

git操作

git工作流程

- 视频教学 十分钟学会正确的github工作流，和开源作者们使用同一套流程_哔哩哔哩_bilibili
2023年3月20日22:27:26

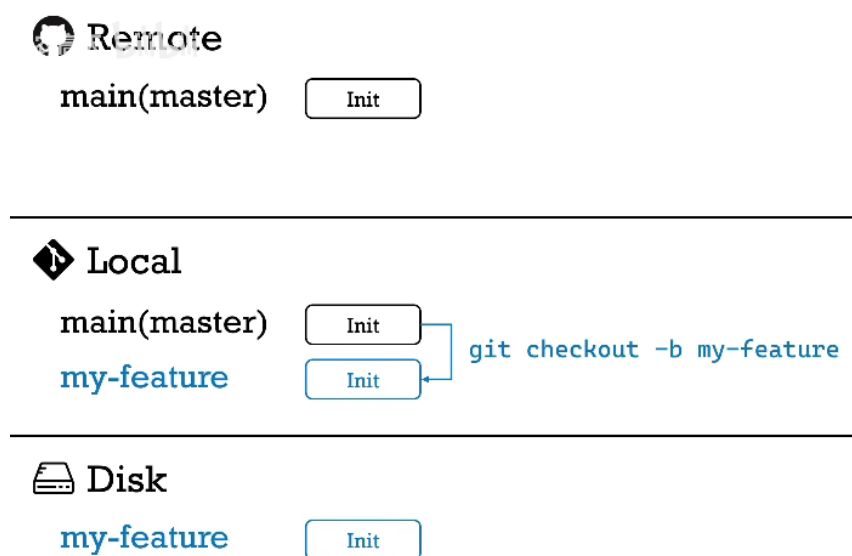
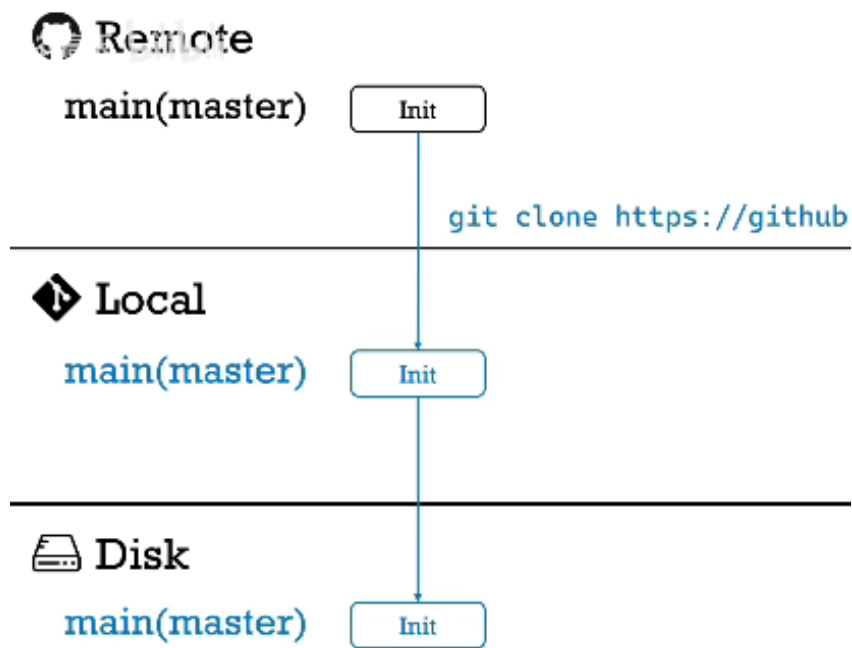
操作代码说明

```
1 1. git clone <project> // 克隆到本地
2 2. git checkout -b <myFeature_branch> // 切换至新的分支 myFeature_branch 仅是git
   的操作，对硬盘文件不影响
3 3. <changes> 修改完本地的代码后（部署在硬盘的源文件代码）
4 4. git diff // 查看一下自己对代码作出的修改
5 5. git add <changed_file> // 上传更新后的代码至暂存区
6 6. git commit // 把所有 git add 的文件提交到 Local里面，使用git commit -m 可以在后
   面输入提交的说明
7 7. git push origin <myFeature_branch> // 将本地的myFeature_branch分支push到
   github上的git的个人branch分支
8
9 /* 当上述第七步的push完后，远端代码出现update 则需要先同步代码，执行下面 第10行，否则
   跳转 第17行*/
10 10. git checkout master // 切换到 main 源代码会变成init状态，而不是修改后的状态
11 11. git pull origin master // 把远端的 main 更新同步到local的 main 和 disk 中
12 12. git checkout <myFeature_branch> // 切换回我们的 myFeature_branch 即自己修改过
   的代码的分支
13 13. git rebase main // 把main最新的修改拿过来，同时在这个基础上把我们的修改合并上去，
   过程中可能会出现rebase conflict，出现了就需要手动选择要保留的代码
14 14. git push -f origin <myFeature_branch> //把我们local上myFeature_branch的修改
   push到github上的个人分支中，-f表示强制提交，对于公共分支坚决不建议
15
16 /* 请求项目管理者 合并你的push代码 */
17 17. pull request //请求项目管理者合并你的修改
18
19 /* 远端完成更新后，删除分支信息 */
20 20. git checkout master // 切换到 mian branch
21 21. git branch -D <myFeature_branch> // 删除本地git中local的myFeature_branch分支
22 22. git pull origin master // 再把远端的最新代码拉至本地，完成一个循环
```

操作代码的流程图

- Remote是远端的git代码仓库（gitee）
- Local 是本地的git代码仓库（拥有所有你告诉git的信息）
- Disk 是磁盘，即源文件真正在磁盘中的样子

克隆仓库，开始添加代码



Remote

main(master)

Init

Local

main(master)

Init

my-feature

Init

f-commit

Disk

my-feature

Init

f-commit

git commit

changes

Remote

main(master)

Init

my-feature

Init

f-commit

Local

main(master)

Init

my-feature

Init

f-commit

git push origin my-feature

Disk

my-feature

Init

f-commit

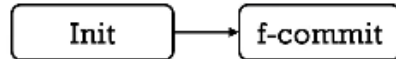
若main分支已经被更新

Remote

main(master)

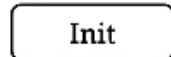


my-feature



Local

main(master)

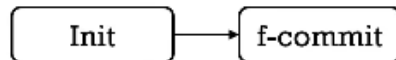


my-feature



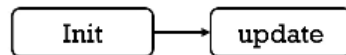
Disk

my-feature

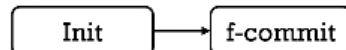


Remote

main(master)

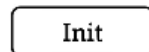


my-feature

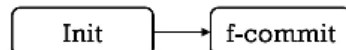


Local

main(master)



my-feature



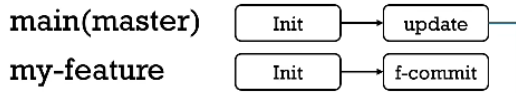
Disk

main(master)



`git checkout main`

Remote



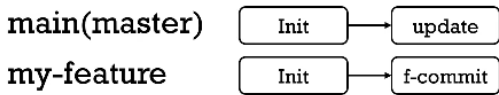
Local



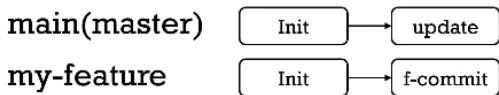
Disk



Remote



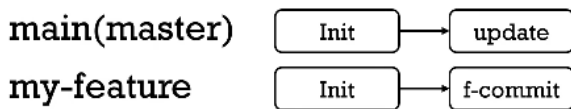
Local



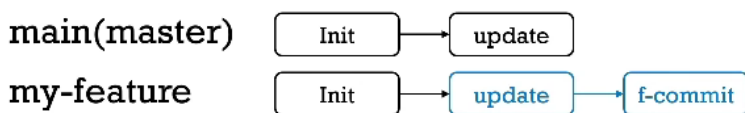
Disk



Remote

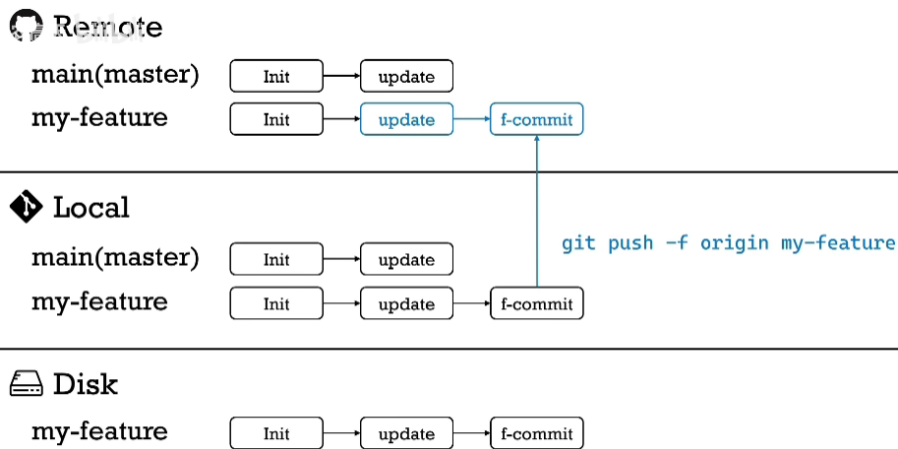


Local

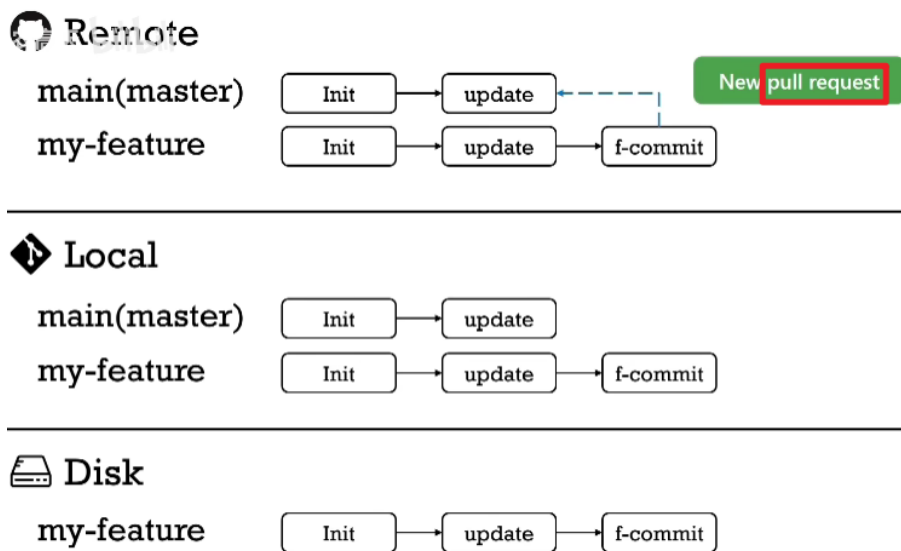


Disk

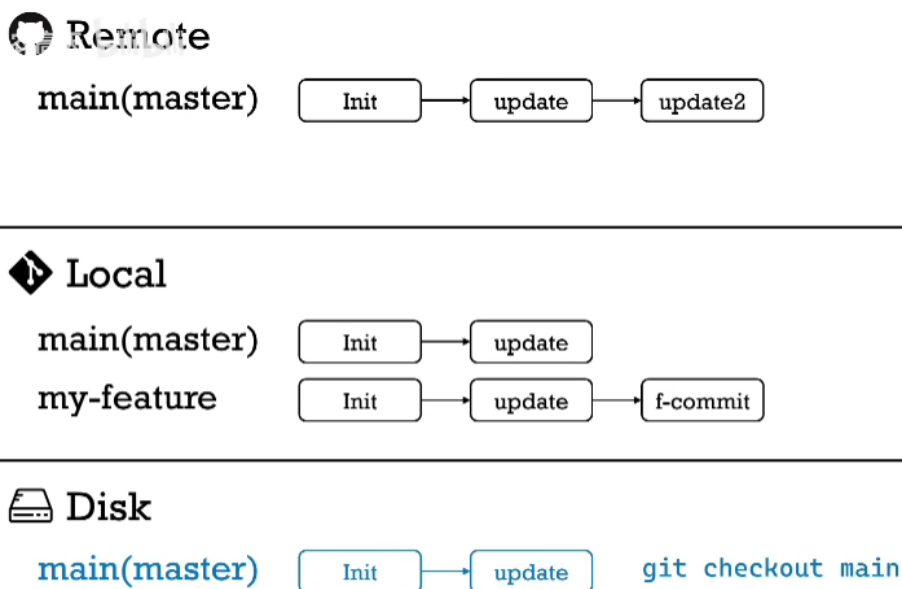




请求项目管理员合并你的分支



删除分支信息



Remote

main(master) Init → update → update2

Local

main(master) Init → update

`git branch -D my-feature`

Disk

main(master) Init → update

Remote

main(master) Init → update → update2

Local

main(master) Init → update → update2

`git pull origin master`

Disk

main(master) Init → update → update2

git撤销操作

- **视频教程** 十分钟学会常用git撤销操作，全面掌握git的时光机_哔哩哔哩_bilibili 2023年3月20日 14:33:51
- 四个区域，设初始状态都是数据一致的，只有一个 init commit
 - a. Disk
 - i. 若你修改完文件后，使用 `git status` 命令，会看见修改的文件被放到 Changes not staged for commit 底下，一般显示为红色
 - ii. 若要，撤销对硬盘上文件的修改撤销

Disk

Init



`git checkout <changed_file>`
(`git restore <changed_file>`)

Staging

Init

Local

Init

```
1 git checkout <changed_file> // 旧版本中 checkout
2 (git restore <changed_file>) // 新版本 restore
3 /*
4 当然，两种方式都是可以的在当前版本下
5 */
```

b. Staging

- 在对 Disk 中的文件使用了 `git add` 后，修改会被同步到暂存区
- 若是现在你使用 `git status` 命令，会看见被修改文件被放到 Changes to be committed 底下，一般为绿色
- 若要，撤销移除暂存区的修改，而保留硬盘上的修改，即撤销 `git add` 操作

Disk

Init

Change

Staging

Init



`git reset <changed_file>`
(`git restore --staged <changed_file>`)

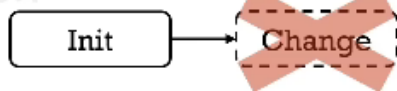
Local

Init

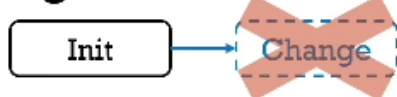

```
1 git reset <changed_file>
2 <git restore --staged <changed_file>>
```

iv. 若要，撤销所有修改，即暂存区和硬盘

Disk

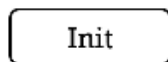


Staging



`git checkout HEAD <changed_file>`

Local

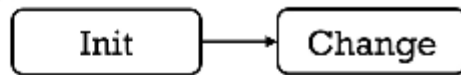


```
1 git checkout HEAD <changed_file> // HEAD 在git中表示最近一次commit, 在这里即代表git add
```

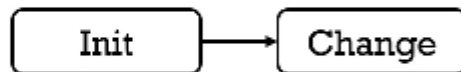
c. Local

- i. 在接着上面的git add 后使用了，git commit，那么修改会被放到 local git
- ii. 若是，单纯撤销这个 commit

Disk



Staging



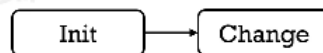
Local



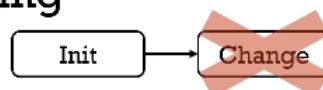
1 `git reset --soft HEAD~1` //~1代表之前一个 ~2即代表回退2个commit, 以此类推

iii. 若是, 同时撤销 local 和 Staging

Disk



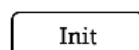
Staging



Local

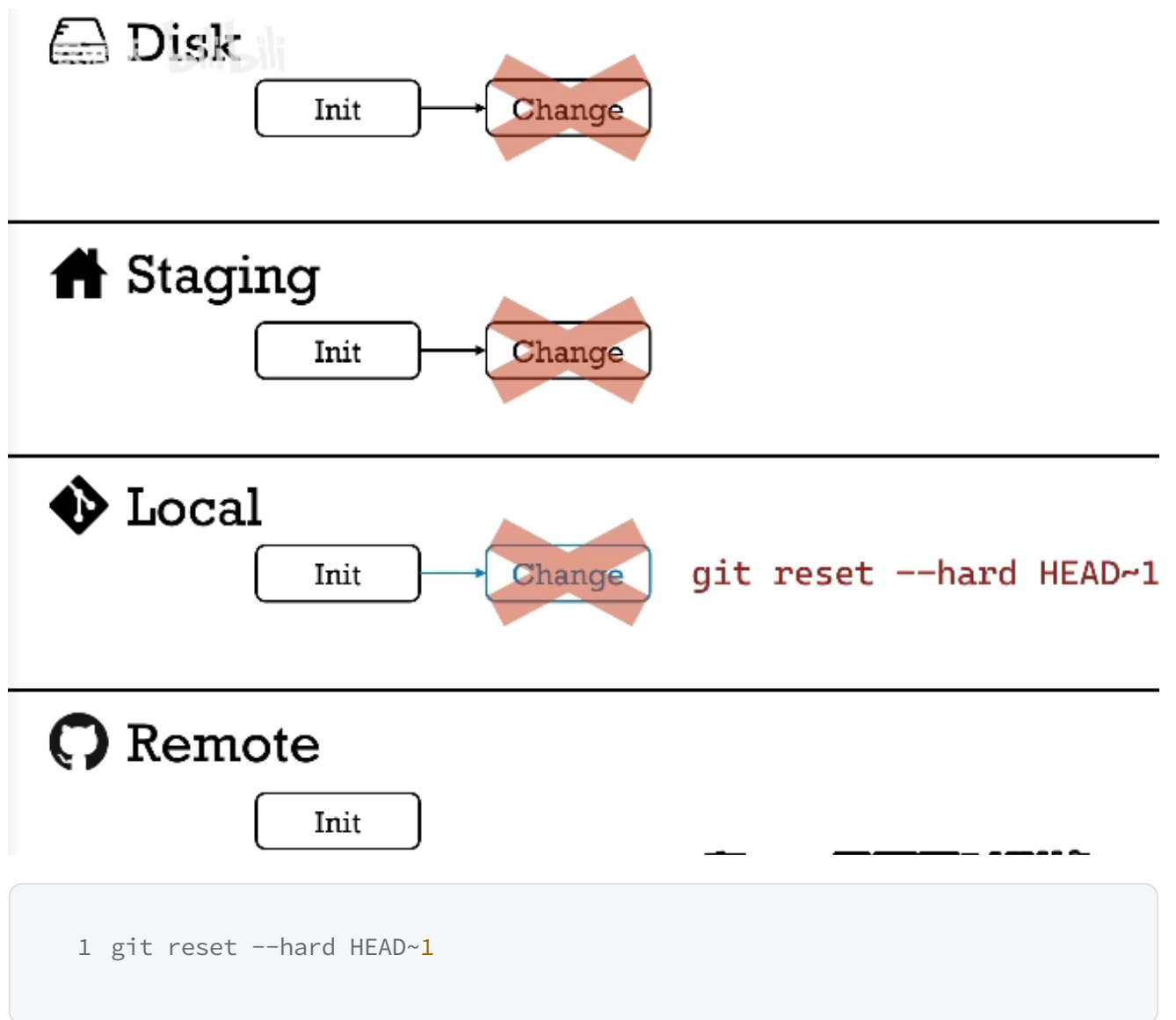


Remote



1 `git reset HEAD~1`
2 (`git reset --mixed HEAD~1`)

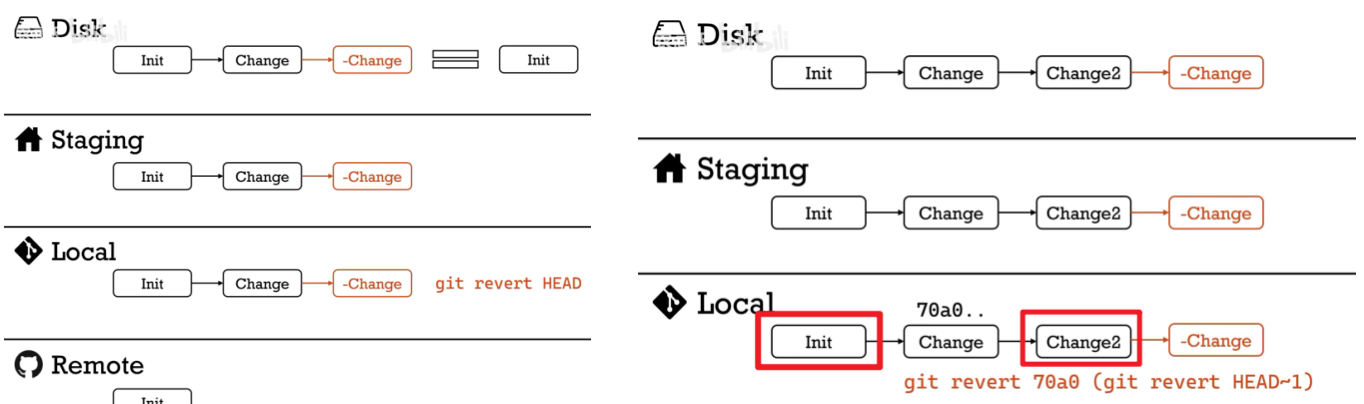
iv. 若是, 撤销全部修改 要谨慎使用



d. Remote

i. 当我们使用 git push 把修改提交到了远端，

1. git revert HEAD 是添加一个 -Change 的commit，即添加的代码就变成减少对应的代码，同样可以回到init

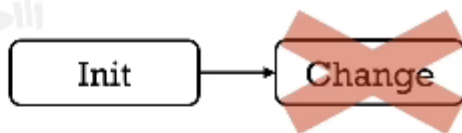


a. 其可以实现，撤销中间修改的效果

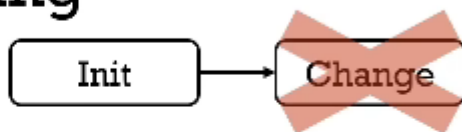
- b. 同时，当修改使用 git push 提交到远端后，由于**公有分支**只支持添加，不支持回退，因此只能使用git revert HEAD 命令，然后使用 git push 提交到远端（因为是添加一个commit，所以可以用push命令）
- c. 若 提交到远端的是个人分支，即只有你一个人用的



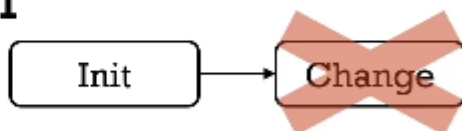
Disk



Staging



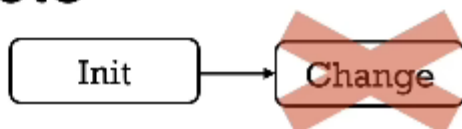
Local



git reset --hard HEAD~1
git push -f



Remote



```
1 git reset --hard HEAD~1
2 git push -f //用来同步到远端，f代表 force
3 /*
4  若是使用 git push 提交，远端的git会发现，
5  你这个分支少了一个commit，会被不允许
6  因此必须使用 -f 强制提交
7  对于公有分支，原则上不允许使用 -f
8  */
```