想要更好的提交树，使用rebase操作会更好一点。

这样可以线性的看到每一次提交，并且没有增加提交节点。

rebase 操作，会中断rebase，同时会提示去解决冲突。

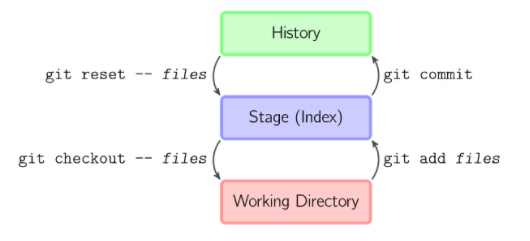
解决冲突后，将修改add后执行git rebase --continue继续操作，

或者git rebase --skip忽略冲突，或者git rebase –abort

git pull = git fetch + git merge FETCH\_HEAD

git pull --rebase = git fetch + git rebase FETCH\_HEAD

git 模型可以抽象为 远程仓库——remote， 本地三级仓库: level1——working directory level2——stage(index) level3——repository(History)



上面的四条命令在工作目录、暂存目录(也叫做索引)和仓库之间复制文件。

git add files 把当前文件放入暂存区域。

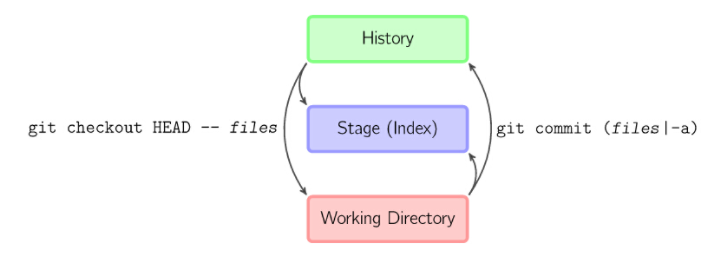
git commit 给暂存区域生成快照并提交。

git reset -- files 用来撤销最后一次git add files，你也可以用git reset 撤销所有暂存区域文件。（操作对象是HEAD）

git checkout -- files 把文件从暂存区域复制到工作目录，用来丢弃本地修改。（目的是working Directory）

你可以用 git reset -p, git checkout -p, or git add -p进入交互模式。

也可以跳过暂存区域直接从仓库取出文件或者直接提交代码， 如下



git commit -a 相当于运行 git add 把所有当前目录下的文件加入暂存区域再运行。git commit.

git commit files 进行一次包含最后一次提交加上工作目录中文件快照的提交。并且文件被添加到暂存区域。

git checkout HEAD -- files 回滚到复制最后一次提交。

注意：

1、git add files 的关键是将文件放入暂存区；不仅仅是新添加一个新文件到暂存区，修改现有的文件，使用git add也是将修改提交到暂存区。

2、git pull –rebase

在提交本地修改文件时，先执行git pull –rebase，将其他人的修改合并到本地，在git push自己修改的代码。

远程仓库的文件test.txt内容是

1111

2222

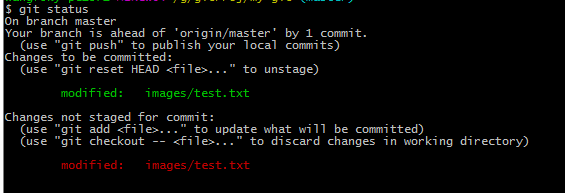
3333

4444

5555

6666

本地三级仓库文件test.txt内容是



history区，有一次提交，text.txt内容是

1111

2222

3333

4444

5555

7777

暂存区，有一次提交，text.txt内容是

1111

2222

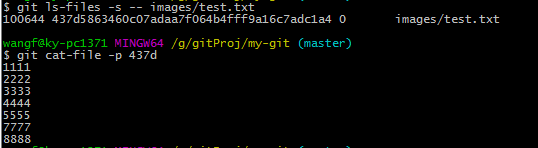
3333

4444

5555

7777

8888



工作区，有一次修改，text.txt内容是

1111

2222

3333

4444

5555

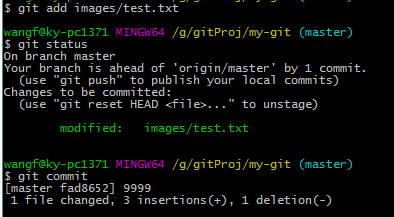
7777

8888

9999



可以看到如果工作区有修改、暂存区有提交记录，git pull –rebase会失败。

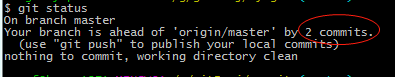


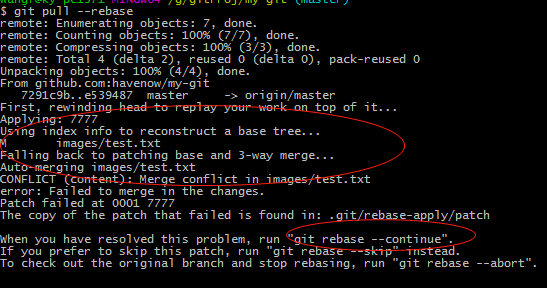
git commit时编辑提交说明

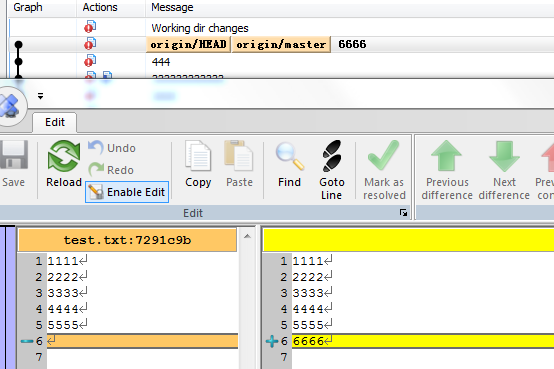
i进入编辑模式

Esc退出编辑模式

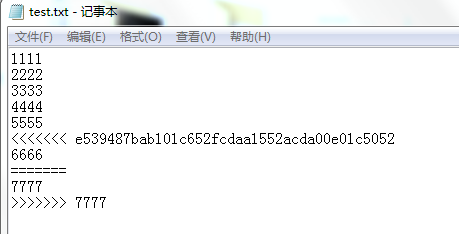
:wq保存退出





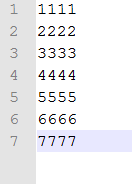


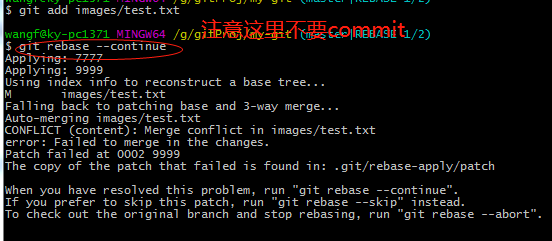
这时要修改工作区的images/test.txt文件

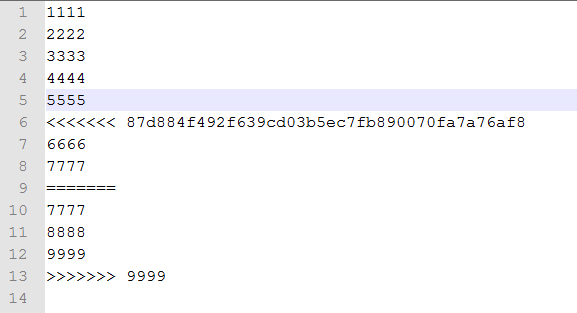


可以看到这是history的第一次提交和远程仓库合并时产生的冲突

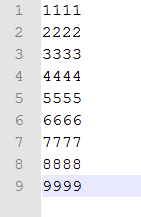
修改images/test.txt文件后

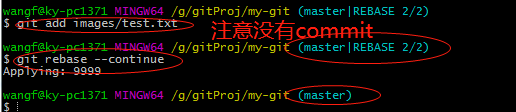






修改为





git pull –rebase后

解决冲突后，git add images/test.txt

接着git rebase –continue

继续解决冲突，git add images/test.txt

本地history区的两次提交都合并后，看到分支提示由(master|REBASE 2/2)变成了(master)

注意没有git commit操作。

最后git push origin master

