Git实验3、4、5实验报告

第三组： 王磊 陆青霞 董财辉 孙怡

一、实验目的

1、远程版本库与远程 Fork 版本库的合并、通过 GitHub 与 http://github.com/chenxhjeo 协

作。

2、掌握版本回退、管理修改、撤销修改、删除文件。

3、实现分支的创建、合并；学习解决分支冲突；了解分支管理策略；掌握与远程仓库的分支交互。

二、实验内容

1、通过 GitHub 协作

2、在工作区和暂存区中管理文件。

3、创建和合并分支

三、实验过程

通过 GitHub 协作：

1、远程版本库与远程 Fork 版本库的合并命令：

1）先克隆项目到本地：

mkdir clone

cd clone

Git init

Git clone https://github.com/自己的用户名/softwareEngineer

cd softwareEngineer

2）添加原作者项目的 remote 地址， 然后将代码 fetch 过来

git remote add sri https://github.com/chenxhjeo/softwareEngineer

git fetch sri //相当于是从远程获取最新版本到本地，不会自动 merge

‘sri’相当于一个别名

辅助命令：例如查看本地项目目录用命令 git remote -v

3）合并

git checkout master

git merge sri/master

如果有冲突的话，需要丢掉本地分支：

git reset –hard sri/master

4）提交到远程库

当你的当前本地的项目变成和原作者的主项目一样了，可以把它提交到你的GitHub 库，参考代码如下：

git commit -am ‘更新到原作者的主分支’

git push origin

git push -u origin master -f –强制提交

有时候提交出错，可以把 origin 先删除，然后再 push，参考代码如下：

git remote rm origin //origin 是别名，可以换成其他的名称，后续命令相应修改

git remote add origin https://username:password@github.com/username/test.git

git push origin master

2、通过 GitHub 与 http://github.com/chenxhjeo 协作

3、常用命令：

1）显示文本文件内容：cat readme.txt

2）编辑文本文件：vi readme.txt

Git 使用 4：

1、版本回退。

$ git add readme.txt

$ git commit -m "append GPL"

在实际工作中，版本控制系统有某个命令可以告诉我们历史记录，在 Git中，我们用 git log 命令查看：

$ git log

如果嫌输出信息太多，看得眼花缭乱的，可以试试加上--pretty=oneline 参数：

$ git log --pretty=oneline

当我们要把当前版本“append GPL”回退到上一个版本“add distributed”，就可以使用 git reset 命令：

$ git reset --hard HEAD^

看看 readme.txt 的内容是不是版本 add distributed：

$ cat readme.txt

还可以继续回退到上一个版本 wrote a readme file，不过且慢，然我们用 git log 再看看

现在版本库的状态：

$ git log

可以指定回到未来的某个版本：

$ git reset --hard 3628164

版本号没必要写全，前几位就可以了，Git 会自动去找。当然也不能只写前一两位，因

为 Git 可能会找到多个版本号，就无法确定是哪一个了。再小心翼翼地看看 readme.txt 的

内容：

$ cat readme.txt

Git 提供了一个命令 git reflog 用来记录你的每一次命令：

$ git reflog

2、管理修改。

第一步，对 readme.txt做一个修改，比如加一行内容：

$ cat readme.txt

然后，添加：

$ git add readme.txt

$ git status

然后，再修改 readme.txt：

$ cat readme.txt

提交：

$ git commit -m "git tracks changes"

提交后，再看看状态：

$ git status

提交后，用 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 diff HEAD -- readme.txt

3、撤销修改。

在 readme.txt 中添加了一行：

$ cat readme.txt

如果发现错误错误发现得很及时，就可以很容易地纠正它。你可以删掉最后一行，手动把文件恢复到上一个版本的状态。如果用 git status 查看一下：

$ git status

你可以发现，Git 会告诉你，git checkout -- file 可以丢弃工作区的修改：

$ git checkout -- readme.txt

现在，看看 readme.txt 的文件内容：

$ cat readme.txt

文件内容果然复原了。

git checkout -- file 命令中的--很重要，没有--，就变成了“切换到另一个分支”的命令，如果还 git add 到暂存区了：

$ cat readme.txt

庆幸的是，在 commit 之前，你发现了这个问题。用 git status 查看一下，修改只是添加到了暂存区，还没有提交：

$ git status

Git 同样告诉我们，用命令 git reset HEAD file 可以把暂存区的修改撤销掉（unstage），

重新放回工作区：

$ git reset HEAD readme.txt

再用 git status 查看一下，现在暂存区是干净的，工作区有修改：

$ git status

还记得如何丢弃工作区的修改吗？

$ git checkout -- readme.txt

$ git status

4、删除文件。

在 Git 中，删除也是一个修改操作，我们实战一下，先添加一个新文件 test.txt 到 Git并且提交：

$ git add test.txt

$ git commit -m "add test.txt"

一般情况下，你通常直接在文件管理器中把没用的文件删了，或者用 rm 命令删了：

$ rm test.txt

这个时候，Git 知道你删除了文件，因此，工作区和版本库就不一致了，git status 命令会立刻告诉你哪些文件被删除了：

$ git status

现在有两个选择，一是确实要从版本库中删除该文件，那就用命令 git rm 删掉，并且 git commit：

$ git rm test.txt

$ git commit -m "remove test.txt"

另一种情况是删错了，因为版本库里还有呢，所以可以很轻松地把误删的文件恢复到最新版本：

$ git checkout -- test.txt

创建和合并分支：

1、实现分支的创建、合并

（1） 首先，进入先前创建的本地仓库testgit目录中（若先前没有创建本地仓库，请从实验一开始再来一次），然后 git checkout –b 分支名 创建分支（分支名可随意定），切换到dev分支上。

（2） 用git branch命令查看自己当前目录下所有的分支，此时可查看到自己创建的分支，当前分支前面会添加一个“\*”

（3） 我们在dev分支上继续做demo，比如我们现在在readme.txt再增加一行7777777。首先我们先用cat 来查看下readme.txt 内容，接着添加内容7777777

（4） 现在dev分支工作已完成，现在我们用git checkout切换到主分支master上，继续查看readme.txt

（5） 现在我们可以把dev分支上的内容合并到分支master上了，可以在master分支上，使用如下命令 git merge dev

（6） 注意到上面的Fast-forward信息，Git告诉我们，这次合并是“快进模式”，也就是直接把master指向dev的当前提交，所以合并速度非常快。合并完成后，我们可以接着删除dev分支了

2、解决分支冲突

（1）下面我们还是一步一步来，先新建一个新分支，比如名字叫fenzhi，在readme.txt添加一行内容8888888，然后提交

（2）接着，我们现在切换到master分支上来，也在最后一行添加内容，内容为9999999，如下所示（和fenzhi分支的操作相同）

（3）现在我们需要在master分支上来合并fenzhi1

（4）修改冲突部分内容，重新提交（add ? commit）,最后可用git log查看分支合并情况。

3、分支管理策略

（1） 创建一个dev分支。

（2） 修改readme.txt内容。

（3） 添加到暂存区。

（4） 切换回主分支(master)。

（5）合并dev分支，使用命令 git merge --no-ff -m “注释” dev。

（6）查看历史记录。

4、将新分支推送到github以及删除github上的分支

5、从远程的分支获取最新的版本

Git中从远程的分支获取最新的版本到本地有这样2个命令：

1）git fetch：相当于是从远程获取最新版本到本地，不会自动merge

git fetch origin master

git log -p master..origin/master

git merge origin/master

上述过程可以用以下更清晰的方式来进行：

git fetch origin master:tmp

git diff tmp

git merge tmp

从远程获取最新的版本到本地的test分支上，之后再进行比较合并

2）git pull：相当于是从远程获取最新版本并merge到本地

git pull origin master

上述命令其实相当于git fetch 和 git merge

四、实验总结