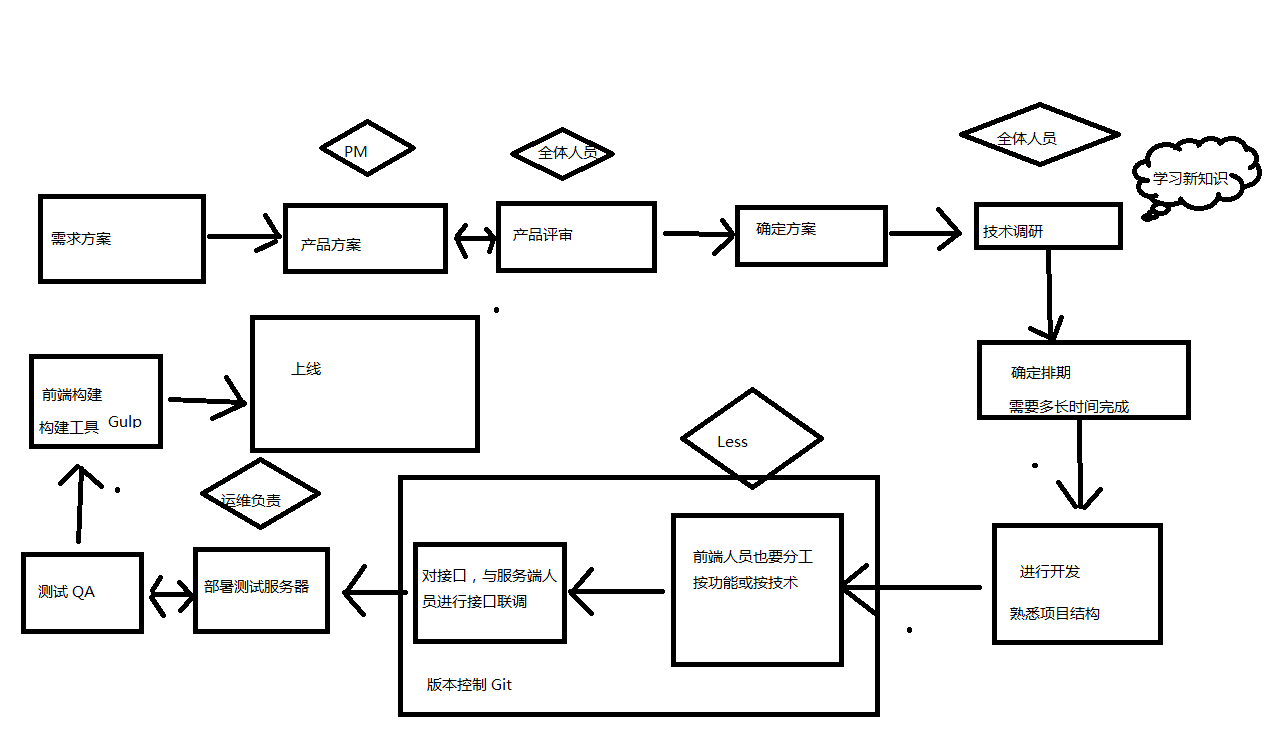
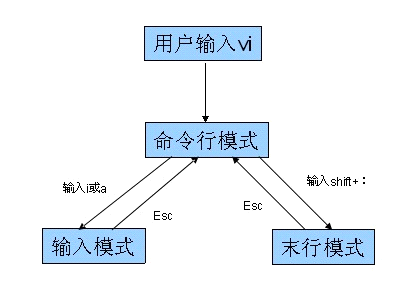
一：说明开发过程中的工作流程

二：什么是shell：**在计算机科学中，Shell俗称壳，用来区别于Kernel（核），是指“提供使用者使用界面”的软件（命令解析器）。它类似于**[**DOS**](http://baike.baidu.com/view/365.htm)**下的command和后来的cmd.exe。它接收用户命令，然后调用相应的应用程序**

**三：vi编辑器：vi编辑器提供了3种模式，分别是命令模式、插入模式、底行模式，每种模式下用户所能进行的操作是不一样的**

**使用vi编辑器**

a) 打开/创建文件， vi 文件路径

b) 底行模式 :w保存，:w filenme另存为

c) 底行模式 :q退出

d) 底行模式 :wq保存并退出

e) 底行模式 :e! 撤销更改，返回到上一次保存的状态

f) 底行模式 :q! 不保存强制退出

------------------------------------------------------------------------------------------------

h) 命令模式 ZZ（大写）保存并退出

i) 命令模式 u辙销操作，可多次使用

j) 命令模式 dd删除当前行

k) 命令模式 yy复制当前行

l) 命令模式 p 粘贴内容

o) 命令模式 i进入编辑模式，当前光标处插入

p) 命令模式 a进入编辑模式，当前光标后插入

q) 命令模式 A进入编辑模式，光标移动到行尾

r) 命令模式 o进入编辑模式，当前行下面插入新行

s) 命令模式 O进入编辑模式，当前行上面插入新行

当我们处在编辑模式的情况下，和我们在Windows编辑器的使用相似

**四：认识bash这个shell**

**bash常见命令**

pwd (Print Working Directory) 查看当前目录

cd (Change Directory) 切换目录，如 cd ./etc

ls (List) 查看当前目录下内容，如 ls –al 查看得更详细

ll 查看当前目录修改的文件的相关信息

mkdir (Make Directory) 创建目录，如 mkdir blog

touch 创建文件，如 touch index.html

cat 查看文件内部内容，如 cat index.html

rm (remove) 删除文件，如 rm index.html、rm -rf 为全删除 //是删除工作区的文件，删除后也要提交（不用add）不然会残留在git仓库里面 rm –cached //删除暂存区的缓存

rmdir (Remove Directory) 删除文件夹，只能删除空文件夹，不常用

mv (move) 移动文件或重命名，如 mv index.html ./demo/index.html

cp (copy) 复制文件，cp index.html ./demo/index.html

head 查看文件前几行，如 head -5 index.html

history 查看操作历史

who am i 查看当前用户

| 管道符

grep 匹配内容，一般结合管道符使用

* 五：版本控制

1.本地版本控制系统

2.集中式版本控制系统-SVN

3.分布式版本控制系统-git

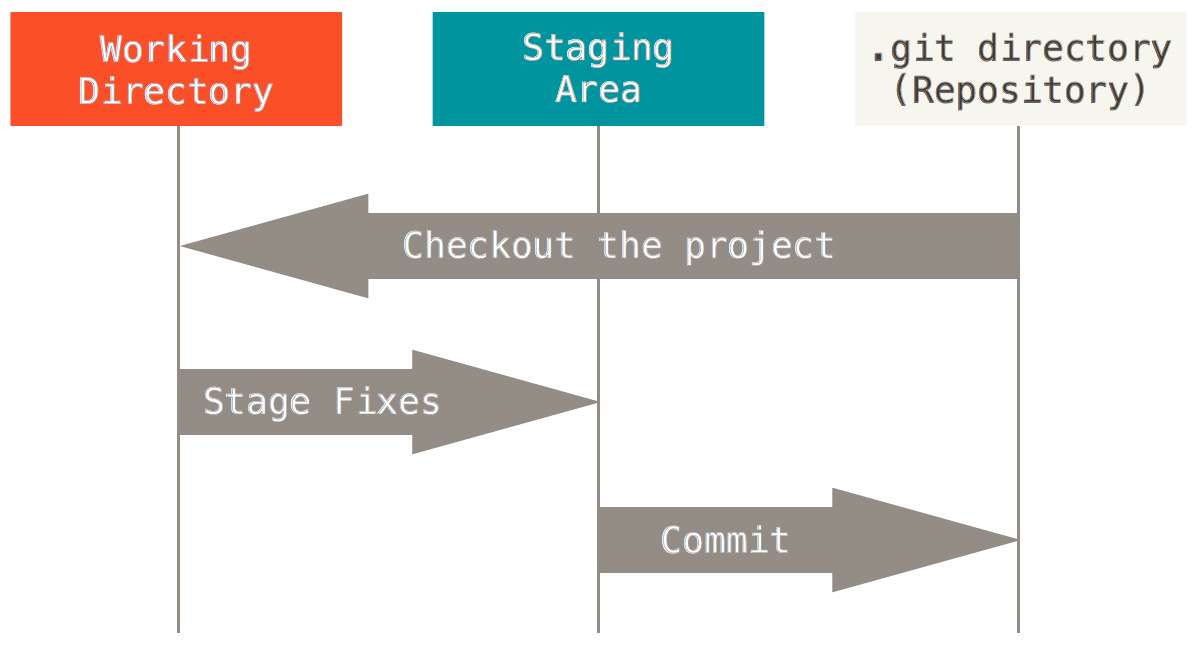
六：git

* git工作原理

Git仓库目录是Git用来保存项目的元数据和对象数据库的地方。 这是Git 中最重要的部分，从其它计算机克隆仓库时，拷贝的就是这里的数据。

工作目录是对项目的某个版本独立提取出来的内容。 这些从Git仓库的压缩数据库中提取出来的文件，放在磁盘上供你使用或修改。

暂存区域是一个文件，保存了下次将提交的文件列表信息，一般在Git仓库目录中。有时候也被称作“索引”（Index），不过一般说法还是叫暂存区域。

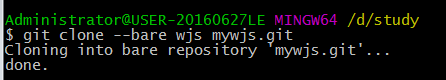
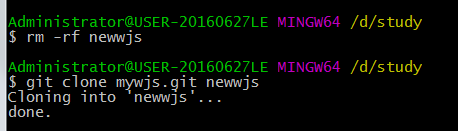
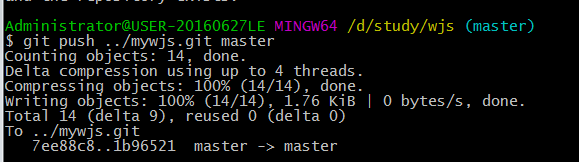
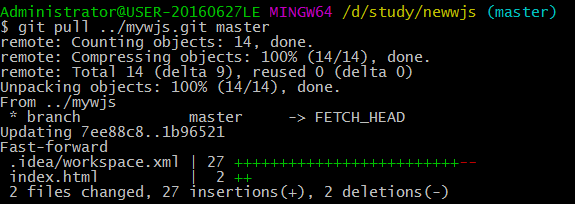


基本的Git工作流程如下：

1、在工作目录中修改文件。

2、暂存文件，将文件的快照放入暂存区域。

3、提交更新，找到暂存区域的文件，将快照永久存储到Git仓库目录

* git安装
* 配置用户名： git config --global user.name "用户名"
* 配置邮箱： git config --global user.email "邮箱地址"
* 查看配置信息： git config --list
* 初始化仓库：
* 命令：git init
* 作用：使用Git管理项目的初始化操作，会创建一个名为.git的隐藏目录,创建git仓库
* 一般是在项目的根目录执行这个命令
* 跟踪文件
* 将文件由 工作区 添加到 暂存区（Index），暂存文件
* 命令：git add 文件路径
* git add --all 或者 git add -A（简写） 添加所有文件,添加跟踪
* 提交更新
* 作用：将文件由 暂存区 添加到git 仓库（HEAD）
* git commit -m "提交说明"
* 查看文件状态
* 命令：git status
* 红色为未被跟踪，绿色为添加到跟踪（add ），白色为提交到git库（commit –m” ”）
* Git工作流程
* 在工作目录中修改某些文件。
* 对修改后的文件进行快照，然后保存到暂存区域。
* 提交更新，将保存在暂存区域的文件快照永久转储到.git目录中。
* 查看提交日志
* 命令：git log 查看详细的提交信息，里面有每次提交后的版本号，版本号为git自动生成的一串数据
* 版本回退
* 作用：恢复到已经提交的某一个版本中
* 命令：git reset --hard 版本号 //版本号很长，一般只输入前几位就好，是从仓库回退到工作区
* 其他方式：git reset --hard head~1 //版本回退到上一个文件提交的状态
* git reset head 文件名 //某个文件的回退到在暂存区
* git checkout – 文件名 //从暂存区回退到工作区
* git checkout HEAD 文件名//从仓库回退到暂存区，和暂存区回退到工作区，不建议使用
* git分支
* 查看分支：git branch，当前分支会标有一个\* //要先提交git仓库里有东西才能新建分支
* 创建分支：git branch [分支名称]
* 切换分支：git checkout [分支名称] //要是没有提交（commit）则新添加的或修改过的文件会出现在其他分支上，要是提交过后只会出现在提交的分支上
* 合并分支：git merge [分支名称]，即：将其他分支合并到当前分支
* 删除分支：git branch -d [分支名称]
* 创建本地共享仓库:仓库就是一个文件夹
* 
* 通过—bare创建裸仓库
* 共享的仓库必须是一个裸仓库
* 通过从共享仓库中复制内容，左边是复制目标，右边是复制后的名字
* 复制目标工作目录到共享仓库后里面显示的和目标的看起来不一样，但从裸仓库中又复制过去的时候，又变成真正的工作目录下的内容
* 修改内容，并且将修改的内容更新（提交）到裸仓库中
* 进入到另外的目录，将之前提交的内容拉取
* clone本地仓库的时候被clone的远端仓库默认被称为 origin，所以如果你想向/从这个远端仓库push/pull 的时候，用 origin 指代这个远端仓库
* git远程仓库

1 在本地创建仓库

git init

git config

2 新建 README.md 文件，并输入任意内容

3 将 README.md 提交到本地

git add

git commit

4 在github中新建仓库，并拿到仓库地址

git remote add origin 远程连接地址 //设置origin映射为远程地

5 使用命令 git push [仓库地址] master 提交内容到github的默认分支

6 刷新github仓库页面，在线修改 README.md 文件，并提交

7 使用命令 git pull [仓库地址] master 获取仓库中的最新内容

git pull命令的作用是，取回远程主机某个分支的更新，再与本地的指定分支合并。

完整格式：$ git pull <远程主机名> <远程分支名>:<本地分支名>