# 了解Git和Github

## Git和Github

**Git**是一个免费、开源的版本控制软件。

|  |  |
| --- | --- |
| * 版本控制是记录一个或若干个文件变化，以便将来查阅特定版本修订情况的系统。 | |
| * 记录文件的所有历史变化 * 随时可恢复到任何一个历史状态 | * 多人协作开发或修改 * 错误恢复 |

Github是面向开源及私有软件项目的全球最大社交编程托管平台。**因只支持 Git 作为唯一的版本库格式进行托管**，故名 GitHub。Github是项目代码托管平台，提供一个web界面（用户名.github.io/仓库名）借助git管理项目代码。

## 基本概念

|  |
| --- |
| * 仓库（Repository）：仓库即你的项目，即用来存放项目代码，每个项目对应一个仓库，多个开源项目则有多个仓库。 * 收藏（Star）：收藏项目，方便下次查看。仓库主页star按钮！ * 复制克隆项目（Fork）：你开源了一个项目，别人想在这个项目上做改进，应用到自己项目中，就可 Fork 你的项目（点击项目主页右上角fork按钮），然后他的 GitHub 主页上就多了个项目，只不过这个项目是基于你的项目（本质上是在原有项目的基础上新建一个分支），他可随心所欲的去改进，但丝毫不会影响原有项目的代码与结构。 * 发起请求（Pull Request）：基于 Fork发起请求，如果别人改进后觉得不错，于是就想把自己的改进合并到原项目，这时就可发起一个 Pull Request（简称PR） ，原有项目创建人，就可收到这个请求，如果觉得OK，就会接受他的PR，这时候改进原有项目。 * 关注（Watch）：关注项目，当项目更新可以接收到通知。 * 事务卡片（Issue）: 发现代码BUG，但是目前没有成型代码，需要讨论时用；他就可以给你提个 Issue，然后你就可去逐个修复，修复ok了就可以一个个的 Close 掉。 |
| * Github主页：账号创建成功或点击网址导航栏github图标都可进入github主页：该页左侧主要显示用户动态以及关注用户或关注仓库的动态；右侧显示所有的git库 * 仓库主页：仓库主页显示项目信息，如：项目代码，版本，收藏/关注/fork情况等 * 个人主页：个人信息：头像，个人简介，关注我的人，我关注的人，我关注的git库，我的开源项目，我贡献的开源项目等信息 |

# 使用github

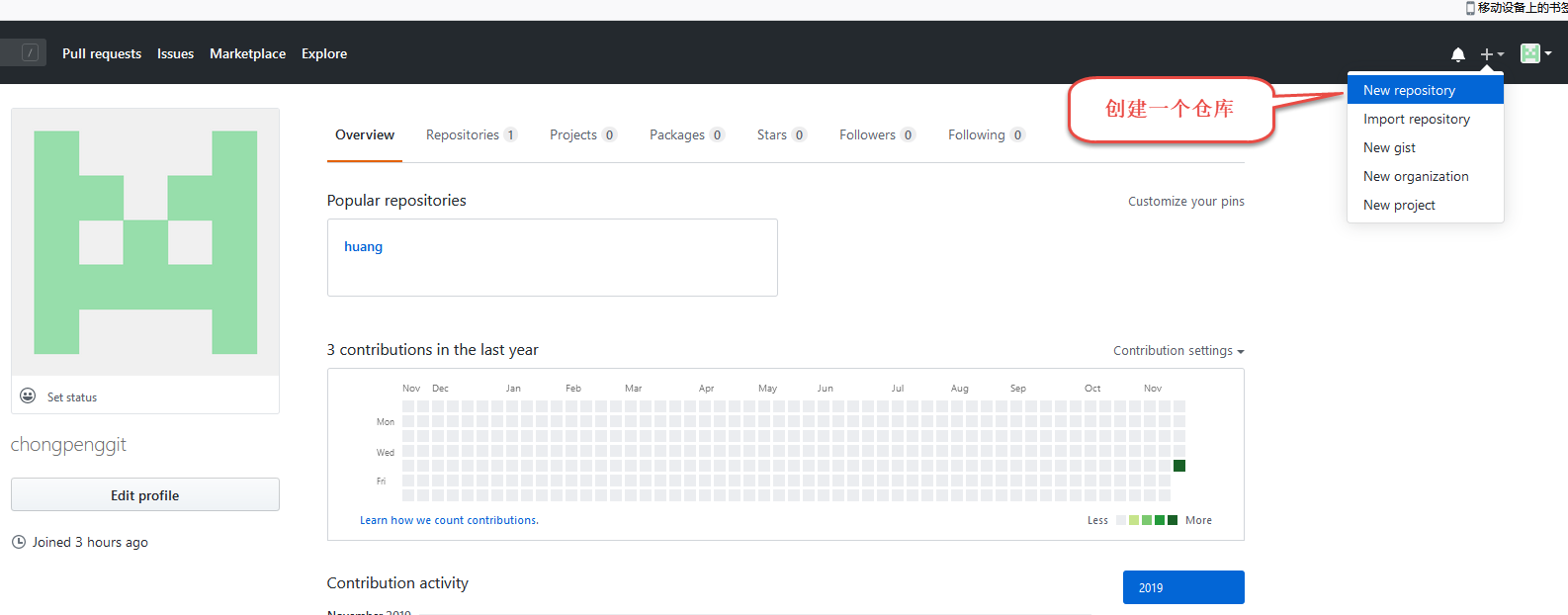
## 注册

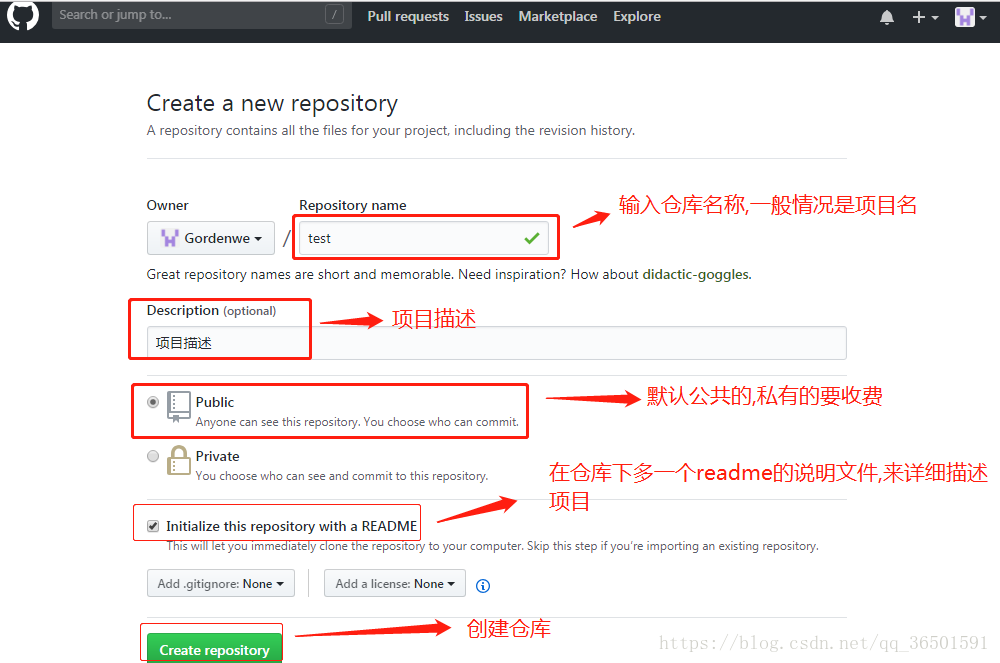
<https://github.com/> ，注册免费版。有时需github上下载翻墙软件**Shadowsocks**。

|  |
| --- |
| * 私有仓库只能自己或者指定的朋友才有权限操作（私有仓库也免费了） * 新注册的用户必须验证邮箱后才可以创建git库仓库（QQ邮箱，可能需要设置域名白名单——在反垃圾中） |

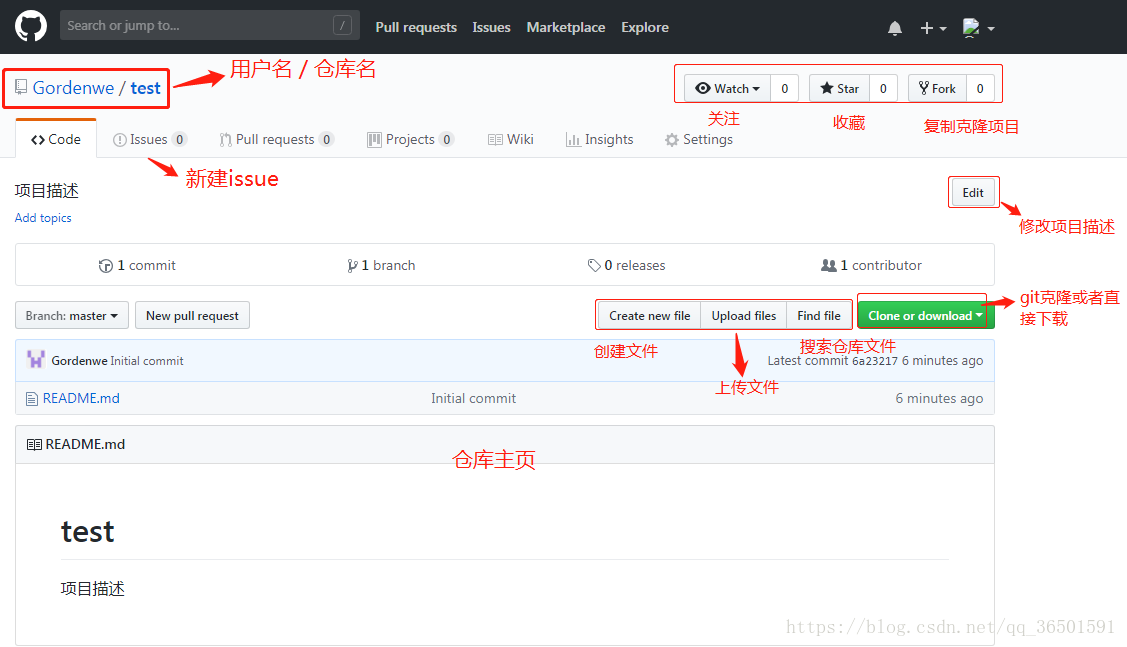
## 创建仓库

* 进入**用户主页**<https://github.com/chongpenggit> （其中chongpenggit是你的用户名），创建仓库。

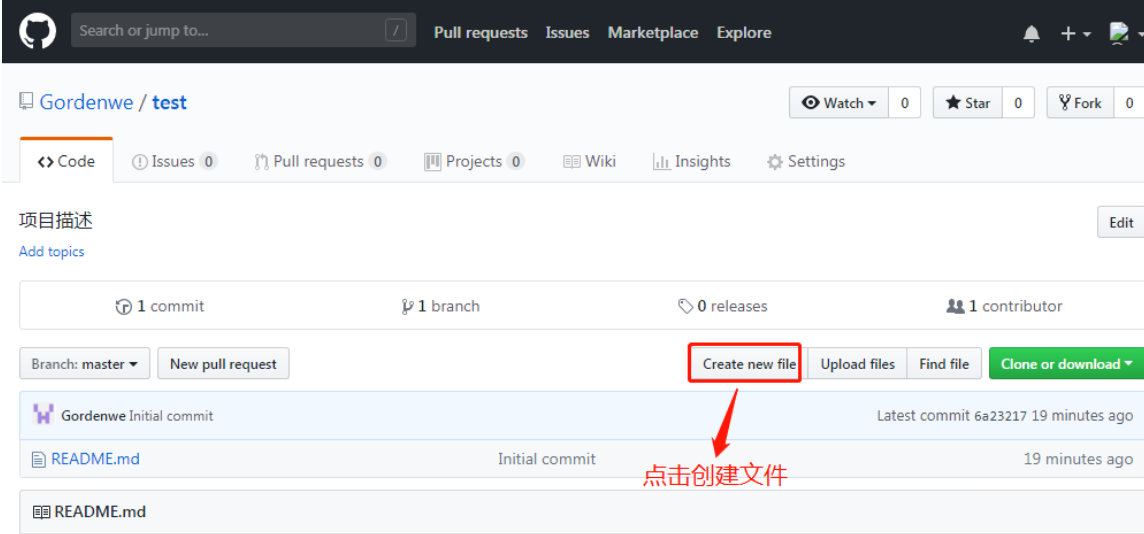


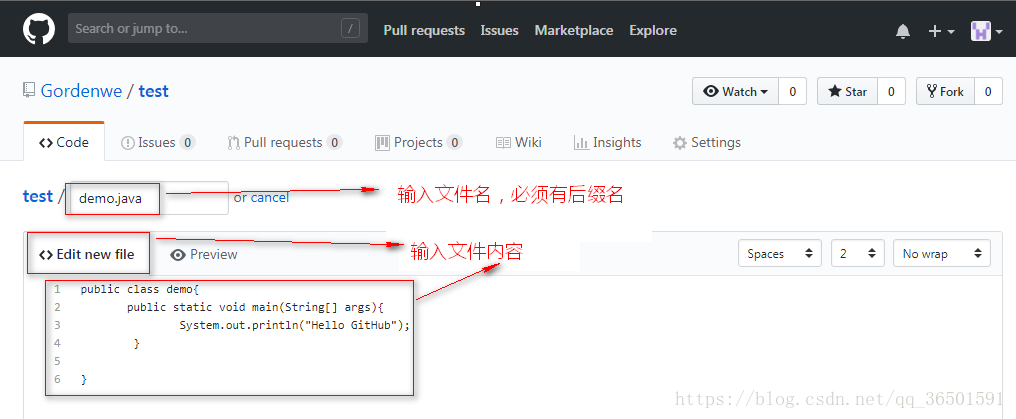


* 仓库主页



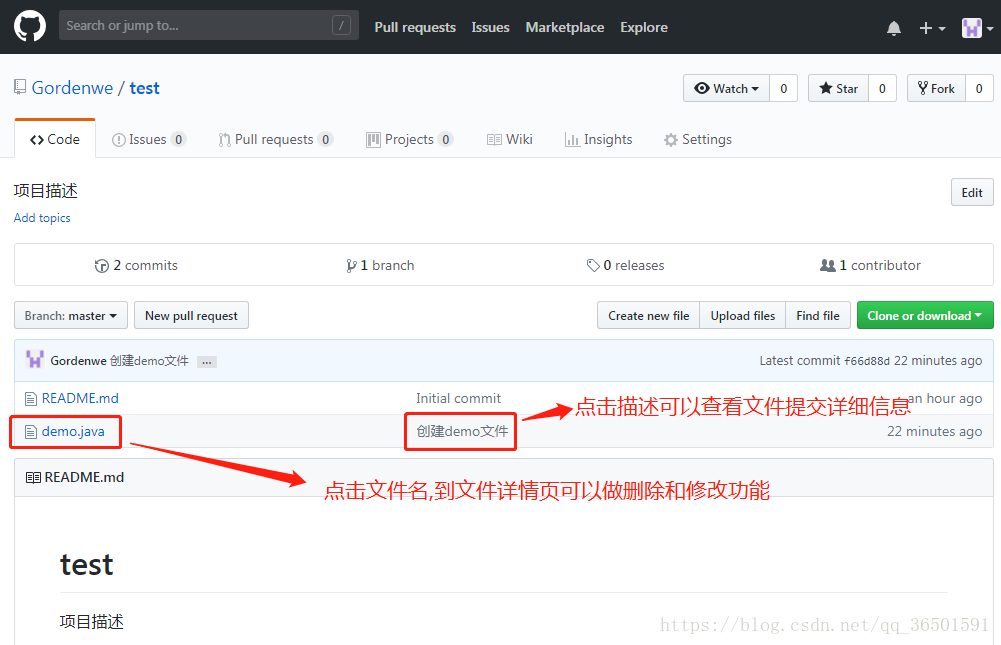
## 创建文件

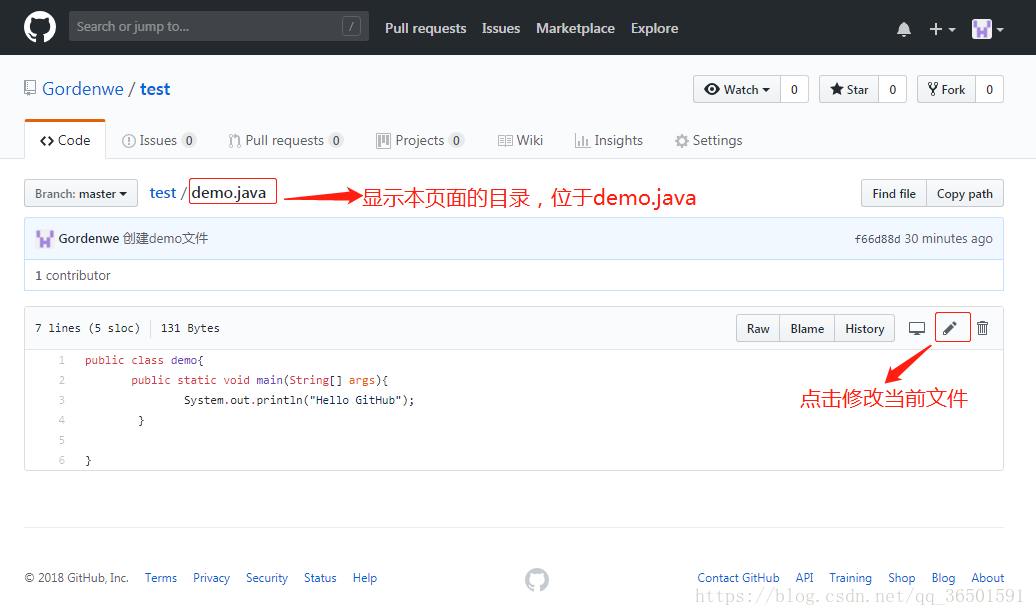


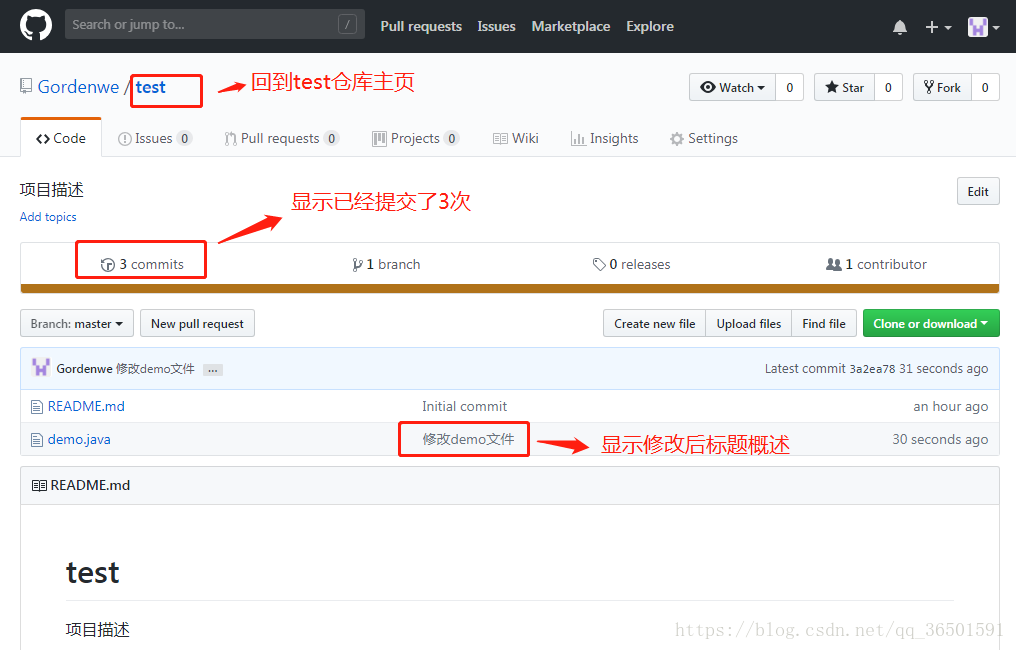




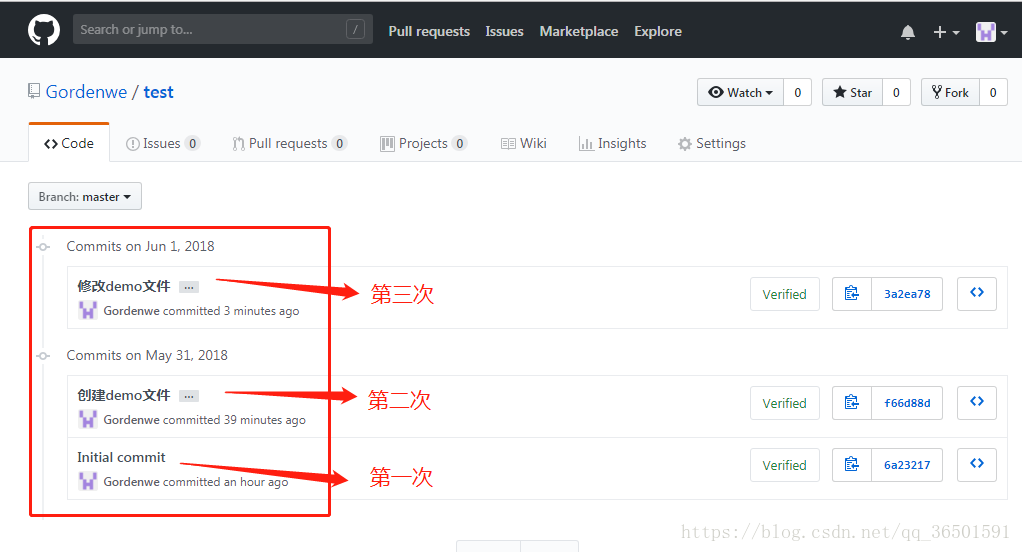
## 修改文件



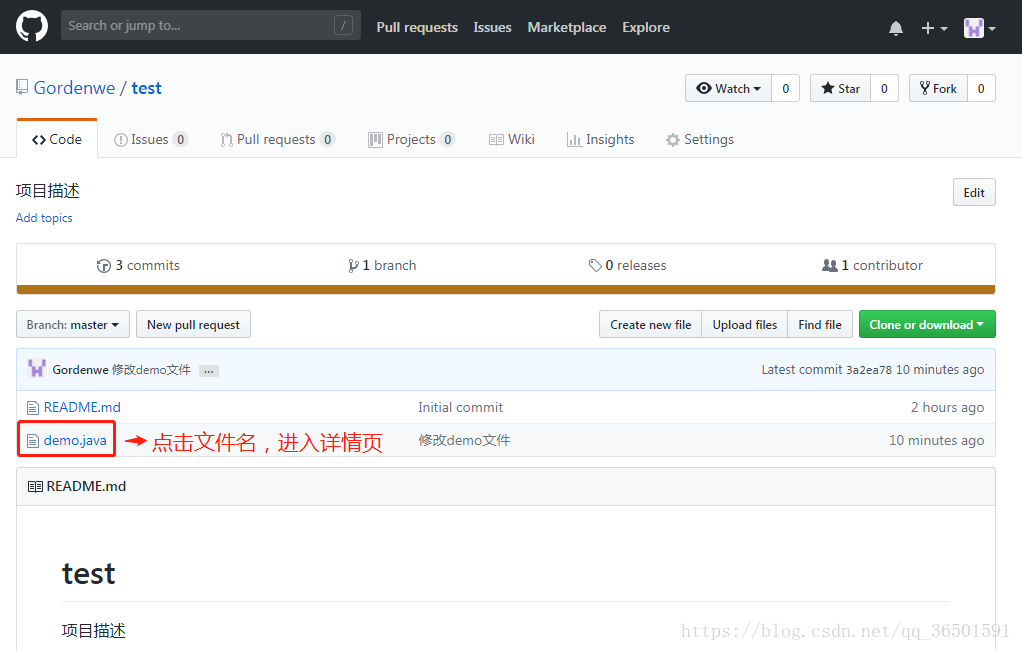


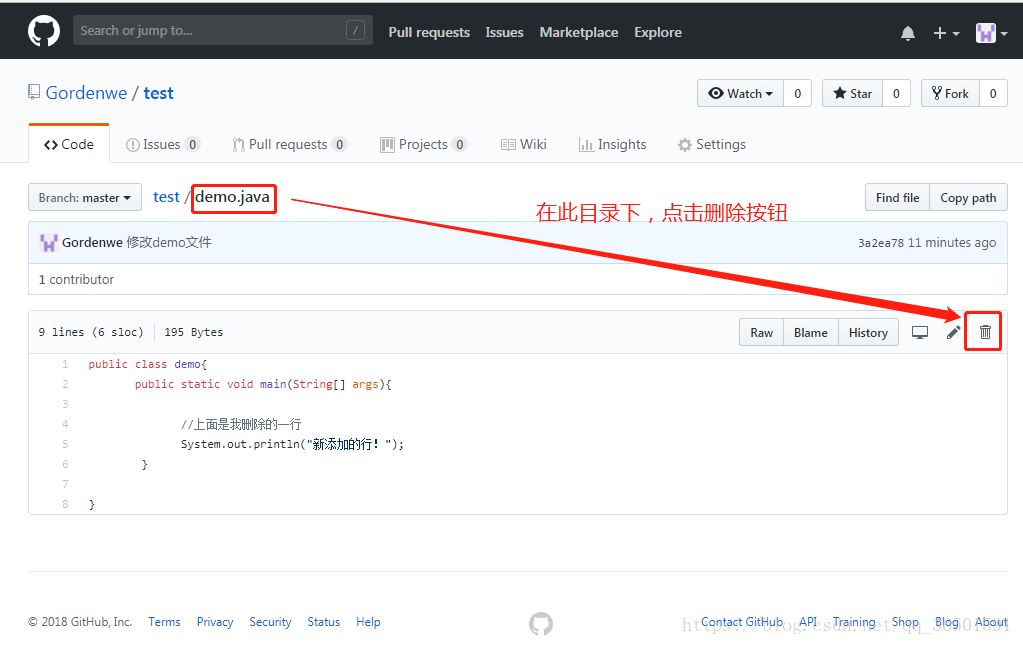


* 查看修改记录

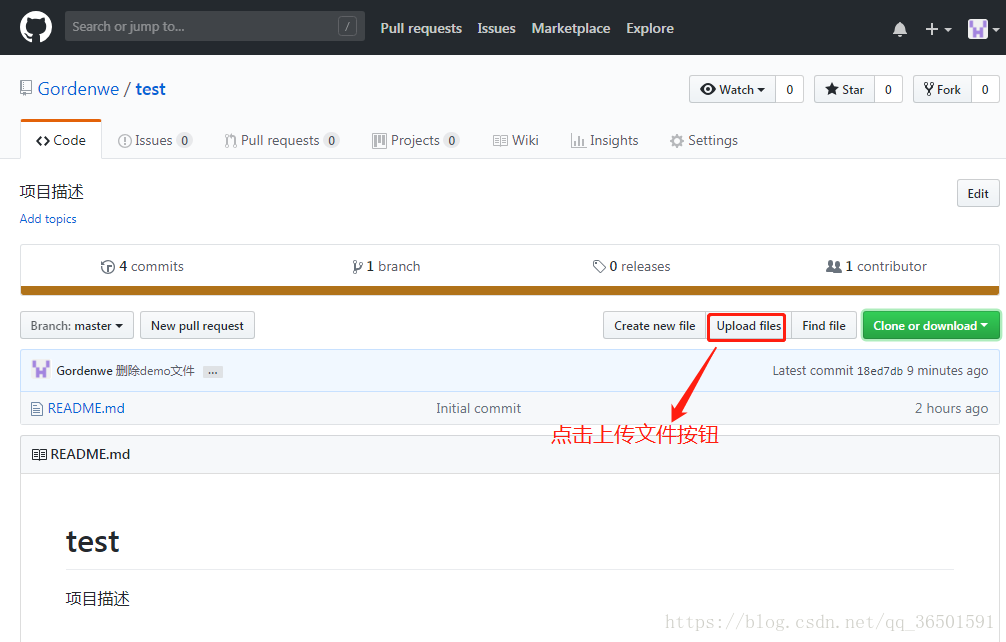


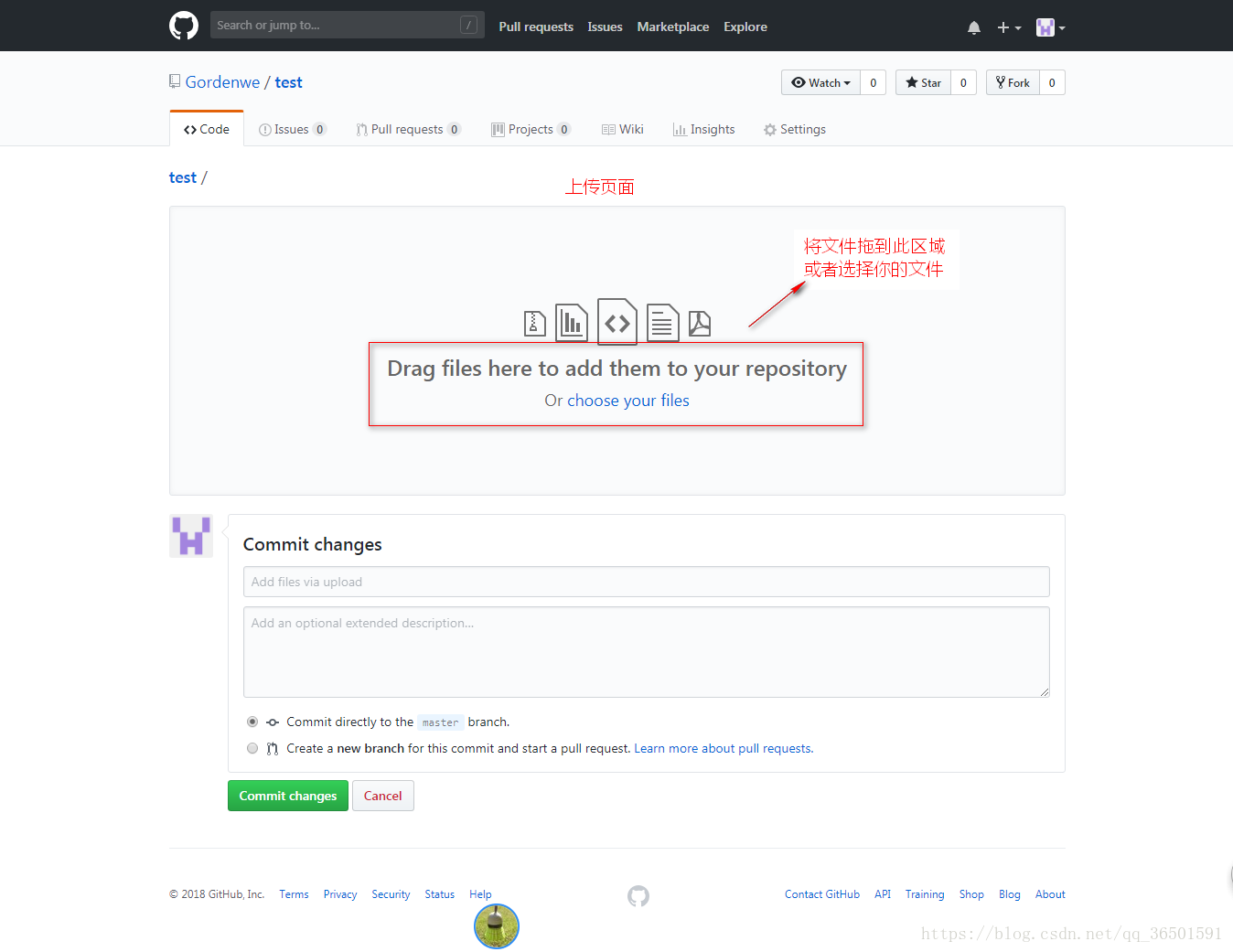
## 删除文件





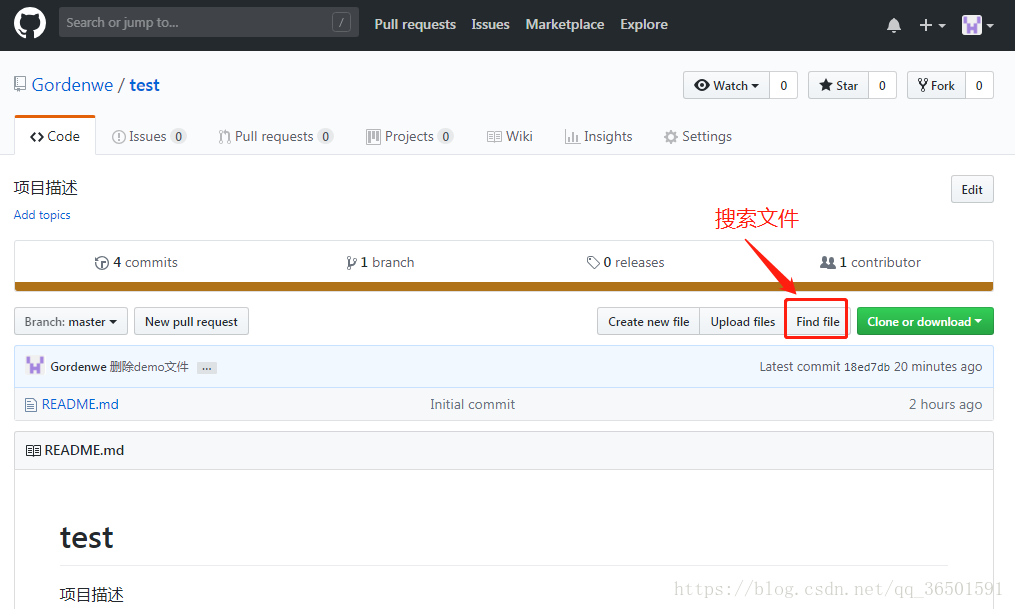
## 上传文件





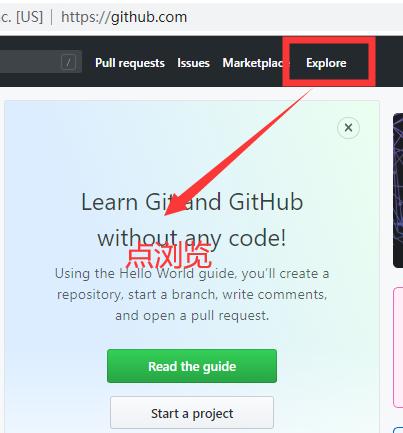
## 搜索仓库文件

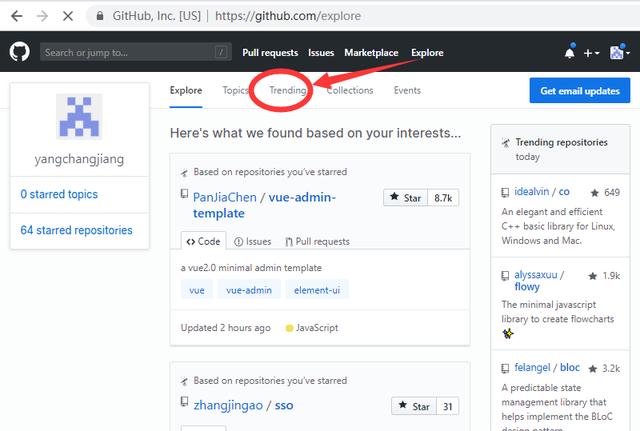
* 快捷键t



# 查看网上资源

## 查看排行榜





## GitHub搜索用法

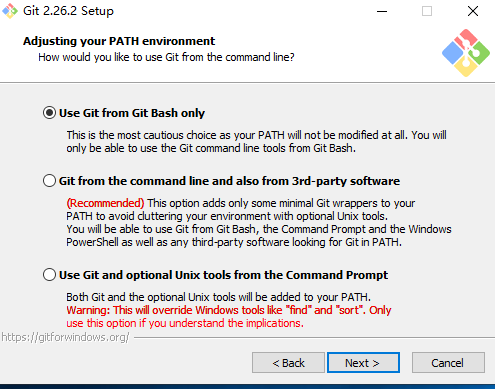
* 公式： XXX关键词 in:name 或 description 或 readme
* xxx关键词 stars 通配符 --:> 或者 :>=

|  |  |
| --- | --- |
| * dubbo in:name | 搜索项目名包含dubbo 的项目 |
| * springboot stars:>=5000 | 查找starts数大于5000的 springboot项目 |
| * springcloud forks:>500 | 查找forks数大于500的springcloud项目 |
| * name java stars:>500 forks:>200 |  |

# Git

## 下载和安装

<https://www.git-scm.com/download/win>



* 安装成功



## Git的初始化

在桌面某个文件夹下操作。

* 初始化用户名和邮箱——该设置在github仓库主页显示谁提交了文件（只需配置一次）

|  |
| --- |
| * 设置用户名：   git config --global user.name ‘chongpenggit’ |
| * 设置用户邮箱：   git config --global user.email ‘746861631@qq.com’ |
| * git config --list |

## 下载

|  |
| --- |
| * git clone https://github.com/chongpenggit/00-git |

## 推送项目

### 基本概念

|  |
| --- |
| * 工作区：就是你在电脑里能看到的目录。 * 暂存区：英文叫stage, 或index。一般存放在 ".git目录下" 下的index文件（.git/index）中，所以我们把暂存区有时也叫作索引（index）。 * 版本库：工作区有一个隐藏目录.git，这个不算工作区，而是Git的版本库。 |



### 推送过程

* 初始化一个仓库

|  |
| --- |
| * 在当前文件夹中创建一个文件夹 （mkdir test） |
| * 进入这个文件夹（cd test） |
| * 初始化一个仓库（git init） |

* 向仓库添加文件

|  |
| --- |
| $ git status |
| **还在工作区:**  (use "git add <file>..." to include in what will be committed)  huang.docx |
| $ git add huang.docx[[1]](#footnote-1) #添加到暂存区 |
| $ git status |
| **进入暂存区:**  (use "git restore --staged <file>..." to unstage)  new file: huang.docx |
| $ git commit -m "add huang.docx" |
| [master 1de7e8c] first commit  1 file changed, 0 insertions(+), 0 deletions(-)  create mode 100644 huang.docx |
| $ git status |
| Your branch is ahead of 'origin/master' by 1 commit.当前分支比远程多了一个commit |
| $ git push |
|  |

# 设置个人站点

## 搭建步骤：

* 创建—>新建仓库（仓库名必须是：用户名.github.io）

1. git add . 为添加全部 [↑](#footnote-ref-1)