#### GIT

**分支合并冲突**

· 在不同分支上，修改同一个文件

· 不同的人，修改了同一个文件

· 不同的仓库，修改了同一个文件

· 冲突只在合并分支的时候才会发生

· 发生冲突并不可怕，冲突的代码不会丢；

· 解决冲突，冲突重新提交 commit -m 时不要给 message;

· 冲突信息格式,

**GIT** **命令**

\1. ls -a 查看当前目录文件

\2. git init 创建本地仓库

\3. git add . 提交到暂存区

\4. git commit -m"提交到远程仓库"

\5. git 远程仓库地址

\6. git -u origin master提交到master分支上 -u 当前

\7. git status 查看状态

\8. git diff 命令 查看修改 内容

\9. vim 修改 a.txt文件名后缀名

\10. git clone 克隆远程仓库 地址名

\11. git pull 刷新当前

\12. git log 查看日志 提交了几次条数

\13. git pull origin master 拉取远端的master分支的更新，并与本地分支合并

\14. git diff 可以查看远端和本地代码的冲突

\15. git stash pop 读取栈中最近一次保存的内容

\16. git merge local 代码合并分支

\17. git branch -a 查看所有分支 -r 显示所有分支 -v查看分支 -d删除分支 -D强制删除分支

\18. git checkout 切换分支

\19. git fetch 获取 所有远程分支

\20. git stash clear 删除所有缓存

\21. git status lit 查看所有缓存 pop 恢复

\22. git reset HEAD test.txt 将暂存区文件撤回到本地

\23. git show 显示提交详情

\24. touch . 创建 .文件名

\25. touch .gitignroe 创建一个文件夹用来忽略不想上传的项目

24.   vim：   
vim test.txt   打开要编辑的文本  
vim +#   打开文件，并定位到#行  
vim 四中模式：  
  正常模式；命名模式；插入模式；可视模式。从其他模式退出到正常模式的时候esc即可。  
正常模式命令：  
  h   向前一个字符  
  l   向后一个字符  
  j   同位置向下走  
  k   同位置向上走  
  n   查找下一个  
  N   查找上一个  
  这几个命令前加上数字，表示向前多少个字符  
​  
  :q 退出编辑  
  :wq 保存并退出  
  :q! 不保存退出  
  :w 保存  
  :w! 强行保存  
​  
  w 移到下一个单词的词首  
  e   跳到当前或下一个单词的词尾  
  b   跳到当前或前一个单词的词首  
  #w 移动#个单词  
​  
  gg 第一行  
  G   最后一行  
  #G 跳到第几行  
​  
  Ctrl+f 向下翻一屛  
  ctrl+b 向上翻一屛  
  ctrl+d 向下翻半屛  
  ctrl+u 向上翻半屛  
​  
  x   删除当前光标所在位置的一个字符  
  #x 删除光标所在位置及向后的#个字符  
  dd 删除当前光标所在行  
      yy   复制一行  
  p     粘贴。粘贴一行的话，放在当前行的下边。  
​  
  u   撤销操作  
正常模式--->插入模式  
  i   在光标所在字符前输入，转到输入模式  
  a 在光标所在字符后输入，转到输入模式  
  o（字母 o） 在光标所在行的下面单独开一行，并转到输入模式  
  s 删除光标所在的字符，并进入到输入模式  
  I（大写 i） 在当前行的行首开始输入文字，并进入输入模式。如果在行首有空格，则在空格后开始插入。  
  A 在行尾进行输入。进入输入模式  
  O（大写 O） 在当前行的前一行插入空行，并进入输入模式  
  S 删除光标所在的一行，并进入输入模式。  
可视模式（就是可以选取一行，或者选取一个区域）：  
  v   进入可视模式，然后使用方向控制就可以进行选取

#### gitgnore文件

* 所有空行或以空格注释符号#开头的都会被**Git**忽略
* 可以使用标准的glob模式匹配
* 匹配模式最后跟( / )说明要忽略目录
* 要忽略指定模式以外的文件目录，可以加上 ！ 号进取反