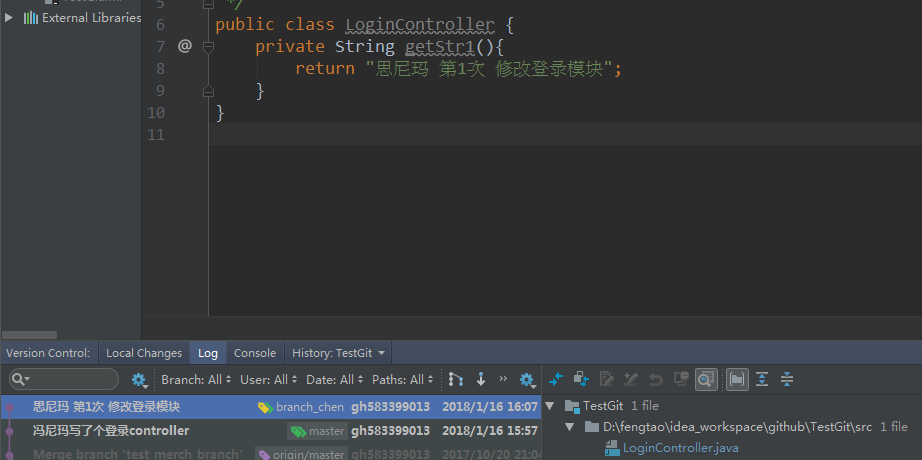
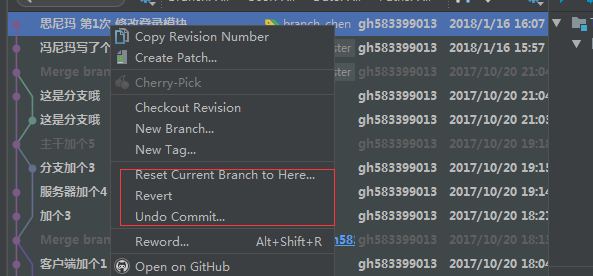
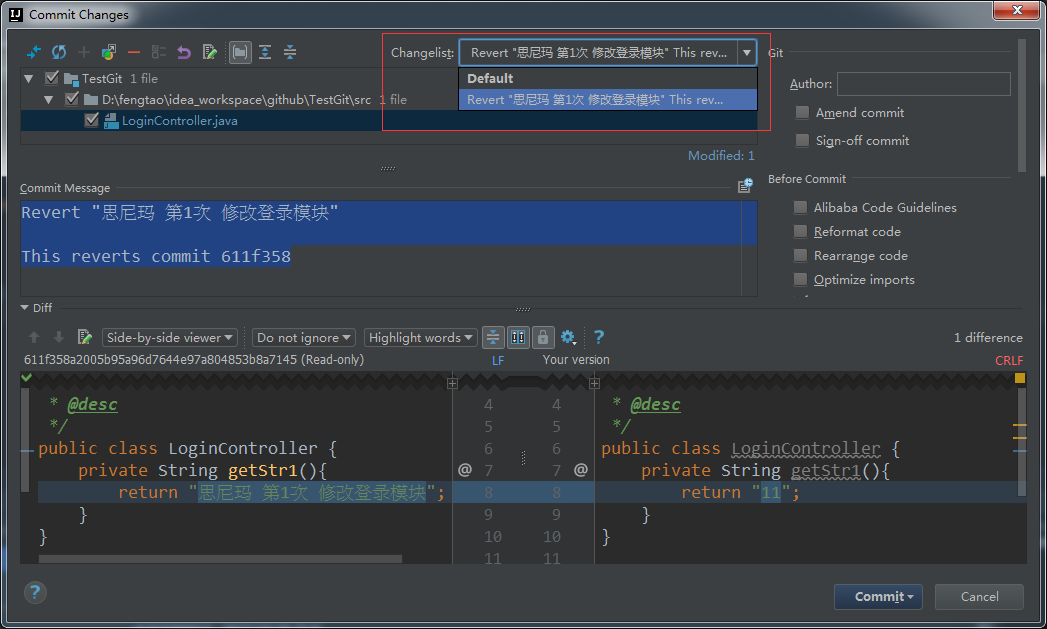
如下图 分支branch\_chen 修改了代码逻辑提交到本地(本地切的分支 并没有push到远程库)



右键提交日志 有三个操作:1 reset current branch to here 2:revert 3:undo commit(最新的commit才会有undo commit 之前的commit是没有的 注意)



我们来看一下revert会是什么效果

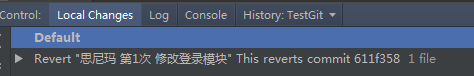


效果就是 本次提交的代码本地回滚了,然后changelist自动生成了一个回退版本 changeList

弹窗就是让你点击去重新提交.(本地代码已经回滚 提交时提交到暂存区)

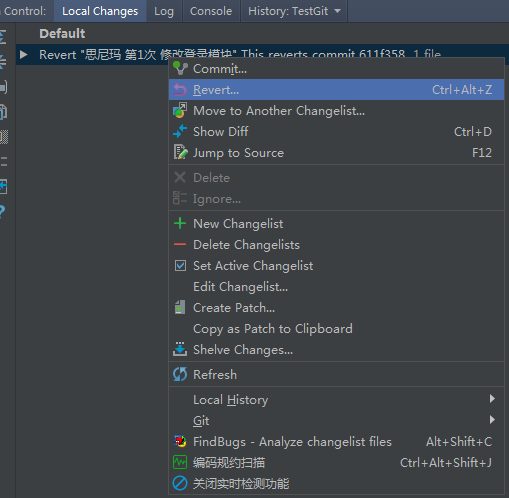
如果提交了 就保存到暂存区(提交回滚) 之后就是push到远程库不谈

如果这时候取消. 我们能在localchanges能看到操作生成的changelist

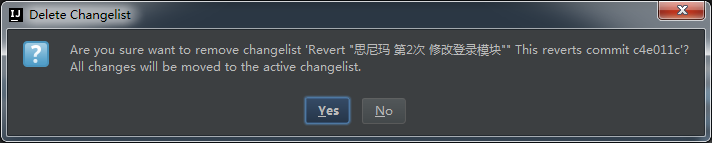


点击右键 revert能够取消这次revert(可以理解为撤销之前的回滚操作,还是提交最新的内容)

如果commit那就是直接回滚了.

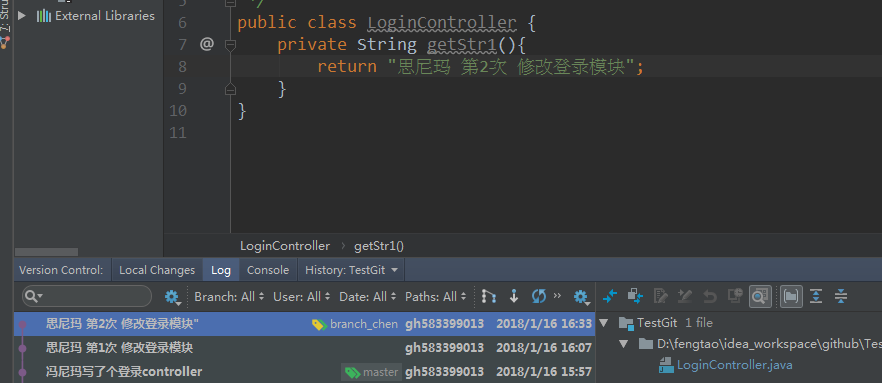


如果不小心delete了 那这个变动就移动到活跃的changeList了 比如上图的Default默认的 这个是没法删除的那你就只能提交这次回滚 或者 撤销这次回滚啦

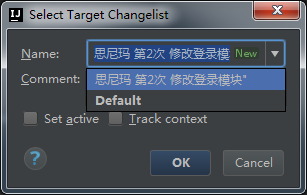


我们来看一下Undo Commit会是什么效果

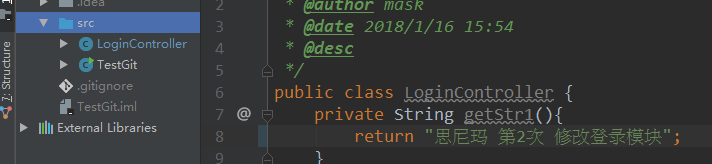
我们先在搞一次提交 第二次提交 内容就是文案变了



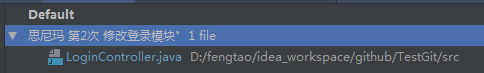
对上图的第2次 的log右键点击Undo Commit 出现弹窗 我们先不选default(选default的话变动内容就到了default)



我们发现代码没有回退 但是这个类提示有更改标记 代表有新的内容需要提交(undo commit就是撤销这一次的commit 但是内容保留) 因此这次的被undo 的 commit日志也没了.



我们再看看local changes



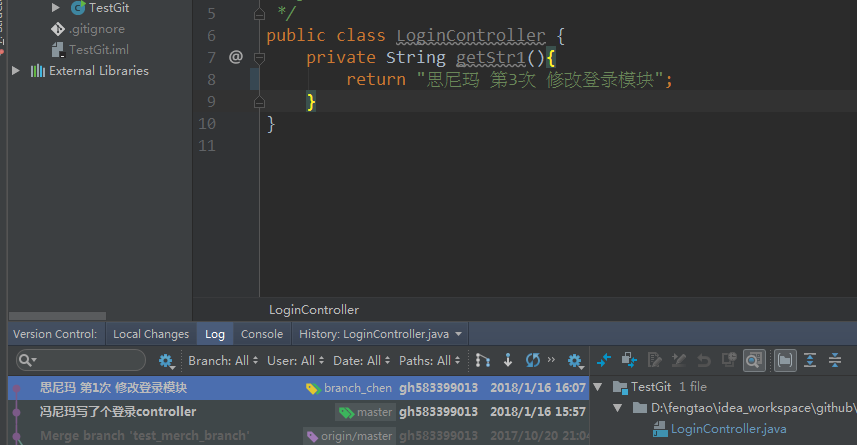
这个类依旧提示有修改内容 如果我们现在点击revert 代码就会回滚

点击删除 已然是移到default 你可以回滚或者重新提交

我们来看一下reset current branch to here会是什么效果

保留了最后一次的提交内容(第三次提交的)

第二次 和 第三次的提交日志没有了 需要重新commit



如果在local changes revert

那么版本回退到选定的版本

总结:

revert 是回滚代码并且贴心自动弹出问你是否提交这次回滚 如果取消了可以在localchanges中继续提交这次回滚 或者 撤回这次回滚

Undo commit是针对最后一次commit 它移除了本次提交的日志 但是代码没有被回滚,你可以选择继续提交 或者回滚当前的修改

reset current branch to here(sort mixed harded区别以后再说)

revert的加强版 可以回滚到指定版本A 相当于A之后的提交都被回滚了

git reset --mixed：此为默认方式，不带任何参数的git reset，即时这种方式，它回退到某个版本，只保留源码，回退commit和index信息

git reset --soft：回退到某个版本，只回退了commit的信息，不会恢复到index file一级。如果还要提交，直接commit即可

git reset --hard：彻底回退到某个版本，本地的源码也会变为上一个版本的内容，此命令 慎用！

https://www.cnblogs.com/chenweichu/articles/5819378.html

git reset --mixed git reset --soft区别在于有没有回退index信息

举个例子 A提交我提交了A.txt B提交 我修改了A.txt 并且 新增了B.txt

现在如果采用soft回退 那么B.txt我们无需再加入版本管理 因为soft保留了index B提交的时候已经把B.txt 写入了index

而mixed index也被回退 虽然B.txt文件任然保留(源码保留 不像hard) 但是如果要提交的话 要add vcs 或者 git 也就是要加入版本库管理

Q&A:

比如A B C 三次提交都是针对read.txt C最后提交 然后我们revert B的提交,因为B之后仍有C修改read.txt 因此此时会出现冲突界面 需要你手动操作合并操作