GitHub工作流

1. 单人任务

在本地新建一个repository，初始就有名为master的branch。然后新建一个分支，不妨取名dev。则代码的版本更新在dev下进行。完成一次小改之后commit一次，在shell里在master下使用使用git merge dev将dev的commit合并到master中，不要使用界面上的update master from dev。这样将产生一个Merge的commit，并无必要。在适当的时候将master上传到github备份即可，也可完全在本地使用而不上传。

1. 多人任务，使用pull request方式

项目主持人(不妨设为甲，jia)建立一个repository并上传，参与者。fork一个。然后clone到本地，此时会有一个自己的master分支，同时有一个jia/master。协作者在自己的master上修改，完成后发起pull request。从自己的master到主持人的master或者dev等（依项目是否需要测试版和稳定版两版而定）。然后主持人请阅后通过pull。

协作者在进行修改前要使自己fork的repository与项目主持人的一致，切换到jia/master，然后git pull (大约等于先git fetch再git merge)获取最新的版本，然后切换到master，然后git merge jia/master，此时本地即更新，再git push origin master将服务器端也更新。客户端工具中的update在此任务中感觉用着有些问题，不如直接用命令行。

以上说明中有只能在命令行中操作的已经给出命令，如切换分支等，可在命令行中用git checkout操作，也可在客户端工作中操作。

客户端中的update会产生Merge...的额外的commit，似乎没必要。不如命令行的git merge，revert也是，不如命令行的git reset –hard

补充

在用git merge dev时，会把dev上所有从master分支后的commit全部都merge上去。此时可以用git merge –squash dev只把文件更新，但是不在master上生成commit，此时可以手动commit一次。

此时如果再用一次git merge dev则会把dev上以前没有merge的commit全部merge上去。

如果要让本地的master保持与远端一致，可以先git branch –d master将本地的删除，然后再git pull origin master将远端的下载下来。