**配置本机用户名和Email地址**

**1** $ git config --global user.name"Your Name"

**2** $ git config --global user.email "email@example.com"

**创建版本库(仓库)**

版本库又叫仓库(repository)，这个目录里面的所有文件都可以被Git管理起来，每个文件的修改、删除都能被跟踪。

在合适的位置直接鼠标右键创建一个空目录作为仓库，然后从Git-Bash命令行进入到该目录，或者也可以使用命令行创建空目录，再进入到该空目录中。

以下给出创建并初始化git仓库的代码：

进入到仓库的位置，我将仓库放在了C:\Android\git-repositories目录下，注意，使用cd命令进入到目录中时，在Git-Bash中应该使用斜线”/”，

而不是反斜线”\”

**1** $ cd C:/Android/git-repositories

**2** $ mkdir new\_repository\_1 创建新的目录

**3** $ cd new\_repository\_1 进入到创建的目录

**使用init命令将当前目录初始化为Git仓库**

**1** $ git init

Initialized empty Git repository in C:/Android/git-repositories/new\_repository\_1/.git/

(显示信息意思为：初始化了一个空的Git仓库，new\_repository\_1目录下多了一个.git目录，时用来管理版本库的)

**添加远程仓库**

**从远程仓库克隆（先有远程库）**

**1** $ git clone git@github.com:username/name.git （或使用<https://github.com/username/name.git>)

**2** $ cd name

**3** $ ls （目录查看）

**本地库与远程库关联（现有本地，去github创建远程库，然后关联）**

1 $ git remote add origin git@github.com:username/name.git （或使用<https://github.com/username/name.git>)

2 $ git push -u origin master （第一次 加'-u '）

**删除远程仓库**

**1** $ git remote （在当前目录下查看）

**2** $ git remote rm 远程仓库名 （删除）

**删除本地仓库**

**1** $ ls -a （找到目录下.git）

**2** $ rm -rf .git （删除.git文件）

**3** $ rm -rf xxxx (在上级目录下删除本地库文件)

**将数据提交到git仓库（本地仓库）**

**第一步：添加文件**

**1** $ git add . 添加所有的文件、文件夹

**2** $ git add <file> 添加指定名称的文件,<>内部写文件全称

注：如果文件没有做出任何修改，则默认不会添加任何文件

**第二步：提交文件**

**1** $ git commit –m “commit info” 提交本次事务，即将add的文件提交到git仓库，引号内部表示本次提交的提示信息

**查询提交状态**

**1** $ git status 显示提交的状态：已经添加，等待提交事务的文件(绿色字体表示)；已经改变但是没有添加(not staged)的文件(红色字体表示)；

**查询该文件和git仓库中的文件的区别，即做了什么修改**

**1** $ git diff <文件全称> 如果已经add了，就打印不出有什么修改了，这一步骤应该在add之前，即添加之前可以用来看看做了什么修改。

**打印历史记录**

**1** $ git log

Commit xxx commit id 版本号

Author:xxx<xxx@xxx.com> 提交人和邮箱

Date：xxx 提交的时间

XXXXXXXXXXXXXX 提交的信息(所以说，提交信息很重要！！！)

**2** $ cat <文件全名称> 显示整个文件的内容

**版本回退**

**1** $ git reset --hard head^

在Git中，HEAD表示当前版本，就是最新提交的版本，即使用git log打印出来的位于第一位的版本，上一个版本就是HEAD^，上上个版本就是HEAD^^，

当前向上100个可以写成HEAD~100。当然，还有一种方式就是直接使用commit id来代替HEAD^，比如版本号是cadab353589f3eef075817b890dafe8b722d802b，

那么就可以直接使用命令：

**2** $ git reset --hard cadab353589f 使用前几位表示即可，git会自动查找

注：版本回退以后，使用git log打印的历史记录都是回退版本之前的数据，之后的都没有了，不过放心，git总有后悔药可以吃的

1.如果命令行窗口没有关闭，直接去前面找commit id即可；

2.如果命令行窗口关闭了，或者第二天后悔了，可以进入到该目录下，使用git reflog命令来查看以前的每一次命令，可以获得每次提交的commit id，

就可以版本回退了。

**3** $ git reflog 可以查看命令历史，包含提交的commit id。

**第三步：删除文件**

一般情况下，你通常直接在文件管理器中把没用的文件删了，或者用rm命令删了：

**1** $ rm test.txt

这个时候，Git知道你删除了文件，因此，工作区和版本库就不一致了，git status命令会立刻告诉你哪些文件被删除了：

**2** $ git status

现在你有两个选择， 一是确实要从版本库中删除该文件， 那就用命令git rm删掉，并且git commit：

**3** $ git rm test.txt

**4** $ git commit -m "remove test.txt"

现在，文件就从版本库中被删除了。

**第四步：撤销删除文件/修改**

**（1）撤销删除文件**

另一种情况是删错了，因为版本库里还有呢，所以可以很轻松地把误删的文件恢复到最新版本：

**1** $ git checkout -- test.txt

**git checkout其实是用版本库里的版本替换工作区的版本，无论工作区是修改还是删除，都可以“一键还原”。**

**（2）撤销修改**

场景1：当你改乱了工作区某个文件的内容，想直接丢弃工作区的修改时，用命令

**1** $ git checkout -- file。

场景2：当你不但改乱了工作区某个文件的内容，还添加到了暂存区时，想丢弃修改，分两步，第一步用命令git reset HEAD <file>，就回到了场景1，第二步按场景1操作。

**1** $ git reset HEAD <file>

**2** $ git checkout -- file。

场景3：已经提交了不合适的修改到版本库时，想要撤销本次提交，参考版本回退一节，不过前提是没有推送到远程库。

**1** $ git reset --hard head^