**学习日志**

姓名： 陈雨舒 学号：1813065 日期： 2020.6.15

**学习内容安排：**

上午：

Git简介、安装Git、创建版本库、版本回退、工作区与暂存区、管理修改、撤销修改、删除文件

下午：

远程仓库、分支管理、标签管理、使用GitHub、忽略特殊文件、配置别名、使用SourceTree

**学习反馈：**

**一．已掌握知识：**

**Git的作用：**

(1)备份文件：可以应用在游戏的存档或者记录历史，比如文件的修改记录。

(2)多端共享：云服务，更利于团队协作，可以解决冲突。

**版本管理工具发展历史：**cvs(1985)-svn(2000)-git(2005)-github(2008)

**集中式和分布式的区别：**

分布式不需要联网，每个人的电脑就是完整的版本库，系统的安全性更高；

集中式必须联网才能工作，版本库集中存放在中央服务器

**Git安装：**

Git可以在Linux、Unix、Mac和Windows几大平台上正常运行。

Git可以安装在Linux,macOS,Solaris,Windows,Raspberry操作系统上。

**创建版本库：**

1. 版本库**repository**又名仓库，相当于目录，目录中所有的文件的修改、删除等操作都能被Git跟踪
2. 初始化一个Git仓库，使用git init命令。
3. 可以用ls –ah命令看见隐藏的.git目录
4. 添加文件到Git仓库共分为两步：

(1)使用命令git add <file>，可反复多次使用，添加多个文件；

(2)使用命令git commit -m <message>，完成。-m后面输入的是本次提交的说明，可以输入任意内容，最好是有意义的。

**版本回退：**

1. HEAD指向的版本就是当前版本，上一个版本是HEAD^，往上一百个版本是HEAD~100。因此，Git允许我们在版本的历史之间穿梭，使用命令git reset --hard commit\_id。
2. 穿梭前，用git log可以查看提交历史，以便确定要回退到哪个版本。

可以用加上--pretty=oneline参数使输出信息更简洁。

1. 要重返未来，用git reflog查看命令历史，以便确定要回到未来的哪个版本。

**工作区和暂存区：**

Git管理的文件分为：工作区，版本库，版本库又分为暂存区stage和暂存区分支master(仓库)

git add把文件从工作区移到暂存区，git commit把文件从暂存区移到仓库，

git diff查看工作区和暂存区差异，

git diff --cached查看暂存区和仓库差异，

git diff HEAD -- readme.txt命令可以查看工作区和版本库里面最新版本的区别

git add的反向命令git checkout，撤销工作区修改，即把暂存区最新版本转移到工作区，

git commit的反向命令git reset HEAD，拉取最近一次提交到版本库的文件到暂存区 并且该操作不影响工作区。

**管理修改和撤销修改：**

1. 每次修改，如果不用git add到暂存区，那就不会加入到commit中。
2. 命令git checkout -- readme.txt意思就是，把readme.txt文件在工作区的修改全部撤销，就是让这个文件回到最近一次git commit或git add时的状态。如果修改后还没有被放到暂存区，就撤销修改回到和版本库一模一样的状态；如果已经添加到暂存区后又作了修改，撤销修改就回到添加到暂存区后的状态。
3. 命令git reset HEAD <file>可以把暂存区的修改撤销掉（unstage），重新放回工作区。

**删除文件：**

1. 如果要删除test.txt文件，可以执行 git rm test.txt 、git commit -m "remove test.txt" 然后文件就被删掉了
2. 如果不应该删test.txt但是误删了，且只执行了rm test.txt，还没有提交，所以可以执行git checkout test.txt将文件恢复。

**远程仓库：**

1. 要关联一个远程库，使用命令git remote add origin [git@server-name:path/repo-name.git](mailto:git@server-name:path/repo-name.git)；
2. 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；
3. 每次本地提交后可以使用命令git push origin master推送最新修改；
4. 要克隆一个仓库，首先必须知道仓库的地址，然后使用git clone命令克隆。

Git支持多种协议，包括https，但ssh协议速度最快。

**分支管理：**

1. 查看分支：git branch
2. 创建分支：git branch <name>
3. 切换分支：git checkout <name>或者git switch <name>
4. 创建+切换分支：git checkout -b <name>或者git switch -c <name>
5. 合并某分支到当前分支：git merge <name>
6. 删除分支：git branch -d <name>
7. 解决冲突：

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。解决冲突就是把Git合并失败的文件手动编辑为我们希望的内容，再提交。

用git log --graph命令可以看到分支合并图。

1. 合并分支时，加上--no-ff参数就可以用普通模式合并，合并后的历史有分支，能看出来曾经做过合并，而fast forward合并就看不出来曾经做过合并。
2. git stash命令保存现场，之后可以创建新的分支修复Bug，修复后git stash pop，回到工作现场；
3. 开发一个新feature，最好新建一个分支；

如果要丢弃一个没有被合并过的分支，可以通过git branch -D <name>强行删除。

1. 查看远程库信息：git remote -v
2. 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；
3. 使用git push origin branch-name从本地推送分支，如果推送失败，先用git pull抓取远程的新提交；
4. 使用git checkout -b branch-name origin/branch-name在本地创建和远程分支对应的分支。
5. 从远程抓取分支，使用git pull，如果有冲突，要先处理冲突。
6. rebase操作可以把本地未push的分叉提交历史整理成直线；

**标签管理：**

1. 命令git tag <tagname>用于新建一个标签，默认为HEAD，也可以指定一个commit id；
2. 命令git tag -a <tagname> -m "message"可以指定标签信息，-a指定标签名，-m指定说明文字。
3. 命令git tag可以查看所有标签。
4. 命令git show <tagname>可以看到说明文字：
5. 命令git push origin <tagname>可以推送一个本地标签；
6. 命令git push origin --tags可以推送全部未推送过的本地标签；
7. 命令git tag -d <tagname>可以删除一个本地标签；
8. 命令git push origin :refs/tags/<tagname>可以删除一个远程标签。

**忽略特殊文件：**

忽略文件的原则是：

1. 忽略操作系统自动生成的文件，比如缩略图等；
2. 忽略编译生成的中间文件、可执行文件等，也就是如果一个文件是通过另一个文件自动生成的，那自动生成的文件就没必要放进版本库，比如Java编译产生的.class文件；
3. 忽略你自己的带有敏感信息的配置文件，比如存放口令的配置文件。

忽略某些文件时，需要编写.gitignore；.gitignore文件本身要放到版本库里。

**配置别名：**

命令git config --global alias.别名 配置别名

**使用SourceTree：**

可以以图形界面操作Git，省去了敲命令的过程，对于常用的提交、分支、推送等操作来说非常方便。

**二．学习心得：**

通过一天的学习，了解了许多Git的命令和操作方法，对Git的了解逐渐深入。学习了GitHub和SourceTree的使用方法，使学习Git更加便捷清晰。Git 是一个分布式管理工具，关心的是文件数据整体的变化，并不保存变化前后的差异数据；Git 可以备份文件和修改记录，十分有利于团队协作。学习好Git对接下来实训中的团队合作以及各种编程知识的学习都有较大帮助。