/\*  
\* git:工具，版本控制工具  
\*  
\* github:程序员的社交网站  
\*  
\* 一.安装git工具  
\*  
\* 二.注册github账号  
\*  
\* 三.绑定git和github  
\*  
\* 四.在github上新建一个项目  
\*  
\* 五.把远程仓库的项目放到本地  
\* cd (进入目录) c:  
\* tab键补全代码  
\* 进入指定目录后，克隆远程仓库的代码  
\* git clone 远程地址  
\* 通过ll或ls来查看目录下的文件  
\*  
\* 六.拖一个html文件进入目录  
\*  
\* 七.本地文件上传到远程:  
\* git add 新添加文件名 (从本地放到缓存区)  
\* git add . (多个文件一起放入缓存区)  
\* git commit -m "注释" (从缓存区放入版本区)  
\* git push origin master(推送 相当于git到github)  
\*  
\* 查看状态:(查看的是本地和缓存区)  
\* git status  
\* git log(查看版本记录) (如果有:要返回 按Q键返回)  
\* git diff(查看本地和缓存区的差异)  
\* git diff --cached(查看缓存区和版本区的差异)  
\* git diff master (查看本地和版本区的差异)  
\* git reflog (查看所有版本号)  
\*  
\* 撤销:  
\* git reset HEAD 文件名 (撤销缓存区文件到本地)  
\* git checkout -- 文件名 (本地内容还原成版本区的内容)  
\*  
\* 当某个文件已经提交到版本区后，还有文件需要提交到同一组文件时x  
\* 先把没提交文件commit下，然后在注释后面--amend  
\* 那么最近提交的版本会和这个文件合并成一个版本  
\*  
\* 删除:  
\* git rm 文件名 (当某个文件已经提交到缓存区,该文件在本地已删除,需要在缓存区内删除)  
\* git rm -f 文件名 (同时删除本地和缓存区的文件)  
\* git rm --cached 文件名 (删除缓存区的文件)  
\*  
\* 恢复:  
\* git checkout 版本号 要恢复的文件 (恢复需要恢复的单个文件)  
\* git reset --hard 版本号(通过版本号来恢复本地文件)  
\*  
\* 同步到远程仓库:  
\* git push origin master  
\* git push (推送)  
\* origin (默认的远程仓库名称)  
\* git remote (查看远程仓库的名字)  
\* git remote -v (查看远程仓库对应的地址)  
\* git remote add 名称 (创建仓库的名字)  
\*  
\*  
\*  
\*  
\* 快速拉取远程仓库的代码 (会覆盖本地代码，小心使用)  
\* git pull  
\*  
\* 参与项目开发  
\* fork 项目  
\* clone 项目  
\* 修改提交项目  
\* New pull request  
\*  
\* 拉分支  
\* 默认分支为 master (主干分支)  
\* git branch (查看分支)  
\* git branch 要拉新分支的名字 (拉新分支)  
\* git checkout -b 要拉新分支的名字 (快速拉新分支并切换)  
\* git checkout 要切换的分支 (切换分支)  
\* git merge 要合并的分支 (合并分支)  
\* git branch -D 要删除的分支(强制删除)  
\* git branch -d 要删除的分支(合并后删除分支)  
\*  
\* 注意:不能在本分支上删除本分支，需要到其他分支或master上去删除  
\*  
\* 个人博客  
\* 一样的创建项目:  
\* 自己的名字.github.io;  
\*  
\* 注意:博客中必须要有一个html的文件，名字必须为index.html(只能有一个)  
 \* 文件或文件夹的名字不能为中文  
\*  
\* \*/  
  
/\*  
\* requirejs.cn  
\*  
\* \*/