**git和github**

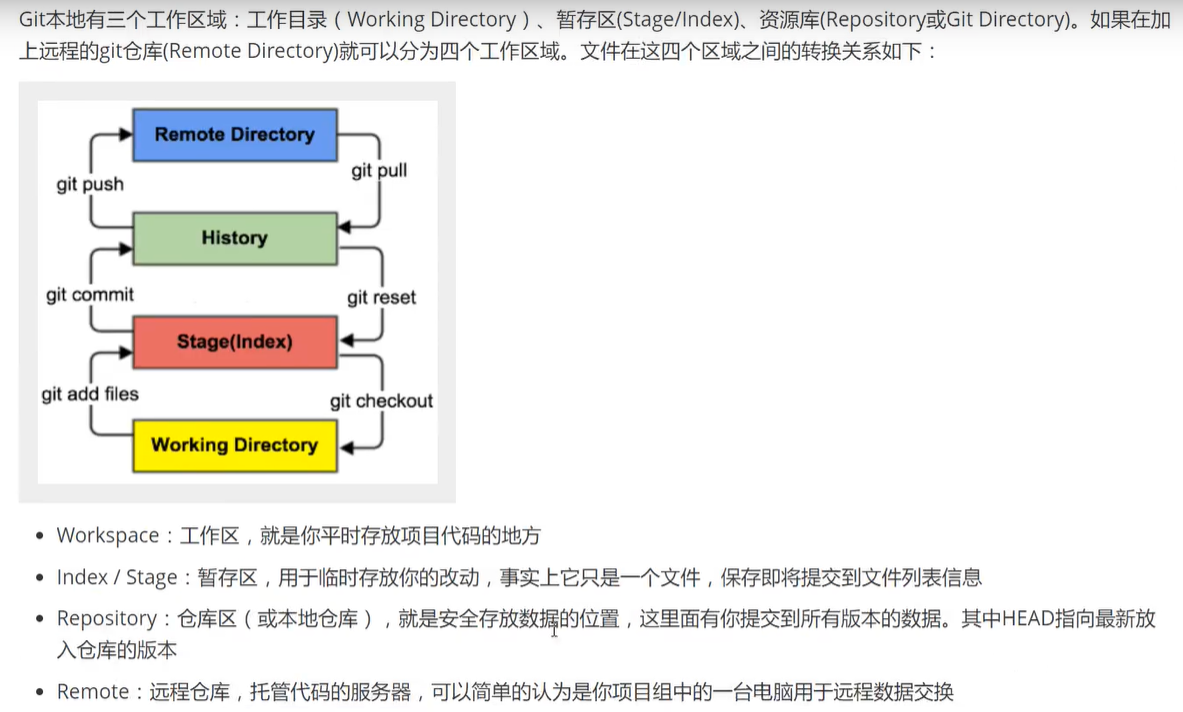
**概述**

git是版本控制软件。git中三个重要的概念：commit(提交)，repository(仓库)，branch（分支）

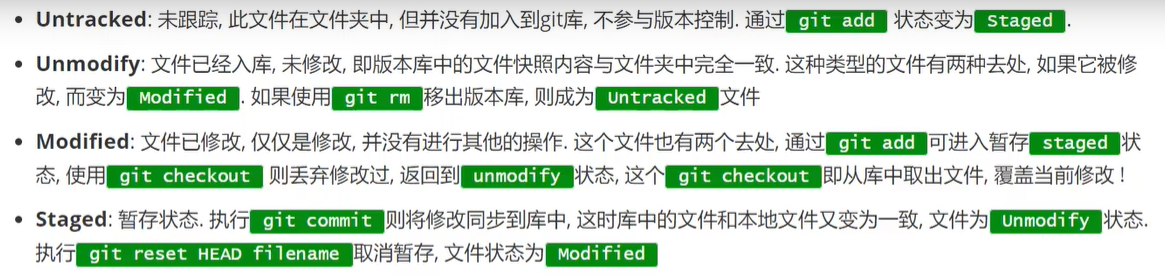
github是基于git开发的一个可以托管代码的网站，也就是托管代码的服务器，托管在该网站的代码可以实现开源共享。

注: git仓库(本地仓库)位于本机中，是一个私有的，无法开放共享的仓库。如果本地库中的资源想要开放共享，就得把本地仓库中的资源同步推送到远程仓库(github仓库)。

**git理论核心**



**文件的状态**



**创建本地仓库**

创建本地库的两种方法

方法一: 将github仓库直接clone到本地。这样在本机中就生成了一个本地

仓库，并且clone来的本地仓库无需再初始化。命令如下:

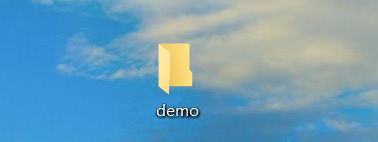
git clone github仓库地址

注: 该方法实际就是将github上的开源项目，clone到本地。clone包含3个操

作: 拉取代码，初始化本地仓库，创建别名

方法二: 直接在本机中创建一个本地仓库。步骤如下:

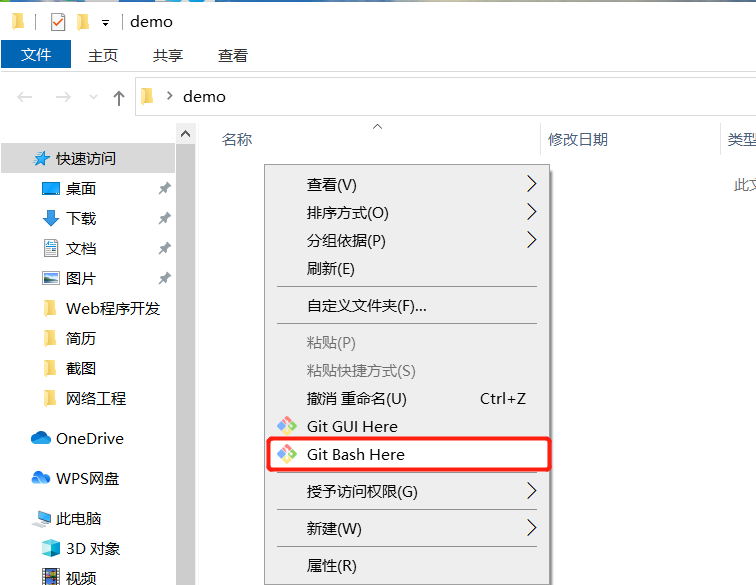
第一步: 先创建一个文件夹。



注意: 这个文件夹就是即将初始化的本地仓库。

第二步: 进入该文件夹，打开git bash，使用git init命令将这个文件夹初始化成

本地仓库。





这两步就已经创建好了一个本地仓库。

**提交文件到本地仓库**

本地仓库创建好后，这里再简单测试将工作区中的t1.java文件提交到本地仓库。

过程: 工作区->暂存区->本地仓库(git仓库)。步骤如下:

第一步: 在工作区(demo目录下)，先创建一个t1.java



第二步: 此时的t1.java文件处于未跟踪状态，需要先使用git add命令将其添加

到暂存区。添加到暂存区中的文件已被跟踪。



注: 可以直接使用git add . 命令，一次添加所有文件到暂存区。

第三步: 最后使用git commit -m ‘描述信息’ 命令，将暂存区的文件提交到本地

仓库。

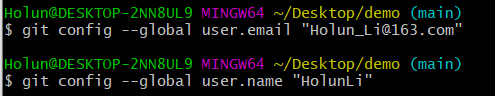
注意: 在将暂存区中的文件提交到本地仓库之前，必须保证已配置好邮箱和用户

名，配置邮箱和用户名的目的是让git软件知道你是谁。

配置邮箱和密码的命令如下:

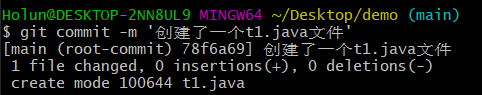
git config --global user.email "github 上设置的邮箱地址"

git config --global user.name " github 上的用户名"



注: 邮箱和用户名可以再初始化本地仓库之前就配置好。

确保邮箱和用户配置完毕后，再提交到本地仓库



**将本地仓库中资源推送(push)到远程仓库**

前面提到过，本地仓库(git仓库有)中的资源想要实现开放共享，可以将本地仓库

中的资源同步推送到远程仓库(github仓库)。

以上面的例子为例，现在将本地仓库中的t1.java文件，推送到远程仓库。过程如下:

第一步: 本地仓库先与某个远程仓库建立连接。命令如下:

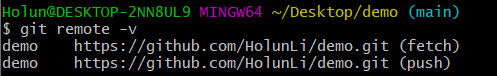
git remote add 别名 github仓库地址



注: github仓库地址很长，可以在前面给它取个别名。别名建议和连接的github仓库名相同。建议本地仓库名也和连接的远程仓库名相同。

与远程仓库连接成功后，可以使用git remote -v 命令，查看当前的本地仓库

已经与哪个远程仓库建立了连接。



这里的两条连接信息需要了解下：

https://github.com/HolunLi/demo.git (fetch) //表示与指定的远程仓库连接成功

后，可以将远程仓库demo中的资源拉取到本地仓库(这里的本地仓库也叫demo)。

https://github.com/HolunLi/demo.git (push) //表示与指定的远程仓库连接成功

后，可以将本地仓库中的资源推送到远程仓库demo

这两条连接信息被记录在config文件中，config文件在.git目录下。

还可以使用git remote remove github仓库的别名 命令，将当前本地仓库与指

定的远程仓库断开连接。断开连接后，会在config文件删除之前记录的连接信

息。

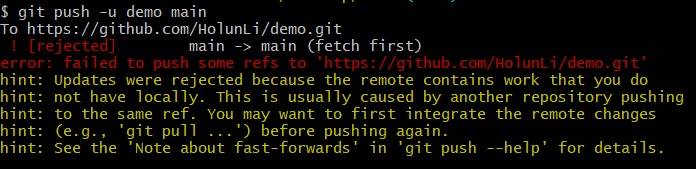
第二步: 连接后，再将本地仓库中的资源(t1.java)推送到远程仓库。命令如下:

git push 将要推送到的github仓库地址的别名 分支名

注: 在推送前，必须先将已连接的github仓库中的README.md文件拉取到本地

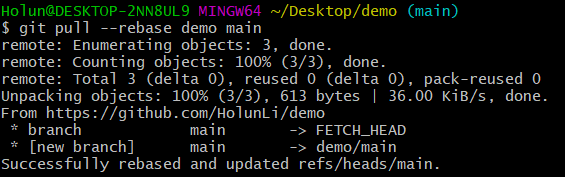
仓库。如果本地仓库未包含README.md文件，就执行推送(push)命令，将会出

现以下错误

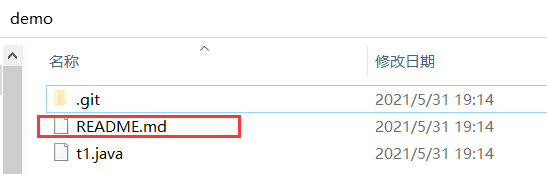


将github仓库中的README.md文件拉取到本地仓库的命令如下:

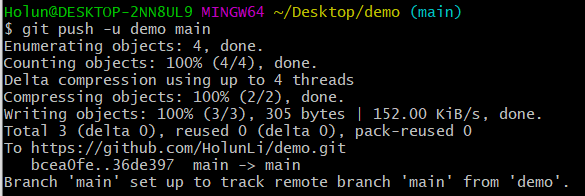
git pull --rebase github仓库的别名 分支名



此时可以看到本地仓库多出了一个README.md文件



保证本地仓库已有README.md文件后，在将本地仓库的资源推送到远程仓库

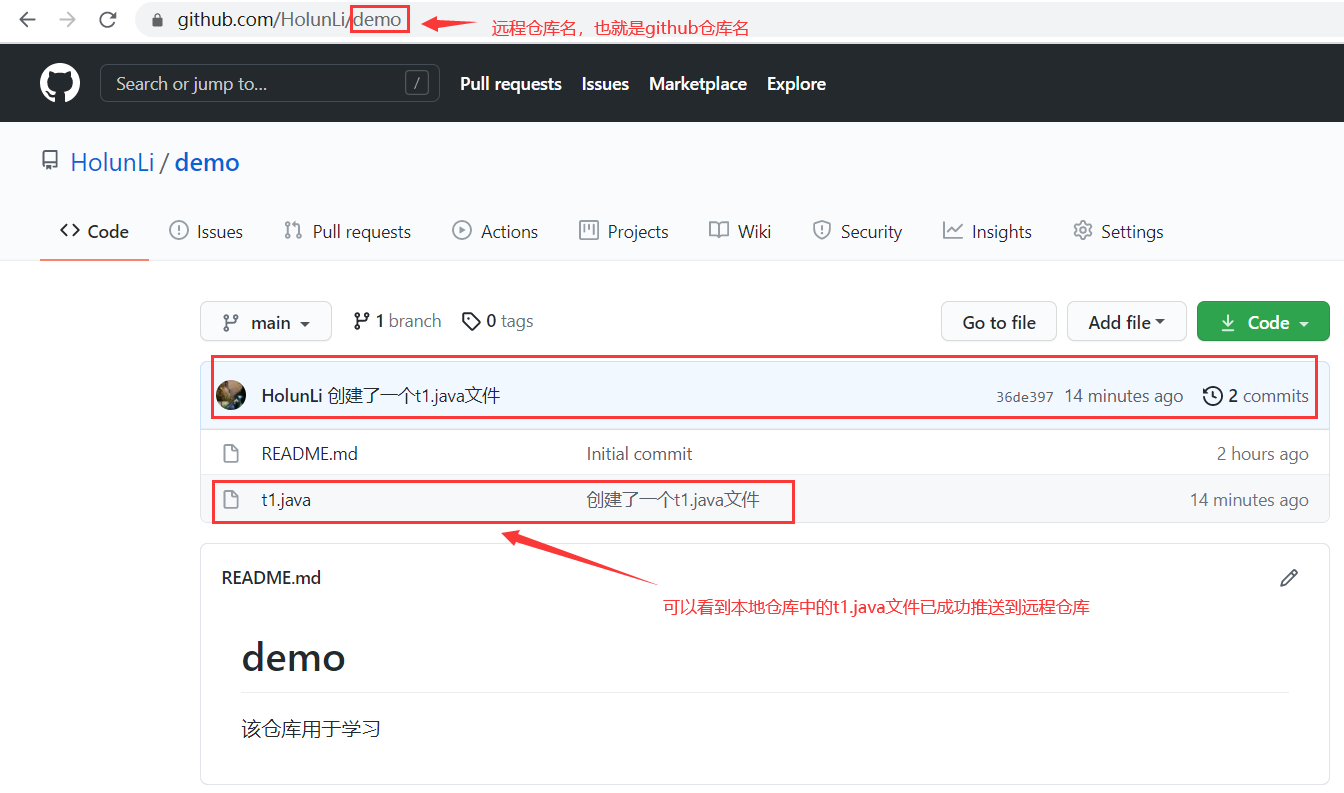


注: 第一次推送时使用git push -u 别名 分支名，因为第一次推送时，要输入

github的账号和密码进行安全验证。后面再推送直接使用git push 别名 分支名

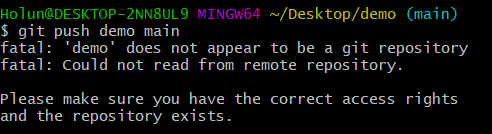
即可。

推送成功的话，在远程仓库(github仓库)中，就能看到t1.java文件



断开连接后，就无法再继续推送





**远程仓库拉取(pull)到本地仓库**

我们知道本地仓库和远程仓库中的资源要做到同步更新，简单将就是本地仓库和

远程仓库中的资源状态要保持一致。一般情况下，是先在本地仓库中对文件内容

进行修改后，再重新push到远程仓库。相反的情况就是，当远程仓库中的文件

中的内容被修改了，但本地仓库没有跟着同步，此时就需要将远程仓库中修改过

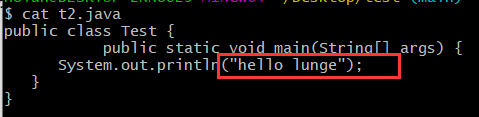
的资源，拉取到本地仓库，覆盖掉本地仓库中修改前的资源。简单理解，就是将

远程仓库中的资源覆盖掉本地仓库中的资源。拉取的命令如下:

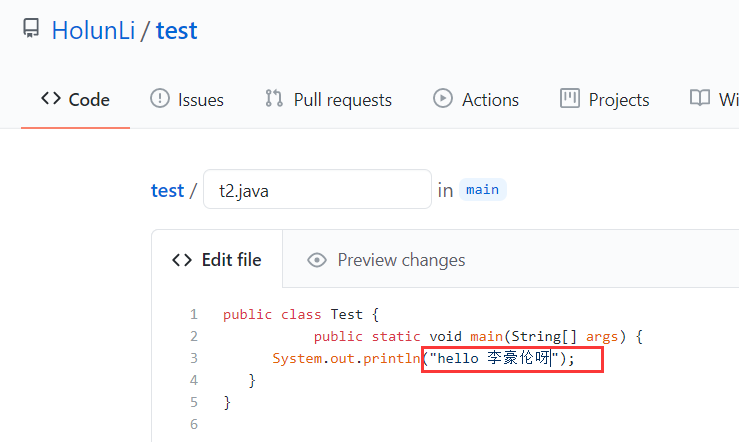
git pull github仓库地址的别名 分支名

举个例子:

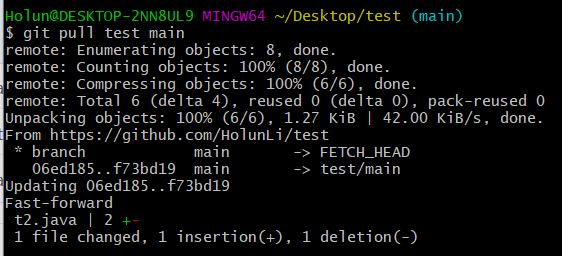
此时，查看本地仓库中的t2.java内容是这样的



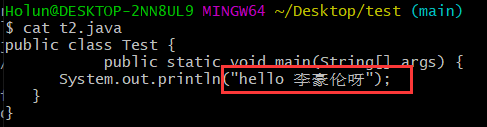
修改远程仓库中的t2.java的内容



从远程仓库拉取到本地仓库



再次查看本地仓库中t2.java的内容，发现内容改变，即覆盖成功



**git中常用的命令**

1. git config --global --list 可以查看全局配置信息(用户自己配置的信息，比如邮箱。密码)，list可以简写成l
2. git config --global user.email "您的 github 上设置的邮箱地址" 配置邮箱

git config --global user.name "您的 github 上的用户名" 配置密码

配置的邮箱和密码被保存在 .gitconfig文件中，.gitconfig文件保存在C:\Users\Holun 目录下。如果在 .gitconfig 文件中将之前配置好的邮箱和密码删除，则需要在git bash中重新配置。

1. git remote -v 可以查看本地仓库(git仓库，又称版本库)和哪个远程仓库(github仓库)建立了连接。
2. git checkout 文件名 从本地仓库中取出文件，覆盖当前修改的文件
3. git status 列出当前目录所有还没有被git管理的文件和被git管理且被修改但还未提交(git commit)的文件.。