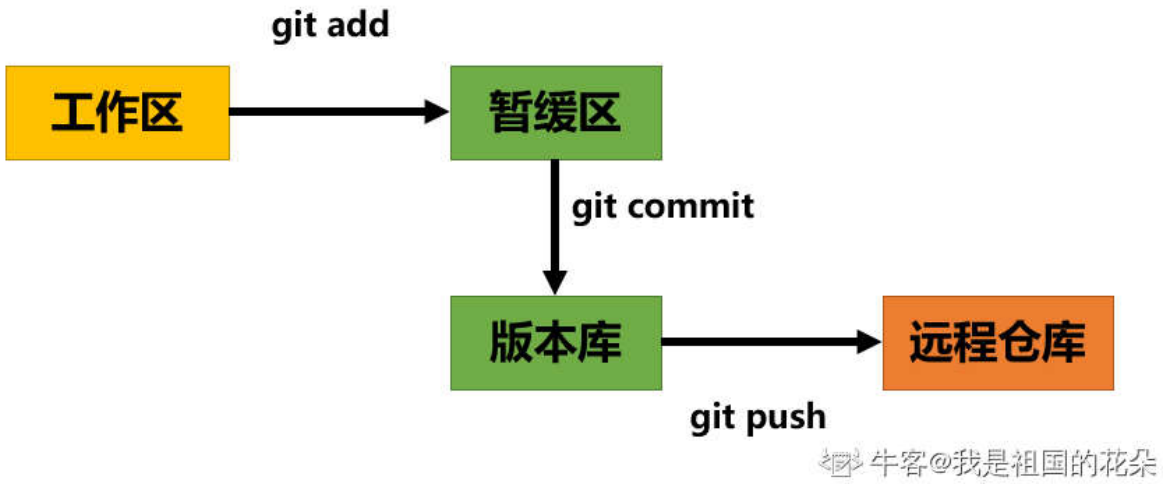
大家好，前面两节我们分别介绍了Linux和Maven的常用命令。本小节中，我们继续介绍版本控制工具Git的相关知识点及其常用命令。

Git是一个优秀的版本控制管理工具，可以帮助我们进行代码版本的管理，目前是市场上主流的版本管理工具。如果你拥有自己的Github，那么相信你一定对git的常用命令比较熟悉了。接下来，我们对Git的相关知识点及其常用命令进行一个简单学习吧。

Git的工作原理

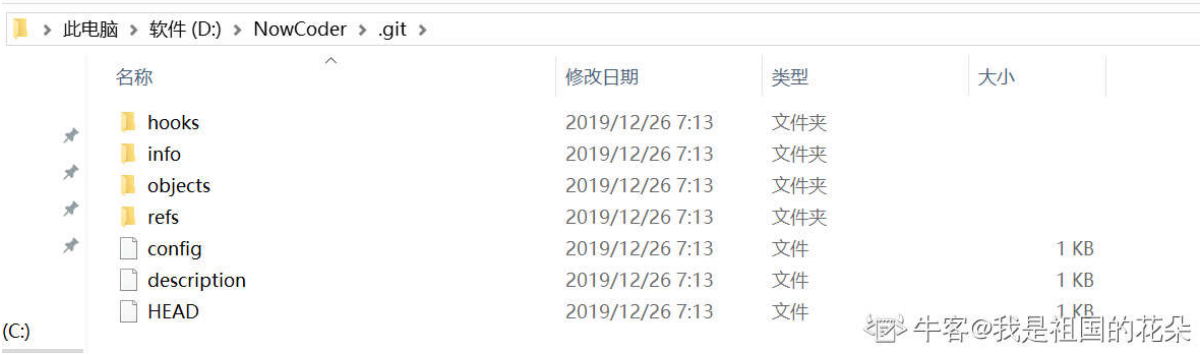
在学习Git之前，我们有必要先了解如下的几个重要概念，并且可以整体上将这副Git工作原理架构图给串联起来。



- **工作区：**也就是在我们本地的一个git仓库目录下，除了隐藏的.git文件之外的所有内容。（说白了，就是我们本地目录里边的内容）
- **暂缓区（Stage）：**数据暂时存放的区域，可在工作区和版本库之间进行数据交互。
- **版本库：**就是我们的Git本地版本仓库，也就是.git目录下就是版本库。通过git push命令可以将本地的版本推送到远程仓库。
- **远程仓库：**是我们在远程进行版本控制的仓库，比如GitLab或者GitHub等服务器上的仓库。

接下来，我们通过示意图来展示下这些重要概念：

这是一个我们本地的git仓库目录，其中.git文件中是本地版本库，其余内容则是我们的**工作区。版本库**内容如下所示：



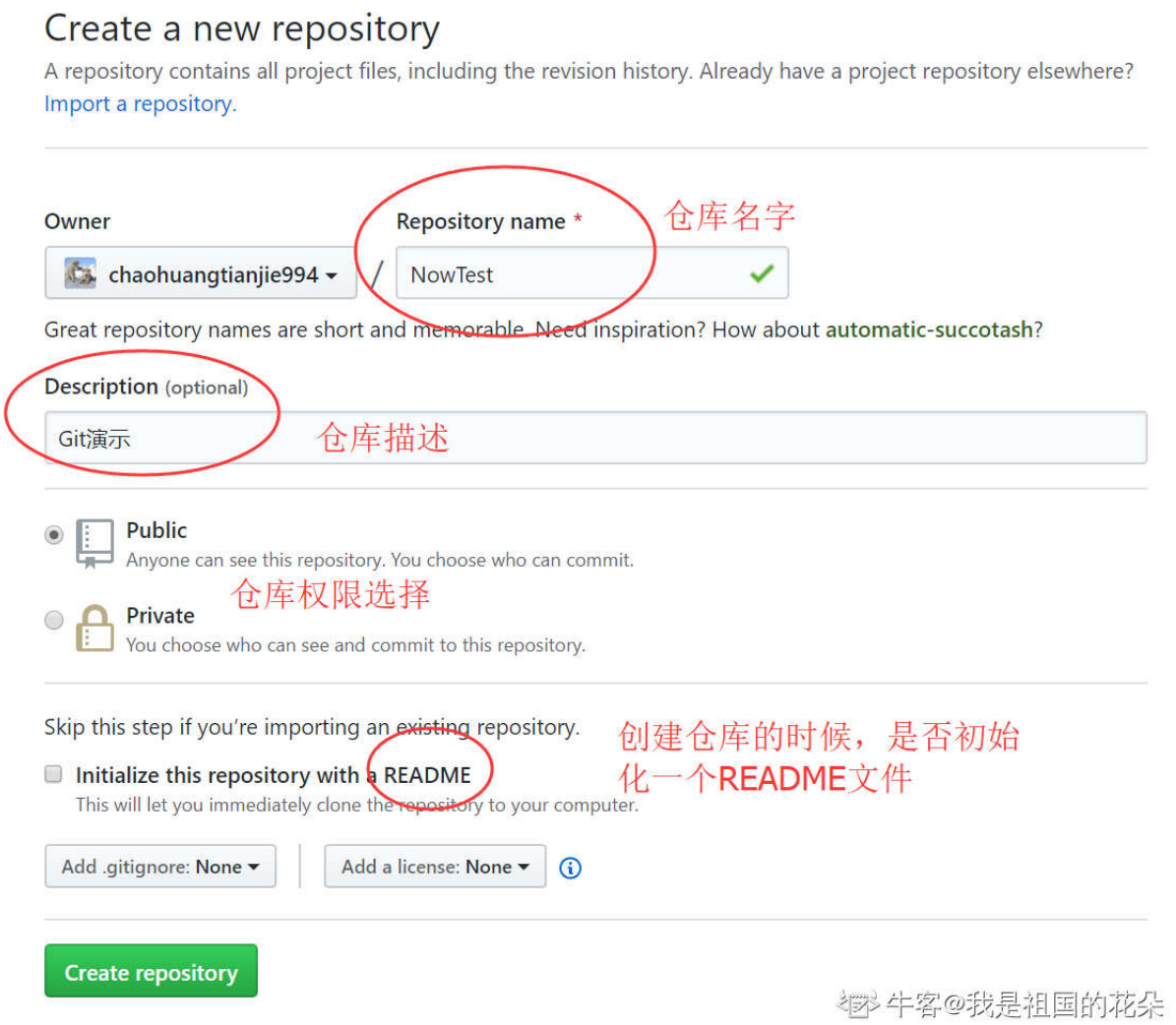
本地版本库中存着分支和版本等有效信息。最后，远程仓库就是我们git服务器上的一个仓库，我们找一个GitHub上的仓库地址如下：

https://github.com/chaohuangtianjie994/BlueTooth-AutoPair.git

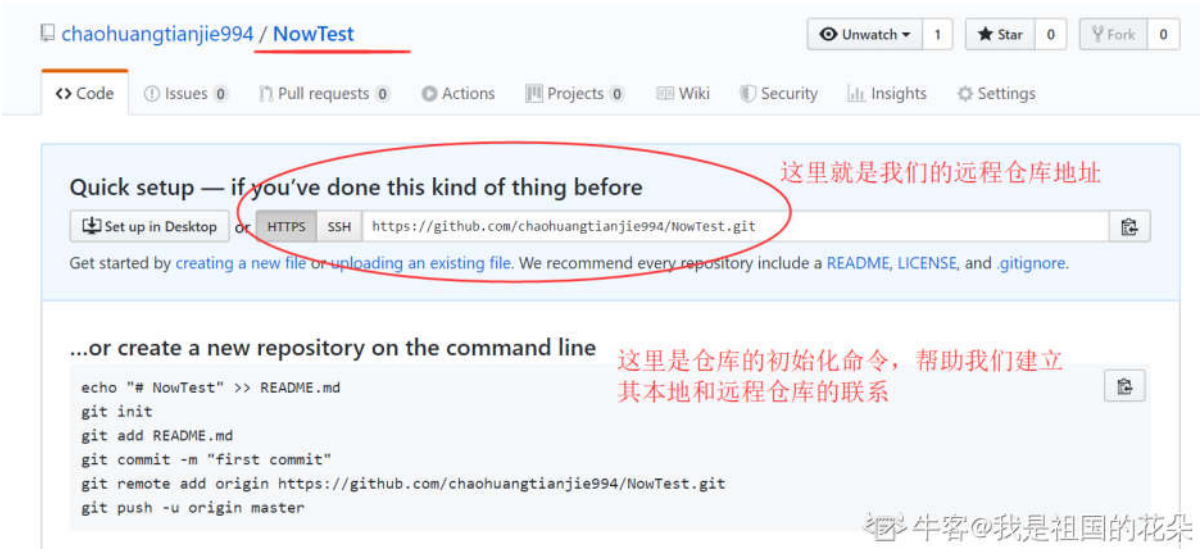
好了，了解了Git的重要概念之后，我们先来进行一个Git仓库的演示。（此处默认大家以及装好了Git，限于篇幅，本小节没有给出如何安装Git，请大家自行学习。

Git仓库Demo演示：

在https://github.com 页面，我们点击New可以进入新建仓库的页面，填入如下的信息：

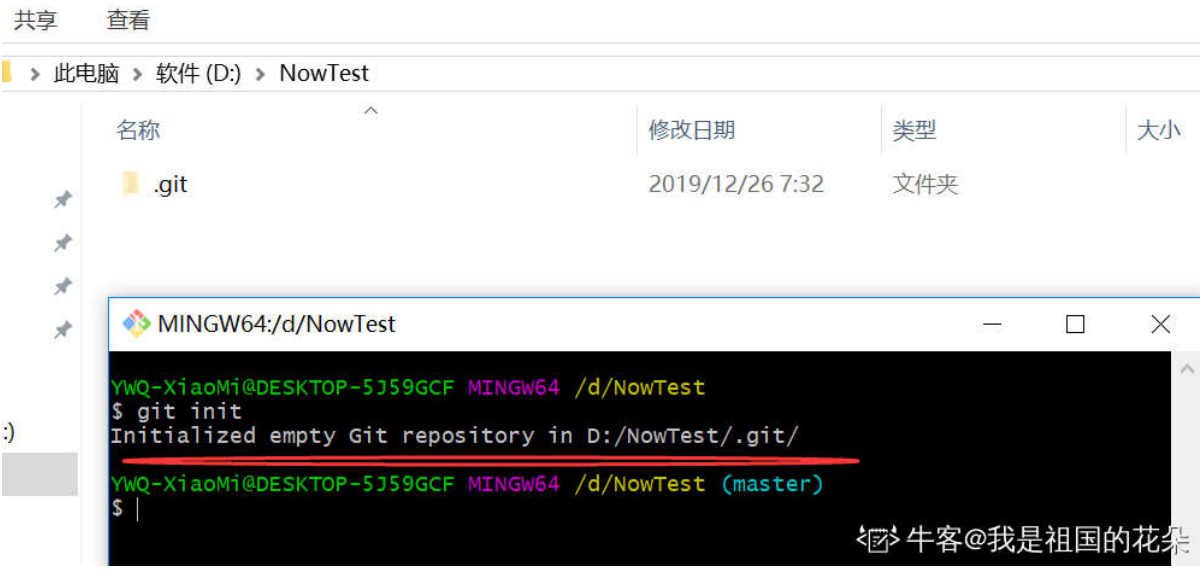


建立成功之后的页面如下所示：

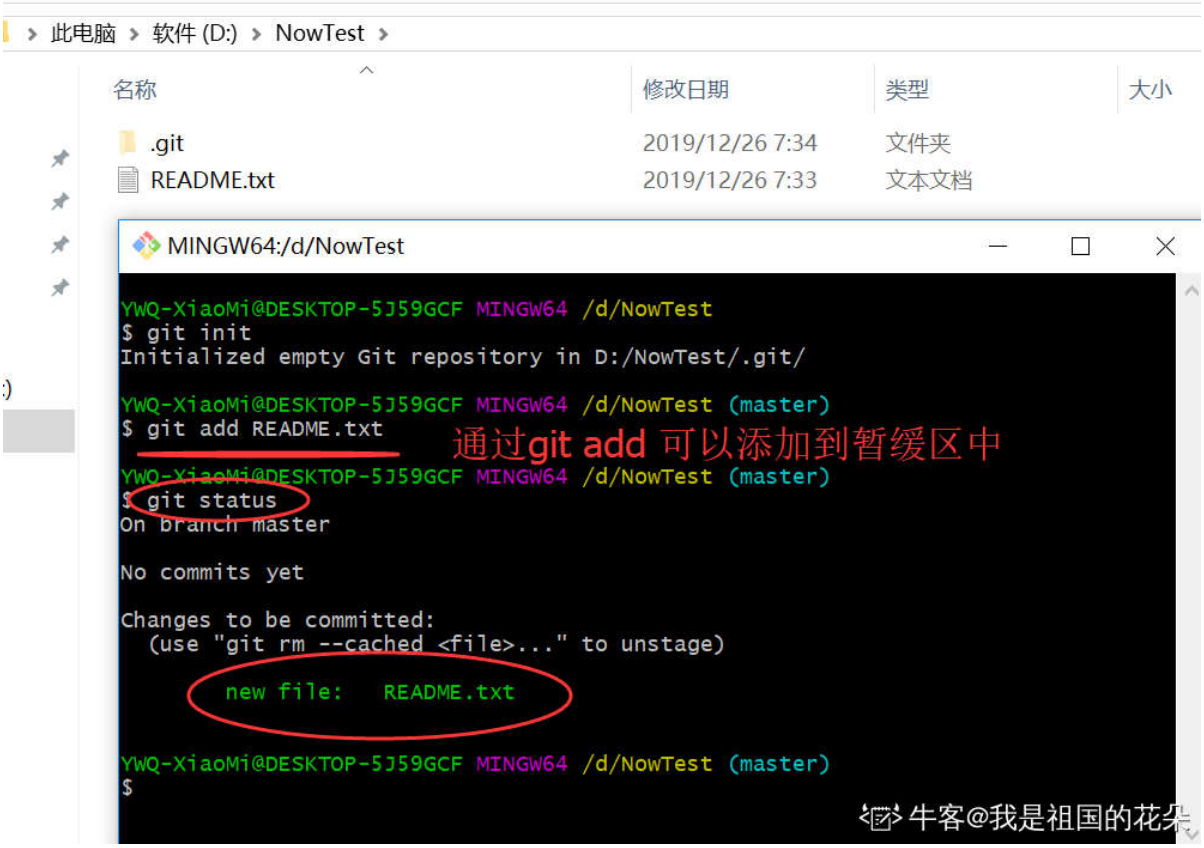


接下来，我们打开一个本地目录，如下所示：

初始化本地仓库：**git init**

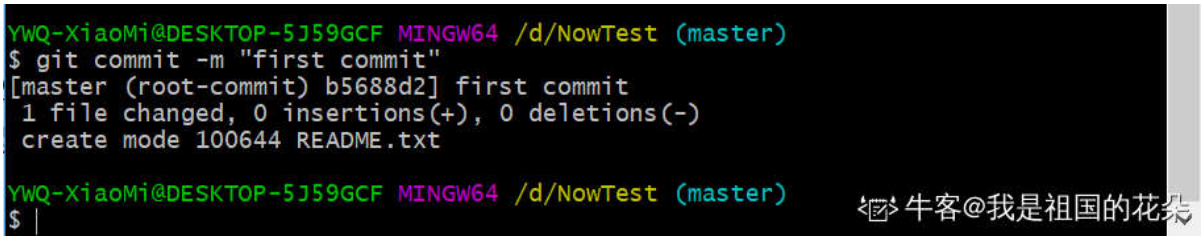


然后，我们添加一个文件README.txt，也就是我们的工作区添加了新的文件，然后进行add操作:**git add**



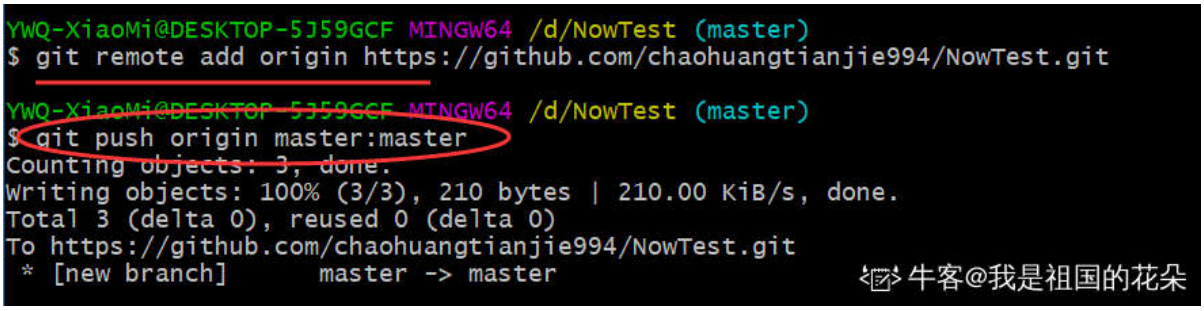
上图中，我们接着执行了git status命令，用于查看当前的状态，看到了当前有一个文件被提交到了暂缓区中，并且我们可以通过提示命令来撤销add操作。

接下来，我们执行**git commit -m "first commit"**来进行提交：



初始化仓库的时候，我们通过**git remote add origin “远程仓库地址”** 来将本地仓库和远程仓库建立连接。

最后，通过 **git push origin master:master**命令来将本地版本库中的文件推送到了远程的仓库中。



执行到这里，你可能会遇到push的时候需要输入用户名和密码进行验证。没错，在使用git之前，我们需要进行设置：

- **配置全局用户名**：git config --global user.name "用户名"（跟踪是谁在修改提交）

- **配置全局邮箱**：git config --global user.email "邮箱"（多人开发间的沟通）

设置完信息之后，我们告诉了Git仓库，我们是谁。当然在提交的时候需要进行身份的验证，不过一般情况下，当我们使用**HTTPS方式的Git地址**可以输入用户名和密码进行验证。

2

如果你不想每次都输入用户名和密码，那么你可以使用**SSH方式的Git地址**，这样只需要生成并且保存你的密钥到Git仓库就可以直接推送代码啦。（限于文章篇幅，这里具体配置大家可以自行学习）

远程仓库的SSH地址如下所示，我们可以自由切换：

接下来，我们介绍Git的常见命令，这也是面试中重点考察的知识点。

Git常用命令

Git是一个优秀的版本控制工具。我们常用的命令如下：

- **git branch newBranch**：在本地新建一个分支(newBranch)
- **git checkout newBranch**：切换到你的新分支(newBranch)
- **git branch**：查看当前的分支，可以看出当前处于哪个分支。

- **git branch –b newBranch**：在本地新建分支(newBranch)，并且切换过去（低版本Git不支持）

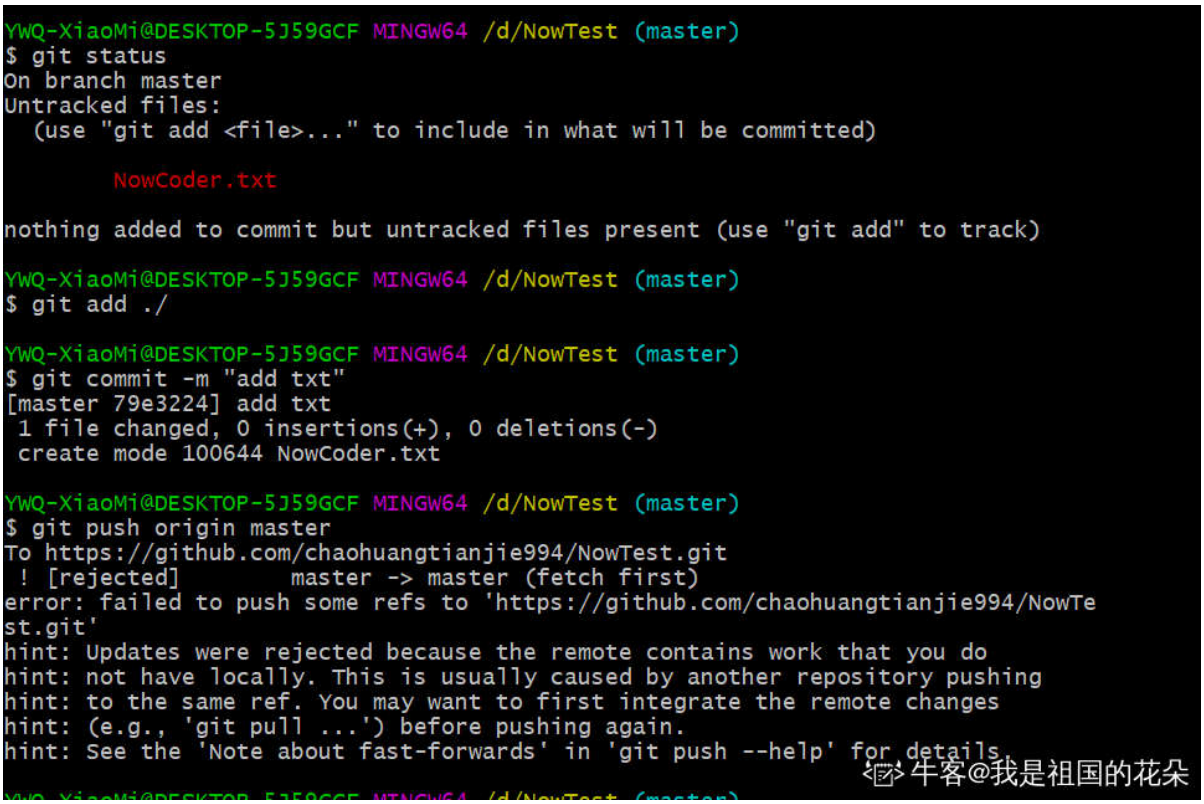
接下来的命令，我们初始化仓库的时候已经进行了演示，这里直接给出作用：

- **git add ./**：将改动提交到暂缓区中
- **git commit –m “this is test”**：将暂缓区中的内容提交到本地分支
- **git push origin newBranch**：将本地分支发布在远程仓库上

接下来，我们演示git pull和git merge 命令。

- **git pull origin master**：将远程master代码更新到本地当前分支

假设远程仓库现在比本地仓库要新，也就是有其余协作开发的同事推送到了远程仓库。这个时候，我们本地进行push会报错如下：



这个时候，我们需要执行git pull命令来更新本地仓库，再次推送，则成功。

```
YWQ-XiaoMi@DESKTOP-5J59GCF MINGW64 /d/NowTest (master)
$ git pull origin master
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
From https://github.com/chaohuangtianjie994/NowTest
 * branch            master     -> FETCH_HEAD
   b5688d2..260ab20  master     -> origin/master
Merge made by the 'recursive' strategy.
 README.txt | 1 +
 1 file changed, 1 insertion(+)

YWQ-XiaoMi@DESKTOP-5J59GCF MINGW64 /d/NowTest (master)
$ git push origin master
Counting objects: 4, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 507 bytes | 253.00 KiB/s, done.
Total 4 (delta 1), reused 0 (delta 0)
remote: Resolving deltas: 100% (1/1), done.
To https://github.com/chaohuangtianjie994/NowTest.git
   260ab20..4d09258  master -> master
```


- **git merge newBranch**：将本地newBranch分支代码合并到当前分支上

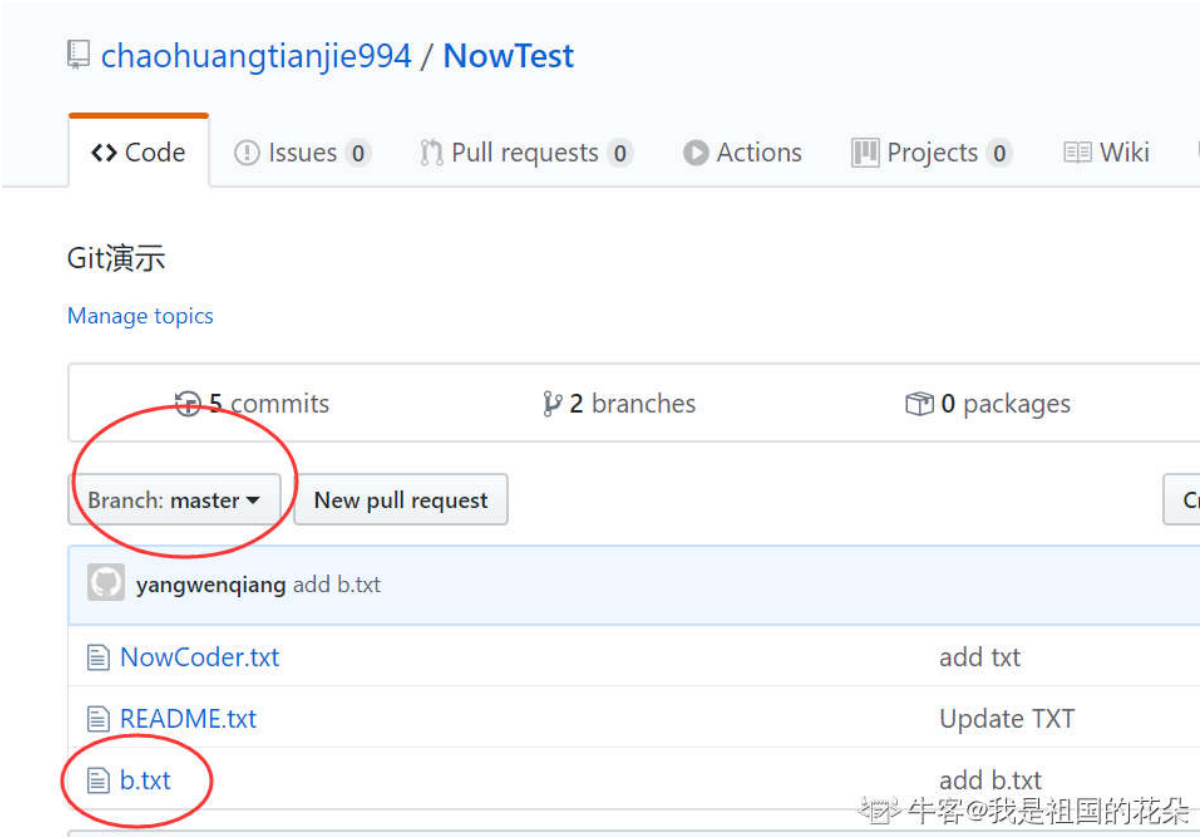
git merge可以将其余分支的内容merge到当前分支上。比如，我们在分支newBranch上创建了一个b.txt文件，需要merge到本地master上。

```
YWQ-XiaoMi@DESKTOP-5J59GCF MINGW64 /d/NowTest (newBranch)
$ git status
On branch newBranch
Untracked files:
  (use "git add <file>..." to include in what will be committed)

        b.txt

nothing added to commit but untracked files present (use "git add" to track)

YWQ-XiaoMi@DESKTOP-5J59GCF MINGW64 /d/NowTest (newBranch)
$ git add ./

YWQ-XiaoMi@DESKTOP-5J59GCF MINGW64 /d/NowTest (newBranch)
$ git commit -m "add b.txt"
[newBranch 27b5499] add b.txt
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 b.txt

YWQ-XiaoMi@DESKTOP-5J59GCF MINGW64 /d/NowTest (newBranch)
$ git push origin newBranch
Counting objects: 2, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (2/2), 227 bytes | 227.00 KiB/s, done.
Total 2 (delta 1), reused 0 (delta 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
remote:
remote: Create a pull request for 'newBranch' on GitHub by visiting:
remote:      https://github.com/chaohuangtianjie994/NowTest/pull/new/newBranch
remote:
To https://github.com/chaohuangtianjie994/NowTest.git
 * [new branch]      newBranch -> newBranch
```


然后，我们切换到master上执行merge命令：

```
YWQ-XiaoMi@DESKTOP-5J59GCF MINGW64 /d/NowTest (master)
$ git merge newBranch
Updating 4d09258..27b5499
Fast-forward
 b.txt | 0
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 b.txt

YWQ-XiaoMi@DESKTOP-5J59GCF MINGW64 /d/NowTest (master)
$ git status
On branch master
nothing to commit, working tree clean

YWQ-XiaoMi@DESKTOP-5J59GCF MINGW64 /d/NowTest (master)
$ git push origin master
Total 0 (delta 0), reused 0 (delta 0)
To https://github.com/chaohuangtianjie994/NowTest.git
   4d09258..27b5499  master -> master

YWQ-XiaoMi@DESKTOP-5J59GCF MINGW64 /d/NowTest (master)
```

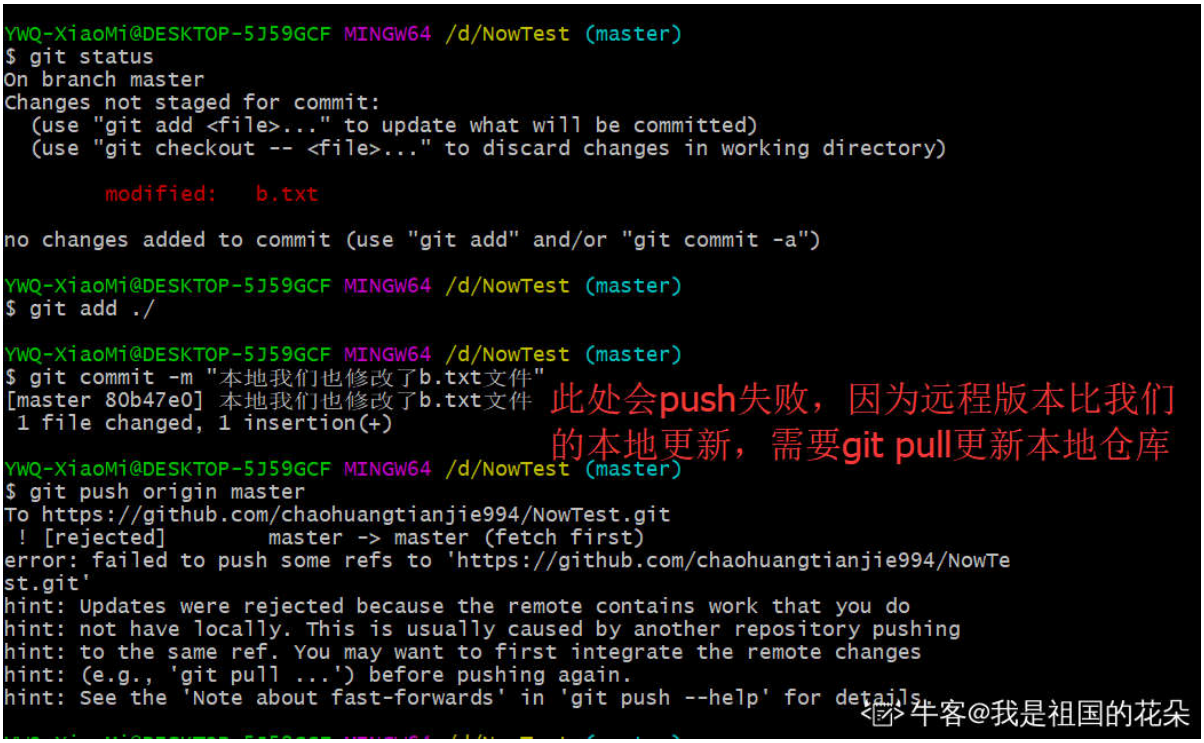

可以看到，远程master分支上也有了b.txt文件：



在git pull origin master将远程master分支更新到本地master的时候，可能会存在冲突。那么，**为什么会产生冲突呢?**

冲突是因为git pull命令会从远程拉内容，并且合并到本地仓库的分支上，如果两个人都对同一个文件进行了修改，在合并的时候不可以自动选择应该以哪个版本为准，也就是产生了冲突，需要我们解决冲突，之后重新add-commit-push。举例如下：

当我们本地也修改了b.txt并且尝试push的时候会出错，如下所示：



这个时候，我们需要通过git pull 命令来更新本地仓库，然后就发生了冲突：

```
YWQ-XiaoMi@DESKTOP-5J59GCF MINGW64 /d/NowTest (master)
$ git pull origin master
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
From https://github.com/chaohuangtianjie994/NowTest
 * branch            master     -> FETCH_HEAD
   27b5499..b4fafdf  master     -> origin/master
Auto-merging b.txt
CONFLICT (content): Merge conflict in b.txt
Automatic merge failed; fix conflicts and then commit the result.
```


那么如何解决冲突呢?

git给出了明确的提示，需要我们去解决冲突，并且重新提交，我们打开b.txt，并且重新修改了该文件之后，重新提交即可：

```
YWQ-XiaoMi@DESKTOP-5J59GCF MINGW64 /d/NowTest (master|MERGING)
$ git status
On branch master
You have unmerged paths.
  (fix conflicts and run "git commit")
  (use "git merge --abort" to abort the merge)

Unmerged paths:
  (use "git add <file>..." to mark resolution)
        both modified:   b.txt      这里是我们解决的冲突文件

no changes added to commit (use "git add" and/or "git commit -a")

YWQ-XiaoMi@DESKTOP-5J59GCF MINGW64 /d/NowTest (master|MERGING)
$ git add ./

YWQ-XiaoMi@DESKTOP-5J59GCF MINGW64 /d/NowTest (master|MERGING)
$ git commit -m "解决冲突"
[master e2cb492] 解决冲突

YWQ-XiaoMi@DESKTOP-5J59GCF MINGW64 /d/NowTest (master)
$ git push origin master
Counting objects: 4, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (4/4), 436 bytes | 436.00 KiB/s, done.
Total 4 (delta 2), reused 0 (delta 0)
remote: Resolving deltas: 100% (2/2), completed with 1 local object.
To https://github.com/chaohuangtianjie994/NowTest.git
   b4fafdf..e2cb492  master -> master
```

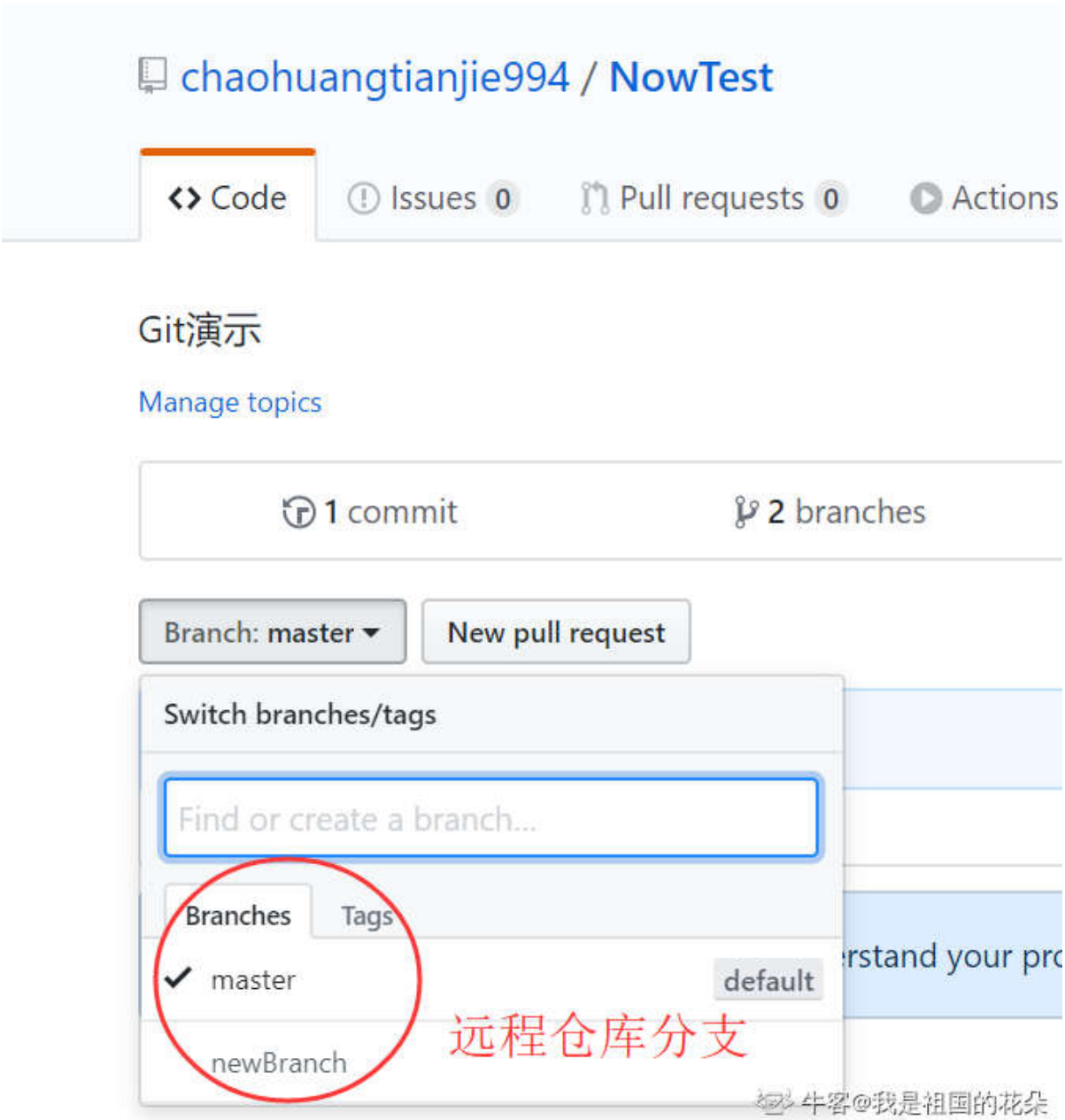

当然了，同样的道理，在merge的时候，遇到冲突，我们还是同样的解决办法。

接下来是删除分支的命令：

- **git branch -d newBranch**：在本地删除一个分支

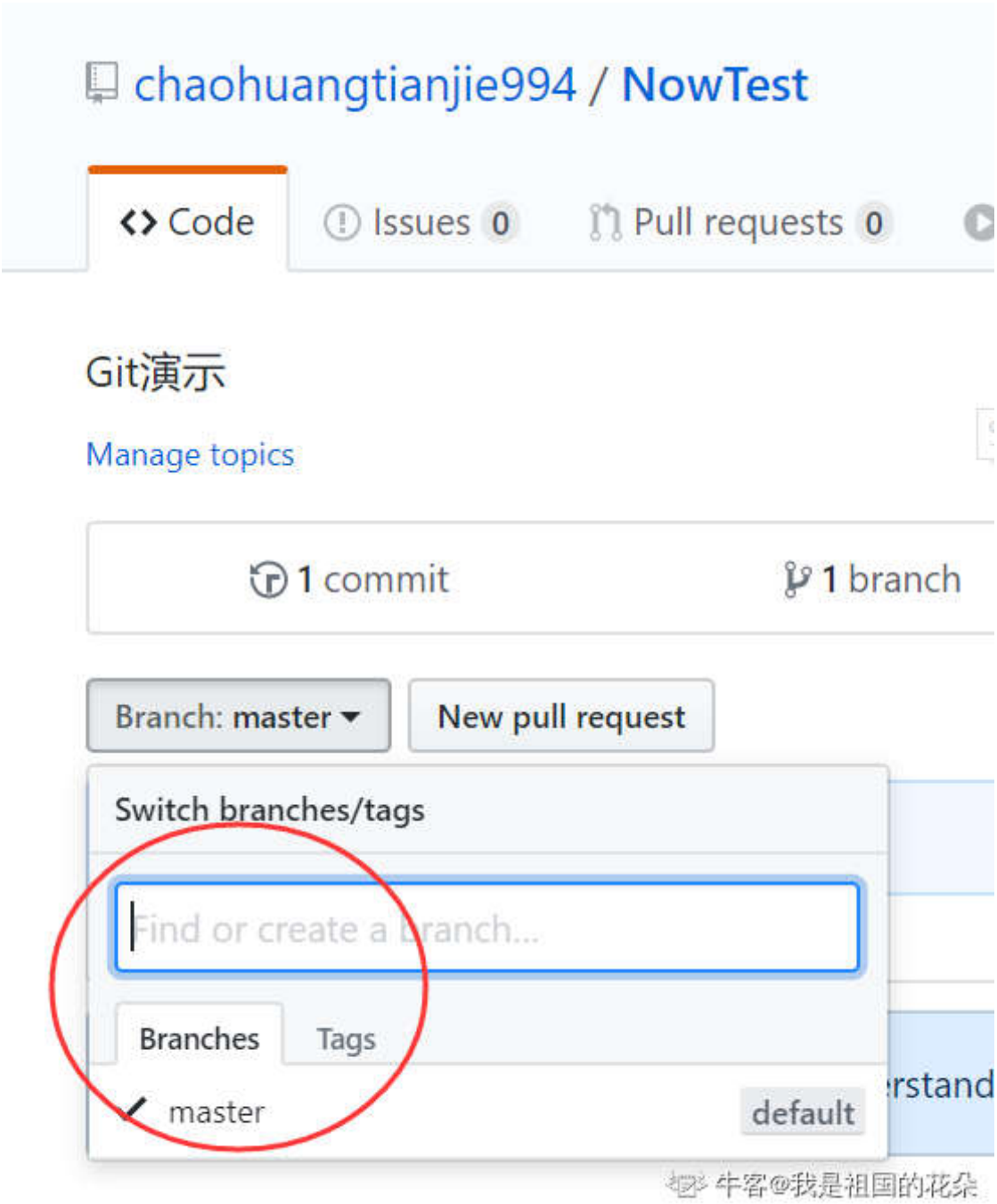
```
YWQ-XiaoMi@DESKTOP-5J59GCF MINGW64 /d/NowTest (master)
$ git branch
* master
  newBranch

YWQ-XiaoMi@DESKTOP-5J59GCF MINGW64 /d/NowTest (master)
$ git branch -d newBranch
Deleted branch newBranch (was b5688d2).

YWQ-XiaoMi@DESKTOP-5J59GCF MINGW64 /d/NowTest (master)
$ git branch
* master
```


- **git push origin :Branch1**：在github远程端删除一个分支 (分支名前的冒号表删除)




```
YWQ-XiaoMi@DESKTOP-5J59GCF MINGW64 /d/NowTest (master)
$ git push origin :newBranch
To https://github.com/chaohuangtianjie994/NowTest.git
 - [deleted]         newBranch

YWQ-XiaoMi@DESKTOP-5J59GCF MINGW64 /d/NowTest (master)
$
```

我们可以看到，远程仓库的newBranch分支已经被删除了：



最后，我们一起看下git中的版本重置命令吧，这个也是面试中经常被问到的知识点。

git reset --hard 和 --soft 的区别是啥？(重点掌握)

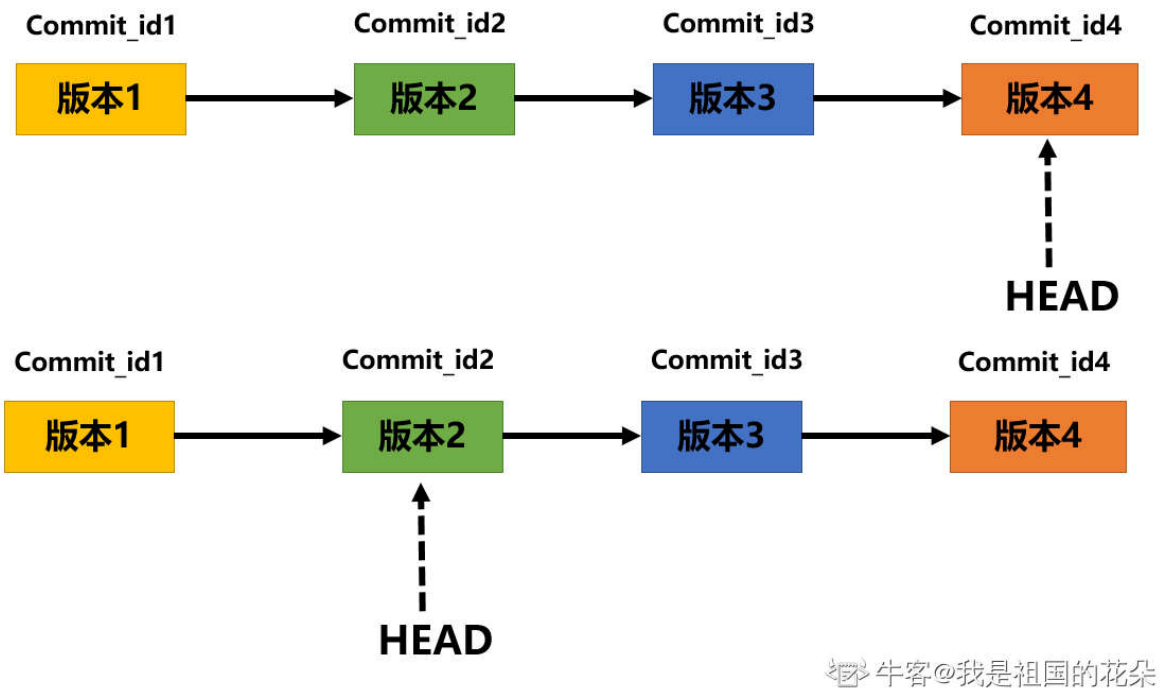
答：通过**git log可以查看当前分支的commit_id提交记录。**当我们想回退到某个版本的时候，可以采用git reset命令，之后通过**git push origin branchName –f** 命令来强制将当前分支push到远程仓库。

- --hard 表示将文件恢复到指定的版本，删除掉了所有的改动，包括工作区和暂缓区。
- --soft 则是将当前分支的HEAD指向指定的版本，并且把指定版本到重置之前的所有改动放入了暂缓区中，经常用来合并多个commit_id。

这里进行一个简单的分析。首先，我们通过git log命令可以看到当前分支上的版本号commit_id，如下所示：

```
YWQ-XiaoMi@DESKTOP-5J59GCF MINGW64 /d/NowTest (master)
$ git log
commit e2cb4920b28cd61cd876ec91248d4b7e49f01ce8 (HEAD -> master, origin/master)
Merge: 80b47e0 b4fafdf
Author: yangwenqiang <yangwenqiang@xiaomi.com>
Date:   Thu Dec 26 20:52:23 2019 +0800

    解决冲突

commit 80b47e0653d8c2821180ac99fda50a053a4e05d7
Author: yangwenqiang <yangwenqiang@xiaomi.com>
Date:   Thu Dec 26 20:46:10 2019 +0800

    本地我们也修改了b.txt文件

commit b4fafdf4339f1899a5d5aaf136b73162009a506e
Author: YWQ <956283501@qq.com>
Date:   Thu Dec 26 20:43:33 2019 +0800

    修改b.txt文件

commit 27b549995e1084d9ae5d51c19656143b5024791e (origin/newBranch, newBranch)
Author: yangwenqiang <yangwenqiang@xiaomi.com>
Date:   Thu Dec 26 09:06:34 2019 +0800

    add b.txt
```


HEAD指向了我们最新的一个版本号，这个时候，我们突然发现版本3和版本4有问题，需要回退到版本2上，那么我们可以执行git reset --hard commit_id2，执行结束之后效果的示意图如下所示：



我们在实际的项目中，给大家演示：

```
YWQ-XiaoMi@DESKTOP-5J59GCF MINGW64 /d/NowTest (master)
$ git reset --hard b4fafdf4339f1899a5d5aaf136b73162009a506e
HEAD is now at b4fafdf 修改b.txt文件

YWQ-XiaoMi@DESKTOP-5J59GCF MINGW64 /d/NowTest (master)
$ git log
commit b4fafdf4339f1899a5d5aaf136b73162009a506e (HEAD -> master)
Author: YWQ <956283501@qq.com>
Date:   Thu Dec 26 20:43:33 2019 +0800

    修改b.txt文件

commit 27b549995e1084d9ae5d51c19656143b5024791e (origin/newBranch, newBranch)
Author: yangwenqiang <yangwenqiang@xiaomi.com>
Date:   Thu Dec 26 09:06:34 2019 +0800

    add b.txt
```


reset之后，我们可以看到最新的两个版本不存在了，当前HEAD处于新的版本号上。

如果，我们现在又后悔了，想要reset回去，那么通过git reflog可以查看所有的版本号（包括刚刚干掉的版本号，保证git保存的数据不会丢失），然后再次reset即可回去。如下所示：

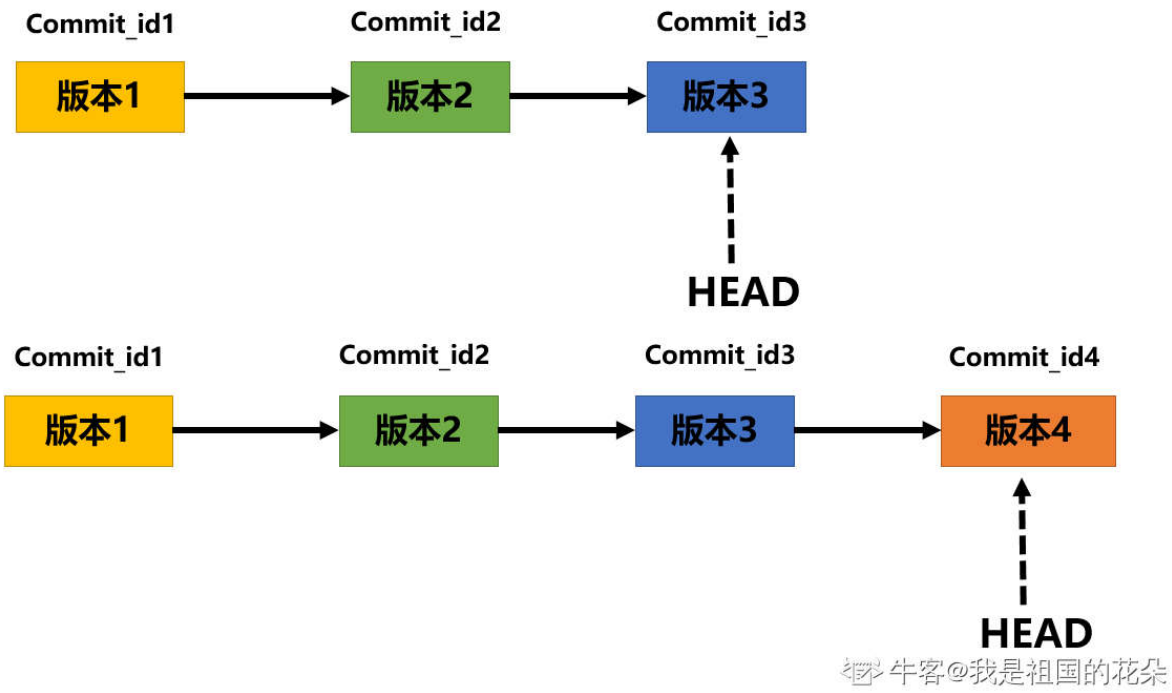
```
YWQ-XiaoMi@DESKTOP-5J59GCF MINGW64 /d/NowTest (master)
$ git reflog
b4fafdf (HEAD -> master) HEAD@{0}: reset: moving to b4fafdf4339f1899a5d5aaf136b73162009a506e
e2cb492 (origin/master) HEAD@{1}: commit (merge): 解决冲突
80b47e0 HEAD@{2}: commit: 本地我们也修改了b.txt文件
27b5499 (origin/newBranch, newBranch) HEAD@{3}: merge newBranch: Fast-forward
4d09258 HEAD@{4}: checkout: moving from newBranch to master
27b5499 (origin/newBranch, newBranch) HEAD@{5}: commit: add b.txt
4d09258 HEAD@{6}: checkout: moving from master to newBranch
4d09258 HEAD@{7}: pull origin master: Merge made by the 'recursive' strategy.
79e3224 HEAD@{8}: commit: add txt
b5688d2 HEAD@{9}: checkout: moving from newBranch to master
b5688d2 HEAD@{10}: checkout: moving from master to newBranch
b5688d2 HEAD@{11}: commit (initial): first commit

YWQ-XiaoMi@DESKTOP-5J59GCF MINGW64 /d/NowTest (master)
$ git reset --hard e2cb492
HEAD is now at e2cb492 解决冲突

YWQ-XiaoMi@DESKTOP-5J59GCF MINGW64 /d/NowTest (master)
$ git log
commit e2cb4920b28cd61cd876ec91248d4b7e49f01ce8 (HEAD -> master, origin/master)
Merge: 80b47e0 b4fafdf
Author: yangwenqiang <yangwenqiang@xiaomi.com>
Date:   Thu Dec 26 20:52:23 2019 +0800

    解决冲突

commit 80b47e0653d8c2821180ac99fda50a053a4e05d7
Author: yangwenqiang <yangwenqiang@xiaomi.com>
Date:   Thu Dec 26 20:46:10 2019 +0800

    本地我们也修改了b.txt文件

commit b4fafdf4339f1899a5d5aaf136b73162009a506e
Author: YWQ <956283501@qq.com>
Date:   Thu Dec 26 20:43:33 2019 +0800

    修改b.txt文件

commit 27b549995e1084d9ae5d51c19656143b5024791e (origin/newBranch, newBranch)
Author: yangwenqiang <yangwenqiang@xiaomi.com>
Date:   Thu Dec 26 09:06:34 2019 +0800

    add b.txt
```


git reset 和 git revert 的区别是啥？（掌握）

答：git reset是通过回退HEAD来完成版本回退。git revert是生成了一次新的提交，新的提交和你想去掉的那次提交的内容刚好相反，本质上是一次新的提交，也就是HEAD继续向前进。

解析：

git revert的意思是版本继续先后前进，当然commit_id4和commit_id3所做的操作刚刚相反。这种情况一般用于，我们只想去掉某一个坏的版本而不是去掉所有的版本。



总结：

本专刊的第六章重点讲述的是常用工具的命令，并不是给大家详细的介绍Linux，Maven以及Git的知识点，本章的目的是在和大家交流**面试中容易被问到的一些基础命令和作用**，当然，对于基础的Maven和Git入门知识点也做了一个简单的介绍与交流。抛砖引玉，每一个命令的熟悉都需要大家多加练习才行。

限于作者水平，文章中难免会有不妥之处。大家在学习过程中遇到我没有表达清楚或者表述有误的地方，欢迎随时在文章下边指出，我会及时关注，随时改正。另外，大家有任何话题都可以在下边留言，我们一起交流探讨。

讨论 评论

刘畅201904211606816 1#

打卡

发表于 2020-01-14 11:51:51 赞(0) 回复(0)

牧水s N 2#

打卡

发表于 2020-03-08 20:19:20 赞(0) 回复(0)