# window10操作

看来不是所有的window软件都有图形交互界面，有一些操作仍然需要用命令行解决，所以我在这里写一些命令行的用法。

### 网站类型

### 软件类型

#### git.bash

github里面大多是代码项目。但也不全是代码项目，一些文本形式的文档也会被保存在里面，有些人打算将一些信息永久保存，我也是其中之一，所以学会用命令行控制我的库是必然的。

在windows上打git代码也需要用下载git软件，这个软件一般可以在官网的[Git - Downloading Package (git-scm.com)](https://git-scm.com/download/win)下载。

软件安装过程中基本不用更改选项。软件安装完成后，找到git.bash，双击可以进入命令行界面。

git.bash必须定位到自己的项目库才能合理运行，在正式开始之前，需要用cd到自己的项目库。

##### cd 进入目录

|  |  |
| --- | --- |
| **git.bash** | |
| 1 | cd ［目录名］ #进入目录。 |

##### git status 查看仓库的状态

|  |  |
| --- | --- |
| **git.bash** | |
| 1 | git status #查看仓库的状态。 |

##### git init 初始化仓库

|  |  |
| --- | --- |
| **git.bash** | |
| 1 | git init #初始化仓库。如果一个预定成为仓库的文件夹还不是仓库，它会在git status的时候报错，使用git init将其初始化为仓库。 |

##### git add 将文件加入临时缓冲区

|  |  |
| --- | --- |
| **git.bash** | |
| 1 | git add ［文件名］ #将文件加入到“临时缓冲区”。 |

初始化仓库后，文件夹内原有的文件会在git status的时候被醒目地展示出来，表示这个文件还不属于库，这时使用这个指令将文件加入“临时缓冲区”，防止之后的命令因为它而报错。

##### git commit 提交文件

|  |  |
| --- | --- |
| **git.bash** | |
| 1 | git commit -m “［文件名］” #将临时缓冲区中的文件加入库。 |

正常的话输入git status会显示nothing to commit, working tree clean字样表示传输完毕。

##### git log 打印提交日志

|  |  |
| --- | --- |
| **git.bash** | |
| 1 | git log #打印git提交日志。 |

主要显示两行信息：Author提交者，Date日期。

##### git branch 分支操作

|  |  |
| --- | --- |
| **git.bash** | |
| 1 | git branch #查看当前库的所有分支。 |
| 2 | git branch ［分支名］ #添加分支。 |
| 3 | git branch -d ［分支名］ #删除特定分支。 |
| 4 | git branch -D ［分支名］ #强制删除。 |

##### git checkout 切换分支

|  |  |
| --- | --- |
| **git.bash** | |
| 1 | git checkout ［分支名/标签名］ #切换到特定分支、标签。 |
| 2 | git checkout -b ［分支名］ #创建并切换到目标分支。 |

##### git merge 合并分支

|  |  |
| --- | --- |
| **git.bash** | |
| 1 | git merge ［分支名］ #将目标分支合并到当前分支。 |

效果是在两分支没有冲突的情况下合并分支。

##### git tag 添加标签

|  |  |
| --- | --- |
| **git.bash** | |
| 1 | git tag ［标签名］ #为当前分支添加标签。 |

##### ssh-keygen 创建ssh密钥

windows系统不自带ssh软件，但是下载git的时候也会顺便安装上，所以ssh指令在git.bash里是可以用的。由于本地的软件需要与github.com相连才能起作用，所以需要与网站建立ssh连接。

|  |  |
| --- | --- |
| **git.bash** | |
| 1 | ssh-keygen -t rsa #指定rsa算法创建密钥、 |

在windows10，密钥文件会保存在C:\Users\［用户名］\.ssh下，其中id\_rsa是密钥文件，id\_rsa.pub是公钥文件。将公钥文件里的内容上传到github的网站里就能用ssh远程连接到用户的库。

这些完成后，可以用ssh -T [git@github.com](mailto:git@github.com)测试成果。

##### git clone 克隆项目

|  |  |
| --- | --- |
| **git.bash** | |
| 1 | git clone https://github.com/［用户名］/［项目名］.git #将github上的项目克隆到当前文件夹。默认只克隆主分支。 |
| 2 | git clone --mirror https://github.com/［用户名］/［项目名］.git #同上，但一开始就克隆全部的分支。 |

##### git remote 远程仓库

远程仓库名默认是origin。

|  |  |
| --- | --- |
| **git.bash** | |
| 1 |  |
| 2 |  |

##### git push 推送

推送是最后一步。如果想要修改远程仓库的文件，需要先修改本地文件夹内的文件，然后将这个文件上传到暂存区，然后将它提交给本地仓库，如下：

|  |  |
| --- | --- |
| **git.bash** | |
| 1 | #修改文件。 |
| 2 | git add ［文件名］ #将文件添加到暂存区。 |
| 3 | git commit -m “［文件名］” #将更改提交到本地仓库。 |

最后再推送：

|  |  |
| --- | --- |
| **git.bash** | |
| 1 | git push ［远程仓库名］ ［分支名］ #将分支的更改从本地仓库推送到远程仓库。 |