# Git操作

## 版本管理的三种方式

第一种方式：本地管理 即CVS 缺点：不能进行多人协作开发 只在本地管理

第二种方式：集中式管理 SVN 这种方式有个中央服务器 协同开发的程序员通过客户端连接到这个中央服务器 取出最新的文件或者提交更新 缺点：依赖于网络 如果中央服务器连接不上 无法提交代码

第三种方式： 分布式管理 git 这种方式拉取的不仅是最新版本的代码 而是代码仓库完整的镜像 这样一来 每次操作就会对代码仓库进行完整的备份

好处： 不依赖网络 本身电脑就是一个服务器 没网时照样可以提交

## Git的下载安装

略

注意：安装后右键鼠标 会出现2个选项

第一个选项：git gui 即图形化操作

第二个选项：git bash 即命令行的方式操作

## Git的版本管理分为3个区

工作区 暂存区 仓库区

工作区：就是程序员写代码的地方

暂存区：表示暂时存放的区域：注意git中 无法将工作区的代码直接提交到仓库区 需要从工作区到暂存区再到仓库区

仓库区：讲存储在暂存区的内容 永久保存到git的仓库中 并且生成版本号 生成版本号之后 可以回退到任何一个版本中

## git的基本指令

### git init

指令作用：这个表示初始化git 会在文件夹下生成一个.git的隐藏文件

实际原理：执行git init 之后 会创建出暂存区 和 仓库区 都在.git文件中

### git add

指令作用：讲工作区的内容添加到 暂存区

可跟参数：

1： 文件名(要加后缀) git add index.html

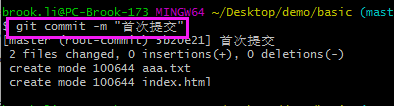
2: 直接跟文件夹 git add js

3： 通配符的形式 git add \*.js

4： 添加所有 git add . 或者 git add -A 或者git add --all

### git commit -m

指令作用：将暂存区的内容提交到仓库区 并且生成版本号 永久保存在git仓库中



注意：如果没有使用 -m 添加提交信息 则会进入vi编辑页面

### git commit -a -m

指令作用：直接将工作区的内容 提交到仓库区

本质：也是通过暂存区再到仓库区

但是注意：这个指令无法操作 还没有追踪过的文件

### git commit --amend -m

指令作用：修改最后一次提交的 提交信息

### git status

指令作用：查看文件的状态

如果是红色：表示文件在工作区

如果是绿色：表示文件在暂存区

### git log

指令作用：查看提交日志

参数 --oneline 在一行显示一个日志

### Git reflog

指令作用：查看所有版本信息 包含回退的版本信息

### git config

可配置相关

常用参数

--list 查看所有配置 git config --list

--global 配置全局配置 git config --global user.name 辉哥

--unset --global 取消全局配置 git config --unset --global user.name

## Git的对比指令

### git diff

指令作用：比较工作区和暂存区

### git diff --cached

指令作用：表示对比暂存区和仓库区的不同

### git diff HEAD

指令作用：表示工作区和仓库区的不同

## Git版本重置

### git reset

指令作用：表示版本重置

可跟参数一: --hard 版本号 表示工作区 暂存区 和仓库区 同时重置

可跟参数二： --soft 版本号 表示只重置仓库区

可跟参数三：--mixed 版本号 表示重置仓库区和暂存区

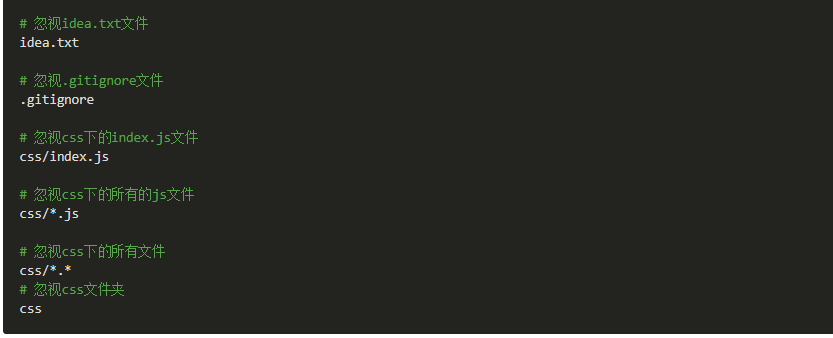
不跟参数 比如 git reset 版本号 这种方式和 --mixed一样

## git的忽略文件

操作步骤

第一步：在.git目录的同级目录下 添加.gitignore文件

在这个文件中 添加忽略文件



注意一：在window系统下 创建.gitignore文件的时 主要加.

注意二：每个忽略的文件 要占一行 不能一行写多个

## git的分支处理

原理：git的分支 其实就是一个指针 每次提交代码后 只是这个分支的指针想回移动 指向最后一次提交的版本 HEAD分支 指向当前分支

### git branch

指令含义：表示查看所有分支

注意：默认时 是没有任何分支的 只有第一次提交代码时 才会出现master分支 即主分支

### git branch 分支名

指令含义：表示创建分支

### git checkout 分支名

切换分支 原理只是把HEAD指针指向了 当前分支

### Git checkout -b 分支名

表示创建并且切换分支

### Git merge 分支名称

合并分支

## Git的远程操作

### Git clone

克隆远程仓库代码到本地 可以加文件创建文件夹的名称 如何添加 则默认是仓库名

Git clone 远程仓库 huige 表示拉取远程仓库的代码 到huige文件夹中

如果不添加huige 则使用远程仓库名 作为本地文件夹的名字

### git push 远程仓库 分支名

git push origin master origin 表示远程仓库 当然也可以使用仓库地址

直接 git push 什么都不加 默认的就是master分支

### git pull

作用 将远程代码下载到本地

### Git remote

给仓库起别名

Git remote add 仓库别名 仓库地址

Git remote add huige <http://.......git>

Git remote remove 仓库别名 删除这个别名

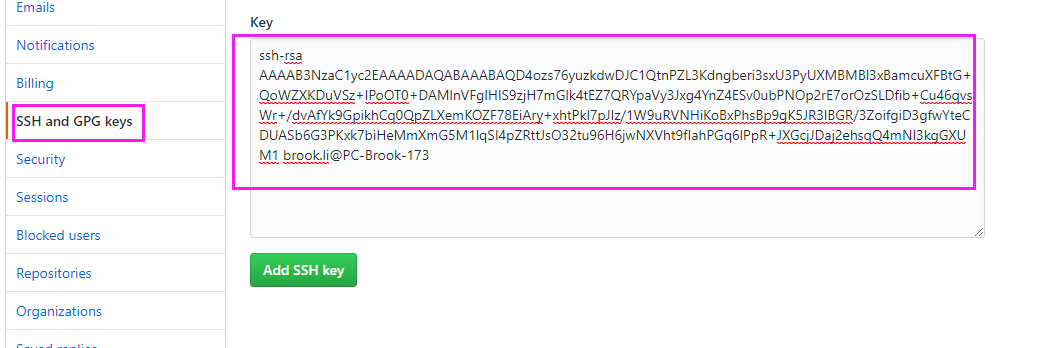
## SSH免密码登录

第一步：创建SSH Key 命令是：ssh-keygen -t rsa ssh-keygen 中间没有空格

第二步：在C盘的当前用户文件夹中找到.ssh目录 里面有2个文件

一个公钥 一个私钥 (公钥的带.pub)

第三步：在github中配置公钥 把公钥内容复制一份 粘贴到这个地方



第四步：创建仓库 找到SSH链接clone 就可以了

## Git提交到非空仓库

### 方式一：

第一步：先拉取远程仓库内容

作用：允许不相关的历史记录

命令：Git pull 地址或者origin --allow-unrelated-histories

命令执行完毕之后 因为是merge的原因 会弹出编辑器 这个时候:q退出就 行了

第二步： 直接push到远程仓库

Git push origin marster

或者git push 地址

### 方式二：

第一步：执行命令

git pull --rebase origin 或者 git pull --rebase 地址

第二步：直接push

Git push origin marster

或者git push 地址