操作

查看历史分支图

git log --oneline --decorate --graph –all

创建新分支并切换到该分支

git checkout –b [branchName]

删除分支

git branch -d hotfix

git branch

查看已经合并到当前分支的分支(这里的分支可以删除，因为已经被合并了，内容不会丢失)

git branch --merged



查看没有合并到当前分支的分支

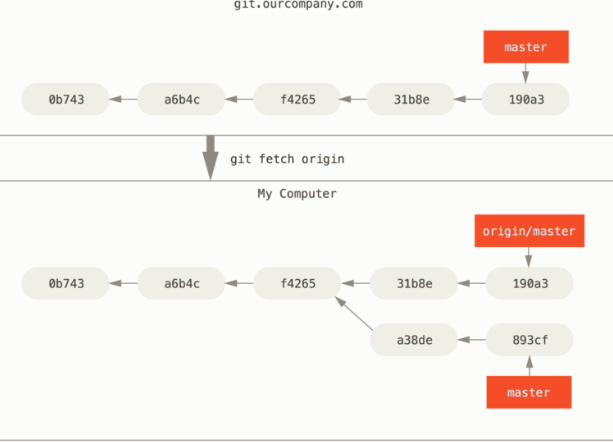
git branch --no-merged

查看远程库信息：

git remote show [branchName]

git fetch origin

这个命令查找 “origin” 是哪一个服务器，从中抓取本地没有的数据，并且更新本地数据库，移动 origin/master指针指向新的、更新后的位置。



git fetch 之后，将远程库分支的指针在本地生成了一份，命名为origin/[branchName]，接下来有两种应用场景：

1. 将远程分支合并到当前分支

git merge origin/[branchName]

1. 从远程分支上检出一个本地分支，用于开发

git checkout –b branchName origin/[branchName]

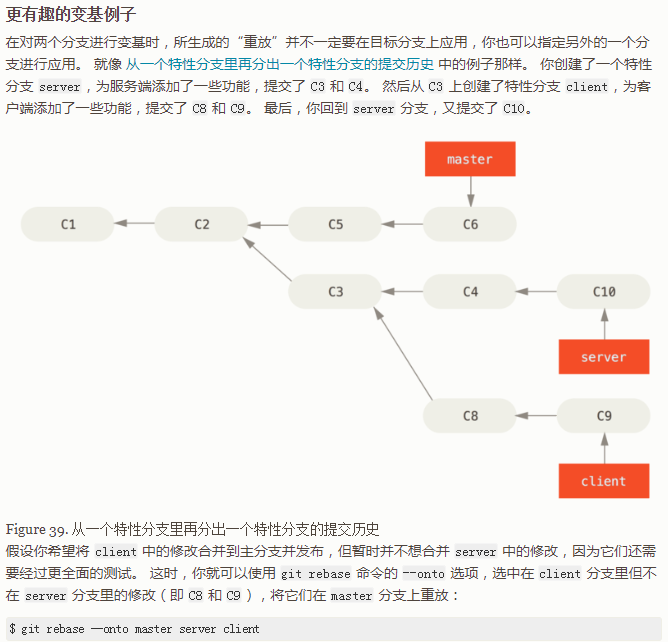
git fetch --all

直接删除远程库的分支（指针）

git push origin --delete [branchName]

git rebase --onto master server client

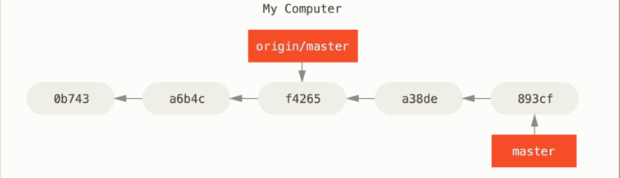
取出 client 分支，找出处于 client 分支和 server 分支的共同祖先之后的修改，然后把它们在 master 分支上重放一遍，下图：



原理：

1. **远程跟踪分支**是远程分支状态的引用。 它们是你不能移动的本地引用，当你做任何网络通信操作时，它们会自动移动。 远程跟踪分支像是你上次连接到远程仓库时，那些分支所处状态的书签。它们以 (remote)/(branch) 形式命名。

只要你不与 origin 服务器连接，你的 origin/master 指针就不会移动。



后续：

1. 校验落后于Master的钩子？
2. Pull比fetch和merge的好处