git使用总结

在本地进行版本修改前，先建立一个分支，如dev。可用界面，或者git checkout -b dev。然后进行编辑，这里可以在两个分支之间切换来比较前后的效果。修改完成后切换到master下，git merge dev将修改的内容合并进来。然后git branch –d dev将dev分支删除。然后git push origin master上传，或者用Sync。如果在master中也commit过一个的话，则merge的时候修改的文件会有冲突，需要手动修改成想要的然后再commit一次。

但是上述提到的切换branch来比较前后工作区的方法必须是要commit以后清空工作区的情况下才能进行。不然会提示说文件被覆盖。此时若另有任务要立即清空工作区的话，可以使用git stash，完成另外的任务后再git stash pop取回之前保存的状态，这样就不必在工作未完成时commit了

更多参考<http://www.liaoxuefeng.com/wiki/0013739516305929606dd18361248578c67b8067c8c017b000>

在本地建立repository后，改动并commit后，用界面上的Sync会将本地的同步到服务器去。

在本地用git reset --hard [id] 可以回滚，但是回滚之后建立的文件，再在本地commit后Sync后，会是在本地最近的一个结点的基础上加上建立的文件，并在服务器上形成一个新的commit。如果建立的文件是之前服务器上最新结点已经有的，则最新的commit不会同步到服务器，如果新建的文件在服务器的最新结点上本来就有，且内容改了，则提示有冲突。如果不改文件直接Sync的话则回滚到服务器端最新结点。

在本地用git reset --hard [id]回滚后，再用git push origin head --force后，则将本地到目前结点为止的树同步到远端服务器，目前结点以后的结点将从服务器上删除（不显示了，不知道是彻底删除了还是能找回）。此时本地仍可以使用git reset --hard [id]滚动到，在界面上那些结点显示就是没有同步到服务器上的圈（服务器上已有的是点）。此时用Sync的话，则会将服务器上的最近结点同步到本地（实为下载，与之前的上传不同），若改了文件并commit的话，也只是在服务器端的基础上再加一个commit。此时要再改变服务器上的树只能再用git push origin head --force。如果要将本地的之后的结点彻底删除，则将整个库删了，再从服务器clone下来即可。

git push origin :分支名 删除远程分支

# 同步一个 fork

搜索 fork sync，就可以看到 GitHub 自己的帮助文档 [Syncing a fork](https://help.github.com/articles/syncing-a-fork/) 点进去看这篇的时候，注意到有一个 Tip: Before you can sync your fork with an upstream repository, you must [configure a remote that points to the upstream repository](https://help.github.com/articles/configuring-a-remote-for-a-fork/) in Git.   
根据这两篇文章，问题迎刃而解！

## 具体方法

### Configuring a remote for a fork

* 给 fork 配置一个 remote
* 主要使用 git remote -v 查看远程状态。

git remote -v

# origin https://github.com/YOUR\_USERNAME/YOUR\_FORK.git (fetch)

# origin https://github.com/YOUR\_USERNAME/YOUR\_FORK.git (push)

* 添加一个将被同步给 fork 远程的上游仓库

git remote add upstream https://github.com/ORIGINAL\_OWNER/ORIGINAL\_REPOSITORY.git

* 再次查看状态确认是否配置成功。

git remote -v

# origin https://github.com/YOUR\_USERNAME/YOUR\_FORK.git (fetch)

# origin https://github.com/YOUR\_USERNAME/YOUR\_FORK.git (push)

# upstream https://github.com/ORIGINAL\_OWNER/ORIGINAL\_REPOSITORY.git (fetch)

# upstream https://github.com/ORIGINAL\_OWNER/ORIGINAL\_REPOSITORY.git (push)

### Syncing a fork

* 从上游仓库 fetch 分支和提交点，传送到本地，并会被存储在一个本地分支 upstream/master   
  git fetch upstream

git fetch upstream

# remote: Counting objects: 75, done.

# remote: Compressing objects: 100% (53/53), done.

# remote: Total 62 (delta 27), reused 44 (delta 9)

# Unpacking objects: 100% (62/62), done.

# From https://github.com/ORIGINAL\_OWNER/ORIGINAL\_REPOSITORY

# \* [new branch] master -> upstream/master

* 切换到本地主分支(如果不在的话)   
  git checkout master

git checkout master

# Switched to branch 'master'

* 把 upstream/master 分支合并到本地 master 上，这样就完成了同步，并且不会丢掉本地修改的内容。   
  git merge upstream/master

git merge upstream/master

# Updating a422352..5fdff0f

# Fast-forward

# README | 9 -------

# README.md | 7 ++++++

# 2 files changed, 7 insertions(+), 9 deletions(-)

# delete mode 100644 README

# create mode 100644 README.md

* 如果想更新到 GitHub 的 fork 上，直接 git push origin master 就好了。

其实fork来的repository clone到本地之后，会自动将stream的git添加到remote里，这时要合并最新的stream的更新的话只要执行后面说的步骤就好了，而后面的步骤可以在界面是点Update from。。。来完成。

用git merge somebranch是把somebranch的所有分支后的commit全都加到master里，而使用界面上的compare之后的update，则是创建一个新的commit，内容是合并分支后的状态。

如果git fetch upstream无法得到正确的stream/master的话，就git branch –d stream/master然后再fetch。但不知为什么再fetch不行，而在界面上再选一次stream/master则是可以的