# 什么是GIT

# 分布式版本控制

# 多个开发人员协调工作

# 有效监听谁做的修改

# 本地及远程操作

# Basic Commands

# git init // 初始化本地git仓库

# git add <file> // 添加文件

# git add \*.html // 添加所有html文件

# git add . // 添加所有文件

# git rm --cached <file> // 删除文件

# git status // 查看状态

# git commit // 提交

# Esc :wq // 退出附文编辑

# git commit -m ‘ ’ // 提交并编辑附文

# git push // 推送到远程仓库

# git pull // 从远程仓库拉取数据

# git clone // 从远程仓库拷贝数据

# git --version // 查看安装的git的版本

# git config --global user.name ‘ ’ // 配置 全局配置 用户名

# git config --global user.email ‘ ’ // 配置 全局配置 邮箱

# GIT的安装方法

# MAC

<http://git-scm.com/download/mac>

# Windows

<http://git-scm.com/download/win>

# Others

进官网找

# 想要忽略某个/些文件

# 创建一个.gitignore文件( 如果没有gitignore文件的话 )

# 在gitignore文件里写入需要忽略的某个.些文件

# someTxt.txt // 某个txt类型文件

# /someDir // 某个文件夹

# 分支的使用和作用

# git branch dyc // 创建一个名为dyc的分支

# git checkout dyc // 切换到一个名为dyc的分支

# git chechout master // 切换到主线

# 在分支中操作的内容不会影响到主线

# git merge dyc // 主线和分支合并( 需要在主线下操作 )

# GitHub的使用

# 点击Start Project或者New repository

# 给个Repository name就可以Creating repository

# git remote // 查看当前的remote



# 如果没有仓库 使用上面的代码

# 如果已经有了仓库 使用下面的代码

# 什么是github

# Git是代码管理工具

# Github是基于Git实现的代码管理平台

# Github的基本使用流程

# 在Github官网创建账号

# 创建远程仓库( 用来存储代码 )

# 安装Github客户端

# 登录客户端并克隆仓库

# 在本地仓库中存储数据

# 提交数据并备注信息

# 同步本地数据到远程仓库

# 安装Github

# Mac

<https://desktop.github.com>

# Windows

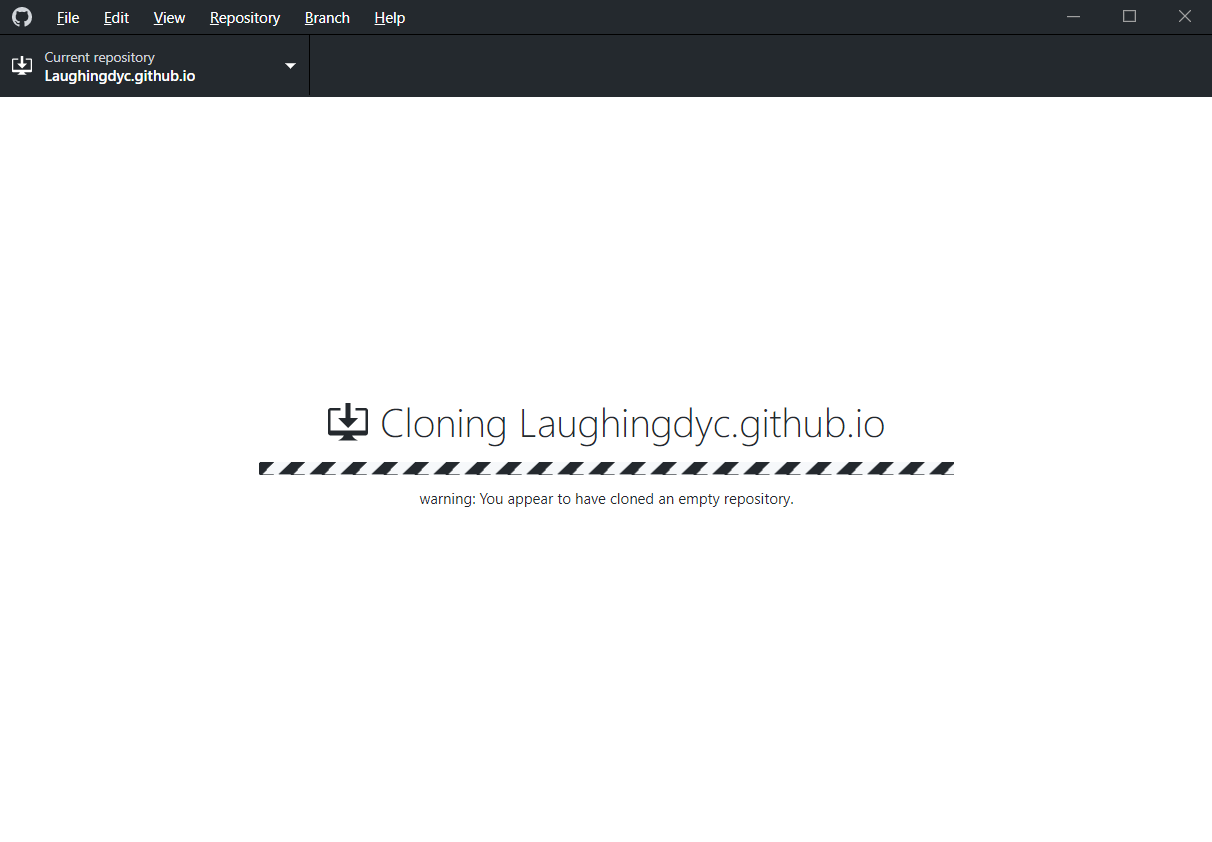
<https://desktop.github.com>

# Laughingdyc.github.io

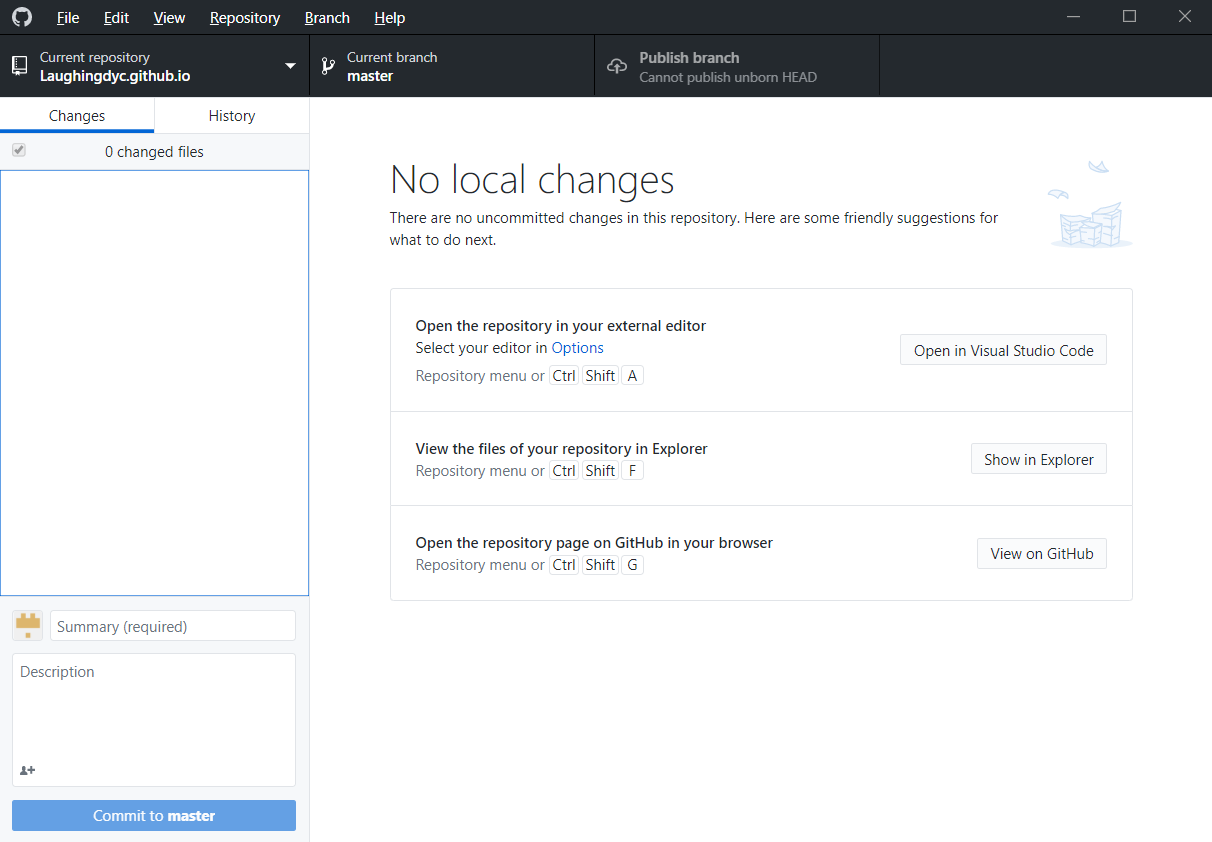
# 创建一个Laughingdyc的repository

# 可以使用它作为服务器,可以把作品展示到服务器里

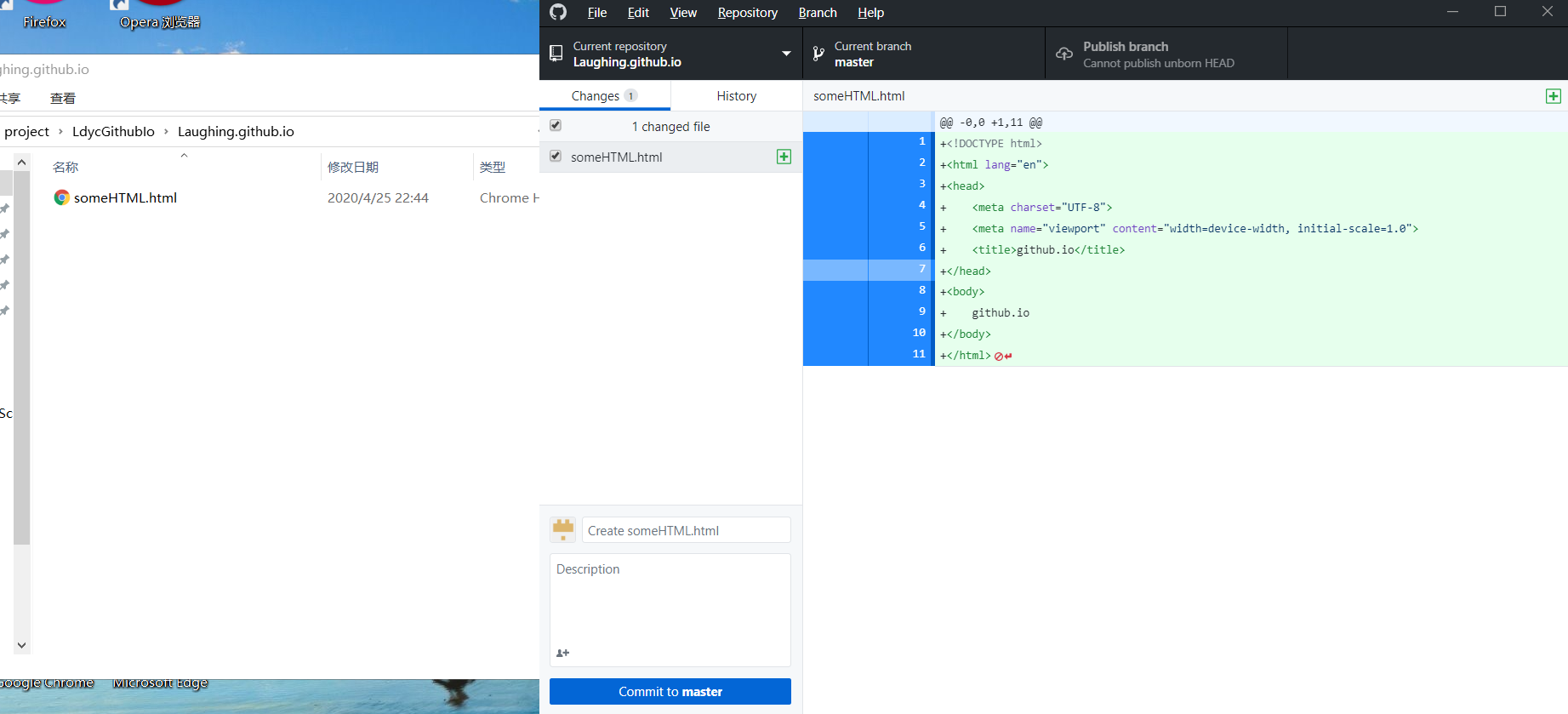
# 创建完之后,点开客户端



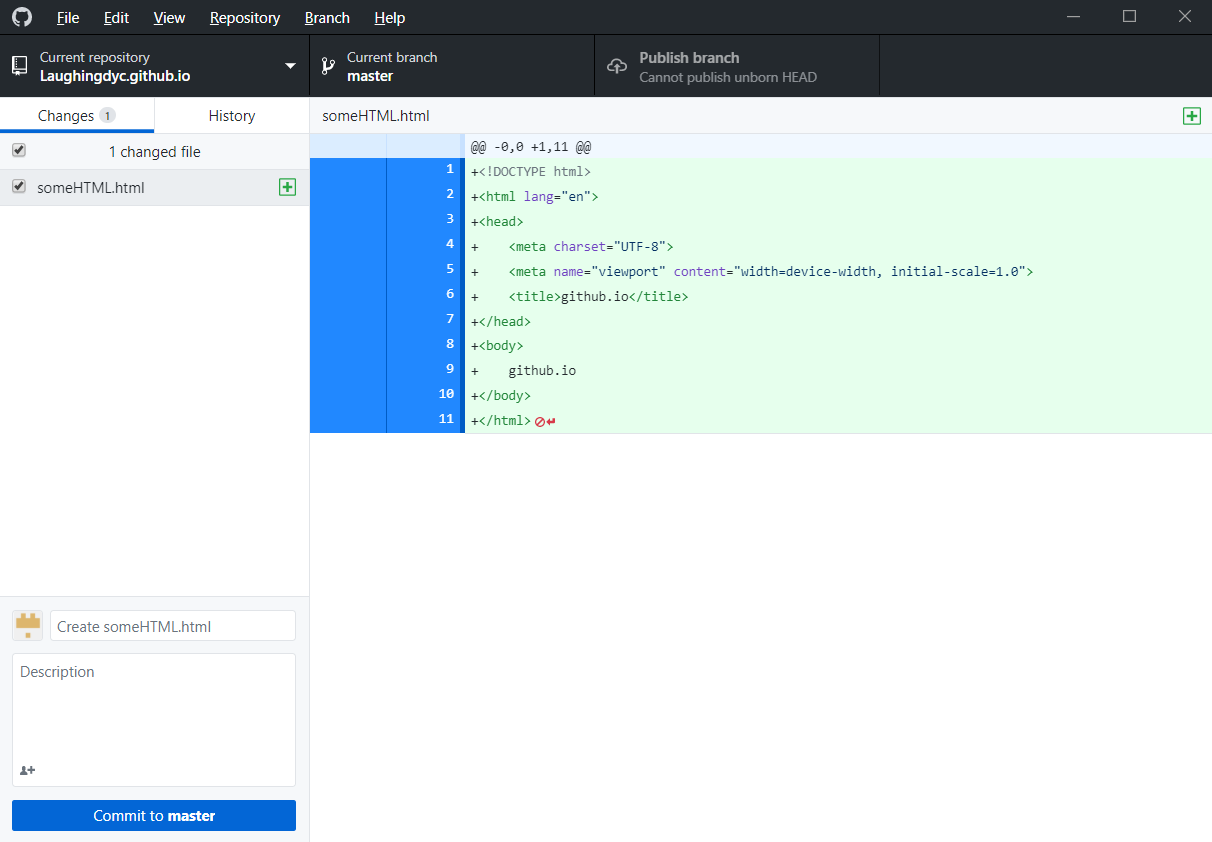
# 完成clone



# 当你在该文件夹里写入什么东西的时候,github客户端回立马显示出来

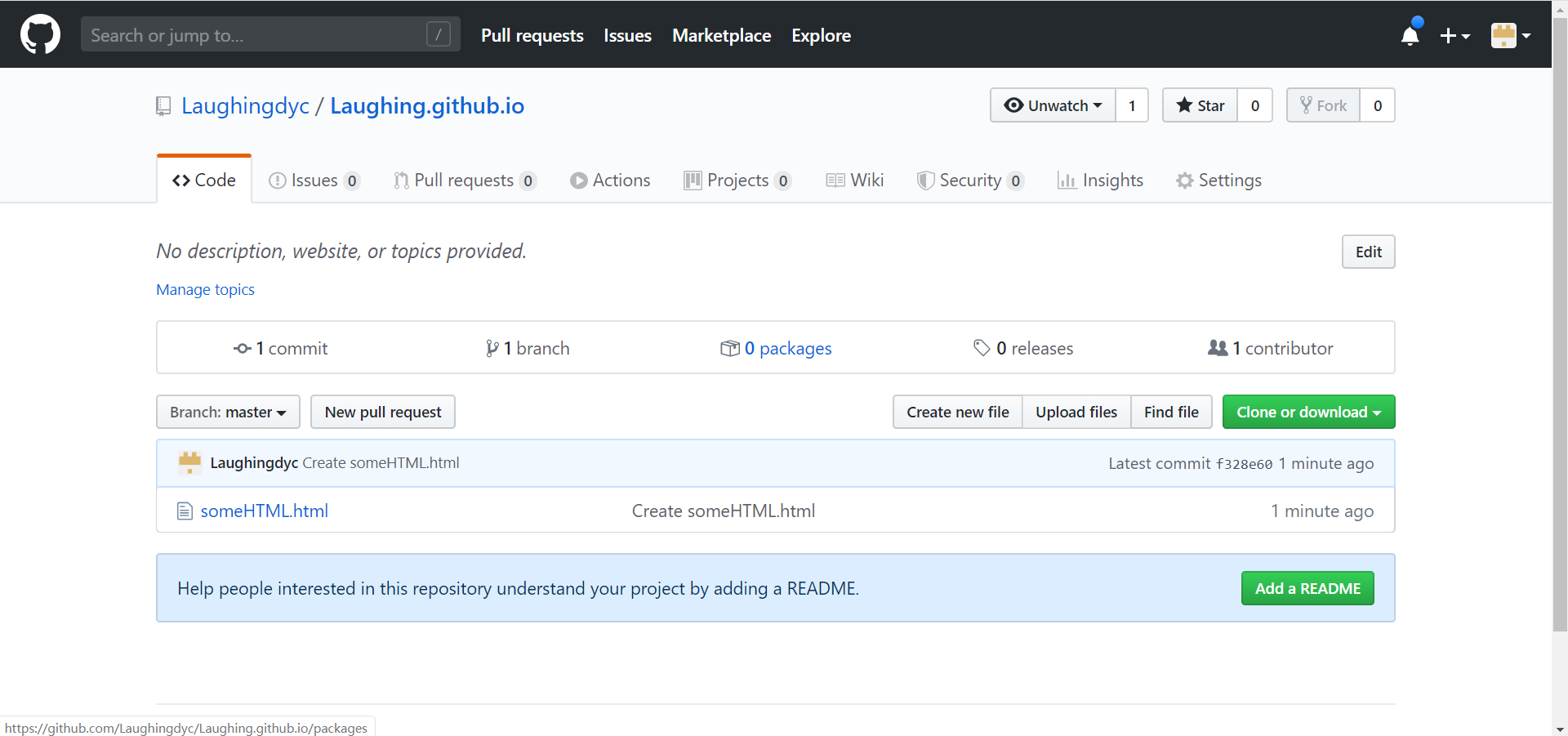


# 在框内写入改写入的摘要和描述,然后提交



# 然后publish,只有这个仓库才会有publish

# 然后该仓库的网页就会显示这个someHTML



# Laughingdyc.github.io的特别之处就是