配置用户信息

$ git config --global user.name "John Doe"

$ git config --global user.email johndoe@example.com

列出所有 Git 当时能找到的配置

git config --list

来检查 Git 的某一项配置

git config <key>

获取 Git 仓库

第一种是在现有项目或目录下导入所有文件到 Git 中；

第二种是从一个服务器克隆一个现有的 Git 仓库。

在现有目录中初始化仓库

如果你打算使用 Git 来对现有的项目进行管理，你只需要进入该项目目录并输入：

$ git init

克隆现有的仓库

git clone [url]

查看状态

git status

添加操作

git add 文件名(将工作区新建/修改的内容添加到暂存区)

提交操作

git commit -m “commit message” 文件名(将暂存区的内容提交到本地库)

查看历史记录

git log

前进和后退

a)基于索引值的操作（推荐做法）

git reset --hard 哈希索引值

示例：找回删除状态已经提交本地库的文件操作。

b)使用^符号 （只能后退，一个^表示后退一步）

git reset --hard HEAD^

c)使用~符号 （只能后退，n表示后退n步）

git reset --hard HEAD~2

比较文件差异

a） git diff [文件名] (将工作区中的文件和暂存区的进行比较)

b） git diff [本地库历史版本] [文件名] (将工作区中的文件和本地库历史记录比较，不带文件名的话，会比较多个文件)

分支管理

在版本控制过程中，使用多条线同时推进多个任务。
分支的优势?
a)同时并行推进多个功能开发，提高开发效率。
b)各个分支在开发过程中，如果某个分支开发失败，不会对其他分支有影响，失败的分支可以删除，然后重新开始即可。

分支常用命令:

a)git branch -v （查看本地库中的所有分支）

b)git branch dev (创建一个新的分支)
c)git checkout dev （切换分支）

分支合并

i)切换到接收修改的分支
git checkout master
ii)执行merge命令
git merge dev
（注：切换分支后，在dev分支中做出的修改需要合并到被合并的分支master上)

冲突解决

当一个分支的内容和另一个分支的内容不同时，此时任一分支合并另一分支过程中就会出现冲突。
冲突的解决办法：
a)编辑文件，删除特殊符号。
b)将文件修改完毕后，保存退出。
c)git add [文件名]。
d)git commit –m “日志信息”。