Git常用命令

1，mkdir demo 【创建文件夹】

2，cd demo【进入文件夹】

3，pwd【显示当前文件夹目录——》/d/workpage/git】

4，git init【初始化当前目录为git管理仓库】

5，ls -ah或者 ls -a【查看当前目录下所有文件（包括隐藏文件）】

6，cmd—>copy con demo.txt回车——>输入内容——>ctrl+Z回车——>生成TXT文件

7，仓库添加文件2步走：

1. git add demo.txt【添加文件】

git add demo.txt demo2.txt demo3.txt【添加多个文件】

（2）git commit -m “注释描述”【提交文件】

8，git status【查看当前文件状态，是否被修改，git add之后查看，然后commit文件】OR git status -s【简短的结果输出】

9，git diff demo.txt【产看修改内容】

10，git log【查看提交版本】

11，git log --pretty=oneline【以版本号的形式查看当前提交版本】——git log --oneline【查看简写的提交记录】

12，git reset --hard HEAD^【返回上一版本】

13，cat demo.txt【查看文件具体内容】

14，git reset --hard 版本ID【返回指定版本】

15，git reflog【查看每次提交后的版本ID】

16，撤销（未git add）：git checkout -- demo.txt【返回到最近一次提交的版本，撤销修改】

17，撤销（已git add）：git reset HEAD demo.txt——>git checkout -- demo.txt

18，rm demo.txt——>git rm demo.txt——>git commit -m “remove”【确实要删除】OR git rm --cached【如果要在工作目录中留着该文件】

19，rm demo.txt——>git checkout -- test.txt【误删可返回】

20，ssh-keygen -t rsa -C “email@qq.com”【创建SSH KEY】

21，本地与github仓库关联：git remote rm origin——>git remote add origin [git@github.com:xxxx(github帐号名)/xxxx(仓库名).git](mailto:git@github.com:xxxx(github帐号名)/xxxx(仓库名).git)

22，git push -u origin master【把本地库所有内容推送到远程库上】

23，git push origin master【推送最新提交】

24，git clone [git@github.com:xxxx/xxxx.git](mailto:git@github.com:xxxx/xxxx.git)【克隆远程库所有文件到本地】

25，ls【查看当前文件夹所有文件】

26，git checkout -b dev【创建并切换至dev分支】

27，git branch【查看当前分支】——》（\*dev master）

28，git checkout master【切换至master分支】

29，git merge dev【把dev分支结果合并到master分支上】

30，git branch -d dev【删除dev分支】

31，git branch dev【创建dev分支】

32，git log --graph --pretty=oneline --abbrev-commit【查看分支合并情况（git log --graph）

33，git merge --no-ff -m “no-ff” dev【合并分支】

34，git stash【储存临时未完成工作】——》git status【查看工作区】——》git checkout master【更换分支（工作区）】——》git checkout -b issue-101【建立bug】——》git merge --no-ff -m “new” issue-101【合并分支】——》git branch -d issue-101【删除issue-101分支】——》git checkout dev【进入dev分支】——》git stash list【查看临时存储区】

35，git stash apply【恢复临时区，存储内容不删除】

36，git stash drop【删除存储内容】

37，git stash pop=35+36【恢复临时区，同时删除内容】——》git stash list【再次查看临时区，无stash内容】

38，git stash apply stash@{index}【恢复指定index的stash区】

39，git branch -D dev【强行删除未合并的分支】

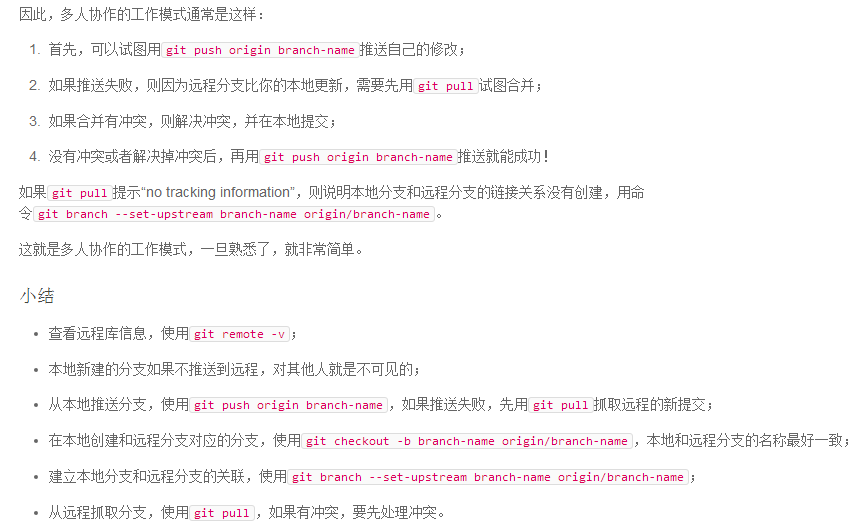
40，git remote【查看当前远程仓库的默认名称，一般：origin】

41，git remote -v【查看更详细的远程仓库信息】（master是主分支，时刻与远程同步，dev是开发分支，bug分支只用于本地修复bug，不同步）

42，git push origin dev【推送至dev分支上】

43，git push origin dev【推送分支到远程仓库】

44，git branch --set-uostream dev origin/dev【设置dev和origin/dev的链接】——》git pull——》git commit——》git push origin dev



45，git tag v0.1【为当期版本添加一个新标签】——》git tag【查看所有标签】——》git log --pretty=oneline --abbrev-commit——》git tag v0.1 ID

55，git show v0.2【显示指定版本详细信息】

56，git tag -a v0.3 -m “version” ID【为指定标签添加文字说明】

57，git tag -s v0.3 -m “self version” ID【通过-s用私钥签名一个标签】

58，git tag -d v0.1【删除指定标签，未push至远程库】

59，git push origin v0.1【推送标签至远程库】

60，git push origin --tag【一次性推送全部本地标签】

61，git tag -d v0.2——》git push origin ：refs/tags/v0.2【删除已推送至远程库的标签】

62，git config --global color.ui true【让git显示不同的颜色】

63，git add App.class【添加.xxx被忽略文件】

64，git add -f App.class【强制用-f添加到Git】

65，git check-ignore -v App.class【检查】

66，git branch -r -d origin/branch-name——》git push origin :branch-name【删除远程分支】

67，git push origin :br-1.0.0【删除远程版本】

68，git branch -a【查看远程分支】

69，git blame demo.js【查看代码修改负责人】

70，git blame -L 1,3 demo.js【查看第一行到第三行修改情况】

71，git blame -L 10,+5 demo.js【查看代码第十行（包括10）总共5行修改情况】

72，git log -p -- demo.js【查看提交历史记录】

73，cat demo.js——》git log -S[demo.js的代码内容] --oneline【查看什么时候引入的（参数S）】——》git log -G[demo.js的代码内容] --oneline【查看什么时候改变的（参数G）】

74，git grep [字符串（具体代码内容）] 【查找指定内容的位置】

75，git grep -n drawImage 【查找drawImage所出现的具体位置和行号】——（参数：[-n出现的行号，--count出现的次数]）

76，git grep -p drawImage \*.js【哪些函数方法包含这个字符串，（\*.js）并限定为js文件】

77，git grep -e ‘name’——》git grep -e ‘name’ --or -e‘age’【正则匹配字符串为name或者age所有出现的位置】

78，git bisect start——》git bisect bad——》git bisect good 75cas58——》git bisect bad——》（直到找到没有问题的版本，才标记为good）git bisect good——》git bisect reset（结束查找）【查找问题引入版本】

79，git fetch 【获取远程仓库所有分支和数据】

Github常用操作

1，Fork【克隆当前项目到自己帐号】——》git clone [git@github.com:xxxx/xxxx.git](mailto:git@github.com:xxxx/xxxx.git)【从自己帐号下clone改项目】

2，

$ git show 12a86bc38 # 根据版本

$ git show v1.0.1 # 根据标签

$ git show feature132 # 根据分支名

$ git show 12a86bc38^ # 一次提交的父节点

$ git show 12a86bc38~2 # 一次提交的祖父节点

$ git show feature132@{yesterday} # 时间相关

$ git show feature132@{2.hours.ago} # 时间相关

Git常规报错

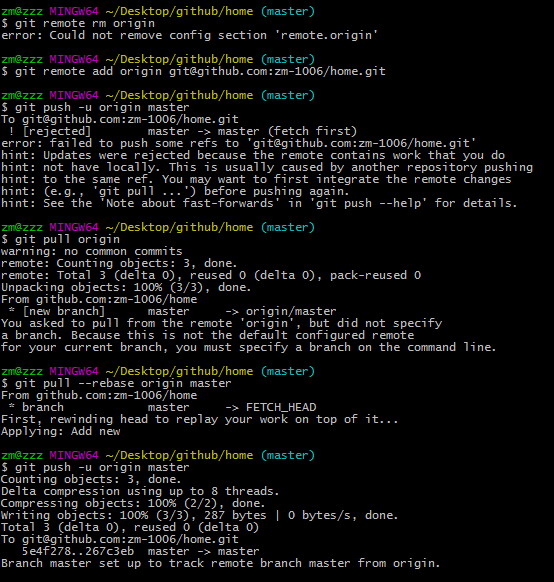
本地库与远程库关联：



——》git init——》git status——》git add “”——》git commit -m “”



——》



——》over【关联，推送成功】

2、error: failed to push some refs to 'git@github.com:yangchao0718/cocos2d.git