# .gitignore

## 参考资料

<https://www.cnblogs.com/kevingrace/p/5690241.html>

<https://mirrors.edge.kernel.org/pub/software/scm/git/docs/user-manual.html>

<https://git-scm.com/book/en/v2>

<http://eagain.net/articles/git-for-computer-scientists/>

## 配置语法

以斜杠“/”开头表示目录；

以星号“\*”通配多个字符；

以问号“?”通配单个字符

以方括号“[]”包含单个字符的匹配列表；

以叹号“!”表示不忽略(跟踪)匹配到的文件或目录；

此外，git 对于 .ignore 配置文件是按行从上到下进行规则匹配的，意味着如果前面的规则匹配的范围更大，则后面的规则将不会生效；

# 查看

## 中文显示为数字

### 参考资料

<https://blog.csdn.net/zhujiangtaotaise/article/details/74424157>

<https://www.cnblogs.com/fallenmoon/p/8659678.html>

### 详细步骤

git config --global core.quotepath false

## 查看当前状态

git status

## 查看远程服务器地址

git remote -v

## 查看分支

git branch

# 新增

## 把文件添加到跟踪中

git add -A

## 撤销git add添加的所有改动

git reset HEAD

# 删除

## 删除一个文件

git rm 文件

## 重命名文件夹

git mv 源文件夹 目标文件夹

## 删除远程服务器中的文件，但是保留本地

### 参考资料

<https://blog.csdn.net/sl543001/article/details/73826375>

### 详细步骤

git rm -r --cached .project

git rm -r --cached .settings/

## 放弃本地修改

### 参考资料

<https://www.cnblogs.com/qufanblog/p/7606105.html>

### 未使用git add添加过

可以使用 git checkout -- filepathname (比如： git checkout -- readme.md ，不要忘记中间的 “--” ，不写就成了检出分支了！！)。

放弃所以的文件修改可以使用 git checkout . 命令。

此命令用来放弃掉所有还没有加入到缓存区（就是 git add 命令）的修改：内容修改与整个文件删除。

但是此命令不会删除掉刚新建的文件。因为刚新建的文件还没已有加入到 git 的管理系统中。所以对于git是未知的。自己手动删除就好了。

### 已经使用git add添加过

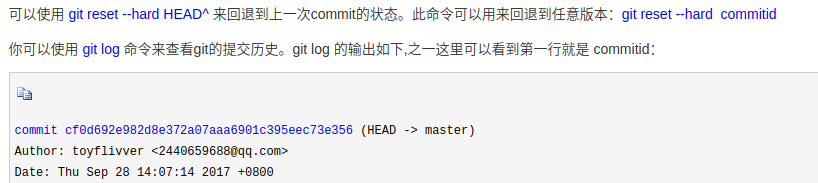
可以使用 git reset HEAD filepathname （比如： git reset HEAD readme.md）来放弃指定文件的缓存，

放弃所有的缓存可以使用 git reset HEAD . 命令。

此命令用来清除 git 对于文件修改的缓存。相当于撤销 git add 命令所在的工作。

在使用本命令后，本地的修改并不会消失，而是回到了如[未使用git add添加过](#_未使用git add添加过)所示的状态。继续用[未使用git add添加过](#_未使用git add添加过)中的操作，就可以放弃本地的修改。

### 已经git commit提交过



# 提交

## 提交到本地

git commit

git commit -m "注释"

## 推送到远程服务器

git push

# 下载

## 克隆（Clone）

### 普通clone

git clone <repository>

### 递归clone

git clone <repository> --recursive

## 从远程服务器下载

git pull

# 分支

## 如何下载分支代码

### 参考资料

<https://segmentfault.com/q/1010000010080086>

<https://blog.csdn.net/u013347784/article/details/73839511>

### Clone方式

如果还没有clone，则可以通过指定分支clone：

Git clone -b 分支 URL

### Checkout切换分支

git checkout -b 远程仓库分支名字 origin/远程仓库分支名字

git pull

# 子模块

## 参考资料

<https://git-scm.com/book/zh/v2/Git-%E5%B7%A5%E5%85%B7-%E5%AD%90%E6%A8%A1%E5%9D%97>

<https://www.cnblogs.com/nicksheng/p/6201711.html>

## .gitmodules

该配置文件保存了项目 URL 与已经拉取的本地目录之间的映射。

## 添加子模块

git submodule add <repository> <path>

## 初始化子模块

git submodule init

## 更新子模块

git submodule update

## 拉取所有子模块

git submodule foreach git pull

# Github相关

## Github提交代码不用每次输入用户名和密码

### 参考资料

<https://segmentfault.com/a/1190000008435592>

### 详细步骤

git config --global credential.helper store

这一步会在用户目录下的.gitconfig文件最后添加：

[credential]

helper = store

push你的代码 (git push), 这时会让你输入用户名和密码, 这一步输入的用户名密码会被记住, 下次再push代码时就不用输入用户名密码!这一步会在用户目录下生成文件.git-credential记录用户名密码的信息。

# 参考资料

<https://www.liaoxuefeng.com/wiki/0013739516305929606dd18361248578c67b8067c8c017b000>

<https://git-scm.com/book/zh/v2>