Git:使用遥控器

约瑟夫·哈利特

2023 年 1 月 11 日

这到底是怎么回事？

上次我们谈到了如何使用Git来管理本地文件的更改。

这次

而不是创建我们自己的新仓库：

- 让我们克隆别人的吧！
- 让我们与其他人分享这些变化！

去中心化是什么意思？

在第一讲中我们说过Git是一个去中心化的版本控制系统

- ▶与SVN/CVS 等集中式相反

这在实践中意味着您的本地存储库应该具有存储库的完整历史记录。 ▶并且应该能够充当存储库的主副本。 ▶ （这又是一个简化，

但还是这样吧……）

假设 Alice 在 ~alice/coursework/ 中有一个课程小组作业的 Git 存储库

- ▶ Bob 希望与 Alice 合作并获得自己的副本

所以鲍勃跑了……

$ git clone ~alice/coursework ~bob/coursework 克隆到“~bob/coursework”...完成。

$ cd ~bob/课程作业； git log --oneline af3818c 陈述有关本课程的真实事实
7e 制作课程作业
cc-O2-管 -o 课程作业 coursework.c

./coursework 软件
工具很酷！
你好世界！ b311c
课程作业初稿

$ make coursework cc
-O2 -pipe -o coursework coursework.c

$ ./coursework 软件工
具很酷！
你好世界！

哦不:一个错误!

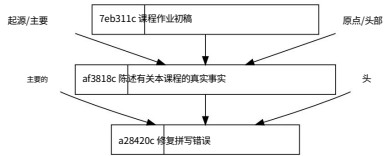
鲍勃可以为爱丽丝解决这个问题!

```
$ ed coursework.c 124
4c

        printf(    软件工具很酷!\n    );
。

wq 124

$ git add coursework.c

$ git commit -m     修复拼写错误     [main a28420c] 修复
拼写错误 1 个文件已更改,1 个插入(+),1 个删除(-)
```

所以现在

在 Alice 的仓库中

在 Bob 的仓库中



那么我们如何将鲍勃的更改返回给爱丽丝呢？

有多种方法！
Bob 可以向 Alice 发送基于补丁的更改Alice 可以基于拉取 Bob 的更改

基于补丁的方法（鲍勃的结局）

这就是Linux 内核和许多其他开源项目管理提交的方式。

Bob 首先准备一个补丁$ git format-patch

origin/main \ --to=alice@bristol.ac.uk 0001-
Fixes-spelling-mistake.patch

并将其发送给爱丽丝

- （查看 git send-email！）

来自 a28420cd5c45d06c9a51625d5a03c37bb77e2ca9 9 月 17 日星期一 00:00 发件人：bob <bob@bristol.ac.uk> 日期:2022 年
11 月 22 日星期二 11:53:04 +0000 主题:[PATCH] 修复拼写错误 收件人:alice@bristol.ac。英国

课程作业.c | 2 +- 1 个文件
更改,1 个插入(+),1 个删除(-)

```
diff --git a/coursework.cb/coursework.c index
2e191f8..8927b2f 100644 --- a/
coursework.c            b/
coursework.c @@ -1,7
+1,7 @@ #include <
 stdio.h>

 int main(void)
-       { printf(    软件工具很酷!\n    ); printf(    软件工
+       具很酷!\n    ); printf(    你好世界!\n    );返回0；


 }
--
2.38.1
```

基于补丁的方法（爱丽丝的结局）

爱丽丝审查鲍勃的补丁,如果他们喜欢它……将其应用到他们的树上。 $ git am ../bob/0001-Fixes-spelling-mistake.patch 应用:修复拼写错误

$ git log --oneline 575dcde
修复拼写错误 af3818c 陈述有关本课程的真实
事实 7eb311c 课程作业初稿

然而,该 ID 与 Bob 最新提交的树不同

- （因为是爱丽丝犯的）。

如果 Bob 想让 ID 与 Alice 保持同步,他们需要重新克隆

- （或 git fetch origin）。

从技术上

讲,git am 实际上是将几个 git 命令合并成一个.....

- 首先,它使用补丁运行 git apply 来暂存它将进行的所有更改
- 然后,它运行 git commit 并使用补丁中提供的提交消息

有很多git命令实际上是链接在一起的较低级别命令 - 小心它们!

另一种选择

当您从事开源工作或想要严格控制如何集成其他人的工作时,基于补丁和电子邮件的工作流程非常有用。

另一种方法是让 Git 为您完成工作并信任其他人。
鲍勃告诉爱丽丝他们修正了一个错误。

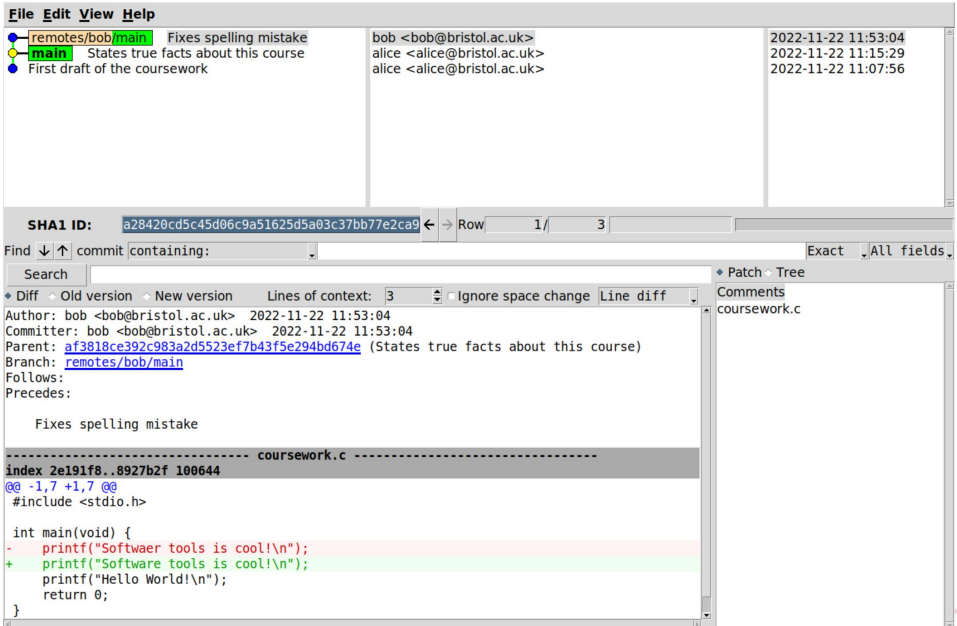
- Alice 将 Bob 的存储库添加为远程 $ git

Remote add bob ~bob/coursework

$ git fetch
Remote:枚举对象:5,完成。远程:计数对象:100% (5/5),
完成。远程:压缩对象:100% (2/2),完成。远程:总共 3 个（增量
1），重用 0 个（增量 0），打包重用 0 解包对象:100% (3/3),255 字
节 | 127.00 KiB/s,完成。

来自 ~bob/coursework * [新
分支] 主要的 -> 鲍勃/手

爱丽丝检查鲍勃的变化.....



如果他们高兴的话……

$ git pull bob main 来自 ~bob/
coursework *branch main 更新
af3818c..a28420c 快进 coursework.c | -> FETCH_HEAD
2 +- 1 个文件更改、1 个插入(+)、1 个删除(-)

再说一遍，

git pull 命令实际上是一个复合命令。以下是等效的： $ git fetch bob

$ git 日志 | cat 提交
a28420cd5c45d06c9a51625d5a03c37bb77e2ca9 作者：bob
<bob@bristol.ac.uk> 日期：2022 年 11 月 22 日星期二 11:53:04 +0000

修正拼写错误

提交 af3818ce392c983a2d5523ef7b43f5e294bd674e 作者：alice
<alice@bristol.ac.uk> 日期：2022 年 11 月 22 日星期二 11:15:29 +0000

陈述有关本课程的真实事实

$ git merge --ff bob/main 更新
af3818c..a28420c 快进

课程作业.c | 2 +- 1 个文件已更改，1 个插入(+

另请注意，这样 Git 提交 ID 就会保持同步：-D。

但是 Github 呢?

Github 为您提供了一个集中式远程（称为forge）：▶您可以注册一个帐户▶为用户设置访问权限▶然后使用 git Push 命令将提交集中发送给每个人▶也可以托管项目页面并构建基础设施

Github 归微软所有：

▶有些人不喜欢这样▶围绕人工智能和开源的一些可疑行为

备择方案：

https://bitbucket.org由 Atlassian 所有https://gitlab.com可以自行托管https://sr.ht由德鲁·德沃特 (Drew DeVault) 拥有……需要花钱自助托管?您所需要的只是一个 SSH 服务器……

▶（搜索裸存储库以了解如何操作）

使用锻造

在鲍勃的仓库中

```
$ git Remote set-url origin \
    git@github.com:alice/coursework

$ git status
Onbranch main 您的
分支比“origin/main”领先 1 个提交。 （使用“git push”发布您的本地提交）


没有什么可提交的,工作树干净

$ git 推送
一切都是最新的
```

在爱丽丝的仓库中

```
$ git Remote -v
                    git@github.com:alice/coursework (fetch) 远程
git@github.com:alice/coursework (push) 远程

$ git pull Remote main 来自
git@github.com:alice/coursework main * 分支 ->
  FETCH_HEAD 更新 af3818c..a28420c 快进 coursework.c | 2 +- 1 个文件更
改,1 个插入(+),1 个删除(-)
```

SSH 密钥要使用伪造，

您通常需要使用 SSH 进行身份验证。这意味着您需要使用密钥：在 Mac/Linux/BSD 上

生成密钥：

$ ssh-keygen 生
成公共/私有 rsa 密钥对。

输入要保存密钥的文件 (/home/joseph/.ssh/id_rsa)： /home/joseph/.ssh/github 输入密码（无密码则为空）： 再次输入相同的密码： 您的身份已保存在 / home/joseph/.ssh/github 您的公钥已保存在 /home/joseph/.ssh/github.pub 密钥指纹为： SHA256:L1Xg2W2I1EgXocqPpuSefRbYlgop/

H2x5TXI4EKs0qA joseph@bristol.ac.uk 密钥的随机艺术图像

为： + ---[RSA 3072]----+ | | .+o+o | |奥博 | =.+ o | | ....o..| | + | 。os=.+ 。 | o E o=O o + .| | oo o* o 。 | | | | +----[SHA256]-----+

在你的 ~/.ssh/config 文件中：

主机 github.com 用
户 git
IdentityFile ~/.ssh/github

确保将 .pub 公钥而不是私钥上传到 Github！

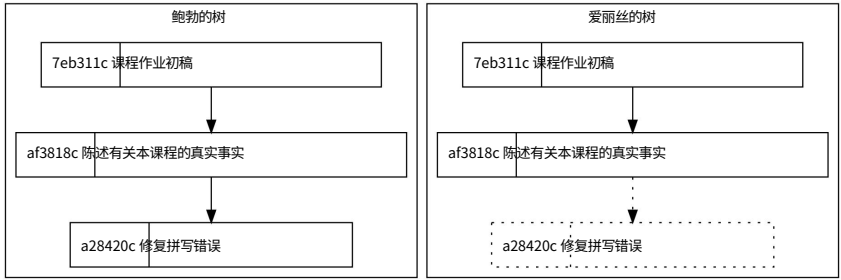
并测试：

$ ssh git@github.com PTY
分配请求失败，您好 uob-jh！您已成功关闭与 github.com 的连接。

关 关 。
.+

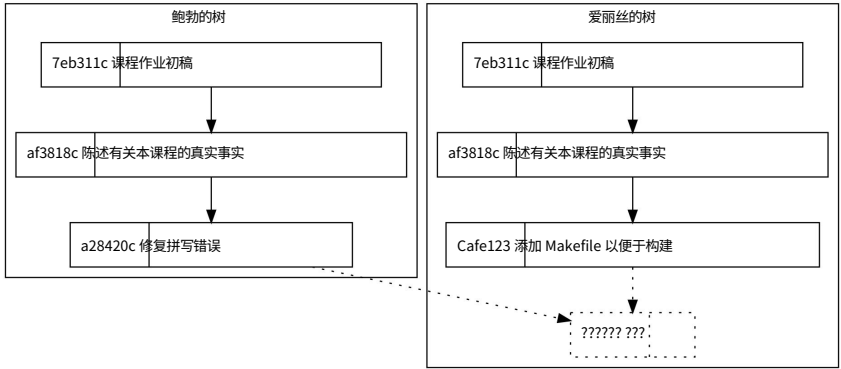
当出现问题时会发生什么？

当爱丽丝早些时候拉动鲍勃的更改时，它们可以快进。 ▶这意味着可以直接将更改拉到并复制到 Alice 的树中。



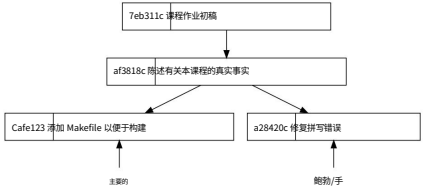
忙、忙、忙……

如果爱丽丝一直很忙并且自己做出了一些承诺怎么办？ ▶现在无法快进，因为树木已经分叉了



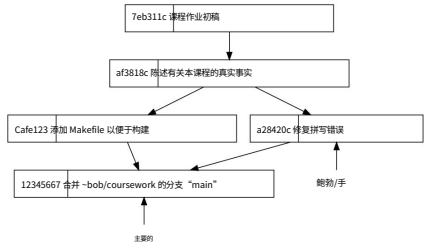
合并

从爱丽丝的角度来看,这就是树的样子




```
$ git merge --no-ff bob/main 提示:正在等
待编辑器关闭文件...
通过“ort”策略进行合并。
 生成文件| 0 1 个文
 件已更改,0 个插入(+),0 个删除(-) 创建模式 100644 Makefile
```

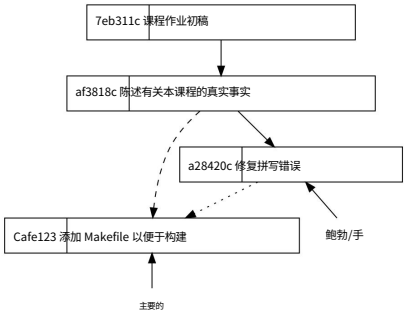
最简单的方法是进行合并并添加显式合并树两个路径中的更改的提交



（但通常它会很聪明,发现您更改了不同的文件并且仍然快进.....）

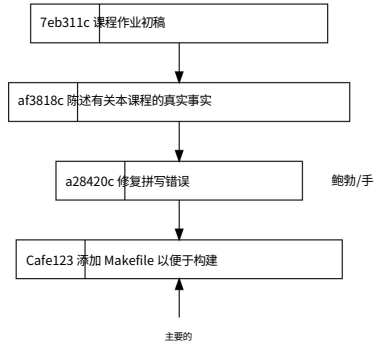
或者,Alice 可以进

行一些时间旅行……让我们假设Alice 的提交是在 Bob 之后的：



$ git rebase bob/main 成功重新建立基础并更新了 refs/heads/main。

然后我们可以像以前一样快进鲍勃的更改,然后重放爱丽丝的新提交。



现在我们又得到了一棵漂亮整齐的直线树！

合并与变基

合并在概念上更简单…… ▶ ……但凌乱的变基更整洁

▶ …但复杂且容易失败▶您还可以使用其他一些巧妙的技巧来使其变得更好（例如 git bisect）

您（和您的雇主)会有意见

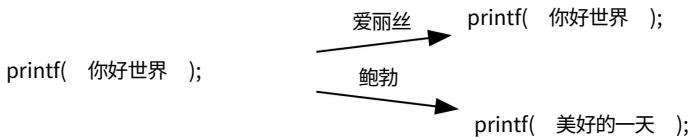
▶这样做。

▶真的没关系▶ （我稍微喜欢合并版本,但我来回切换……）

到目前为止,我们的合并很容易。

- Alice 和 Bob 对不同的文件进行了编辑
- 更改全部可以由 Git 自动完成。

  如果 Alice 和 Bob都更改同一文件中的相同行,会发生什么情况?



$ git merge bob/main 自动
合并 coursework.c CONFLICT
(content): 合并 coursework.c 中的冲突 自动合并失败;修复冲突,然后
提交结果。

让我们解决冲突

Git 发现有两组更改,但它无法确定应该选择哪一组……

如果我们遵循 Git 的说明， coursework.c 看起来像：

```
#include <stdio.h>

int 主函数（无效){
<<<<<<< 头部
    printf(   你好世界\n   );
=======
    printf(   美好的一天\n   );
>>>>>>> 全部/主要返
    回 0;
```

修复文件,然后在看起来不错时运行 git add / git commit... ▶不要只删除它的一侧。 ▶ ……说真的……我见过有人因此被解雇。

```
$ git add coursework $ git
commit [main
16d3aa6] 合并远程跟踪分支    bob/main
```

包起来

▶使用 git Remote 和 git clone 与其他人一起工作▶使用 git fetch 或 git pull 或修补文件来获取其他人的工作▶使用 git merge 或 git rebase 来集成更改▶使用 git Push 将工作发送回 forge ▶合并冲突是一种痛苦,但你必须处理它们

合并工具如果您发现
自己经常处理合并冲突……有一些工具可以帮助您处理它们

https://meldmerge.org处理合并的好工具（我使用 Emacs。）

命令