一 shell命令回顾：

cd 更改目录

pwd 显示当前目录

mkdir 创建一个空的文件夹

touch 创建一个空的文件

cat 读取文件内容，翻页显示

二 git安装

在windows下安装git需要Cygwin这样的模拟环境。msysgit已经将此模拟环境和Git打包好了。安装这个exe程序即可。

在开始菜单看到git-git bash就证明安装上了。

因Git是分布式版本控制系统，所以每个机器必须自报家门：你的名字和Email地址。

故需要配置：

$ git config --global user.name "YOUR NAME"

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

注意 git config命令的--global参数，表示你这台机器上所有的Git仓库都会使用这个配置，当然也可以对某个仓库指定不同的用户名和Email地址。

三 创建仓库，将文件添加到版本库

$ git init #这个是将当前目录变成git可以管理的仓库

$ git add file(或者文件夹名) #这个是将file给添加到仓库中。在commit之前可以多次add不同的文件

$ git commit -m "本次的提交说明" #将添加到仓库中的提交到仓库。

只有git add和git commit两步都操作了，才算将文件放到仓库里了

四 掌握工作区状态

$ git status # 掌握当前工作区状态，如文件已被修改过但还未储存，文件准备被提交，没有需要提交的修改，等等

$ git diff file # 查看file这个文件做了哪些修改。diff是difference的缩写

五 版本管理

$ git log #当前版本提交历史，显示每次commit的ID号/时间/提交说明（用于回退很好用）

$ git log --pretty='oneline' #显示当前版本的每次commit，每行显示一个提交，有ID号/提交说明

$ git reflog #命令历史，显示所有每次操作的ID号/提交说明（用于回到未来的哪个版本很好用）

git用HEAD表示当前版本，HEAD^表示上一个版本，HEAD^^表示上上一个版本，多的话，HEAD~100表示往上第100个版本

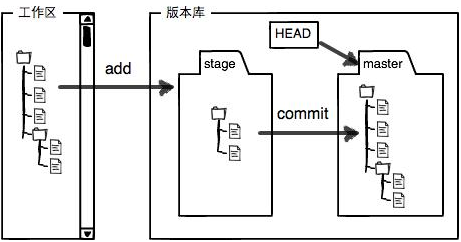
其实HEAD相当于一个指针，当回溯或更改版本时，改的是HEAD的指向

$ git reset --hard HEAD^ #回溯到上一个版本

$ git reset --hard commit\_ID号 #回溯到特定版本，若想回溯的commit ID 是d4d1940，则$ git reset --hard d4d1940

六 工作区与暂存库

工作区（working directory）就是电脑里能看到的目录



.git这个文件就相当于版本库，里面有：

stage（或者index）叫暂存区，

Git自动为我们创建的第一个分支master，

指向master的HEAD指针

七 只要修改而不add，就不会commit到版本库。

如一个文件，先修改再add，然后再修改，然后commit。结果发现第二次修改的没有被commit进去。这就是因为只要修改了，git都会记录，但修改后一定也要add并commit

Git内部存储的不只是difference，是整个文件。

八 撤回工作区的修改

当在工作区修改后，想撤回。用：

$ git checkout -- file #回到最近的一个add或commit命令的版本，其实是版本库里的版本替换工作区里的版本

如，在工作区修改后，还未add，使用上述命令可以将修改删掉

若在工作区修改后，已经add，则此命令不起什么作用，但想unstage掉，这时要用reset，

使用$ git reset HEAD file即可

reset可以回退版本，也可以把暂存区里的修改回退到工作区。使用HEAD时表示最新的版本

然后再使用$ git checkout -- file即可

九 删除文件

在工作区添加了一个文件，而且add并commit到了git库。

在工作区删除了这个文件，然后git status发现提示这个文件在工作区被删除了：

1.也想把git库的删除，则可以

$ git rm file # 记得也要commit

$ git commit -m '删除文件file' # 这时此文件才从git库删除

2.发现只是误删，想恢复回来，则

$ git checkout -- file #这样工作区的file文件就回来了，实际上就是git库里的那个版本

十 远程仓库

以上都是讲如何在git库里对文件操作，分布式git的魅力之一就是远程仓库。

git分布式版本管理系统，同一个git库，可以分布在不同的机器上，而且没有主次之分。

可以找一台电脑充当服务器的角色，其他人都可以推送或者提取。

服务器可以自己搭建，也可以使用GitHub，它提供git仓库托管服务。

本地git仓库和github仓库之间的传输是通过SSH加密的，故

1.在本机用户主目录下查看是否有.ssh目录，里面是否有id\_rsa和id\_rsa.pub这两个文件，没有的就：

$ ssh-keygen -t rsa -C "your@email.com"

然后一路回车即可。key可以不设置密码。

之后用户主目录可看到id\_rsa和id\_rsa.pub文件，id\_rsa是私钥，不要告诉别人；id\_rsa.pub是公钥，可以随便告诉别人。

2.登陆GitHub，点击setting->ssh->ssh key->add key，将id\_rsa.pub里的公钥粘到内容里即可

可以把你每台电脑的key都添加到GitHub中去，这样你所有电脑都可以将内容推送到GitHub了。

Git库是公开的，要么交钱私有化要么自己大件git服务器。

十一 添加远程仓库

在GitHub上添加一个repository，然后按照上面的提示将本地的推送过去

$ git remote add origin https://github.com/queenaLou/GitLearning.git

关联一个远程库

$git push -u origin master

这样就将本地库添加到远程库了。

注意因首次添加仓库，用了-u参数，有两个作用，1.git把本地的master分支内容推送到远程的新的master分支内容；2.把本地master分支和远程master分支关联起来，在以后的推送或拉取时简化命令。

之后只要本地做了修改，就可以用$ git push origin master同步到GitHub了。

若每次都让输入账号密码，极有可能是开始添加origin时用的HTTPS协议，需要改变登录的协议。 创建ssh的key之后用如下命令更改为ssh

$git remote rm origin

$git remote add origin "Git仓库的ssh格式地址"

$git push origin master

SSH协议比HTTPS协议要快。

十二 从远程库克隆

$ git clone git地址(如git@github.com:queenaLou/GitLearning.git)