# 一、Git的简单使用

（参考链接：<https://blog.csdn.net/javaandroid730/article/details/53522872> ）

git使用最好的参考资料是Pro Git book。<https://git-scm.com/book/zh/v2>

***第一步 下载Git for Windows***

1. 在官网点击Download，下载对应的exe文件，注意你的操作系统是32位还是64位。
2. 双击安装，中间不用做任何改动，一直下一步就行。如果你想修改安装位置，请放在纯英文路径下。

***第二步 创建一个本地hello-world仓库***

1. 安装成功，你现在就可以使用git命令行工具了。在你想要下载代码的路径，点击鼠标右键，选择Git Bash here。注意，你的代码路径也应是纯英文的。（打开命令行（cmd）或者在想要创建repository的地方右键鼠标并点击 Git Bash Here 打开窗口）
2. 在命令行输入 mkdir hello-word，创建一个新文件夹。你可以使用ls命令来查看当前目录下有哪些文件和文件夹。
3. 输入cd hello-world进入新文件夹，注意在输入命令时，你可以用Tab键来自动补全。（也就是hello-world才是仓库）
4. 输入git init初始化Git仓库。此时用ls -a查看当前目录，可以看到多了一个.git/的文件夹，此文件夹保存了版本控制的所有相关信息。

注意，在此处使用的所有命令，如果你是在Linux环境下开发，用法都是完全一样的。所以对于完全没有Linux使用经验的学员，这也是一个开始接触Linux工作方式的好方法。 接下来，让我们创建一份简单的说明文件，并提交到版本库中。

1. 输入echo "This is a simple practise" > readme.txt，创建一个readme.txt文件。（ps:> + 文件名 可创建文件）
2. 输入git status查看当前版本库状态，在Untracked files(未跟踪文件)下，会出现红色的readme.txt，代表此文件还未被Git所管理。
3. 使用git add readme.txt，将该文件加入缓冲区，如果你确定所有的修改都需要提交，可以使用git add .来加入所有修改。现在用git status查看，将看到文件名变为绿色。（add .多一点，因为不用去看文件名）
4. 使用git commit -m "This is my first commit via Git!"来提交修改，-m后面所带的参数是本次提交信息，一般用来记录本次提交的主要意图。
5. 提交成功后，可以用git log查看历史提交记录。每个记录都会有提交id，作者和提交日期。
6. 你可以用git branch查看当前有哪些分支，当然，因为我们没有创建任何分支，目前只会有一个master分支。
7. 使用git checkout -b feature创建一个名为feature的分支，再用git branch查看一下。 以上是最最基本的Git操作，大家可以在此hello-world项目中随意尝试各种其他Git命令，最好的参考资料是Pro Git book。<https://git-scm.com/book/zh/v2>

***第三步：版本回退***

1. git reset --hard HEAD^

HEAD^表示上一个版本，HEAD^^表示上上一个版本，当然往上100个版本写100个^比较容易数不过来，所以写成HEAD~100。

1. git reset --hard 版本号

怎么找版本号

a. git log 可以

b. git reflog 找寻所有历史版本号，满足您的所有需求

***git小结***

--git init

--git add .

--git commit -m "注释"

--git status

--git log

--git diff

--git reset --hard +

注意：学会Git的唯一方式是在实际使用中学习，切记不要尝试先记住一大堆理论知识或者Git命令。

Git基础可以看下面两个链接，特别是上面没有的版本回退，还有SSH的创建

<https://www.jianshu.com/p/296d22275cdd>

<https://www.cnblogs.com/schaepher/p/5561193.html>

git和GitHub的就我做自己的补充，因为两位大佬说的都不全符合我的想法

## 二、 Github与Git的关联（综合了两个链接的方法）

0 前期工作

a. 到Github注册账号。

b. 本地配置用户名和邮箱（如果已经设置好，跳过该步）：

git config --global user.name "你的用户名"

git config --global user.email "你的邮箱"

c. 生成ssh key

运行 ssh-keygen -t rsa -C "你的邮箱" ，它会有三次等待你输入，直接回车即可。

将生成的ssh key复制到剪贴板，执行 clip < ~/.ssh/id\_rsa.pub （或者到上图提示的路径里去打开文件并复制）：

d. 打开Github，进入Settings--点击左边的 SSH and GPG keys--点new SSH key--将ssh key 粘贴到右边的Key里面，Title随便命名即可

--点击下面的---Add SSH key 就添加成功了---测试一下吧，执行 ssh -T git@github.com

***最简单的方法（先远程建仓-克隆本地建立关联-初始化提交-普通提交）***

1. 建GitHub仓库
2. 本地clone（建立关联了） git clone + github克隆链接 这样就不用git remote add origin 你复制的地址
3. 直接本地git有修改即add、commit后，提交远程github 你的修改 git push -u origin master（初次）
4. 然后后面的修改提交就都只用git push

***第二种关联方式（本地建与远程同名仓ps 本地和远程谁先建都可-用命令建立关联-初始化提交-普通提交）***

1. 本地建仓或已有仓（建仓就是文件夹git init）
2. 远程建同名仓
3. git remote add origin 你复制的地址->建立关联 如果你在创建 repository 的时候，加入了 README.md 或者 LICENSE (就是远程仓比本地仓高版本)，那么 github 会拒绝你的 push 。 你需要先执行 git pull origin master。（pull命令 从远程仓库同步，在本地版本低于远程仓库版本的时候，获取远程仓库的commit）
4. 直接本地git有修改即add、commit后，提交远程github 你的修改 git push -u origin master（初次）
5. 然后后面的修改提交就都只用git push

**大佬总结**

1. 设置用户名 config --global user.name "你的用户名"
2. 设置邮箱 config --global user.email "你的邮箱"
3. 生成ssh key ssh-keygen -t rsa -C "你的邮箱" 这条命令前面不用加git 4.clone 地址 从网络上某个地址拷贝仓库(repository)到本地
4. 添加远程仓库 remote add origin 你复制的地址 设置origin
5. 上传并指定默认 push -u origin master 指定origin为默认主机，以后push默认上传到origin上
6. 提交到远程仓库 push 将当前分支增加的commit提交到远程仓库
7. 从远程仓库同步 pull 在本地版本低于远程仓库版本的时候，获取远程仓库的commit