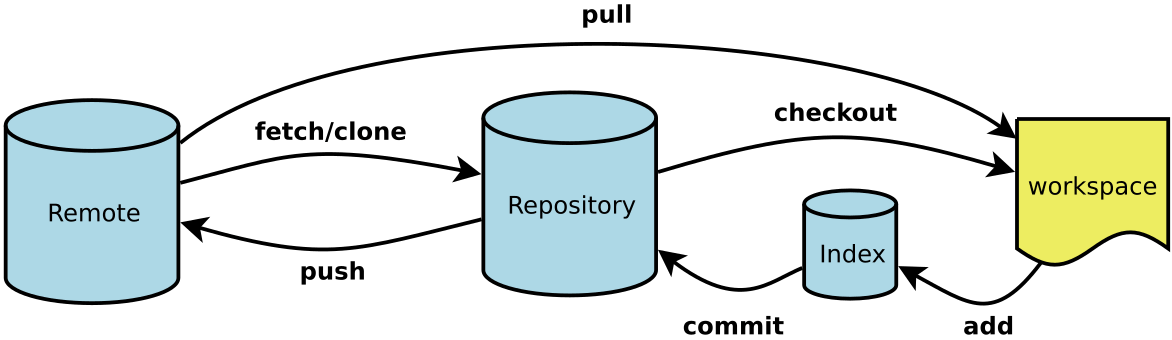
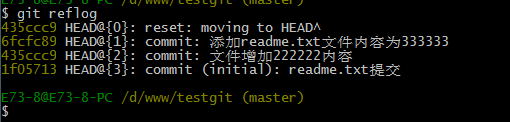
# Git



* Workspace：工作区
* Index / Stage：暂存区
* Repository：仓库区（或本地仓库）
* Remote：远程仓库
* git diff <file> : 比较workspace和index
* Commit后，可以执行git reset --hard HEAD^ 回退到前一个版本
  + 如果要回退到上上版本，可使用HEAD^^，以此类推
  + 如果要回退到前100个版本，可以使用HEAD~100
* 回退了之后，可能你又想返回到最新的版本，怎么办？
  + 可以通过git reflog 获取版本号（6fcfc89)
    - 
  + 然后执行git commit --hard 6fcfc89

**六：创建与合并分支**。

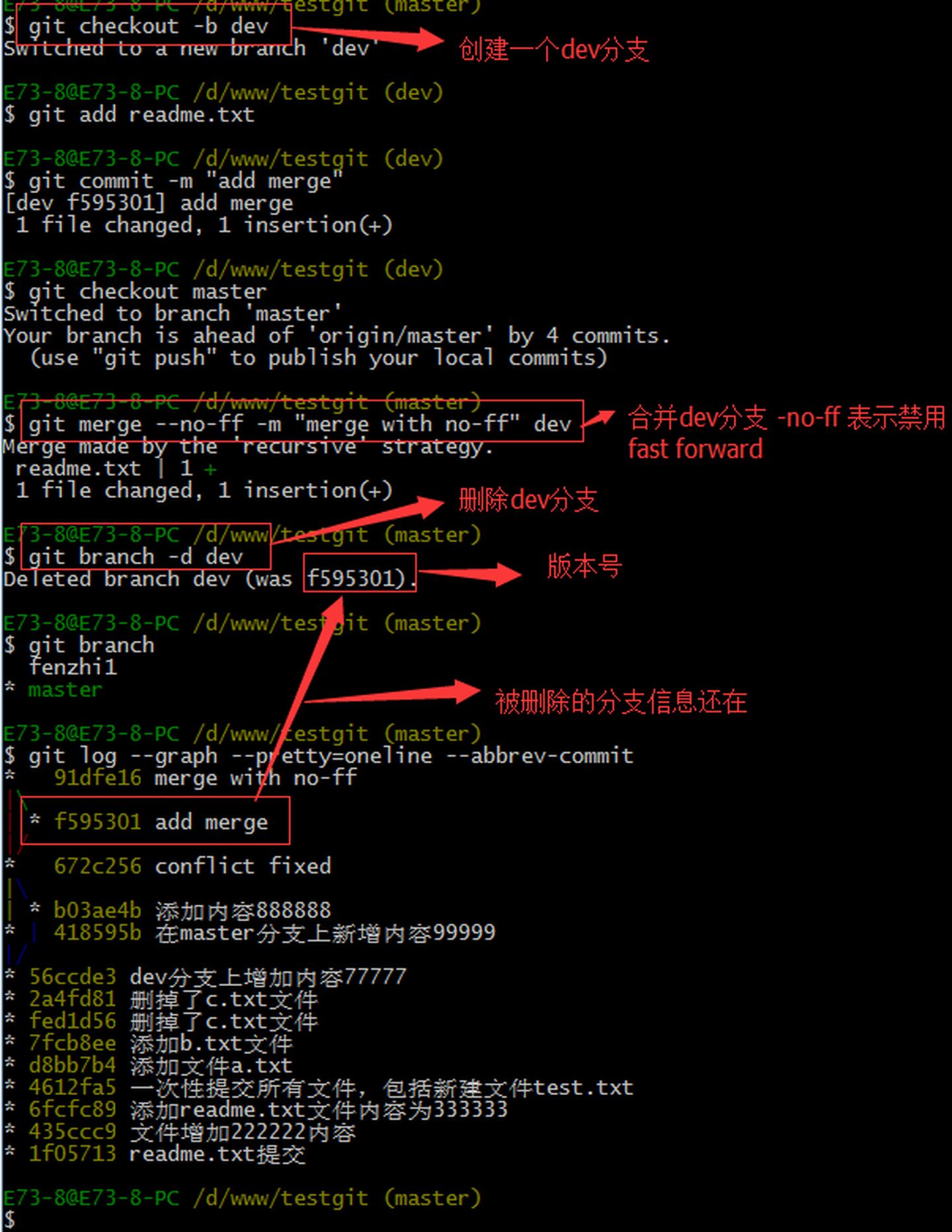
* 基本情况

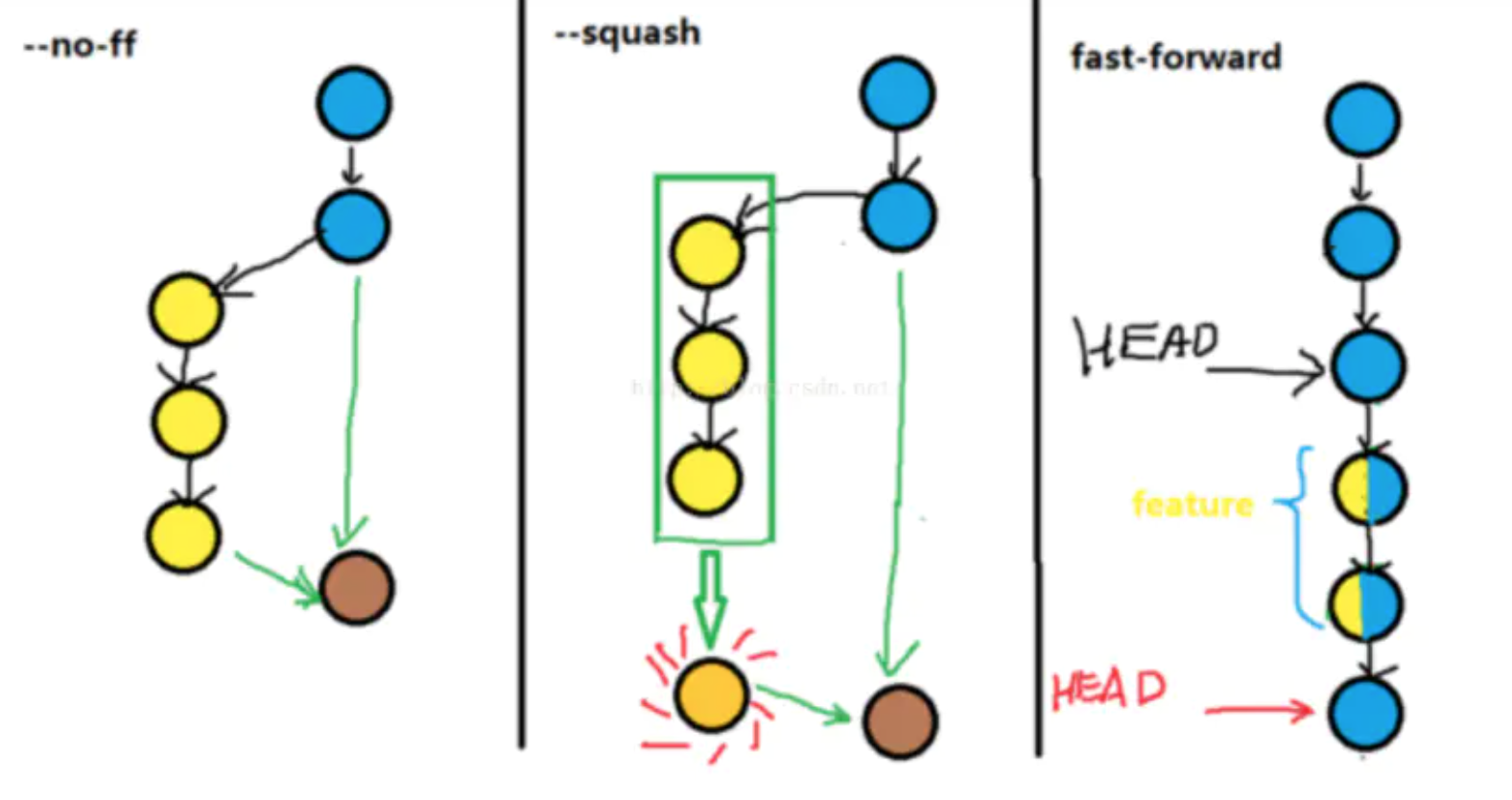
|  |
| --- |
| git checkout -b dev  (make change to readme.txt)  git add readme.txt  git commit -m “submit changes”  git checkout master (此时，不会看到新的改变）  git merge dev  - 可看到回复包含“fast-forward”（快进模式），也就是直接把master指向dev的当前提交，所以合并速度非常快。  git branch -d dev (删除dev分支） |

* Merge Conflict

|  |
| --- |
| IMG_256  IMG_256  IMG_256  Git用<<<<<<<，=======，>>>>>>>标记出不同分支的内容，其中<<<HEAD是指主分支修改的内容，>>>>>fenzhi1 是指fenzhi1上修改的内容，我们可以修改下如下后保存：  IMG_256  git log 来查看log |

* 分支管理策略
  + 使用no-ff后，会多生成一个commit 记录，并强制保留develop分支的开发记录（而fast-forward的话则是直接合并，看不出之前Branch的任何记录）





<https://www.jianshu.com/p/26edea85062d>

**七：bug分支：**

* 可能在开发一个feature时，有一个bug要立刻解决，但你的feature还没写完，那怎么办？
  + 用git stash**把当前工作储存，然后解决bug，然后用git stash apply恢复stash内容**

|  |
| --- |
| git checkout -b dev  (make changes)  **git stash # 把当前工作现场 ”隐藏起来”，等以后恢复现场后继续工作**  git status # 会看到no changes to commit  git checkout -b issue-404 # 创建issue-404分支来修复bug  (修复bug)  git checkout dev # 此时，还是看不到之前在dev分支做过的修改  **git stash list # 查看stash内容**  **git stash apply <stash id> # 恢复stash内容** |

* 此外，也有：
  + git stash pop：恢复的同时把stash内容也删除了
  + git stash drop：把stash内容删除

**八：多人协作**。

当你从远程库克隆(git clone)时候，实际上Git自动把本地的master分支和远程的master分支对应起来了，并且远程库的默认名称是origin。

多人协作时，有可能几个人都在更改同一个分支的同一个文件。

|  |
| --- |
| （同事）  git clone <link>  git checkout -b dev origin/dev #复制远程origin的dev分支到本地  （更改文件）  git add, commit ...  git push origin dev # 把dev分支推到远程去  （同时候，我也推出我的修改，怎么办？）    由上面可知：推送失败，因为我的**小伙伴最新提交的和我试图推送的有冲突**，解决的办法也很简单，上面已经提示我们，先用**git pull**把最新的提交从origin/dev抓下来，然后在本地合并，解决冲突，再推送。    git pull也失败了，原因是没有指定本地dev分支与远程origin/dev分支的链接，根据提示，设置dev和origin/dev的链接，如上。  这回git pull成功，但是合并有冲突，需要手动解决，解决的方法和分支管理中的 解决冲突完全一样。解决后，提交，再push。  IMG_256  因此：多人协作工作模式一般是这样的： 首先，可以试图用git push origin branch-name推送自己的修改. 如果推送失败，则因为远程分支比你的本地更新早，需要先用git pull试图合并。 如果合并有冲突，则需要解决冲突，并在本地提交。再用git push origin branch-name推送。 |

<https://www.zhihu.com/question/22285059>