http://blog.163.com/zhulp0372@yeah/blog/static/115894479201241545917697/

**关于git的操作命令**

2012-05-15 16:59:17|  分类： [技术篇](http://blog.163.com/zhulp0372@yeah/blog/#m=0&t=1&c=fks_087071082080088074087082083095085087084075093082082068093) |  标签：[git(2)](http://blog.163.com/zhulp0372@yeah/blog/#m=0&t=3&c=git(2))  |字号 订阅

1）赋权限

进入目录 cd .ssh/

命令：  ssh-keygen

获取key:  vim id\_rsa.pub

2）

安装   apt-get install git-core

获得   git clone git://url

访问   http://url/gitweb

3）纳入版本控制:

git add \*.txt     //添加指定文件

git add README    //添加单个文件

git add .         //添加所有文件包括子目录，但不包括空目录

4）提交：

git commit -m “no1”             //全部提交

git commit -m “no1” someFile    //提交指定文件

git commit -C HEAD -a —amend    //复用HEAD留言，增补提交（修改小错误，而不增加提交记录，掩盖自己的小马虎）

-m “提交的说明”

-a 动把所有已经跟踪过的文件暂存,并提交.(工作目录中修改过的文件都提交到版本库，不需一个一个手动add了)

—amend 增补提交

-C 复用指定提交的提交留言

-c 打开编辑器在已有的提交基础上编辑修改

5）查看提交历史:

git log

这时“j”向下浏览，“k”向上浏览，“q”退出

6）问责：查明谁修改了代码

git blame hello.html            //查看修改过文件的人

git blame -L 12,+10 hello.html //12到22行  用"-L"参数在命令(blame)中指定开始和结束行:

blame还可以跟踪内容复制，文件复制

7）撤销缓存区的修改（没有commit的）

git checkout head 文件名 //撤销暂存区的修改

git checkout head readme.txt todo.txt

git checkout head \*.txt

git checkout head . //撤销所有

8）反转提交：

git revert HEAD //创建一个反向的新提交抵消原来的提交改动

如果需要反转多个，必须从最后的开始反转, 加 -n可以不马上提交，之后一起提交。

git revert -n HEAD

git revert -n 54efhds

git commit -m “revert head and 54efhds”

9）复位：还没有commit，让工作目录回到上次提交时的状态

git reset —hard HEAD //所有未提交的内容清空，这会让"git diff" 和"git diff —cached"命令的显示法都变为空

git reset —soft HEAD //复位版本库，暂存差异，便于提交中发现错误需要更改时有用（例如私人密码放到里边了）

10）分支：

在当前分支末梢建立分支：

git branch RB\_1.0（建立分支不会自动切换过去）

切换分支：

git checkout RB\_1.0（切换到RB\_1.0分支）

创建并切换分支：

git checkout -b RB\_1.0（简化上边2步操作）

删除分支：

git branch -d RB\_1.0

基于某次提交、分支或标签创建新分支:

git branch RB\_1.0 master

git branch RB\_1.0 6fe57de0

git branch Rb\_1.01 1.0

查看分支：

git branch -r //显示远程分支

git branch -a //列出所有分支

分支重命名：

git branch -m master mymaster

-M 大写M会覆盖同名的分支

合并分支：

直接合并：

git merge 想合并到当前分支的源分支名

git merge —no-commit 分支 //合并但不提交

压合合并：将分支压合成一条commit记录，并合并过来

git merge —squash 某bug分支

git commit -m “修复某bug”

拣选合并：只合并一个提交

git cherry-pick 321d76f

如果需要连续拣选，就需要加 -n参数

然后再git commit ，但不要加-m参数，编辑器就会使用刚拣选的提交留言作为现在的留言。

11）标签Tag:

查看标签:

git tag

创建标签：

git tag 1.0 //在当前分支最后一次提交创建标签

git tag 1.0 RB\_1.0 //基于RB\_1.0分支的最新踢脚创建标签

git tag 1.0 ae468d8kt //为某次提交创建标签

检出标签：

git checkout 1.0 //检出标签与检出分支一样操作，但检出标签后用git branch查看本地分支会发现你现在不再任何分支上

这时你不应该修改，而应该立即基于此标签创建一个分支

git checkout -b from-1.0

12）变基：

1）git rebase RB\_1.01 //也许修改过一个bug，希望新版本变基到RB\_1.01分支上

2）手动解决冲突 //如果解决不了直接git rebase -skip或-abort来跳过特定提交或完全放弃变基

3）git add xxx.html //冲突解决

4）git rebase —continue

//—onto参数可以改写历史抹掉中间的参数，将倒数第一个参数变基到倒数第3个参数，为防止出错建议在试验性分支上先试验。

rebase -i 可以排序历史记录，多个提交合并为1个，一个提交分解成多个提交 ，

详见版本控制之道p86 ，需要编辑器支持，windows记事本不行

13）远程相关：

git clone git://github.com/schacon/grit.git //从现有仓库克隆

git clone git://github.com/schacon/grit.git mygrit //换名,唯一区别就是新建的目录成了mygrit,其他都一样

添加远程仓库:

git remote add pb git://github.com/paulboone/ticgit.git

clone会默认添加origin仓库，如果原本用git init创建的版本库，后来又想提交到远程版本库，就可以用下边的办法

git remote add origin git@example.com:/xxxxxx

查看远程分支：

git remote -v //查看远程仓库,默认clone后,应该有一个origin仓库,-v显示对应的clone地址

git remote show origin //查看远程仓库信息

远程仓库重命名和删除:

git remote rename pb paul

git remote rm paul

获取数据：

现在pb/master可以在本地访问了,你可以合并到自己的某个分支,或者切换到这个分支看看有什么有趣的更新

git pull 抓取数据合并到工作目录中当前分支

推送数据：

git push [remote-name] [branch-name] //默认为 git push origin master

git push origin serverfix //推送分支，其实是下边一句的简化,提取我的 serverfix 并更新到远程仓库的 serverfix

git push origin serverfix:serferfix

git push origin :serverfix //这个语法用于删除,只要把分号前留空

14）其他：

git gc //垃圾回收，每隔一段时间例如一个月运行一次可以减少磁盘占用空间。

git reflog //最后的保障,列出误删的东东

git bisect //二分查找，版本控制之道p124页，略

归档版本库，导出压缩包:

git archive —format=格式 —prefix=目录/ 版本>压缩包.zip

git archive —format=zip head>test.zip

git archive —format=tar —prefix=mysite-1.0/ 1.0 | gzip>mysite-1.0.tar.gz

git archive —format=zip —prefix=mysite-1.0/ 1.0 >mysie-1.0.zip

通过git pull更新仓库，使用git init-db初始化自己的仓库。

commit:

git commit -a -e        提交全部修改文件，并调用vim编辑提交日志。

git reset HEAD^ or

git reset HEAD~1        撤销最后一次提交。

git reset --hard HEAD^  撤销最后一次提交并清除本地修改。

git reset SHA1          回到SHA1对应的提交状态。

add/delete/ls:

git add -a              添加所有文件。除了.gitignore文件中的文件。

git rm file             从git仓库中删除文件。

git commit              添加或是删除后要提交。

git ls-files -m         显示修改过的文件。

git ls-files            显示所有仓库中的文件。

這些事情都可以先在本地開 local branch 做，而不需要立即 Push 分享給別人。

git branch name       建立本地 local branch

git branch -m old\_name new\_name    改名字 (如果有同名會失敗，改用 -M 可以強制覆蓋)

git branch           列出目前有那些 branch 以及目前在那個 branch

git checkout name      切換 branch (注意到如果你有檔案修改了卻還沒 commit，會不能切換 branch，解法稍後會談)

git checkout -b name       本地建立 branch 並立即 checkout 切換過去

git branch -d  name       刪除 local branch

git merge name     合併另一個 branch，若沒有 conflict 衝突會直接 commit。若需要解決衝突則會再多一個 commit。

git merge --squash <branch\_name> 將另一個 branch 的 commit 合併為一筆，特別適合需要做實驗的 fixes bug 或 new feature，最後只留結果。合併完不會幫你先 commit。

git cherry-pick 321d76f 只合併特定其中一個 commit。如果要合併多個，可以加上 -n 指令就不會先幫你 commit，這樣可以多 pick幾個要合併的 commit，最後再 git commit 即可。

查看 分支版本

git branch -a

除了master之外，我们还可以随便创建分支，然后push到服务器上去。

$ git add .

$ git commit -m ""

$ git pull origin camp

$: git push origin camp

远程分支和本地分支需要区分，所以，在从服务器上拉取特定分支的时候，需要指定本地分支名字。

$: git branch product origin/product

更新分支到本地

$: git pull origin camp