1. 查询Git配置及所在的文件夹

git config --list --show-origin

1. 设置用户名和邮件地址

C:\Users\sunguiquan>git config --global user.name "felix sun"

C:\Users\sunguiquan>git config --global user.email [sunguiquan1980@sina.com](mailto:sunguiquan1980@sina.com)

1. 如针对特定项目使用不同的用户名称与邮件地址时，可以在那个项目目录下运  
   行没有 --global 选项的命令来配置。
2. 设置编辑器

C:\Users\sunguiquan>git config --global core.editor "'F:\applicationfolder\npp.7.8.9.bin.x64.exe' -multiInst -notabbar -nosession -noPlugin"

1. 检查配置

C:\Users\sunguiquan>git config –list

1. 检查某一项配置，如名称：

C:\Users\sunguiquan>git config user.name

1. 检查哪个文件最后设置了某个值，如名称：

C:\Users\sunguiquan>git config user.name

1. 查找某个用法，也可以到<https://freenode.net/>寻求帮助。

C:\Users\sunguiquan>git help config

C:\Users\sunguiquan>git config –help

1. 在已存在的目录中初始化仓库，如目录G:\gitprojects 执行：

G:\gitprojects>git init

Initialized empty Git repository in G:/gitprojects/.git/

1. 设置代理和取消（如不使用代理，则不必设置）

G:\gitprojects>git config --global http.proxy <http://20.0.22.20:8080>

G:\gitprojects>git config --global https.proxy <http://20.0.22.20:8080>

git config --global --unset http.proxy

git config --global --unset https.proxy

1. 拉取项目，本例如下：

G:\gitprojects>git clone <https://github.com/guardiangel/gitstudy>

这会在当前目录下创建一个名为 “gitstudy” 的目录，并在这个目录下初始化一个 .git 文件夹， 从远程仓库拉取下所有数据放入 .git 文件夹，然后从中读取最新版本的文件的拷贝。 如果你进入到这个新建的 gitstudy文件夹，你会发现所有的项目文件已经在里面了，准备就绪等待后续的开发和使用 如重新命名，则可以使用如下命令

G:\gitprojects>git clone <https://github.com/guardiangel/gitstudy> newName

1. 查看文件状态

G:\gitprojects\gitstudy>git status

1. 跟踪一个文件

G:\gitforstudy>git add readme.txt

G:\gitforstudy>git status readme.txt

On branch master

Changes to be committed:

(use "git restore --staged <file>..." to unstage)

new file: readme.txt

1. 修改一个文件，再放入暂存区

G:\gitforstudy>git add readme.txt

G:\gitforstudy>git status readme.txt

On branch master

Changes to be committed:

(use "git restore --staged <file>..." to unstage)

new file: readme.txt

1. 任何修改之后，都需要运行git add。否则执行第一次git add之后，再修改文件，类似于如下：

G:\gitforstudy>git status readme.txt

On branch master

Changes to be committed:

(use "git restore --staged <file>..." to unstage)

new file: readme.txt

Changes not staged for commit:

(use "git add <file>..." to update what will be committed)

(use "git restore <file>..." to discard changes in working directory)

modified: readme.txt

1. 简写显示模式

G:\gitforstudy>git status –s readme.txt

AM readme.txt

G:\gitforstudy>git status –short readme.txt

AM readme.txt

1. 查看修改了哪些内容 git diff 文件名

G:\gitforstudy>git diff readme.txt

git diff 本身只显示尚未暂存的改动，而不是自上次提交以来所做的所有改动。 所以有时候你一下子暂存了所有更新过的文件，运行 git diff 后却什么也没有，就是这个原因

1. 若要查看已暂存的将要添加到下次提交里的内容，可以用 git diff --staged 命令

G:\gitforstudy>git diff –staged

1. 如使用git commit ，则当前受控文件，即便未进行add，也会被提交到本地库.
2. 提交所有文件

G:\gitforstudy>git commit -m "commit all existed files"

[master 670747b] commit all existed files

3 files changed, 27 insertions(+)

create mode 100644 src/main/java/com/gitforstudy/gittest/GittestApplication.java

create mode 100644 src/main/resources/application.properties

create mode 100644 src/test/java/com/gitforstudy/gittest/GittestApplicationTests.java

1. 如不使用gid add，直接对修改文件提交，可采用git commit –a命令
2. 移除已跟踪文件 git rm 文件名

G:\gitforstudy>git rm readme.txt

rm 'readme.txt'

G:\gitforstudy>git status

On branch master

Changes to be committed:

(use "git restore --staged <file>..." to unstage)

deleted: readme.txt

1. 如果删除之前修改过的或放入到暂存区的文件，使用git rm –f,这样的数据不能被git 恢复。
2. 只删除跟踪记录，文件保存在磁盘。

G:\gitforstudy>git rm --cached readmeforsave.txt

rm 'readmeforsave.txt'

G:\gitforstudy>git status

On branch master

Changes to be committed:

(use "git restore --staged <file>..." to unstage)

deleted: readmeforsave.txt

Untracked files:

(use "git add <file>..." to include in what will be committed)

readmeforsave.txt

1. git rm 命令后面可以列出文件或者目录的名字，也可以使用 glob 模式。比如

G:\gitforstudy>git rm log/\\*.log

注意到星号 \* 之前的反斜杠 \， 因为 Git 有它自己的文件模式扩展匹配方式，所以我们不用 shell 来帮忙展开。此命令删除 log/ 目录下扩展名为 .log 的所有文件

1. 删除所有名字以 ~ 结尾的文件

git rm \\*~

1. 文件移动

G:\gitforstudy>git mv readmeforsave.txt readmetwo.txt

G:\gitforstudy>git status

On branch master

Changes to be committed:

(use "git restore --staged <file>..." to unstage)

renamed: readmeforsave.txt -> readmetwo.txt

1. 提交之后，发现需要再提交一次的情况，并且两次提交认为是一次。使用git commit –amend命令。第二次提交会代替第一次提交的结果。

G:\gitforstudy>git commit --amend -m "recommit readmetwo.txt,forget it just now"

当你在修补最后的提交时，并不是通过用改进后的提交 **原位替换** 掉旧有提交的方式来修复的， 理解这一点非常重要。从效果上来说，就像是旧有的提交从未存在过一样，它并不会出现在仓库的历史中。修补提交最明显的价值是可以稍微改进你最后的提交，而不会让“啊，忘了添加一个文件”或者 “小修补，修正笔误”这种提交信息弄乱你的仓库历史

1. 重新设置暂存区数据

G:\gitforstudy>git status

On branch master

Changes to be committed:

(use "git restore --staged <file>..." to unstage)

modified: readmetwo.txt

G:\gitforstudy>git reset readmetwo.txt

G:\gitforstudy>git status readmetwo.txt

On branch master

Changes not staged for commit:

(use "git add <file>..." to update what will be committed)

(use "git restore <file>..." to discard changes in working directory)

modified: readmetwo.txt

git reset 命令比较危险，慎用。

1. 查看远程仓库

git remote –v

1. 创建项目并推送远端github网站
2. 在github上建立一个项目gitforstudy，记住地址，本例如：

https://github.com/guardiangel/-gitforstudy.git

2．本地建立一个项目，并在项目根目录下用命令行执行 git init

3.在项目根目录dos下执行

G:\gitforstudy>git remote add origin <https://github.com/guardiangel/-gitforstudy.git>

G:\gitforstudy>git branch -M main

G:\gitforstudy>git push -u origin main

Logon failed, use ctrl+c to cancel basic credential prompt.

Username for 'https://github.com': blueskywei@yeah.net

Password for 'https://blueskywei@yeah.net@github.com':

Enumerating objects: 34, done.

Counting objects: 100% (34/34), done.

Delta compression using up to 4 threads

Compressing objects: 100% (23/23), done.

Writing objects: 100% (34/34), 3.27 KiB | 1.09 MiB/s, done.

Total 34 (delta 3), reused 0 (delta 0), pack-reused 0

remote: Resolving deltas: 100% (3/3), done.

To https://github.com/guardiangel/-gitforstudy.git

\* [new branch] main -> main

Branch 'main' set up to track remote branch 'main' from 'origin'.s

以上第三步需要输入用户名和密码

4.在github上面检查验证。

31.查看远程仓库

G:\gitforstudy>git remote

origin

G:\gitforstudy>git remote -v

origin https://github.com/guardiangel/-gitforstudy.git (fetch)

origin https://github.com/guardiangel/-gitforstudy.git (push)

32添加远程仓库

G:\gitforstudy>git remote -v

origin https://github.com/guardiangel/-gitforstudy.git (fetch)

origin https://github.com/guardiangel/-gitforstudy.git (push)

G:\gitforstudy>git remote add pb https://github.com/guardiangel/-gitforstudy.git

G:\gitforstudy>git remote -v

origin https://github.com/guardiangel/-gitforstudy.git (fetch)

origin https://github.com/guardiangel/-gitforstudy.git (push)

pb https://github.com/guardiangel/-gitforstudy.git (fetch)

pb https://github.com/guardiangel/-gitforstudy.git (push)

1. 添加远程仓库后，可以拉取远程仓库中独有的信息。你可以在命令行中使用字符串 pb 来代替整个 URL。 例如，如果你想拉取 Paul 的仓库中有但你没有的信息，可以运行 git fetch pb。必须注意 git fetch 命令只会  
   将数据下载到你的本地仓库——它并不会自动合并或修改你当前的工作。 当准备好时你必须手动将其合并入你的工作。

git fetch pb

1. 抓取文件到当前分支

G:\gitforstudy>git pull origin

Logon failed, use ctrl+c to cancel basic credential prompt.

Username for 'https://github.com': blueskywei@yeah.net

Password for 'https://blueskywei@yeah.net@github.com':

G:\gitforstudy>git status

git clone 命令会自动设置本地 master 分支跟踪克隆的远程仓库的 master 分支（或其它名字的默认分支）。 运行 gitpull 通常会从最初克隆的服务器上抓取数据并自动尝试合并到当前所在的分支。

1. 推送到远程仓库git push <remote> <branch>当你想要将 master 分支推送到 origin 服务器时（再次说明，克隆时通常会自动帮你设置好那两个名字）origin是远端分支，本地分支可以在intellij中，右键-git-repository中看到，也就是第31步中所设置的名称。

G:\gitforstudy>git push origin main

Logon failed, use ctrl+c to cancel basic credential prompt.

Username for 'https://github.com': blueskywei@yeah.net

Password for 'https://blueskywei@yeah.net@github.com':

Everything up-to-date

1. 查看远端分支

G:\gitforstudy>git remote show origin

Logon failed, use ctrl+c to cancel basic credential prompt.

Username for 'https://github.com': blueskywei@yeah.net

Password for 'https://blueskywei@yeah.net@github.com':

\* remote origin

Fetch URL: https://github.com/guardiangel/-gitforstudy.git

Push URL: https://github.com/guardiangel/-gitforstudy.git

HEAD branch: main

Remote branch:

main tracked

Local branch configured for 'git pull':

main merges with remote main

Local ref configured for 'git push':

main pushes to main (up to date)

1. 远程仓库的命名和删除

G:\gitforstudy>git remote rename pb paul

G:\gitforstudy>git remote

1. 移除一个远程仓库（实际上是在本地，非远端仓库名称）

G:\gitforstudy>git remote remove paul

G:\gitforstudy>git remote

Origin

1. 列出已有的标签

G:\gitforstudy>git tag

1. 创建标签，分为轻量标签和附注标签。轻量标签很像一个不会改变的分支——它只是某个特定提交的引用。而附注标签是存储在 Git 数据库中的一个完整对象， 它们是可以被校验的，其中包含打标签者的名字、电子邮件地址、日期时间， 此外还有一个标签信息，并且可以使用 GNU Privacy Guard （GPG）签名并验证。 通常会建  
   议创建附注标签，这样你可以拥有以上所有信息。

G:\gitforstudy>git tag -a v1.0 -m "my version 1.0"

G:\gitforstudy>git tag

G:\gitforstudy>git show v1.0

轻量标签：

轻量标签本质上是将提交校验和存储到一个文件中——没有保存  
任何其他信息。 创建轻量标签，不需要使用 -a、-s 或 -m 选项，只需要提供标签名字

G:\gitforstudy>git tag v1.4-lw

补打标签：

G:\gitforstudy>git log --pretty=oneline

8bce2c13778a469d350cc3ca8e438c1500619585 <E6><8F><90><E4><BA><A4>git<E5><AD><A6><E4><B9><A0>

559bd6bd1a0451d6ee4d180e8dbc172194924cb5 Create addfromremotehost.txt

f72a7a139f5978dea877cd12d3fea348fcdce94f Create README.md

f53dfcd9fbda649967037e25b67e3ec37cc06f3b recommit readmetwo.txt,forget it just now

db46e8fcbc446df2bcc0a5ccc898789e4cab9d3c commit readmeforsave.txt

a0de35c1459d91523b9d858f05958d672cf81dc1 test commit readme.txt

670747b434c6564e222a61014f375ac2fc7b9696 commit all existed files

481d977266ac6c2cb6e4a83b9e799cbee5be6fb9 commit readme.txt

G:\gitforstudy>git tag -a v0.9 add7bf4a2208480d1faf3189ac8a1 -m "add another old tag"

40.推送所有标签到git

G:\gitforstudy>git push origin –tags

使用 git push <remote> --tags 推送标签并不会区分轻量标签和附注标签， 没有简单的选项能够让你只选择推送一种标签

1. 删除标签:

G:\gitforstudy>git tag -d v1.4-lw

1. 设置别名：

G:\gitforstudy>git config --global alias.co checkout

G:\gitforstudy>git config --global alias.br branch

G:\gitforstudy>git config --global alias.ci commit

G:\gitforstudy>git config --global alias.st status