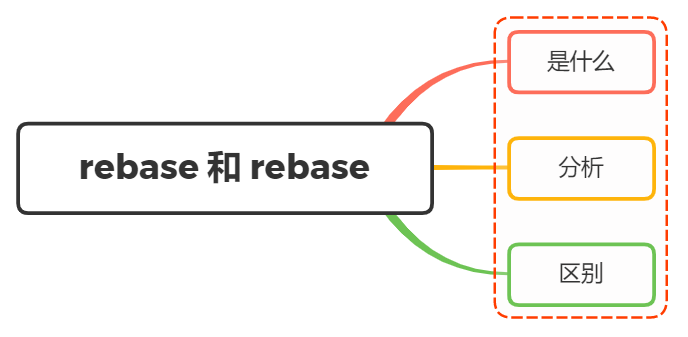
# 面试官：说说你对git rebase 和 git merge的理解？区别？



## 一、是什么

在使用 git 进行版本管理的项目中，当完成一个特性的开发并将其合并到 master 分支时，会有两种方式：

* git merge
* git rebase

git rebase 与 git merge都有相同的作用，都是将一个分支的提交合并到另一分支上，但是在原理上却不相同

用法上两者也十分的简单：

### git merge

将当前分支合并到指定分支，命令用法如下：

git merge xxx

### git rebase

将当前分支移植到指定分支或指定commit之上，用法如下：

git rebase -i <commit>

常见的参数有--continue，用于解决冲突之后，继续执行rebase

git rebase --continue

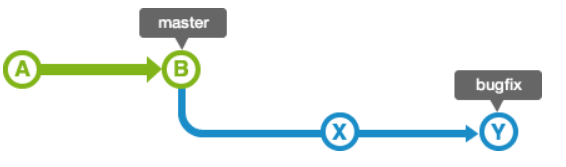
## 二、分析

### git merge

通过git merge将当前分支与xxx分支合并，产生的新的commit对象有两个父节点

如果“指定分支”本身是当前分支的一个直接子节点，则会产生快照合并

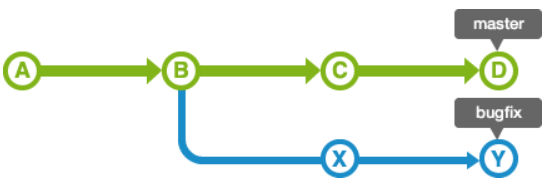
举个例子，bugfix分支是从master分支分叉出来的，如下所示：



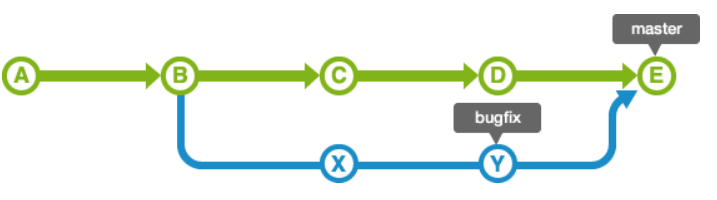
合并bugfix分支到master分支时，如果master分支的状态没有被更改过，即 bugfix分支的历史记录包含master分支所有的历史记录

所以通过把master分支的位置移动到bugfix的最新分支上，就完成合并

如果master分支的历史记录在创建bugfix分支后又有新的提交，如下情况：



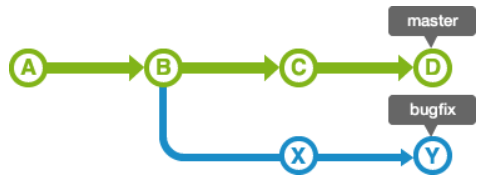
这时候使用git merge的时候，会生成一个新的提交，并且master分支的HEAD会移动到新的分支上，如下：



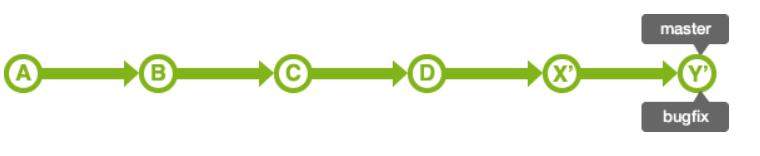
从上面可以看到，会把两个分支的最新快照以及二者最近的共同祖先进行三方合并，合并的结果是生成一个新的快照

### git rebase

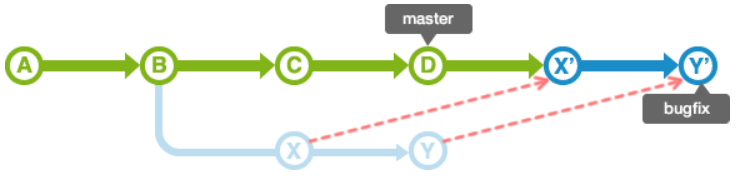
同样，master分支的历史记录在创建bugfix分支后又有新的提交，如下情况：



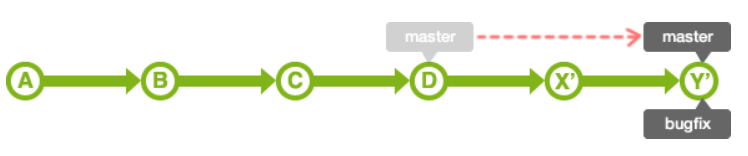
通过git rebase，会变成如下情况：



在移交过程中，如果发生冲突，需要修改各自的冲突，如下：



rebase之后，master的HEAD位置不变。因此，要合并master分支和bugfix分支



从上面可以看到，rebase会找到不同的分支的最近共同祖先，如上图的B

然后对比当前分支相对于该祖先的历次提交，提取相应的修改并存为临时文件（老的提交X和Y也没有被销毁，只是简单地不能再被访问或者使用）

然后将当前分支指向目标最新位置D, 然后将之前另存为临时文件的修改依序应用

## 三、区别

从上面可以看到，merge和rebasea都是合并历史记录，但是各自特性不同：

### merge

通过merge合并分支会新增一个merge commit，然后将两个分支的历史联系起来

其实是一种非破坏性的操作，对现有分支不会以任何方式被更改，但是会导致历史记录相对复杂

### rebase

rebase会将整个分支移动到另一个分支上，有效地整合了所有分支上的提交

主要的好处是历史记录更加清晰，是在原有提交的基础上将差异内容反映进去，消除了 git merge所需的不必要的合并提交

## 参考文献

* https://zhuanlan.zhihu.com/p/361182707
* https://yuweijun.github.io/git-zh/1-git-branching.html#\_rebasing
* https://backlog.com/git-tutorial/cn/stepup/stepup1\_4.html