# Git使用

## 1、Git基础(添加、提交文件)

> Tips（小贴士 ）

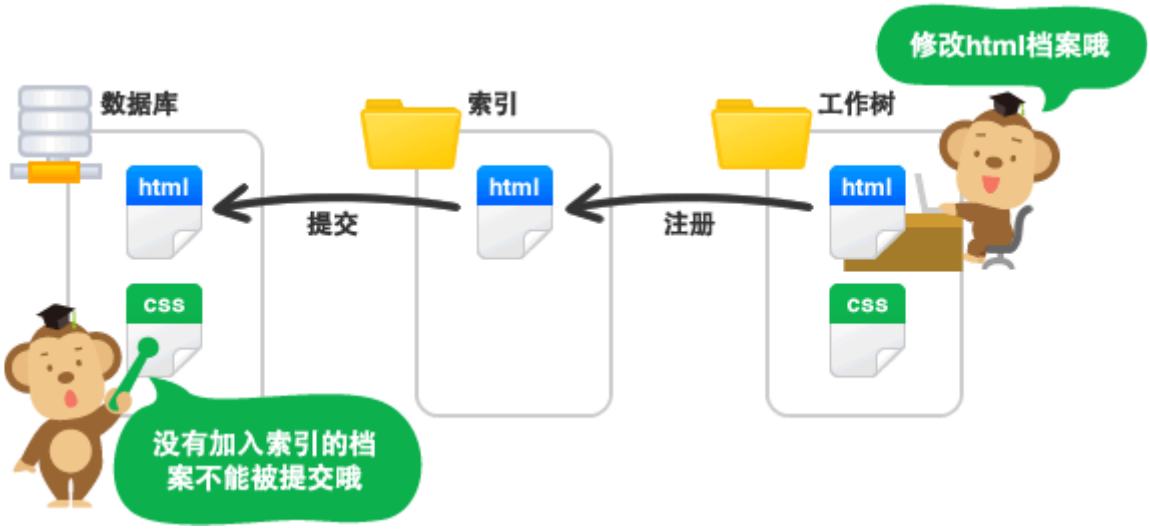
**不同类别的修改（如：Bug修复和功能添加）要尽量分开提交**，以方便以后从历史记录里查找特定的修改内容。

> Tips（小贴士 ）

执行提交时，系统会要求输入提交信息。请务必输入提交信息，因为在空白的状态下执行提交会失败的。

查看其他人提交的修改内容或自己的历史记录的时候，提交信息是需要用到的重要资料。所以请用心填写修改内容的提交信息，以方便别人理解。以下是**Git的标准注解**：

```
第1行：提交修改内容的摘要
第2行：空行
第3行以后：修改的理由
```



Git在执行提交的时候，不是直接将工作树的状态保存到数据库，而是将设置在中间索引区域的状态保存到数据库。因此，要提交文件，首先需要把文件加入到索引区域中。**（就是更改的文件，加入中间索引区域引之后，提交）**

> 新建数据库

首先在任意一个地方创建tutorial目录。然后使用init命令把该tutorial目录移动到本地Git数据库。

```
$ mkdir tutorial
$ cd tutorial
$ git init // 初始化
Initialized empty Git repository in
E:/tutorial/.git/
```

> 添加文件 **git add xxx.xx**

```
$ touch sample.txt //创建文件
$ git status //确认工作树和索引的状态
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will
be committed)
        sample.txt

nothing added to commit but untracked files present
(use "git add" to track)
```

将文件加入到索引，要使用**add命令**。在指定加入索引的文件。用空格分割可以指定多个文件。**指定参数「.」，可以把所有的文件加入到索引。**

```
$ git add sample.txt

$ git status
On branch master

No commits yet

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

> 提交文件 **git commit -m "xxx"**

```
$ git commit -m "first commit"
[master (root-commit) 5b9f247] first commit
 1 file changed, 1 insertion(+)
 create mode 100644 sample.txt

$ git status
# On branch master
nothing to commit (working directory clean)
```

从status响应我们可以看到没有新的变更要提交。

> 查看记录 **git log**

使用log命令，我们可以在数据库的提交记录看到新的提交。

```
$ git log
commit 5b9f247d0c3a0363cf664207b856c96e4ba93e33
(HEAD -> master)
Author: manigoat <1161075573@qq.com>
Date:   Tue Apr 16 17:24:58 2024 +0800

    first commit
```

# 2、远程仓库GitHub

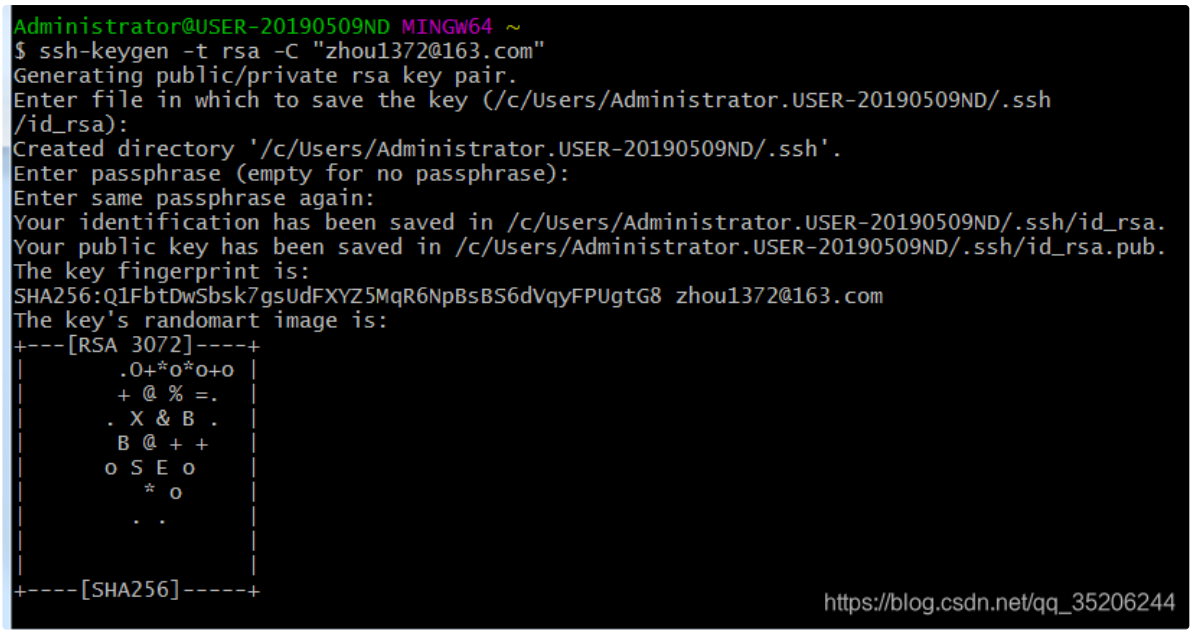
## 2.1、配置本地SSH

```
git config --global user.name "注册名"
```

```
git config --global user.email "注册邮箱"
```

> **生成SSH**

```
ssh-keygen -t rsa -C "自己的邮箱"
```




## 2.2、github配置SSH

> 打开**id_rsa.pub文件**，全选，复制全文

- 打开id_rsa.pub文件，全选，复制全文
- github→账户→setting
- 选择SSH and GPGkeys，New SSH key
- 自定义一个title，然后粘贴从公钥文件中拷贝的key

> 测试SSH连接

- ssh -T git@github.com
  按照提示输入yes，回车，提示successfully之类的就说明SSH连接正常，github上的钥匙也会变成绿色
- 至此，本地git客户端和远程github建立了联系。

## 2.3、clone复制仓库

```
git clone git@github.com:用户名/仓库名.git
```

## 2.4、add文件加入索引

**注意:add有多种形式，可以add某个文件，某个文件夹，或直接add当前仓库下所有文件**

```
git add 单个文件
git add 文件夹1/ 文件夹2/ ……多个文件夹之间空格隔开
git add .
```

## 2.5、commit提交修改

```
git commit -m “注释”
```

## 2.6、push推送仓库

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

**在Git执行推送(Push)操作。执行Push之后，本地的修改记录会被上传到远程数据库。**所以远程数据库的修改记录就会和本地数据库的修改记录保持同步。

- **origin/master**表示远程数据库“origin”的分支“master”的位置。
- **当在克隆的数据库目录执行推送时，您可以省略数据库和分支名称。**
  **git push**

## 2.7、pull拉取远程仓库的更新

```
git pull origin master

remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 1), reused 0 (delta 0), pack-
reused 0
Unpacking objects: 100% (3/3), 908 bytes | 227.00
KiB/s, done.
From github.com:Manigoat/mygit
 * branch            master     -> FETCH_HEAD
   d4a46d3..e7d4e57  master     -> origin/master
Updating d4a46d3..e7d4e57
Fast-forward
 test.txt | 1 +
 1 file changed, 1 insertion(+)
```

2.8、合并修改记录冲突

**如果远程数据库和本地数据库的同一个地方都发生了修改**的情况下，因为无法自动判断要选用哪一个修改，所以就会发生冲突。

Git会在发生冲突的地方修改文件的内容，如下图。所以我们需要手动修正冲突。


=分割线上方是本地数据库的内容， 下方是远程数据库的编辑内容。

如下图所示，修正所有冲突的地方之后，执行提交。


> 实例

用tutorial进行的操作请执行以下指令。

```
$ git pull origin master
```

显示合并时发生冲突的讯息。

用tutorial进行的操作，**讯息显示「Merge conflict in sample.txt」**。请打开sample.txt文件，我们看到Git已添加标示以显示冲突部分。请为Git无法完成主动合并的部分做以下的修改。

```
连猴子都懂的Git命令
add 把变更录入到索引中
<<<<<<< HEAD
commit 记录索引的状态
=======
pull 取得远端数据库的内容
>>>>>>> 4c0182374230cd6eaa93b30049ef2386264fe12a
```

用tutorial进行的操作


**导入两方的修改，并删除多余的标示行以解决冲突。**

```
连猴子都懂的Git命令
add 把变更录入到索引中
commit 记录索引的状态
pull 取得远端数据库的内容
```

文件的内容发生了修改，所以需要进行提交。

```
$ git add sample.txt
$ git commit -m "合并"
```

这样就完成了从远程数据库导入最新的修改内容。

我们可以用log命令来确认数据库的历史记录是否准确。指定–graph选项，能以文本形式显示更新记录的流程图。指定–oneline选项，能在一行中显示提交的信息。

```
$ git log --graph --oneline
*   d845b81 合并
|\
| * 4c01823 添加pull的说明
* | 95f15c9 添加commit的说明
|/
* 3da09c1 添加add的说明
* ac56e47 first commitxxxxxxxxxx $ git log --graph --oneline*   d845b81 合并|\| * 4c01823 添加pull的说明* | 95f15c9 添加commit的说明|/* 3da09c1 添加add的说明* ac56e47 first co$ git log --graph --oneline* d845b81 合并|\| * 4c01823 添加pull的说明* | 95f15c9 添加commit的说明|/* 3da09c1 添加add的说明* ac56e47 first commitmmit
```