# Git 常用指令

## 初始化 init

首先需要创建一个repo，这是cd到文件夹底下，然后init。

***$ git init***

## 添加文件 add

在文件系统里面的文件如果不添加到git的repo里面，不会自动被git辨识，需要手动添加，这也是初始化repo是必须要做的事。

***$ git add Documentation/\\*.txt***

***$ git add git-\*.sh***

## 删除文件 rm

删除文件，分为从git cached里面删除（即从index tree里面删除），何在文件系统中删除，如果不加—cached，就是直接从文件系统里面彻底删除。

***$ git rm [-f | --force] [-r] [--cached] <file>***

***-f 强制删除。***

***-r 针对文件夹的，递归删除。***

***--cached 仅仅从index tree里面删除，不把他从文件系统里面删除。***

***<file> 是file的pattern。***

## 重命名 mv

***$ git mv [-f] <source> <destination>***

***$ git mv [-f] <source> ... <destination directory>***

## 回滚/撤销 reset/revert

Reset是回退到某一版本，然后删除比他新的所有版本。Revert把这一版本的行为撤销相当于ctrl+z但是是那个版本的行为，不是最近的，作为一个新的commit。

***$ git reset [ –hard] [hashcode]回到前一个保存的状态然后删除之后的所有修改。***

***$ git revert [hashcode]撤销每一步的行为，作为新的commit***

## 分支 checkout/branch

分支的作用在git里面非常重要，是和svn的一个比较大的区别。当需要写一些脏代码的时候，我们创建一个新的分支，写代码、测试，然后我们回到原来的分支，所有的一切都消失了。另外，我们还有merge等命令，让两个分支可以合并。主要的使用方法如下所示。

***$ git checkout HEAD~3：回退到三个版本之前的分支。***

***$ git checkout –b branchName [hashcode]：创建并进入分支,在hashcode的基础上创建分支。***

***$ git checkout master：回到master分支。***

***$ git merge desBranch：把目标branch合并到当前branch。***

***$ git branch –d rmBranch：删除分支。***

***$git push origin –delete b1：删除远程分支。***

## 上传和更新 push/pull

Push用于把修改提交到remote上（github），类似checkin。Pull吧版本更新到remote一样新（在开始工作之前干这个事情相当于update）。

***$ git push origin master1:master2： origin是站点名， master1是本地分支名，master2是远程分支名。***

***$ git pull origin master：直接更新***

另外fetch很好用，首先我们把remote上的版本下载到某一个分支上，我们先查看分支的diff，然后merge。Fetch和pull的区别就是pull直接把remote的版本merge到本地了。示例如下：

***$ git fetch origin master:tmp***

***$ git diff tmp***

***$ git merge tmp***

## 标签tag

在程序开发到一定阶段时需要创建tag(v1.0.0)，用于release。

***$ git tag –a “v1.0.0” –m “first release”：创建tag，a表示annotation***

***$ git tag –d “tag name”：删除本地tag。***

***$ git push origin –tag master：吧本地所有tag上传。***

***$ git push origin –delete “tagName”：删除远程tag。***

## 查看更新 diff

这个可以用来查看两个版本之间的差异，可以直接导出patch。

***$ git diff [<commit>] <commit> [<path>…]：比较两个版本***

***如果需要patch：> patch就ok了***

## 忽略文件 .gitignore

用于忽略不想加入git repo中的文件，这个命令在类似eclipse project的时候很有用处，可以把bin文件夹加入到gitignore文件当中，这样可以直接add –a文件而不会受影响。注意这个文件要起作用必须先add到repo里面，要是没有加入会导致不起作用，一起添加入repo。

***$ cd /home/user/doc***

***$ vim .gitignore***

***$ git add .gitignore***

***$ git commit***

另外，文件的格式是每一行是一中ignore pattern。例如我不想添加docx的临时文件可以用下面的pattern表示。这在本memo里面有示例。

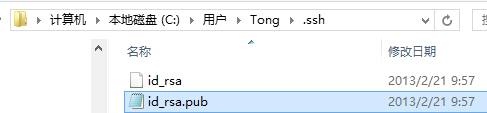
***~$\****

# Git Hub 配置：

## 生成SSH Key

***$ ssh-keygen -t rsa -C tong.wu.stap@gmail.com***

然后会有说存在那个文件里面，直接回车。输密码。下面这个文件里面存的是ssh key

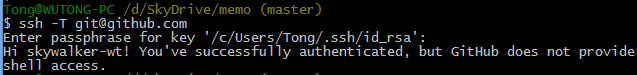


## 在github上设置ssh key

复制ssh key到github的account settings -> SSH Keys里

## 测试配置是否成功

***$ ssh -T git@github.com***



这样的回复是成功的。

## 设置自己的邮件地址和用户名

***$ git config --global user.name "your name"***

***$ git config --global user.email "your\_email@youremail.com"***

这里可以设置—global 为local这里需要继续了解下区别。

## 建立上传repo，添加到remote站点

这里首先要在github上面建立一个repo例如<https://github.com/skywalker-wt/test.git>

***$ git remote add origin*** [git@github.com:skywalker-wt/test.git](mailto:git@github.com:skywalker-wt/test.git)

这里origin 是站点的名字，可以有多个站点，添加完成之后在.git文件夹里面config文件里面可以找到相关的配置。也可以直接通过修改这个配置文件来修改remote站点的修改。

## 简单的上传

***$ git push origin master***

关键就是这个push，把当前的commit提交到origin站点的master branch。