一、新建代码库（本地仓库）

1.git init在当前目录新建一个Git代码库

2.git clone [地址]从远程仓库克隆一个项目和它的整个代码历史

二、配置

1.git config --list显示当前的Git配置

2.git config -e --global编辑Git配置文件

3.设置提交代码时的用户信息

git config --global user.name "[name]"

git config --global user.email "[email address]"

三、增加/删除文件

1.git add [文件名1] [文件名2] ...添加指定文件到暂存区

2.git add [目录名]添加指定目录到暂存区，包括子目录

3.git add .添加当前目录所有文件到暂存区，包括子目录

4.git rm [文件名1] [文件名2] ...删除工作区文件，并且将这次删除放入暂存区

5.git rm --cached [文件名]停止追踪指定文件，但该文件会保留在工作区

6.git mv [原名] [现名]改名文件，并且将这个改名放入暂存区

四、代码提交

1.git commit -m [提交信息]提交暂存区到本地仓库区

2.git commit [文件名1] [文件名2] ... -m [提交信息]提交暂存区的指定文件到本地仓库区

3.git commit -a提交工作区自上次commit之后的变化，直接到本地仓库区

4.git commit -v提交时显示所有diff信息

5.git commit --amend -m [新的提交信息]使用一次新的commit，替代上一次提交（如果代码没有任何新变化，则用来改写上一次commit的提交信息）

五、分支

1.git branch列出所有本地分支

2.git branch -r列出所有远程分支

3.git branch -a列出所有本地分支和远程分支

4.git branch [分支名]新建一个分支，但依然停留在当前分支

5.git checkout -b [分支名]//git switch -c [分支名]新建一个分支，并切换到该分支

6.git switch [分支名]直接切换到已有的分支

7.git branch [分支名] [commit]新建一个分支，指向指定commit

8.git branch --track [分支名] [远程分支]新建一个分支，与指定的远程分支建立追踪关系

9.git checkout [分支名]切换到指定分支，并更新工作区

10.git checkout -切换到上一个分支

11.git branch --set-upstream [分支名] [远程分支]建立追踪关系，在现有分支与指定的远程分支之间

12.git merge [分支名]合并指定分支到当前分支

13.git cherry-pick [commit]选择一个commit，合并进当前分支

14.git branch -d [分支名]删除本地分支

15.git branch -D [分支名]强制删除未合并本地分支

15.git push origin --delete [分支名]删除远程分支

六、标签

1.git tag列出所有tag

2.git tag [标签名]新建一个tag在当前commit

3.git tag [标签名] [commit]新建一个tag在指定commit

4.git tag -d [标签名]删除本地tag

5.git push origin :refs/tags/[标签名]删除远程tag

6.git show [标签名]查看tag信息

7.git push [remote] [标签名]提交指定tag

git push [remote] --tags提交所有tag

git checkout -b [分支名] [标签名]新建一个分支，指向某个tag

七、查看信息

1.git status显示有变更的文件

2.git log显示当前分支的版本历史

3.git log --stat显示commit历史，以及每次commit发生变更的文件

4.git log -S [keyword]搜索提交历史，根据关键词

5.git log [标签名] HEAD --pretty=format:%s显示某个commit之后的所有变动，每个commit占据一行

6.git log [标签名] HEAD --grep feature显示某个commit之后的所有变动，其"提交说明"必须符合搜索条件

7.git log --follow [文件名]//git whatchanged [文件名]显示某个文件的版本历史，包括文件改名

8.git log -p [文件名]显示指定文件相关的每一次diff

9.git log -5 --pretty --oneline显示过去5次提交

10.git shortlog -sn显示所有提交过的用户，按提交次数排序

11.git blame [文件名]显示指定文件是什么人在什么时间修改过

12.git diff显示暂存区和工作区的差异

13.git diff --cached [文件名]显示暂存区和上一个commit的差异

14.git diff HEAD显示工作区与当前分支最新commit之间的差异

15.git diff [first-branch]...[second-branch]显示两次提交之间的差异

16.git diff --shortstat "@{0 day ago}"显示今天你写了多少行代码

17.git show [commit]显示某次提交的元数据和内容变化

18.git show --name-only [commit]显示某次提交发生变化的文件

19.git show [commit]:[文件名]显示某次提交时，某个文件的内容

20.git reflog显示当前分支的最近几次提交

八、远程同步

1.git fetch [remote]下载远程仓库的所有变动

2.git remote -v显示所有远程仓库

3.git remote show [remote]显示某个远程仓库的信息

4.git remote add [shortname] [url]增加一个新的远程仓库，并命名

5.git pull [remote] [分支名]取回远程仓库的变化，并与本地分支合并

6.git push [remote] [分支名]上传本地指定分支到远程仓库

7.git push [remote] --force强行推送当前分支到远程仓库，即使有冲突

8.git push [remote] --all推送所有分支到远程仓库

九、撤销

1.git checkout [文件名]恢复暂存区的指定文件到工作区

2.git checkout [commit] [文件名]恢复某个commit的指定文件到暂存区和工作区

3.git checkout .恢复暂存区的所有文件到工作区

4.git reset [文件名]重置暂存区的指定文件，与上一次commit保持一致，但工作区不变

5.git reset --hard重置暂存区与工作区，与上一次commit保持一致

6.git reset [commit]重置当前分支的指针为指定commit，同时重置暂存区，但工作区不变

7.git reset --hard [commit]重置当前分支的HEAD为指定commit，同时重置暂存区和工作区，与指定commit一致

8.git reset --keep [commit]重置当前HEAD为指定commit，但保持暂存区和工作区不变

9.git revert [commit]新建一个commit，用来撤销指定commit（后者的所有变化都将被前者抵消，并且应用到当前分支）

10.git stash//git stash pop暂时将未提交的变化移除，稍后再移入

十、其他

1.git archive生成一个可供发布的压缩包

2.cd改变目录cd /c改变到c盘cd /d改变到d盘cd [文件名]改变到该目录下文件所在地址

3.cd ..改变到上一级目录

4.pwd显示当前所在路径

5.ls//ll列出当前目录所有文件ll更详细

6.touch新建一个文件

7.reset重新初始化终端/清屏

8.clear清屏

9.history查看命令历史

10.help帮助

11.exit退出

12.#表示注释

Workspace：工作区

Index / Stage：暂存区

Repository：仓库区（或本地仓库）

Remote：远程仓库