git stash 一个强大却容易被忽视的命令

**例子**：假如我们现在正在开发一套新的功能需求时，原有的出现了BUG，按照我们以前的做法的步骤：（相对于在主分支上开发）

1) git add . 2) git commit –m ‘XXXX’

执行以上命令之后在开始处理BUG，这样做的就会让代码中存在坏味道。后续结果有可能在运行 git push命令的时候将该段未完成的代码同样提交到远程仓库，这样的后果……

**现在发现一个新的git命令可以解决此问题**

git stash命令可以用于保存和恢复工作进度。

运行该命令后会使工作区间的代码和上一次提交内容时的代码完全一样，将现在处理的问题给缓存起来(**暂且这样理解，因为我也不知道到底是放到哪里了**)，现在就可以处理BUG问题，处理完之后按照常用的流程提交的远程仓库。

这之后就可以恢复到未处理完的功能：

git stash pop 或者 git stash apply

运行此命令后就会返回到未完成的代码下。

(**如果处理的是相同的文件相同部位的话则会产生冲突，此时就需要手工解决冲突了**)

**git stash pop 和 git stash apply区别**

假设我们 git stash 之后栈里面有一条记录，如果我们使用 git stash pop ，虽然也能恢复到入栈之前的改动，但是栈里的记录没了，如果这时我们 ‘git checkout .’，那么所有的改动都没了，而且也不能再次运用 git stash pop 来恢复，而如果我们用git stash apply 不仅能恢复入栈前的改动，栈里面的记录还保存着，即使我们 ‘git checkout .’ 也能再次运用 git stash apply 来恢复。如果觉得栈里面的记录太多，而且也不需要栈里记录的改动的话，大家可以用 git stash clear 清理一下。

**git stash list 显示所有的可以将当前的Git栈信息打印出来**

**git –exec-path 查找git所有命令存放的位置**

**附表**(git 分页器常用热键 例如:git log命令)

q 退出分页器

h 显示分页器帮助

按空格键下翻一页，按字母b上翻一页

字母d和u 分别代表向下翻动半页和向上翻动半页

字母j和k 分别代表向上翻一行和向下翻一行

如果行太长被截断，可以用左箭头和右箭头使窗口内容左右滚动

输入 /pattern 向下寻找和pattern匹配的内容

输入 ?pattern 向上寻找和pattern匹配的内容

字母n和N 代表向前或向后继续寻找

字母g 跳到第一行；字母G：跳到最后一行；输入数字+g ：跳转到对应的行

输入<!command>：可以执行shell命令