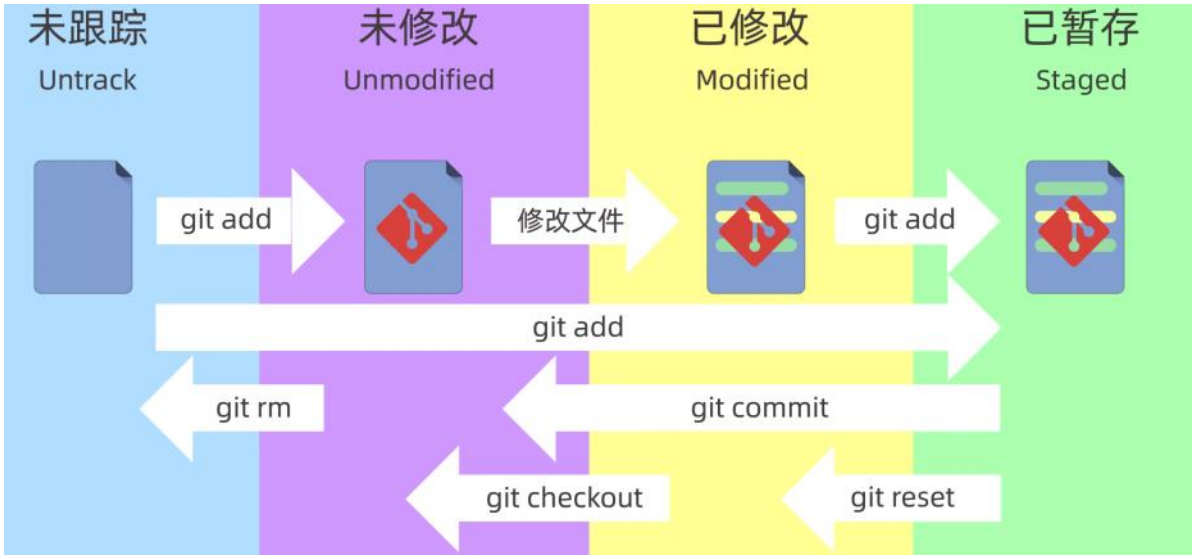
git学习

2024年9月9日　　11:11

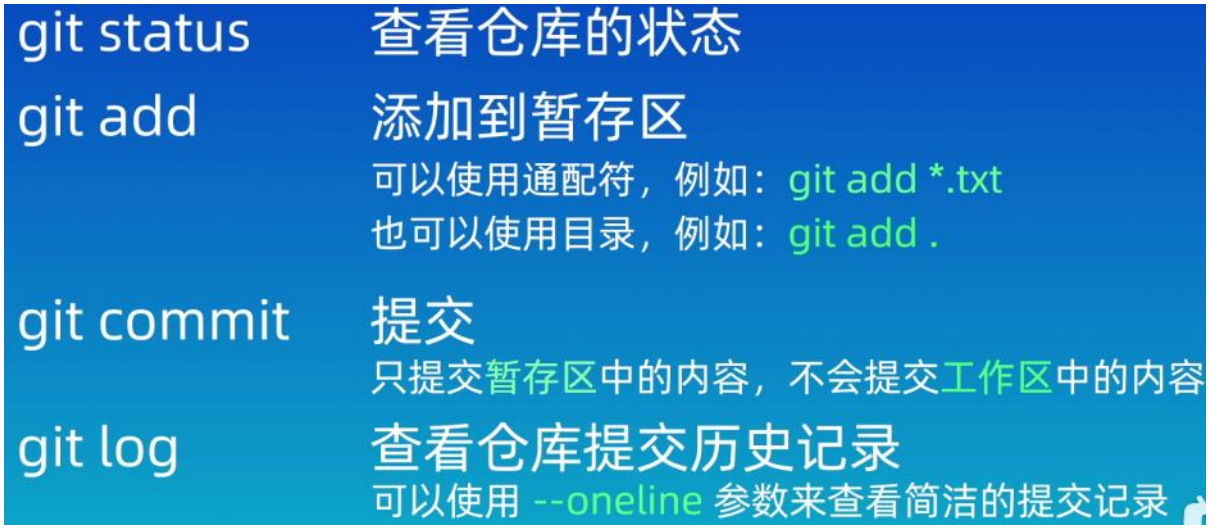
创建仓库

git的工作区域和文件状态



添加和提交文件

创建<temp>文件夹



-am 提示信息

```
git commit -am "delete other.log
```

>>追加 some change 到 other.log 的尾部

```
echo " some change " >> other.log
```

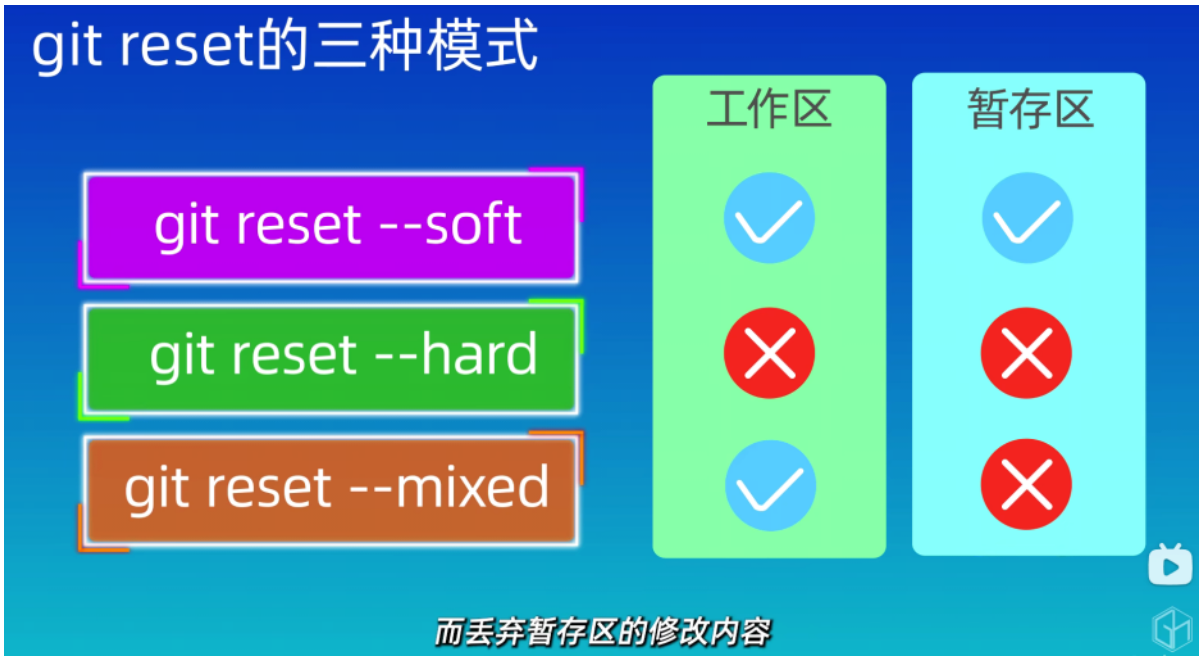
查看状态 -s == -short

```
git status -s
```

Git add 出现 warning: in the working copy of 'file1.txt', LF will be replaced by CRLF the next time Git touches it

解决：https://blog.csdn.net/lxw1844912514/article/details/136052432

回退版本



比较当前版本和上一个版本的差距

```
git diff HEAD~ HEAD
```

删除文件（工作区 、 暂存区）

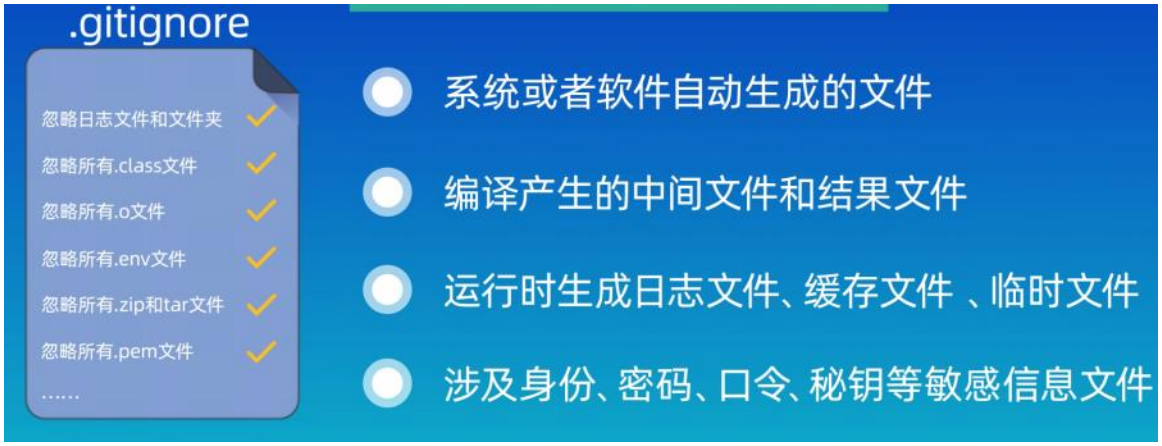
方法一： rm <file>

方法二：

```
~/learn-git/my-repo  main +  git rm file2.txt
rm 'file2.txt'
~/learn-git/my-repo  main +  git status
On branch main
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        deleted:    file1.txt
        deleted:    file2.txt
```

Last:提交以下版本库

gitignore忽略文件:



删除版本库中的文件 而不删除本地的文件

```
git rm --cached other.log
```

ignore 自顶向下匹配 匹配规则如下:

```
空行或者以#开头的行会被Git忽略。一般空行用于可读性的分隔, #一般用作注释
使用标准的Blob模式匹配, 例如:
      星号 * 通配任意个字符
      问号 ? 匹配单个字符
      中括号 [ ] 表示匹配列表中的单个字符, 比如: [abc] 表示a/b/c
两个星号 ** 表示匹配任意的中间目录
中括号可以使用短中线连接, 比如:
      [0-9] 表示任意一位数字, [a-z]表示任意一位小写字母
感叹号！ 表示取反
```

例子:

```
# 忽略所有的 .a 文件
*.a

# 但跟踪所有的 lib.a，即便你在前面忽略了 .a 文件
!lib.a

# 只忽略当前目录下的 TODO 文件，而不忽略 subdir/TODO
/TODO

# 忽略任何目录下名为 build 的文件夹
build/

# 忽略 doc/notes.txt，但不忽略 doc/server/arch.txt
doc/*.txt

# 忽略 doc/ 目录及其所有子目录下的 .pdf 文件
".gitignore" 17L, 383B
```

SSH链接

配置SSH密钥

```
$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/c/Users/Accan/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Passphrases do not match.  Try again.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in
Your public key has been saved in  .pub
The key fingerprint is:
```

私钥文件

```
-rw-r--r-- 1 Accan 197121 3434 Sep  9 19:58 ''$'\033''[C'
-rw-r--r-- 1 Accan 197121  739 Sep  9 19:58 ''$'\033''[C.pub'
```

公钥文件

第二次创建SSH密钥 需要的操作

```
~/.ssh  tail -5 config
# github
Host github.com
HostName github.com
PreferredAuthentications publickey
IdentityFile ~/.ssh/test
```

在github上添加公钥后，去本地仓库clone github上的仓库

```
Accan@qixiang MINGW64 ~/qixiang
$ git clone git@github.com:Z7712066/android.git
Cloning into 'android'...
Enter passphrase for key '/c/Users/Accan/.ssh/totle':
warning: You appear to have cloned an empty repository.
```

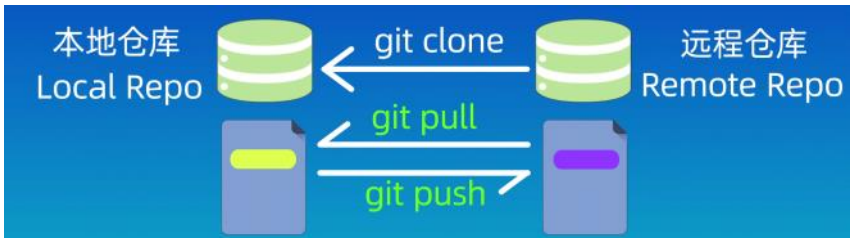
clone成功，尝试添加文件 并且push到远程仓库。

echo 123 > test.txt 生成文件，以及上传、push等

```
Accan@qixiang MINGW64 ~/qixiang
$ cd android

Accan@qixiang MINGW64 ~/qixiang/android (master)
$ ls -ltr
total 1
-rw-r--r-- 1 Accan 197121 7 Sep 11 10:35 hello.txt

Accan@qixiang MINGW64 ~/qixiang/android (master)
$ git add .

Accan@qixiang MINGW64 ~/qixiang/android (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   hello.txt


Accan@qixiang MINGW64 ~/qixiang/android (master)
$ git commit
[master (root-commit) d03b3d8] 这是android的第一个文件
 1 file changed, 1 insertion(+)
 create mode 100644 hello.txt
```

```
Accan@qixiang MINGW64 ~/qixiang/android (master)
$ git push
Enter passphrase for key '/c/Users/Accan/.ssh/totle':
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 246 bytes | 246.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:Z7712066/android.git
 * [new branch]      master -> master
```

本地仓库和远程仓库



总结：

本地仓库上传到github

```
Accan@qixiang MINGW64 ~/qixiang/ac (master)
$ git remote add origin git@github.com:Z7712066/remote_p.git

Accan@qixiang MINGW64 ~/qixiang/ac (master)
$ git remote -v
origin  git@github.com:Z7712066/remote_p.git (fetch)
origin  git@github.com:Z7712066/remote_p.git (push)

Accan@qixiang MINGW64 ~/qixiang/ac (master)
$ git branch -M first

Accan@qixiang MINGW64 ~/qixiang/ac (first)
$ git push -u origin main:first
```

-u upstream origin是仓库别名 main是本地仓库的分支 first是远程仓库的分支

git push origin [本地分支名：远端分支名]

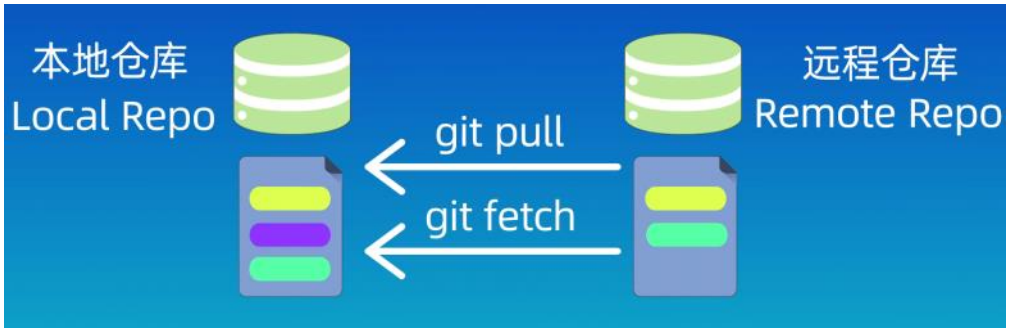
解决 本地仓库更新到远程仓库的push的问题

```
$ git push origin main
error: src refspec main does not match any
error: failed to push some refs to 'github.com:Z7712066/remote_p.git'

Accan@qixiang MINGW64 ~/qixiang/ac (main)
$ git  remote -v
origin  git@github.com:Z7712066/remote_p.git (fetch)
origin  git@github.com:Z7712066/remote_p.git (push)

Accan@qixiang MINGW64 ~/qixiang/ac (main)
$ git remote rm origin

Accan@qixiang MINGW64 ~/qixiang/ac (main)
$ git remote add origin git@github.com:Z7712066/remote_p.git

Accan@qixiang MINGW64 ~/qixiang/ac (main)
$ git  remote -v
origin  git@github.com:Z7712066/remote_p.git (fetch)
origin  git@github.com:Z7712066/remote_p.git (push)

Accan@qixiang MINGW64 ~/qixiang/ac (main)
$ git commit -m "test"
[main (root-commit) 2951f9a] test
 1 file changed, 1 insertion(+)
 create mode 100644 test.txt

Accan@qixiang MINGW64 ~/qixiang/ac (main)
$ git push -u origin main

Enter passphrase for key '/c/Users/Accan/.ssh/totle':
git@github.com: Permission denied (publickey).
fatal: Could not read from remote repository.

Please make sure you have the correct access rights
and the repository exists.

Accan@qixiang MINGW64 ~/qixiang/ac (main)
$ git push -u origin main
Enter passphrase for key '/c/Users/Accan/.ssh/totle':
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
```

拉取远程仓库的内容到本地：

Git pull：

Git f etch: 只获取远程仓库的修改，并不会自动合并到本地仓库。



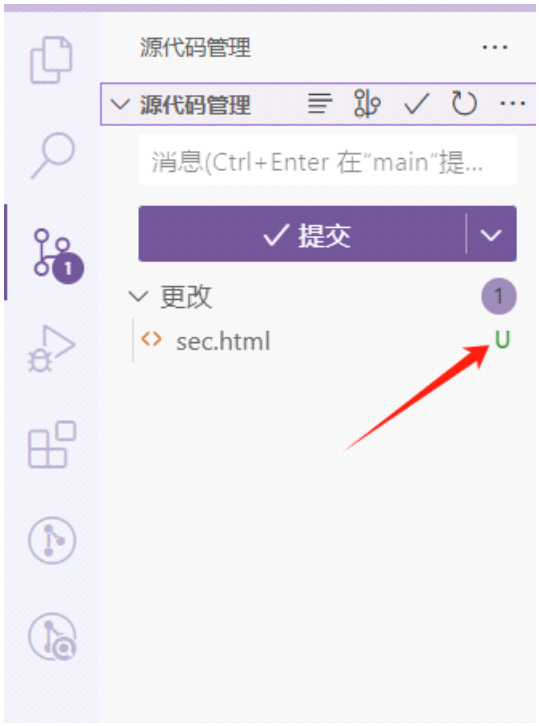
Git remote -v 查看目前关联的远程仓库地址

添加其他的远程仓库地址

```
git remote add gitlab git@gitlab.example.com:gitlab-insta
```

配合VScode使用 配置从git控制台使用 code . 开启

在vscode中字母表示的状态



?? (Untracked) : 未跟踪
M (Modified) : 已修改
A (Added) : 已添加暂存
D (Deleted) : 已删除
R (Renamed) : 重命名
U (Updated) : 已更新未合并

Git 的分支简介和基本操作

Git branch <dev> :创建dev分支

Git checkout <dev> :切换到dev分支

目前防止歧义 使用：git switch <main>

```
git branch dev
git branch

git checkout dev
```

合并分支

```
branch-demo  main  git merge dev
```

把dev分支合并到main分支中

Merge 后面的名称是 <dev> 将要被合并的分支

查看分支图

```
git log --graph --oneline --decorate --all
```

-d 删除已经合并的分支

```
git branch -d dev
```

若分支没有被合并 则可以通过 -D 强制删除分支

```
git branch -D branch-name
```

合并分支时如何解决冲突

Git commit -a -m "可跳过添加暂存 直接提交" -a -m == -am

```
git commit -am "main:6"
```

\

可使用git merge 终止合并命令

```
use "git merge --abort" to abort the merge)
```

Git rebase

Git rebase main　把dev变更到main上

alias命令将其定义一个别名

```
~/learn-git/branch-demo  dev  alias graph="git log --one
line --graph --decorate --all"
```

Note:

```
cp -rf branch-demo rebase1
```

cp命令是linux下面用来复制文件夹的

Rebase

优点：不会新增额外的提交记录，形成线性历史，比较直观和干净；

缺点：会改变提交历史，改变了当前分支branch out的节点。
避免在共享分支使用。