据同步到远程库中

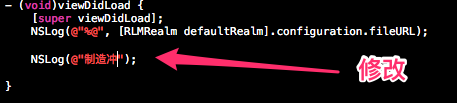
git push origin branch1

这里我在演示一下git解决冲突，演示流程：

1. 制造冲突  
   1.1 新建分支,并切换到分支， 打开其中的一个文件修改， 保存提交

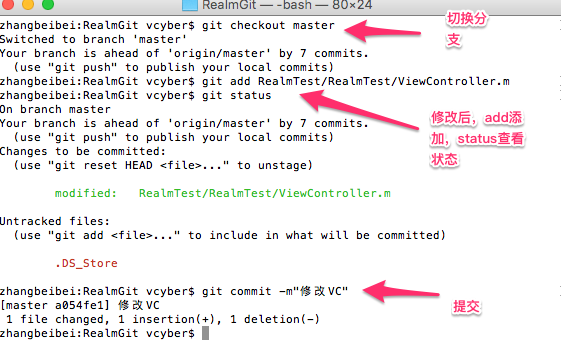


制造冲突1

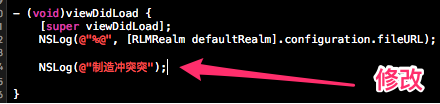


修改

1.2 切换到master分支，打开同一文件，在同一位置修改，保存提交

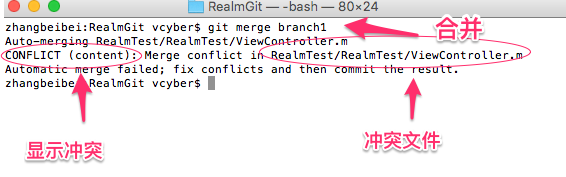


制造冲突



修改

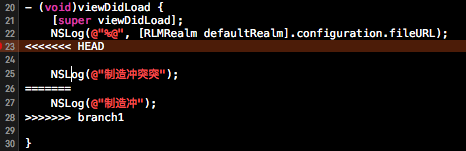
1.3 merge分支



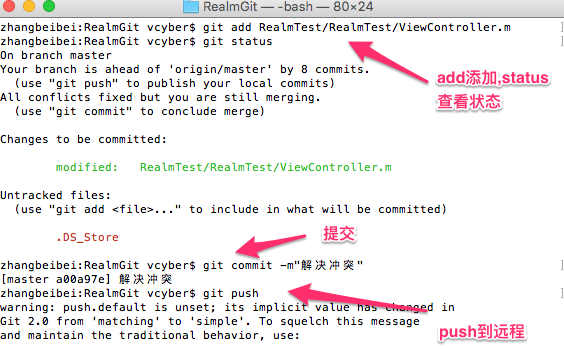
制造冲突

其实还可以通过status来查看冲突文件.

1. 解决冲突(手动解决)  
   打开冲突文件，删除<<<<<<< HEAD ======= >>>>>>> branch1这三行  
   再修改成想要的，添加，提交即可。



解决冲突



解决冲突

有兴趣的小伙伴可以来一波喜欢和关注，有什么错误，欢迎指正。。。