**Git**

安装git

1. 设置用户名
2. 创建版本库
3. 添加到版本库：

1).建立一个文本文件，不能用windows自带的文本编辑器；

2).将文件添加进仓库 add readme.txt

3).将文件提交进仓库:git commit -m “wrote a reame file”(因为git一次提交多个文件，所以，采用添加和提交两步)

5.git status命令可以让我们时刻掌握仓库当前的状态，上面的命令告诉我们，readme.txt被修改过了，但还没有准备提交的修改。

虽然Git告诉我们readme.txt被修改了，但如果能看看具体修改了什么内容，自然是很好的。比如你休假两周从国外回来，第一天上班时，已经记不清上次怎么修改的readme.txt，所以，需要用git diff这个命令看看：

提交修改和提交新文件是一样的两步

要随时掌握工作区的状态，使用git status命令。

如果git status告诉你有文件被修改过，用git diff可以查看修改内容。

1. git log / git log pretty=oneline查看历史记录 ：回到过去
2. git reset --hard HEAD^ 回溯到上一个版本
3. git reflog:记录每一个命令 ：重返未来
4. git reset --hard “commitid” 回溯到固定版本(id写前几位即可)
5. 添加licence文件
6. git管理的是修改，每次修改要先add到暂存区再commit才能被git跟踪；
7. Git checkout -- readme.txt:撤销未add到stage的修改
8. git reset HEAD readme.txt:撤销add到stage的修改，然后再用12撤销；

又到了小结时间。

(12.13和之前版本回溯的区别：并没有提交就发现有问题，想回到上一次编辑的内容)

场景1：当你改乱了工作区某个文件的内容，想直接丢弃工作区的修改时，用命令git checkout -- file。

场景2：当你不但改乱了工作区某个文件的内容，还添加到了暂存区时，想丢弃修改，分两步，第一步用命令git reset HEAD file，就回到了场景1，第二步按场景1操作。

场景3：已经提交了不合适的修改到版本库时，想要撤销本次提交，参考版本回退一节，不过前提是没有推送到远程库。

删除文件：

1.提交删除的修改

git rm test.txt

git commit -m ”remove test.txt”

1. 撤回删除:

git checkout -- test.txt（事实上checkout就是将工作区的文件恢复到版本库里的版本）

GitHub：

要关联一个远程库，使用命令git remote add origin git@server-name:path/repo-name.git；

关联后，使用命令git push -u origin master第一次推送master分支的所有内容；

此后，每次本地提交后，只要有必要，就可以使用命令git push origin master推送最新修改；

分布式版本系统的最大好处之一是在本地工作完全不需要考虑远程库的存在，也就是有没有联网都可以正常工作，而SVN在没有联网的时候是拒绝干活的！当有网络的时候，再把本地提交推送一下就完成了同步，真是太方便了！

Git鼓励大量使用分支：

查看分支：git branch

创建分支：git branch <name>

切换分支：git checkout <name>

创建+切换分支：git checkout -b <name>

合并某分支到当前分支：git merge <name>

删除分支：git branch -d <name>

当Git无法自动合并分支时，就必须首先解决冲突。解决冲突后，再提交，合并完成。

非fast forward 模式合并，保留分支信息

Git checkout -b dev

准备合并dev分支，请注意--no-ff参数，表示禁用Fast forward：

$ git merge --no-ff -m "merge with no-ff" dev

Merge made by the 'recursive' strategy.

readme.txt | 1 +

1 file changed, 1 insertion(+)

Bug

Feature

多人协作

小结

查看远程库信息，使用git remote -v；

本地新建的分支如果不推送到远程，对其他人就是不可见的；

从本地推送分支，使用git push origin branch-name，如果推送失败，先用git pull抓取远程的新提交；

在本地创建和远程分支对应的分支，使用git checkout -b branch-name origin/branch-name，本地和远程分支的名称最好一致；

建立本地分支和远程分支的关联，使用git branch --set-upstream branch-name origin/branch-name；

从远程抓取分支，使用git pull，如果有冲突，要先处理冲突。