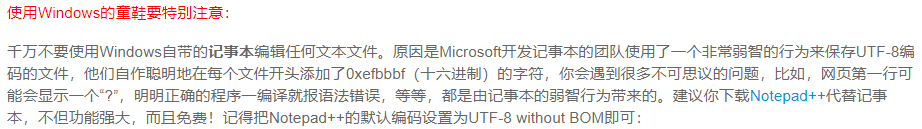
1 创建版本库





2 时光机穿梭



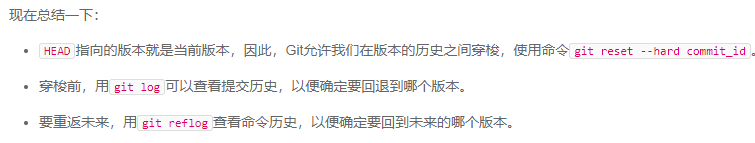
#### 2.1 版本回退

每当你觉得文件修改到一定程度的时候，就可以“保存一个快照”，这个快照在Git中被称为commit。

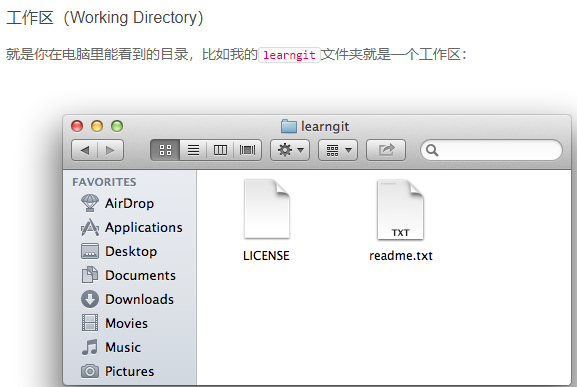
每提交一个新版本，实际上Git就会把它们自动串成一条时间线。

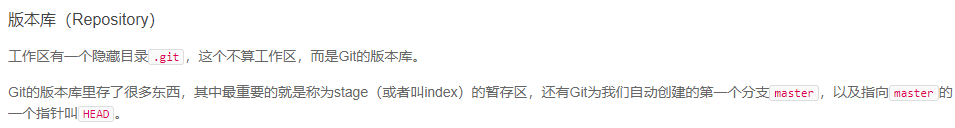


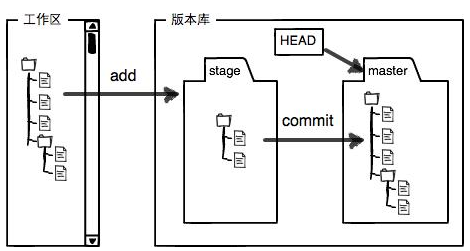


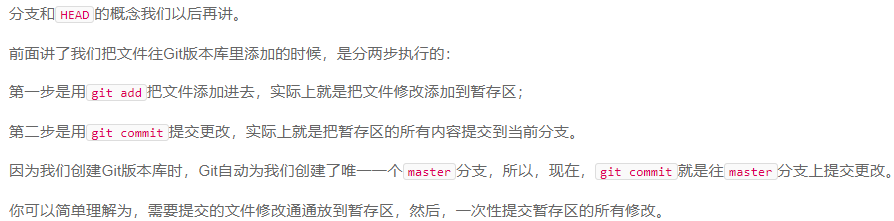


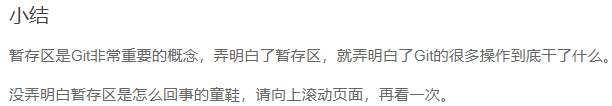
#### 2.2 工作区和暂存区



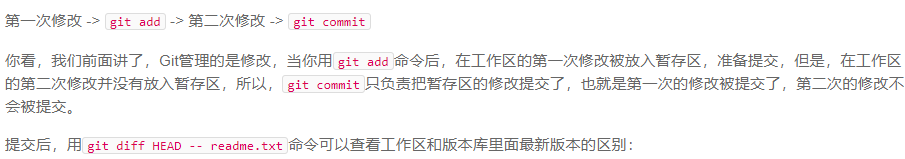


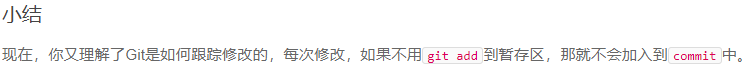




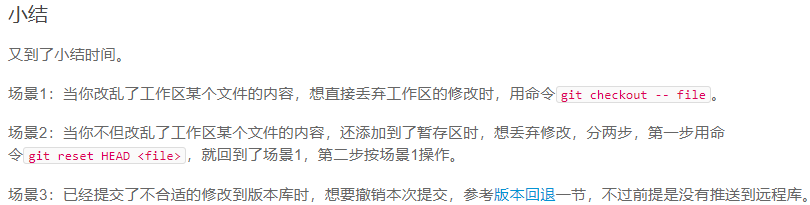


#### 2.3 管理修改

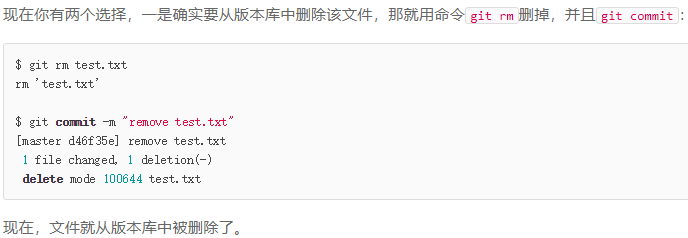


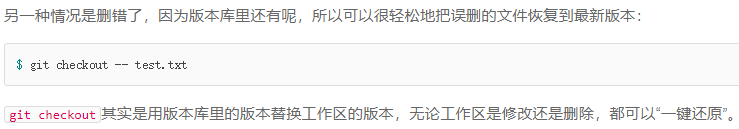


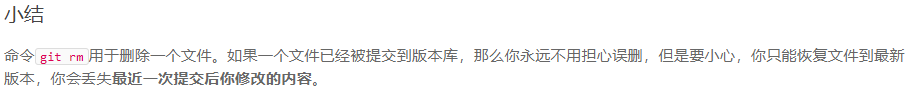
#### 2.4 撤销修改



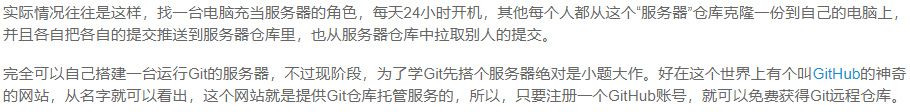
#### 2.5 删除文件



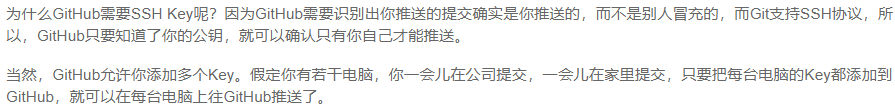


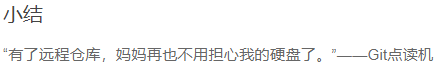


3 远程仓库

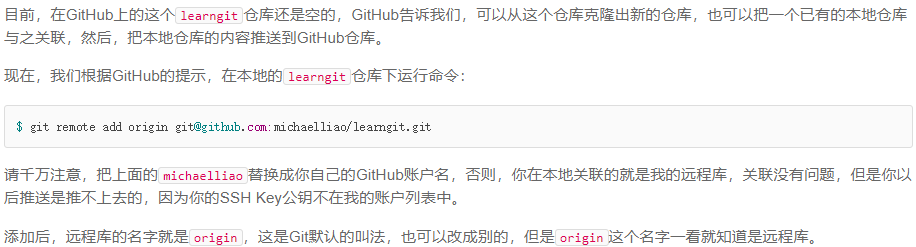


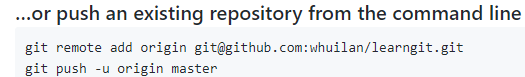






#### 3.1 添加远程库（推送至远程库）



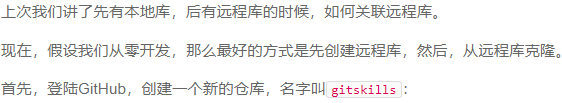




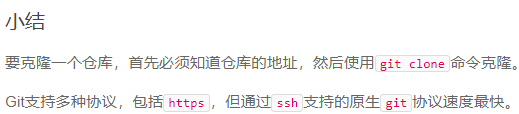




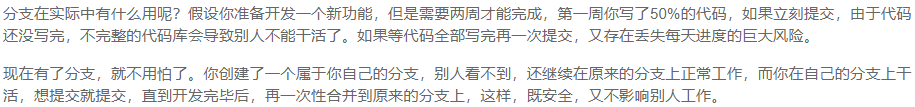
#### 3.2 从远程库克隆



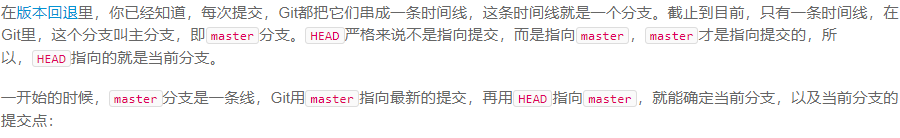


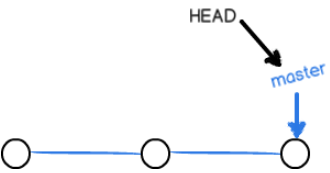


4 分支管理

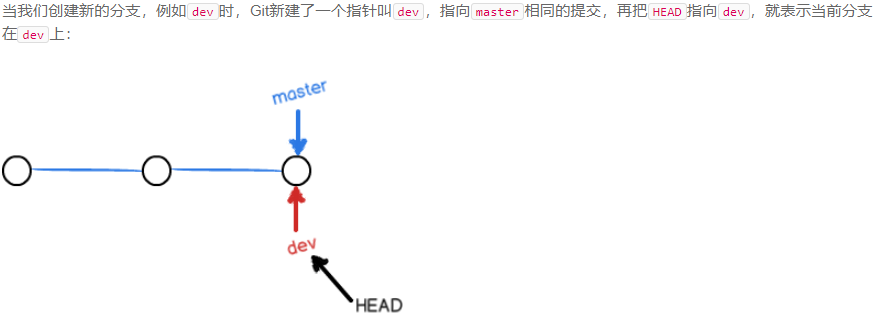


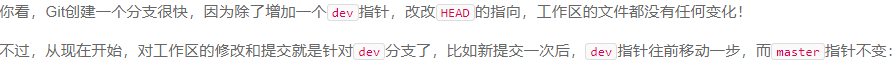
#### 4.1 创建与合并分支

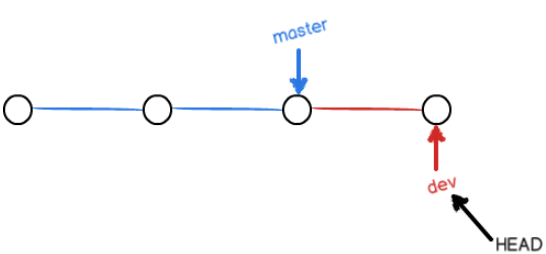


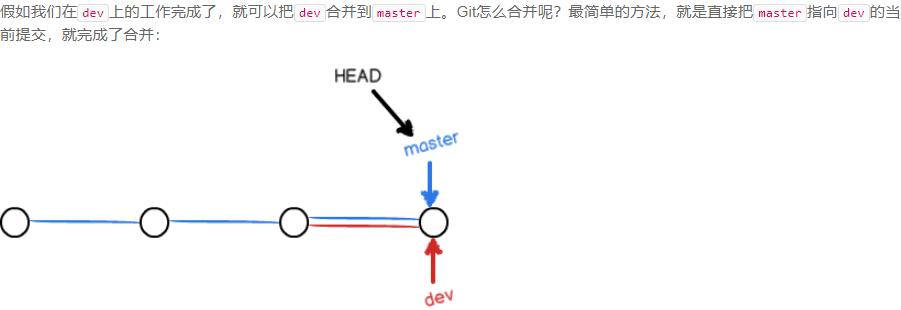


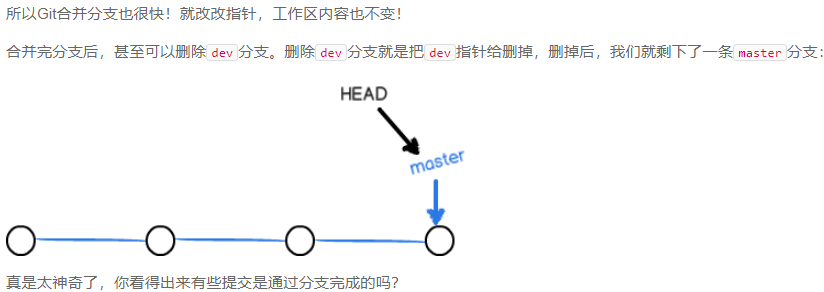






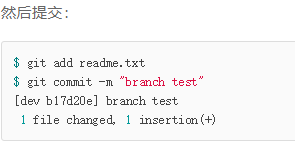


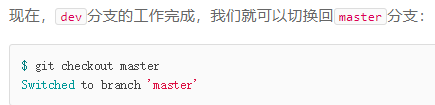


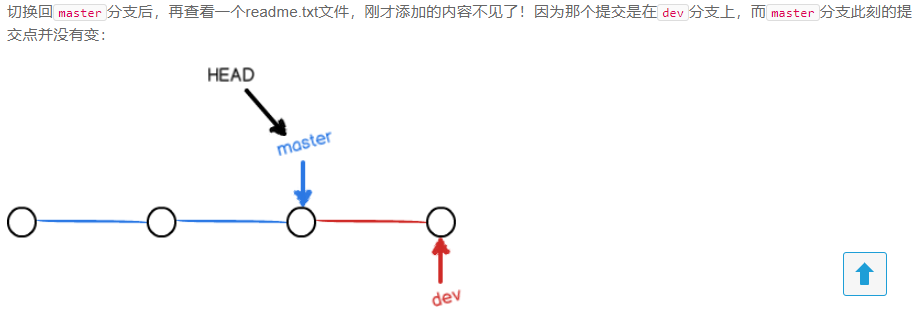


以上过程教程上有视频图示，可以参考看看

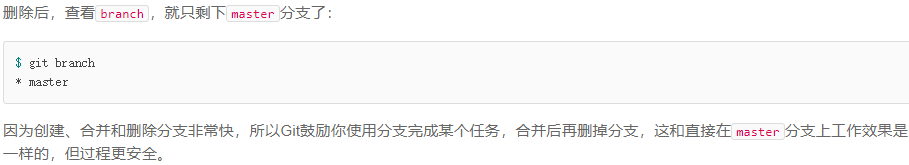












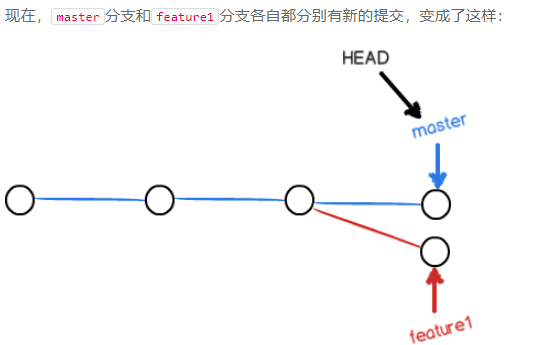
小结

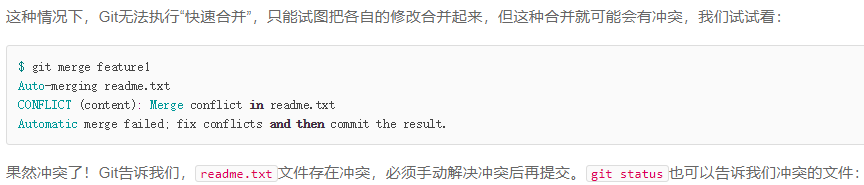


#### 4.2 解决冲突



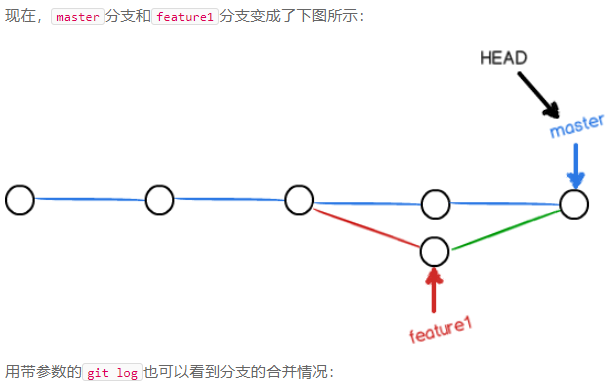




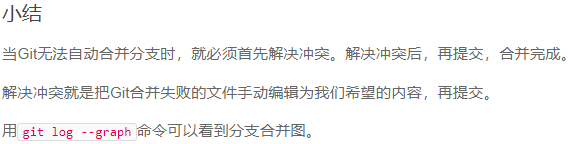




注意手动解决冲突修改工作区代码之后再再次合并之前需要先提交，即git add和git commit



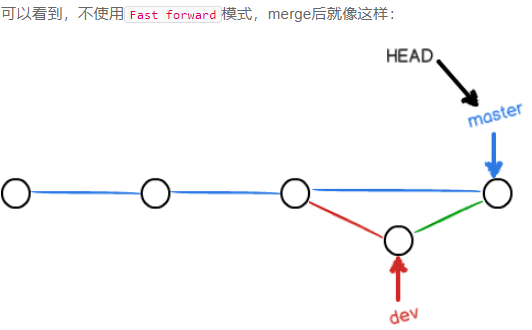




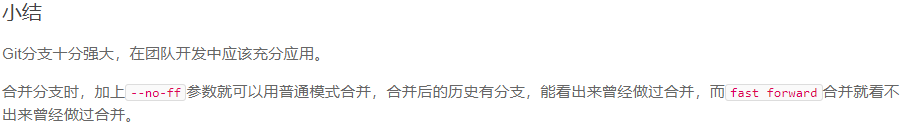
#### 4.3 分支管理策略



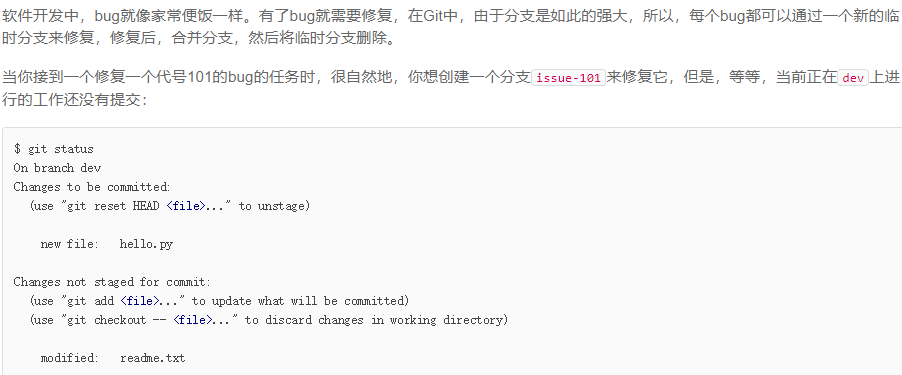








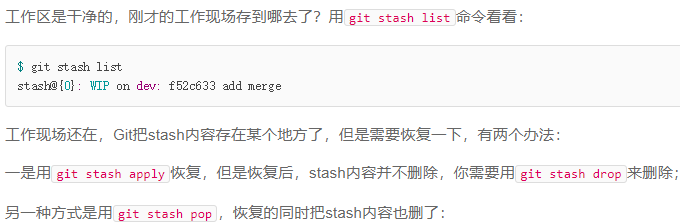
#### 4.4 Bug分支





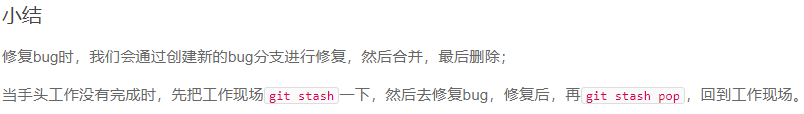




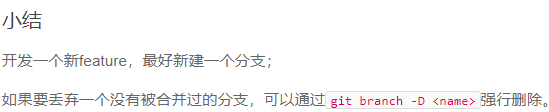




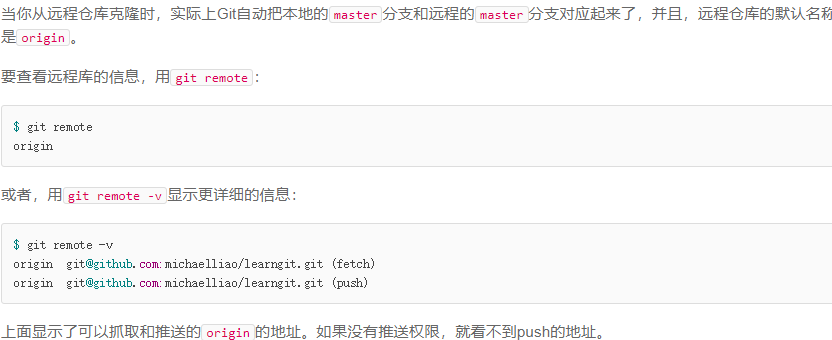
小结：



#### 4.5 Feature分支

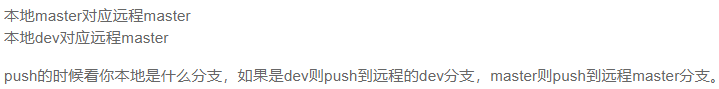


#### 4.6 多人协作





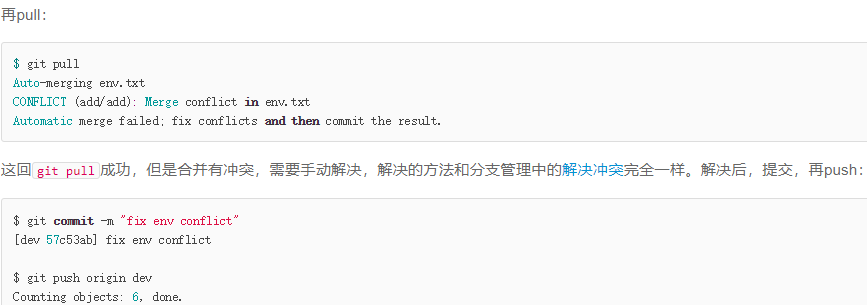
注意这里的git push origin dev就会在远程库origin上自动新建一个dev同名分支，并且把本地dev分支上的提交推送到远程的dev分支。（但是二者之间还没有关联起来）

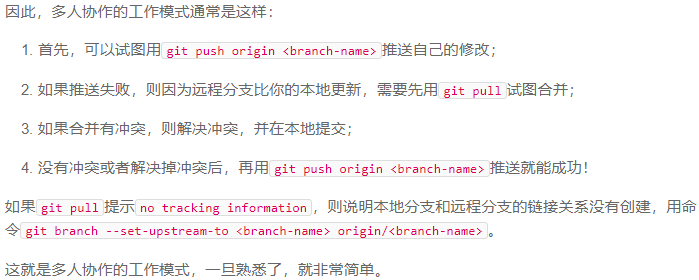








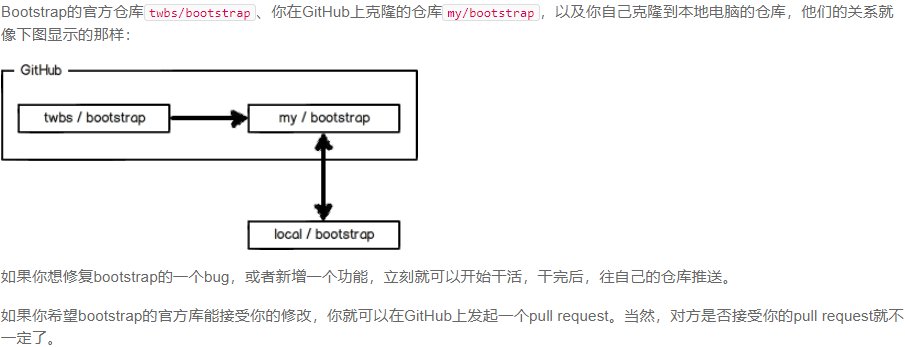






使用GitHub





5 补充

#### 1关于pull request

