# Git

## 1. 查看当前所属分支branch

1、git branch -vv

2、git config --lis

## 2. pull报错解决方法

#### 4.2.1 Git fetch和git pull的区别, 解决Git报错:error: You have not concluded your merge (MERGE\_HEAD exists).

##### Git fetch和git pull的区别:

### 都可以从远程获取最新版本到本地

### **1.Git fetch:只是从远程获取最新版本到本地,不会**merge**(合并)**

$:git fetch origin master //从远程的origin的master主分支上获取最新版本到origin/master分支上$:git log -p master..origin/master //比较本地的master分支和origin/master分支的区别$:git merge origin/master //合并

* 1
* 2
* 3

### **2.Git fetch:从远程获取最新版本并**merge**(合并)到本地**

$:git pull origin master //相当于进行了 git fetch 和 git merge两部操作

* 1

### 实际工作中,可能git fetch更好一些, 因为在merge前,可以根据实际情况决定是否merge

## **再说导致报错:**

## error: You have not concluded your merge (MERGE\_HEAD exists).**的原因可能是在以前pull下来的代码自动合并失败**

### **解决办法一:保留本地的更改,中止合并->重新合并->重新拉取**

$:git merge --abort$:git reset --merge$:git pull

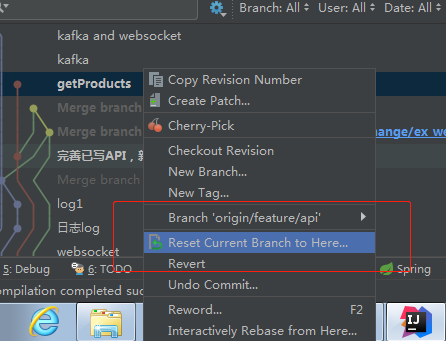
* 1
* 2
* 3

### **解决办法二:舍弃本地代码,远端版本覆盖本地版本(慎重)**

$:git fetch --all$:git reset --hard origin/master$:git fetch

* 1
* 2
* 3

## idea git 将当前分支变为orign上的指定版本



然后选择hard模式完全覆盖就行了。

## 拉取远端的新的分支

git checkout -b mapper-refacter origin/feature/mapper\_refactor

## 远程有了新的分支，给拽下来

### 命令方法

先 git fetch

然后git checkout -b 本地分支名 origin/远程分支名

### Idea操作方法

idea 项目右键-git-fetch，之后右下角就有了新的分支，然后在新的分支上右键-checkout as...之后弹出窗口让命名，名完名后确认，就拽到本地了。

## 命令行操作

* 1. 修改文件后，查看修改的具体内容:

Git diff

* 1. 修改文件后，查看都有哪些文件修改了

Git status（这个命令顺便还展示出了当前所处分支名字、与远端对应的分支名字、以及修改了的但是还没有add的文件）

* 1. 修改文件后，想要提交到本地

1. 先add：

Git add 修改的文件的路径（可以粘贴status上展示出来的地址）

1. 再commit[ -m 传递修改的描述信息]

Git commit -m “该了个文件”

## 7. 将本地仓库创建到远程去

### A.干净的上传

### 先有自己的项目

### 初始化git

在自己的项目的根目录执行

*Git init*

### 把自己项目的内容加入到git管理

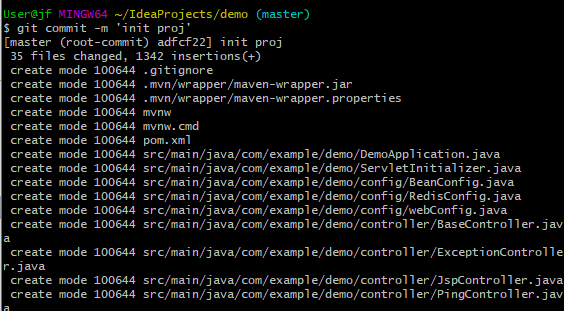
*Git add .*

说明：

“.”的意思就是全部的文件

### 把项目本文件地提交

*Git commit -m ‘init proj’*

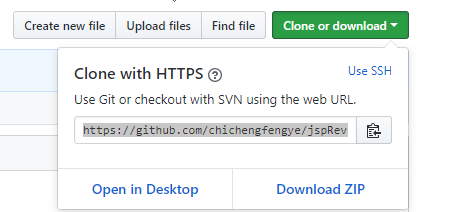


### 在远程创建仓库

不用多说，打开github后点点就可以了。

### 复制远程仓库的地址

如:



### 本地添加远程仓库地址

*Git remote add origin 远程仓库地址*

如：



### 推送上去

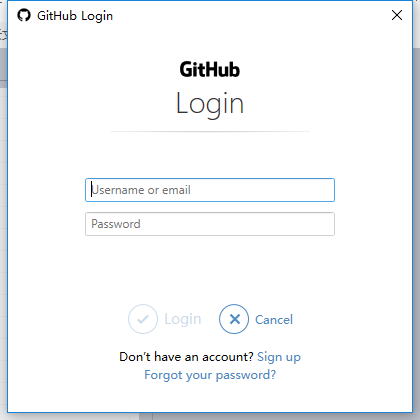
Git push -u origin master

说明：

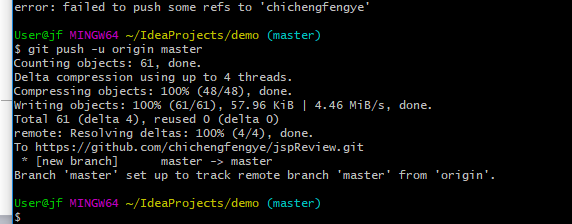
1. Origin就是远程仓库，也就是你在添加的时候指定的那个远程仓库的名字
2. Master就是你将要在远程那里创建的分支的名字

### 输入用户名密码

如果之前git连接的别的git地址，在执行了步骤八之后就会出现一个对话框，现在就需要给新的地址输入账户密码：



### 完成



### 创建本地项目到远程时候，不小心给远程加了一个文件，导致不能把本地的推上去

#### 问题描述

创建远程时候，不小心加了一个readme.txt，导致远程在我本地的前面，这样一来本地的必须先更新才能push，但是本地不能pull，原因是”unrelated”-不关联。

#### 解决办法

在pull的时候，添加一下参数：

--allow-unrelated-histories

就可以pull成功了，接下来就可以push了。

## 从远程拉去branch

之前都是在添加远程git后直接clone命令，这样会导致.git文件与git地址上的项目名在同一个层级，也就是相比于远程的项目的.git文件，向上了一层（简单的说： 真正的.git因该与项目内容在一个层级，而我的.git与项目名称一个层级了--项目名称里面是项目内容）。

按以下方法就可以正确使用：

1. 在一个文件夹处初始化git

Init git

1. 在此处添加远程git

Git remote add origin <https://git.wer.wedsf.git>

1. 拉取远程信息

Git fetch

1. 从远程拉取并新建分支

Git checkout -b master origin/master

## 放弃本地修改，完全用远程的数据覆盖

Git reset --hard origin/master

## Gitignore

让Git忽略部分文件，不管理他们的办法，就是创建”.gitignore”文件，并添加符合规则的内容用来控制即可。

但是，如果git已经管理了那些不想它管理的文件，我们就可以先清空git的cache，这样的清空数据不会历史提交记录给清空，只是刷新管理的目录而已。

git rm -r --cached .

“.”是指全部， 也可以指定部分文件

“-r”表时撤销文件夹

然后再把我们想要它管理的给“add”进去就好了。

Git add readme.txt

## 撤销已经add的文件

git **reset** **HEAD** .

## 删除本地分支

$ git branch -d <BranchName>

## 查看项目的分支们(包括本地和远程)

命令行 : $ git branch -a

## Rebase

## Cherry pick

## 修改分支名称

Git branch -m old new

如：git branch -m release-1.2.6-rc release-1.2.7-rc

## 舍弃本地commit，使用远程的覆盖本地

首先，先发现远程的信息数据：

git fetch –all

然后，充值本地内容

git reset –hard origin/master （前提是你的远程叫做 ‘origin’， 一般情况下都是这个名字）