一、上传文件合并命令

1.1 第一次创建 git 仓库全部步骤：

基本命令就是如下：

$ cd myproject #定位到要上传文件的文件夹，一般情况下右键》git bush here 就是可以了

$ git init # 把此文件夹内容初始化

$ git add . # 把内容加入到 git 缓冲中去（注意 add 与“.”之间有空格）

$ git commit -m 'initial commit' # 添加评论内容（此内容不写，没有文件写入）

$ git remote add origin https://github.com/DreamsFuture/Javase-source-code.git #获得远程 GitHub 的地址，这样可以 push 到这个仓库中去，这里容易出现的问题就是 name“origin”这个本地库名重名，一般一个项目一个仓库“name”，如果想修改名字，可以参考如下：

上面这句指令执行了之后，就会出现用户名和密码输入，然后就执行上传

$ git push origin master

1.2 在已经创建好的仓库时，更新（修改和删除）和增加内容：

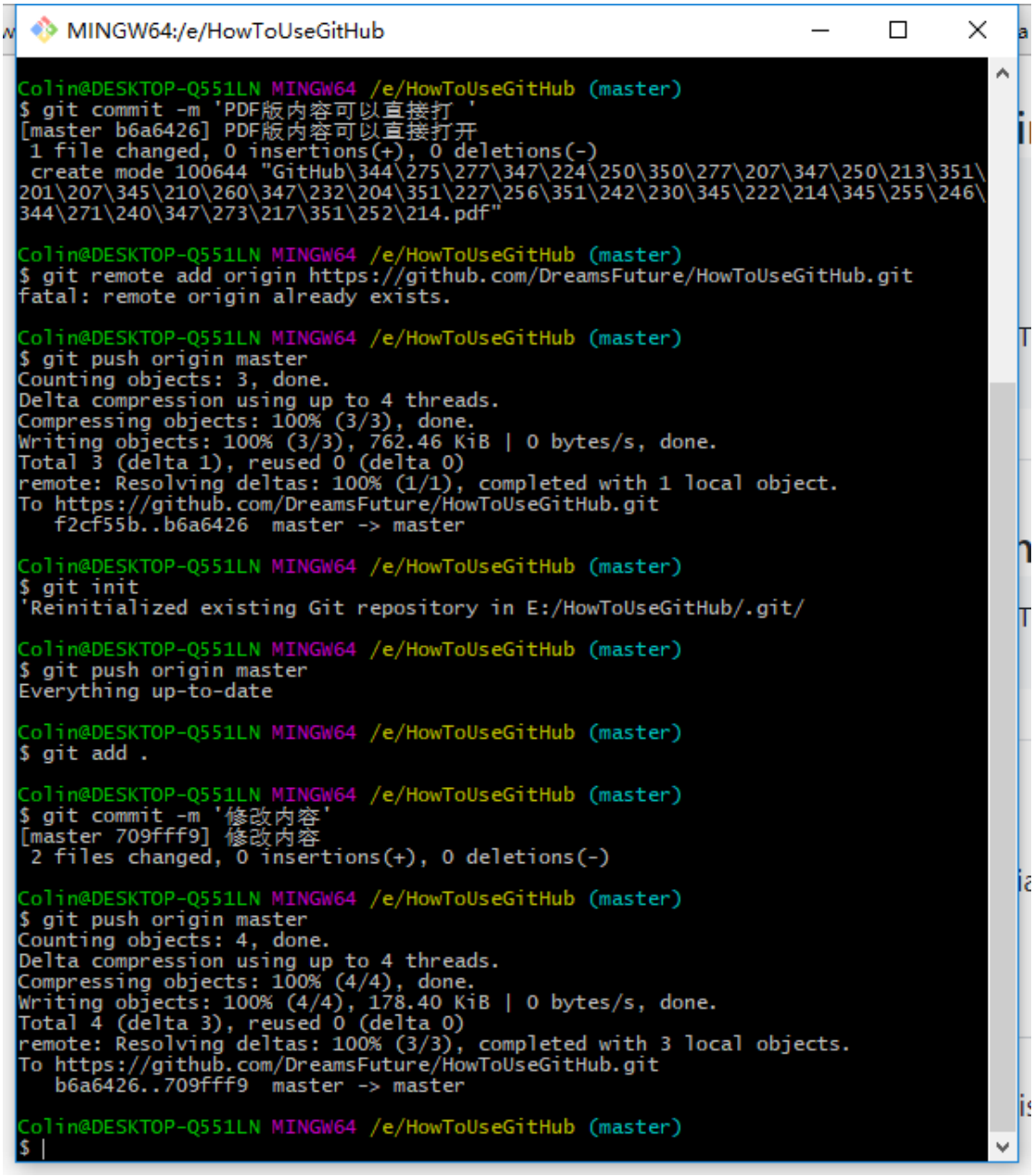
$ cd myproject #定位到要上传文件的文件夹，一般情况下右键》git bush here 就是可以了

$ git add . # 把内容加入到 git 缓冲中去（注意 add 与“.”之间有空格）

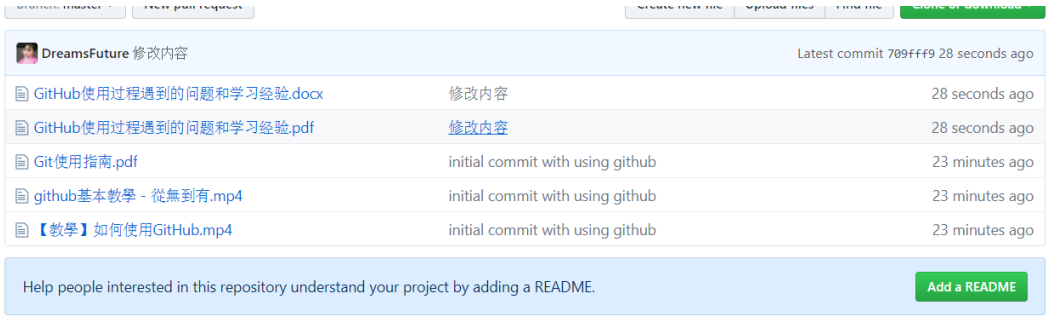
Add 的内容自动识别修改过的文件这样就可以对应添加 comment

$ git commit -m '修改内容' # 添加评论内容（此内容不写，没有文件写入）

$ git push origin master


MINGW64:/e/HowToUseGitHub
Colin@DESKTOP-Q551LN MINGW64 /e/HowToUseGitHub (master)
$ git commit -m 'PDF版内容可以直接打 '
[master b6a6426] PDF版内容可以直接打开
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 "GitHub\344\275\277\347\224\250\350\277\207\347\250\213\351\201\207\345\210\260\347\232\204\351\227\256\351\242\230\345\222\214\345\255\246\344\271\240\347\273\217\351\252\214.pdf"
Colin@DESKTOP-Q551LN MINGW64 /e/HowToUseGitHub (master)
$ git remote add origin https://github.com/DreamsFuture/HowToUseGitHub.git
fatal: remote origin already exists.
Colin@DESKTOP-Q551LN MINGW64 /e/HowToUseGitHub (master)
$ git push origin master
Counting objects: 3, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 762.46 KiB | 0 bytes/s, done.
Total 3 (delta 1), reused 0 (delta 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/DreamsFuture/HowToUseGitHub.git
f2cf55b..b6a6426 master -> master
Colin@DESKTOP-Q551LN MINGW64 /e/HowToUseGitHub (master)
$ git init
'Reinitialized existing Git repository in E:/HowToUseGitHub/.git/
Colin@DESKTOP-Q551LN MINGW64 /e/HowToUseGitHub (master)
$ git push origin master
Everything up-to-date
Colin@DESKTOP-Q551LN MINGW64 /e/HowToUseGitHub (master)
$ git add .
Colin@DESKTOP-Q551LN MINGW64 /e/HowToUseGitHub (master)
$ git commit -m '修改内容'
[master 709fff9] 修改内容
2 files changed, 0 insertions(+), 0 deletions(-)
Colin@DESKTOP-Q551LN MINGW64 /e/HowToUseGitHub (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), 178.40 KiB | 0 bytes/s, done.
Total 4 (delta 3), reused 0 (delta 0)
remote: Resolving deltas: 100% (3/3), completed with 3 local objects.
To https://github.com/DreamsFuture/HowToUseGitHub.git
b6a6426..709fff9 master -> master
Colin@DESKTOP-Q551LN MINGW64 /e/HowToUseGitHub (master)
$

结果如下：


DreamsFuture 修改内容
Latest commit 709fff9 28 seconds ago
GitHub使用过程遇到的问题和学习经验.docx 修改内容 28 seconds ago
GitHub使用过程遇到的问题和学习经验.pdf 修改内容 28 seconds ago
Git使用指南.pdf initial commit with using github 23 minutes ago
github基本教學 - 從無到有.mp4 initial commit with using github 23 minutes ago
【教學】如何使用GitHub.mp4 initial commit with using github 23 minutes ago
Help people interested in this repository understand your project by adding a README.
Add a README

1.3 在另外一台电脑或者另外一个系统中使用相同账号处理一个仓库

$ cd myproject #定位到要上传文件的文件夹，一般情况下右键》git bush here 就是可以了

1.4 从远程仓库拉取数据到本地

$ cd myproject #定位到要上传文件的文件夹，一般情况下右键》git bush here 就是可以了

git fetch https://github.com/DreamsFuture/HowToUseGitHub.git

1.5 Git 基础 - 远程仓库的使用

远程仓库的使用

要参与任何一个 Git 项目的协作，必须要了解该如何管理远程仓库。远程仓库是指托管在网络上的项目仓库，可能会有好多个，其中有些你只能读，另外有些可以写。同他人协作开发某个项目时，需要管理这些远程仓库，以便推送或拉取数据，分享各自的工作进展。 管理远程仓库的工作，包括添加远程库，移除废弃的远程库，管理各式远程库分支，定义是否跟踪这些分支，等等。本节我们将详细讨论远程库的管理和使用。

查看当前的远程库

要查看当前配置有哪些远程仓库，可以用 `git remote` 命令，它会列出每个远程库的简短名字。在克隆完某个项目后，至少可以看到一个名为 origin 的远程库，Git 默认使用这个名字来标识你所克隆的原始仓库：

```
$ git clone git://github.com/schacon/ticgit.git

Cloning into 'ticgit'...

remote: Reusing existing pack: 1857, done.

remote: Total 1857 (delta 0), reused 0 (delta 0)

Receiving objects: 100% (1857/1857), 374.35 KiB | 193.00 KiB/s, done.
```

```
Resolving deltas: 100% (772/772), done.
Checking connectivity... done.
$ cd ticgit
$ git remote
origin
```

也可以加上 `-v` 选项（译注：此为 `--verbose` 的简写，取首字母），显示对应的克隆地址：

```
$ git remote -v
origin  git://github.com/schacon/ticgit.git (fetch)
origin  git://github.com/schacon/ticgit.git (push)
```

如果有多个远程仓库，此命令将全部列出。比如在我的 Grit 项目中，可以看到：

```
$ cd grit
$ git remote -v
bakkdoor  git://github.com/bakkdoor/grit.git
cho45     git://github.com/cho45/grit.git
defunkt   git://github.com/defunkt/grit.git
koke      git://github.com/koke/grit.git
origin    git@github.com:mojombo/grit.git
```

这样一来，我就可以非常轻松地从这些用户的仓库中，拉取他们的提交到本地。请注意，上面列出的地址只有 origin 用的是 SSH URL 链接，所以也只有这个仓库我能推送数据上去（我们会在第四章解释原因）。

添加远程仓库

要添加一个新的远程仓库，可以指定一个简单的名字，以便将来引用，运行 `git remote add [shortname] [url]`：

```
$ git remote
origin
```

```
$ git remote add pb git://github.com/paulboone/ticgit.git

$ git remote -v

origin  git://github.com/schacon/ticgit.git

pb  git://github.com/paulboone/ticgit.git
```

现在可以用字符串 `pb` 指代对应的仓库地址了。比如说，要抓取所有 Paul 有的，但本地仓库没有的信息，可以运行 `git fetch pb`：

```
$ git fetch pb

remote: Counting objects: 58, done.

remote: Compressing objects: 100% (41/41), done.

remote: Total 44 (delta 24), reused 1 (delta 0)

Unpacking objects: 100% (44/44), done.

From git://github.com/paulboone/ticgit

 * [new branch]      master     -> pb/master

 * [new branch]      ticgit     -> pb/ticgit
```

现在，Paul 的主干分支（master）已经完全可以在本地访问了，对应的名字是 `pb/master`，你可以将它合并到自己的某个分支，或者切换到这个分支，看看有些什么有趣的更新。

从远程仓库抓取数据

正如之前所看到的，可以用下面的命令从远程仓库抓取数据到本地：

```
$ git fetch [remote-name]
```

此命令会到远程仓库中拉取所有你本地仓库中还没有的数据。运行完成后，你就可以在本地访问该远程仓库中的所有分支，将其中某个分支合并到本地，或者只是取出某个分支，一探究竟。（我们会在第三章详细讨论关于分支的概念和操作。）

如果是克隆了一个仓库，此命令会自动将远程仓库归于 origin 名下。所以，`git fetch origin` 会抓取从你上次克隆以来别人上传到此远程仓库中的所有更新（或是上次 fetch 以来别人提交的更新）。有一点很重要，需要记住，fetch 命令只是将远端的数据拉到本地仓库，并不自动合并到当前工作分支，只有当你确实准备好了，才能手工合并。

如果设置了某个分支用于跟踪某个远端仓库的分支（参见下节及第三章的内容），可以使用 `git pull` 命令自动抓取数据下来，然后将远端分支自动合并到本地仓库中当前分支。在日常工作中我们经常这么用，既快且好。实际上，默认情况下 `git clone` 命令本质上就是自动创建了本地的 master 分支用于跟踪远程仓库中的 master 分支（假设远程仓库确实有 master 分支）。所以一般我们运行 `git pull`，目的都是要从原始克隆的远端仓库中抓取数据后，合并到工作目录中的当前分支。

推送数据到远程仓库

项目进行到一个阶段，要同别人分享目前的成果，可以将本地仓库中的数据推送到远程仓库。实现这个任务的命令很简单：`git push [remote-name] [branch-name]`。如果要把本地的 master 分支推送到 `origin` 服务器上（再次说明下，克隆操作会自动使用默认的 master 和 origin 名字），可以运行下面的命令：

```
$ git push origin master
```

只有在所克隆的服务器上有写权限，或者同一时刻没有其他人在推数据，这条命令才会如期完成任务。如果在你推数据前，已经有其他人推送了若干更新，那你的推送操作就会被驳回。你必须先把他们的更新抓取到本地，合并到自己的项目中，然后才可以再次推送。有关推送数据到远程仓库的详细内容见第三章。

查看远程仓库信息

我们可以通过命令 `git remote show [remote-name]` 查看某个远程仓库的详细信息，比如要看所克隆的 `origin` 仓库，可以运行：

```
$ git remote show origin
* remote origin
  URL: git://github.com/schacon/ticgit.git
  Remote branch merged with 'git pull' while on branch master
    master
  Tracked remote branches
    master
    ticgit
```

除了对应的克隆地址外，它还给出了许多额外的信息。它友善地告诉你如果是在 master 分支，就可以用 `git pull` 命令抓取数据合并到本地。另外还列出了所有处于跟踪状态中的远端分支。

上面的例子非常简单，而随着使用 Git 的深入，`git remote show` 给出的信息可能会像这样：

```
$ git remote show origin
* remote origin
  URL: git@github.com:defunkt/github.git
  Remote branch merged with 'git pull' while on branch issues
    issues
  Remote branch merged with 'git pull' while on branch master
    master
  New remote branches (next fetch will store in
remotes/origin)
    caching
  Stale tracking branches (use 'git remote prune')
    libwalker
    walker2
  Tracked remote branches
    acl
    apiv2
    dashboard2
    issues
    master
    postgres
  Local branch pushed with 'git push'
    master:master
```

它告诉我们，运行 `git push` 时缺省推送的分支是什么（译注：最后两行）。它还显示了有哪些远端分支还没有同步到本地（译注：第六行的 `caching` 分支），哪些已同步到本地的远端分支在远端服务器上已被删除（译注：`Stale tracking branches` 下面的两个分支），以及运行 `git pull` 时将自动合并哪些分支（译注：前四行中列出的 `issues` 和 `master` 分支）。

远程仓库的删除和重命名

在新版 Git 中可以用 `git remote rename` 命令修改某个远程仓库在本地的简称，比如想把 `pb` 改成 `paul`，可以这么运行：

```
$ git remote rename pb paul

$ git remote

origin

paul
```

注意，对远程仓库的重命名，也会使对应的分支名称发生变化，原来的 `pb/master` 分支现在成了 `paul/master`。

碰到远端仓库服务器迁移，或者原来的克隆镜像不再使用，又或者某个参与者不再贡献代码，那么需要移除对应的远端仓库，可以运行 `git remote rm` 命令：

```
$ git remote rm paul

$ git remote

origin
```

2. Github 自动给出的指令方法：

2.1 …or create a new repository on the command line

```
echo "# Test" >> README.md
git init
git add README.md （本地应该有才能增加）
git commit -m "first commit"
git remote add origin https://github.com/DreamsFuture/Test.git
git push -u origin master
```

2.2 push an existing repository from the command line

```
git remote add origin https://github.com/DreamsFuture/Test.git
    git push -u origin master
```

自己理解的深度

问题分析

如下图 Figure 1 所示，选中文件夹，上传的不是本文件夹，而是此文件夹的信息，所以要根据具体内容来上传；

https://github.com/DreamsFuture/Javase-source-code.git 信息是从 Figure 1 右上框获取的，所以必须先在 GitHub 网址上先创建内容，才能把东西上传上去；

至于下面的那个 **md** 文件，就是我创建 **GitHub** 仓库时附带的，但是我自己使用 **commit** 这个命令时出现问题，导致我之前一直没有做成功过；

但是最后我再一次重新试了一遍，然后跳过了一些步骤之后，就成功了，如下图 Figure 2 所示

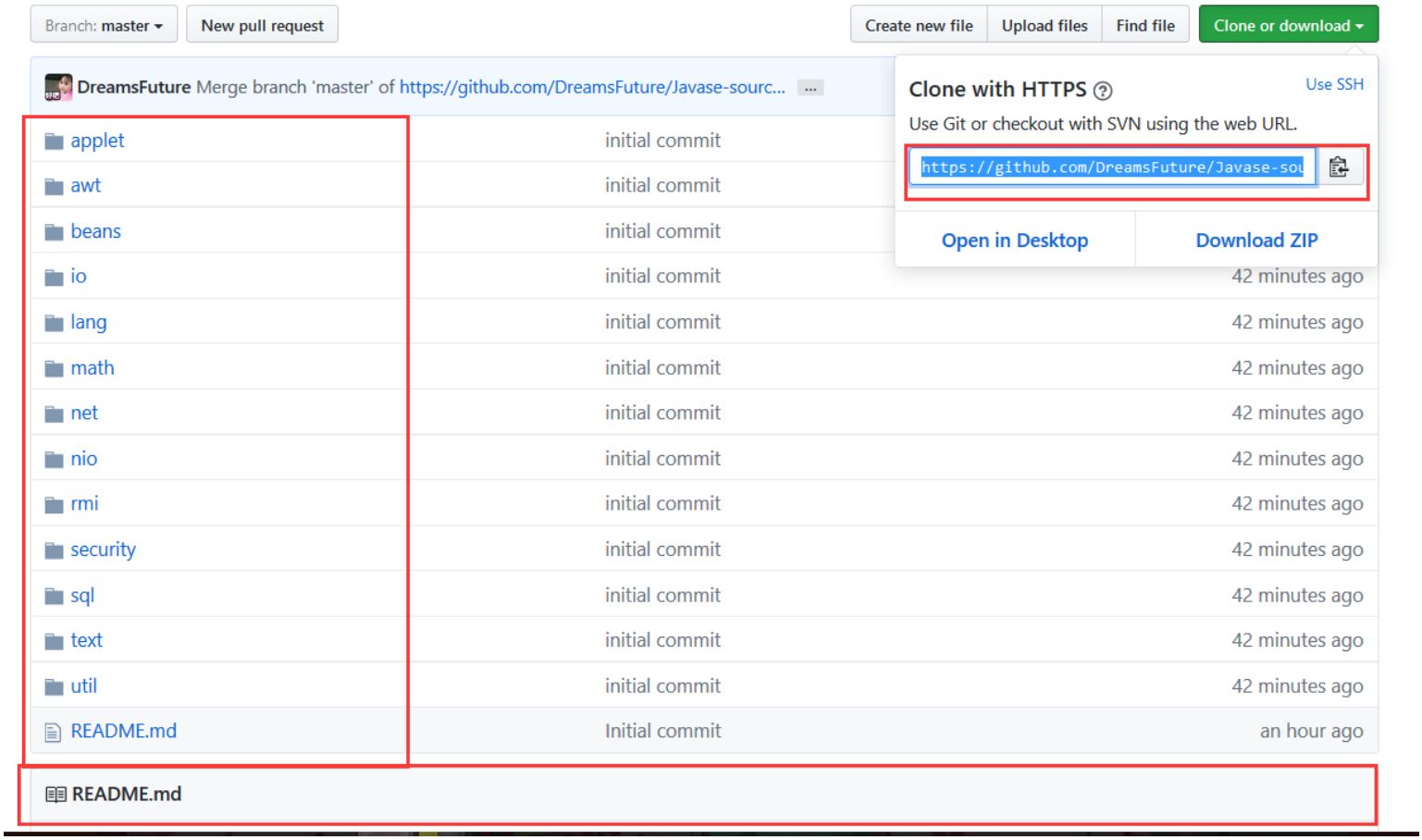


Figure 1 GitHub upload files problem

中间没有画出框的内容“initial commit”是我之前做过的一次的评论和说明，但是之前问题很多，不知道什么原因，现在也不是很清楚；

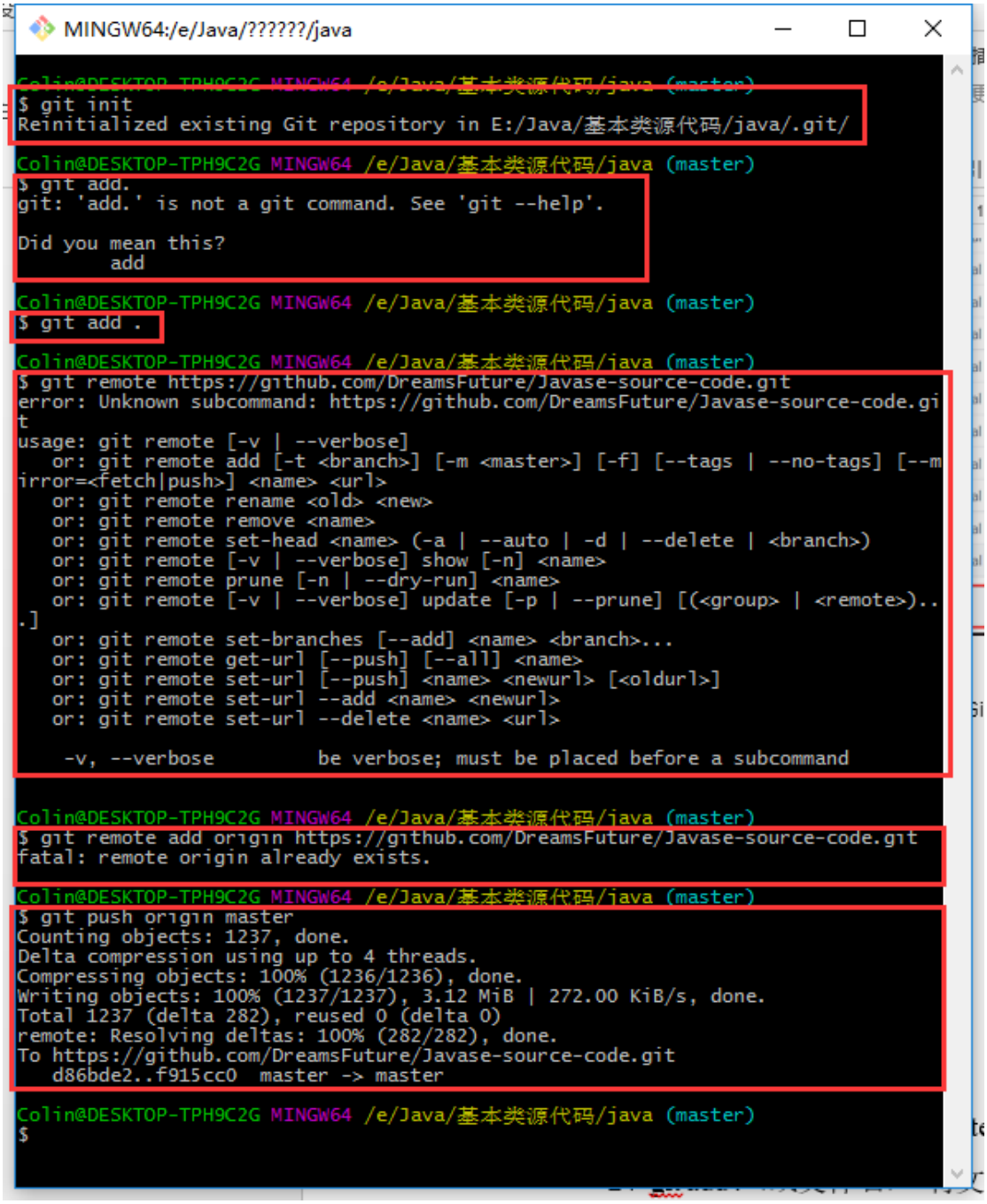


Figure 2　在已经做过一遍上传但是有 errors 的情况下，利用 GitHub 上传文件的整个过程

三、本地库的操作

1、在 wongya 文件夹下创建 test.txt，输入 111 并保存。

2、git add .（或文件名） 将文件添加到缓存区，git commit （可添加文件名）-m "此处添加注释"将文件上传到本地版本库。

3、git branch 分支名 添加分支，git checkout 分支名 可以切换分支。add、commit 到版本库之后切换回 master 主分支，git merge 子分支名 将子分支上的更改合并到 master 主分支上。git checkout -b 分支名在子分支上对 master 主分支进行更改，git branch -d 分支名 可以删除分支。

四、将本地库上传到远程

1、git remote add origin +个性网址。

2、git pull origin master 从远程下载代码合并到本地的 master 主分支上确保远程与本地不冲突。

3、git push origin master 将本地的代码合并到远程分支。

五、总结

git --version 查询版本号

git config --global user.name" " 名字

git config --global user.email" " 邮箱

git add 文件名 将文件添加到缓存区

git add . 将所有文件添加到缓存区

git commit 文件名 -m"注释" 将文件提交到版本库

git status 有没有没提交的更新

git diff 具体怎么改的

git rm -f 文件名 删除文件，要提交才行

git log 显示日志 git branch 查看分支，带*的是当前分支,创建分支

git branch -a 包括远程分支

git checkout 分支 切换到分支

git checkout -b 创建并切换分支

git merge 分支 master 合并 merge 分支，分支的修改要相互不影响

git push origin 分支名字 上传到远程分支

git clone 第一次拿远程的代码

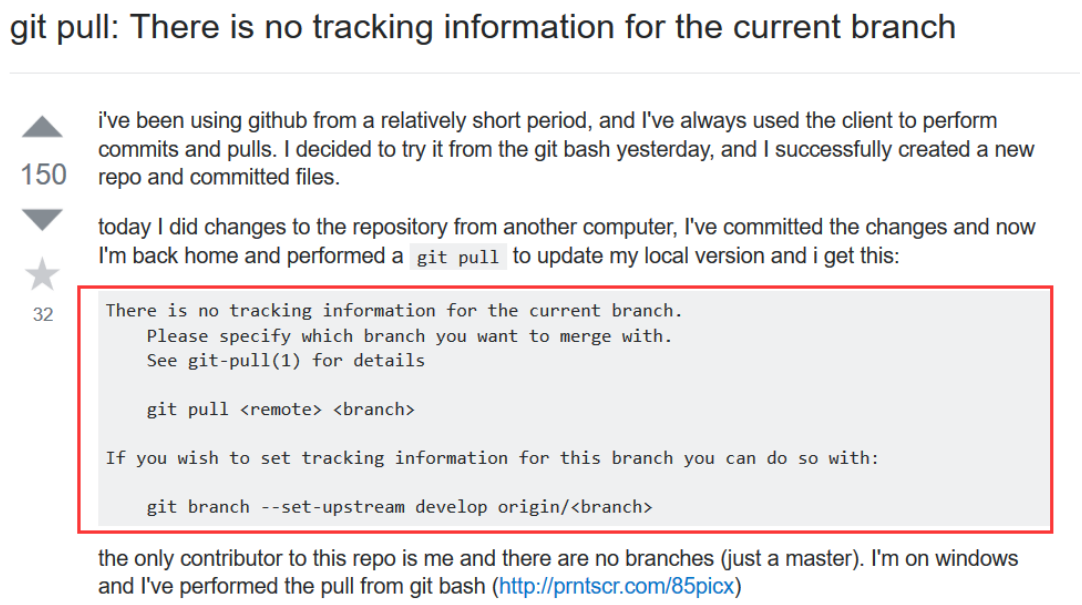
git pull 之后拿代码

git remote add origin 添加新的远程仓库

git pull origin master 将远程的代码直接合并到本地 master 上确保远程与本地不冲突，在将本地的子分支合 并到本地主分支确保本地之间不冲突，再把 master 合并到远程分支

很多不懂的问题，以后要高明白

问题一：git pull: There is no tracking information for the current branch



问题二：Issue pushing new code in Github

Issue pushing new code in Github

49

20

I created a new repository on Github which has only Readme.md file now.

I have a newly created RoR project which I wanted to push to this repository. Following are the commands I gave in my Terminal to execute this along with the error I am getting.

```
git remote add origin https://github.com/aniruddhabarapatre/learn-rails.git
```

After which I entered my username and password

```
git push -u origin master
```

Error ---

```
To https://github.com/aniruddhabarapatre/learn-rails.git
 ! [rejected]        master -> master (fetch first)
error: failed to push some refs to 'https://github.com/aniruddhabarapatre/learn-rails.git'
hint: Updates were rejected because the remote contains work that you do
hint: not have locally. This is usually caused by another repository pushing
hint: to the same ref. You may want to first merge the remote changes (e.g.,
hint: 'git pull') before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.
```

This is my first time pushing my code to a Github repository and I'm lost with the errors. I searched few other questions that are asked here, but none of them had issues first time.

问题三：fatal: The current branch master has no upstream branch

fatal: The current branch master has no upstream branch

36

12

I'm trying to push one of my projects to github, and I keep getting this error:

```
peeplesoft@jane3:~/846156 (master) $ git push

fatal: The current branch master has no upstream branch.
To push the current branch and set the remote as upstream, use

    git push --set-upstream origin master
```

问题四：Git refusing to merge unrelated histories

问题五： error:src refspec master does not match any

这个问题，我之前也遇到过，这次又遇到了只是时间间隔比较长了，为了防止以后再遇到类似问题，还是把这个方法简单记录在此。

当然，是通过搜索引擎找到的答案，开始用谷歌，我以为 stackoverflow 会很权威的，结果在这上面没有找到合适的。

http://stackoverflow.com/questions/21264738/error-src-refspec-master-does-not-match-any
反倒是换用百度输入，查看中文的东西才解决了这个问题。

问题产生

原因分析

引起该错误的原因是，目录中没有文件，空目录是不能提交上去的

解决方法

```
touch README
git add README
git commit -m 'first commit'
git push origin master
```