### gitHub学习之windows客户端安装使用总结（基础篇）

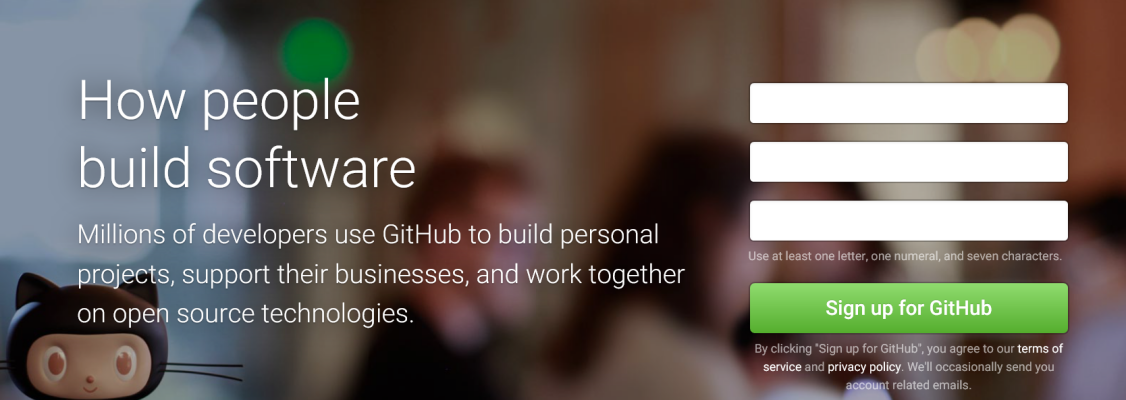
**一直想通过github来管理自己平时学习总结的项目，由于各种原因至今才开始网上查询各种资料学习，最后还是选择了基于window系统的github for window客户端工具（mac系统这里不做说明），而且目前网上教程比较落后，主要是这个软件更新了多个版本界面好多东西跟以前不一样，所以写这个教程，希望帮助跟我一样初学github的同学，希望与大家共勉。**

在这主要分为以下几个步骤：

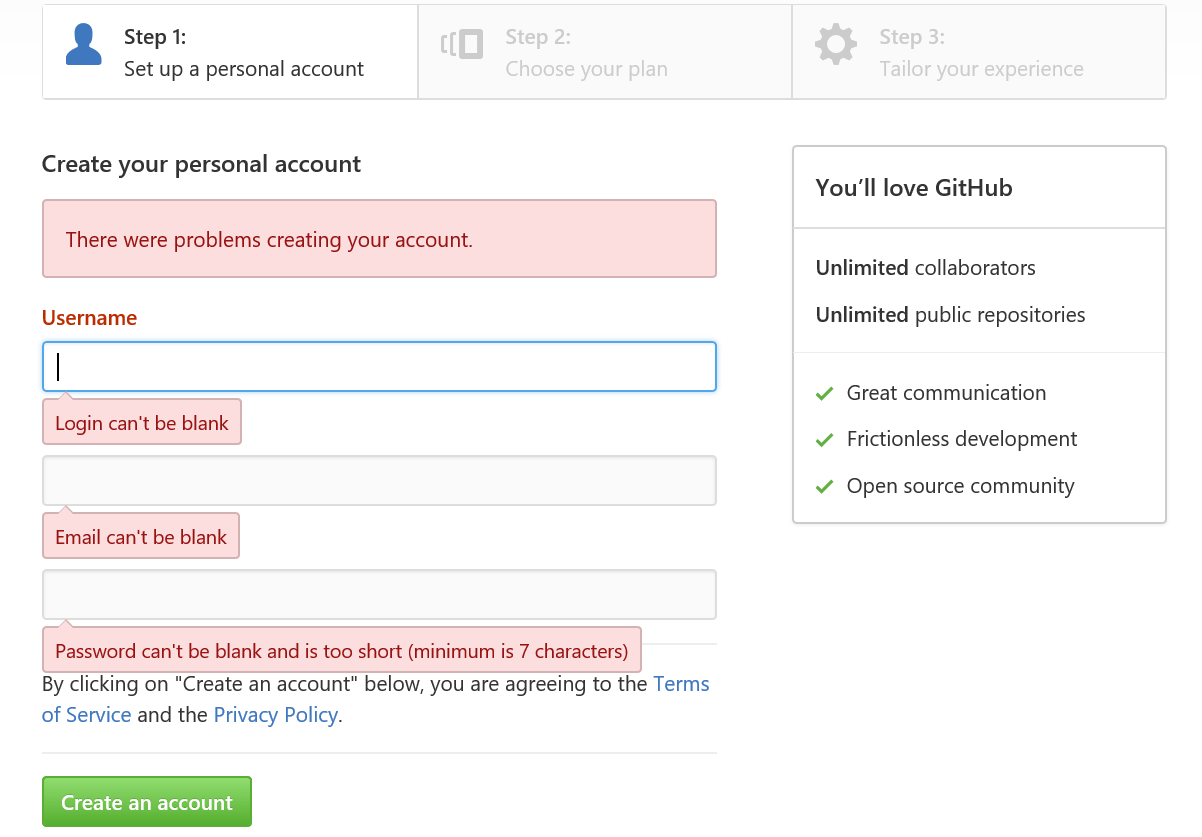
1. **注册github账号**
2. **网页上新建仓库**
3. **安装客户端（v3.3.2版本）**
4. **客户端的使用**
5. **删除仓库**

以下内容对以上几个步骤进行具体说明。有不足之处还望大家谅解。

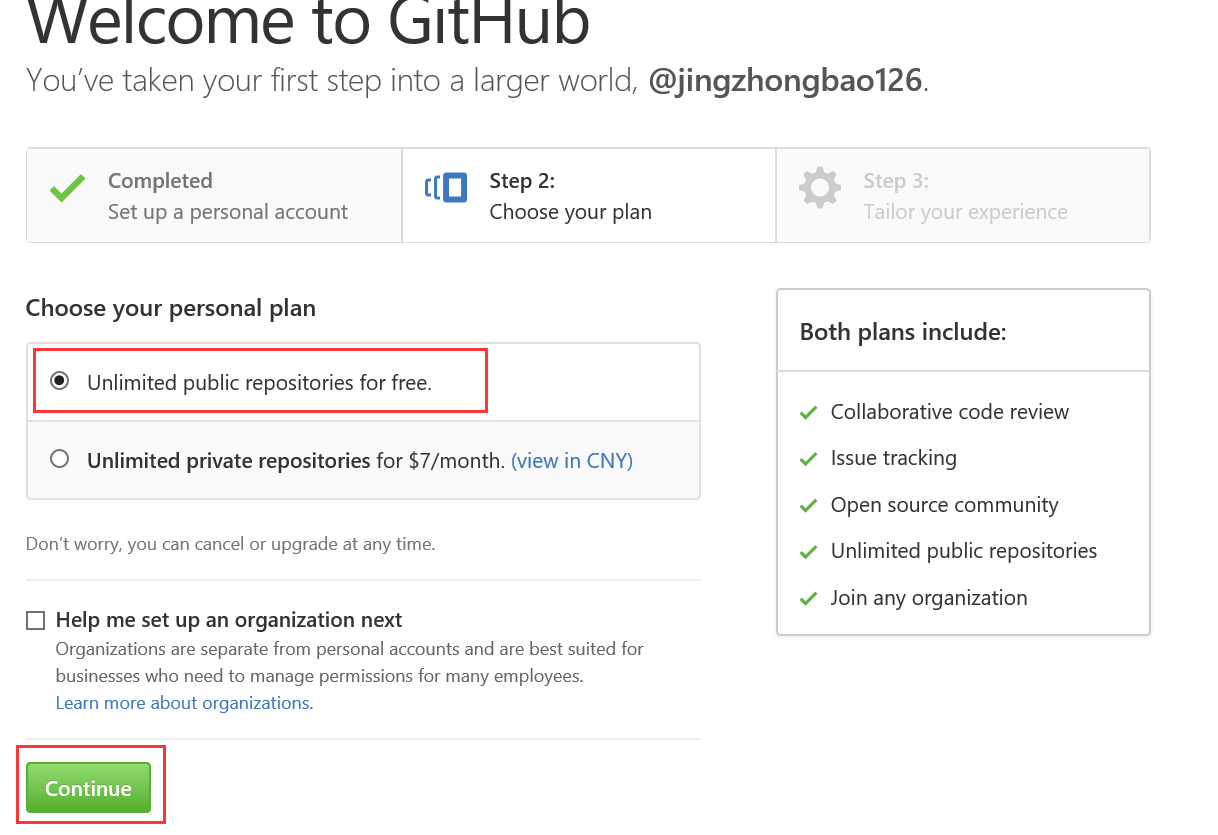
1. 注册github账号
2. 进入github官网（<https://github.com/>）如下图：



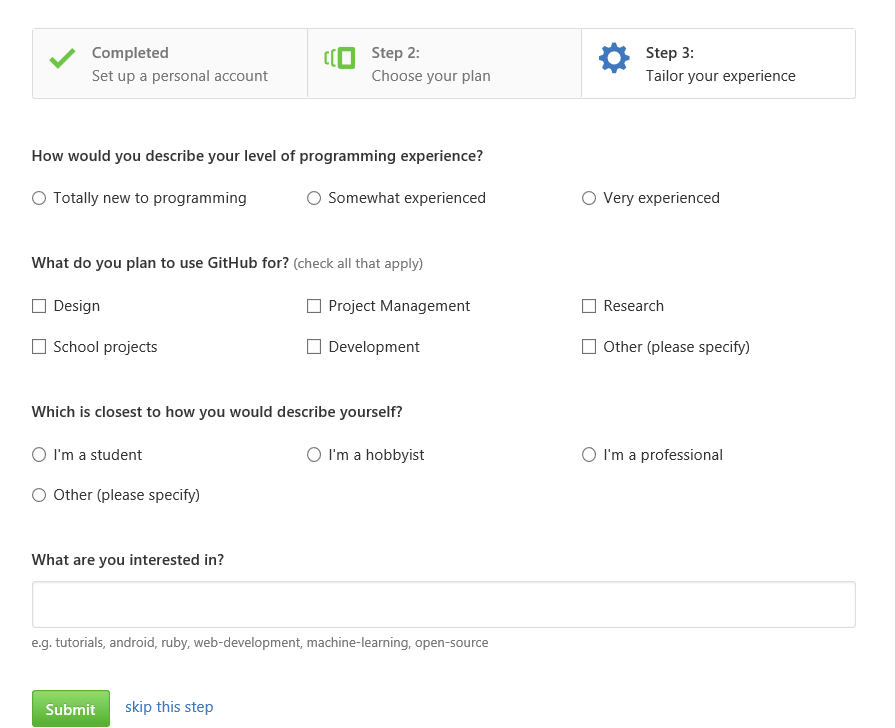
1. 点击绿色按钮sign up for github进入注册页面，注册很简单只需输入用户名密码邮箱即可，如下图：



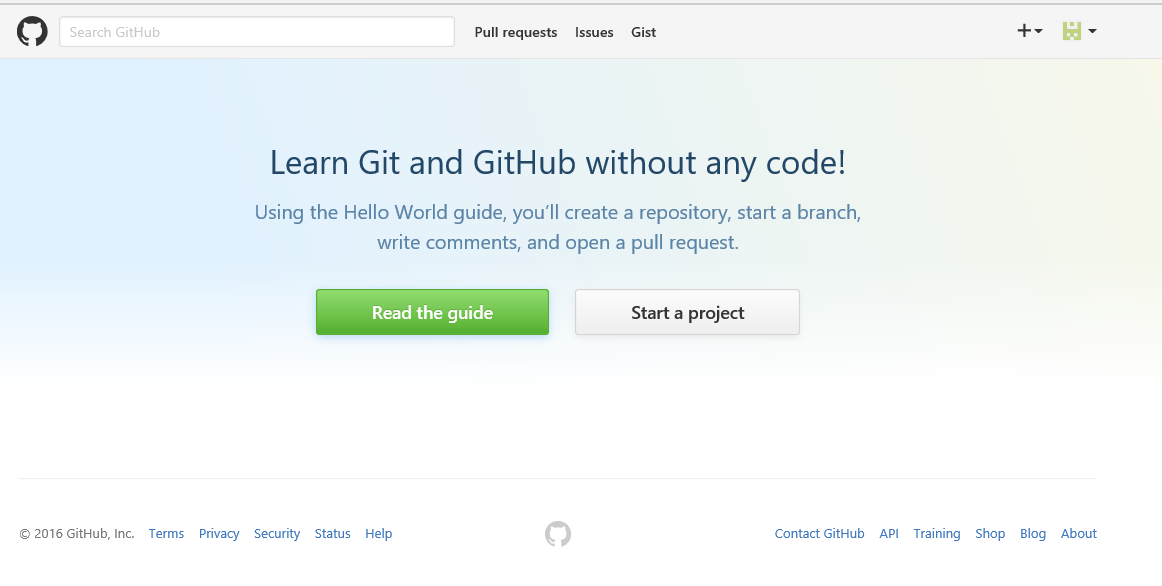
1. 输入完成后create进入下一步，默认选项不用管，直接下一步



4、不用管这些偏好设置直接提交，至此账户就创建好啦

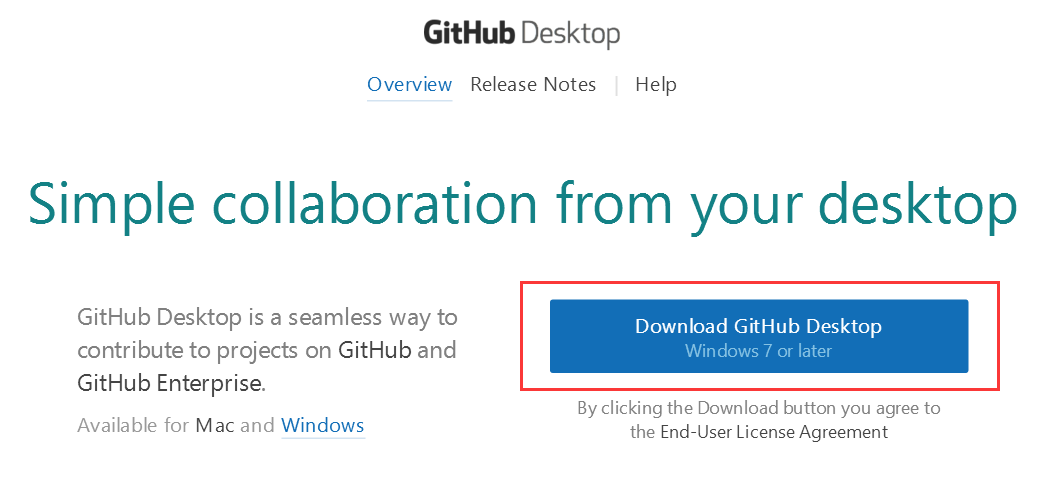


5、成功后展示的页面，是不是so easy。



1. **安装客户端（v3.3.2版本）**
2. **下载**

**在官网可以下载：<https://desktop.github.com/>**



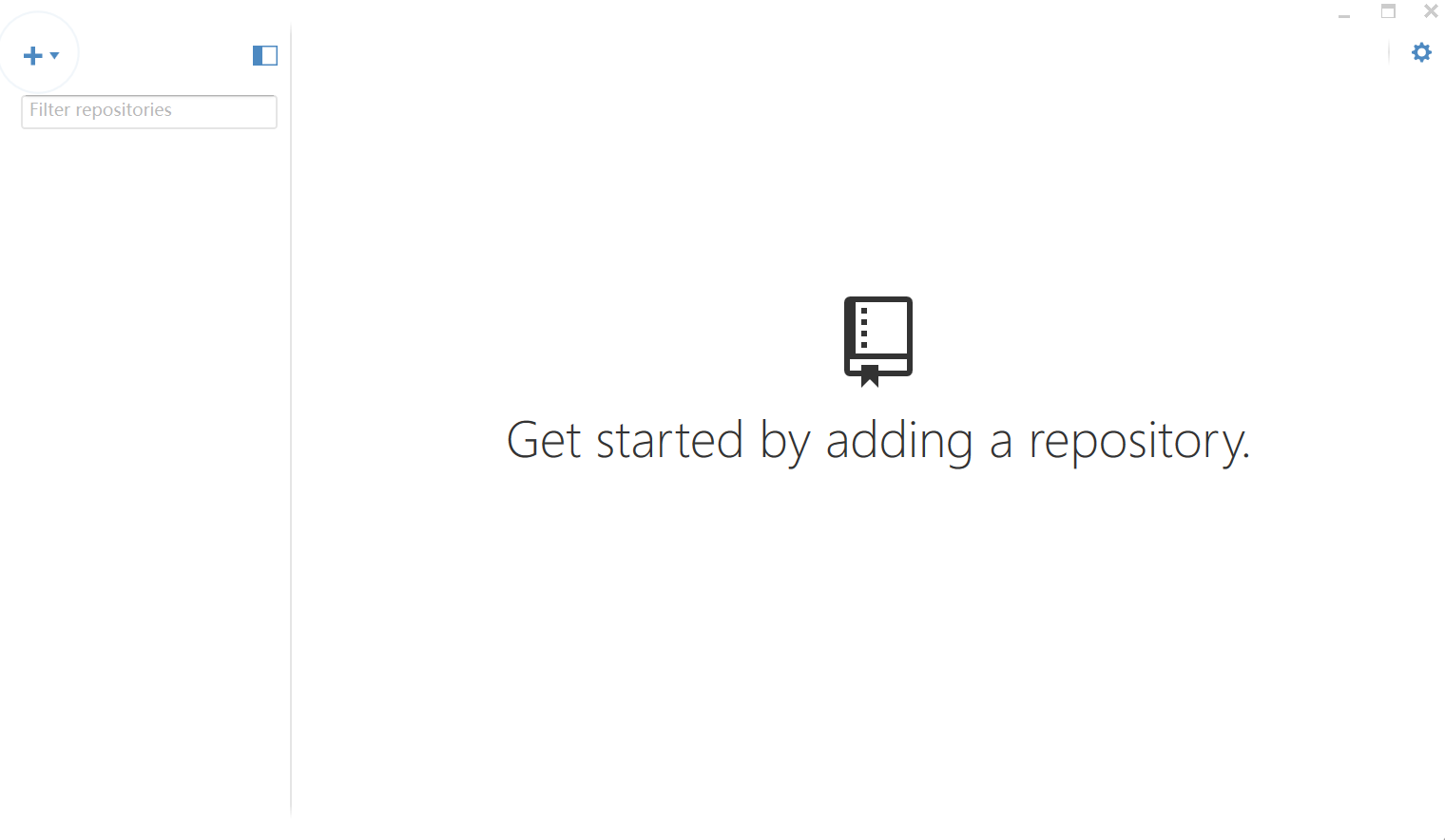
1. **安装**

**安装过程就不在累赘了，安装完成后桌面有二个图标**



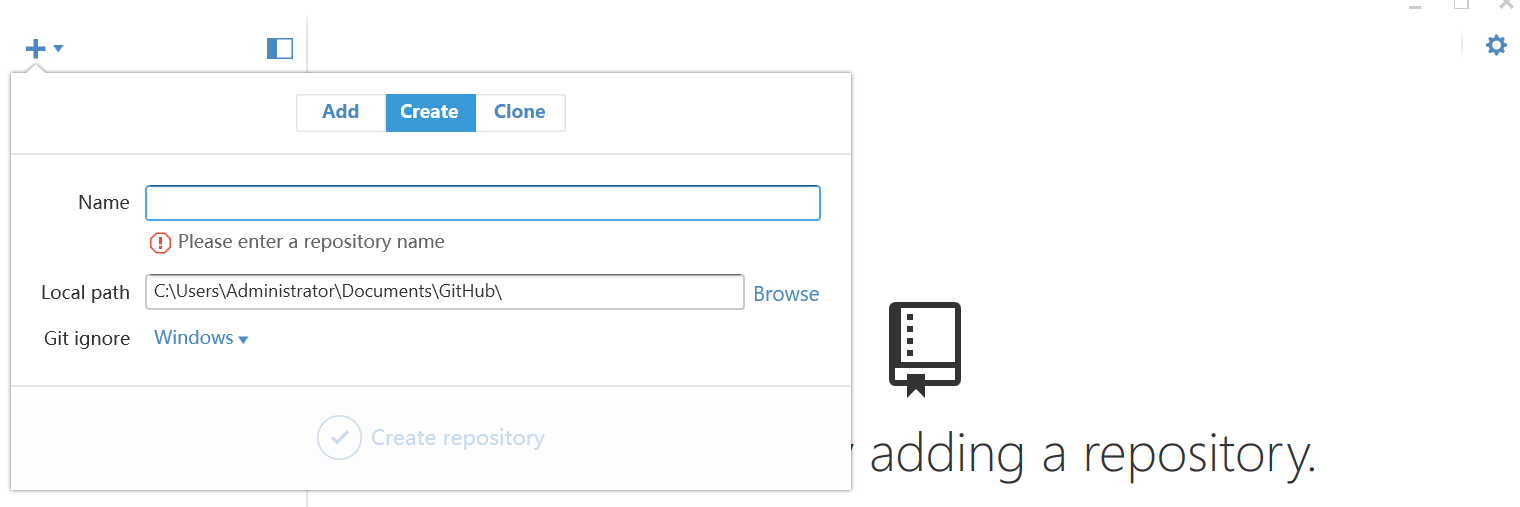
1. **页面介绍**

安装好后打开 用你的GitHub账号登陆。(需要注意的是，Github for Windows会帮你创建SSH Key，你打开邮件应该会收到邮件帮你创建SSH Key的信息，这也省了不少事)   
整体的界面如下：



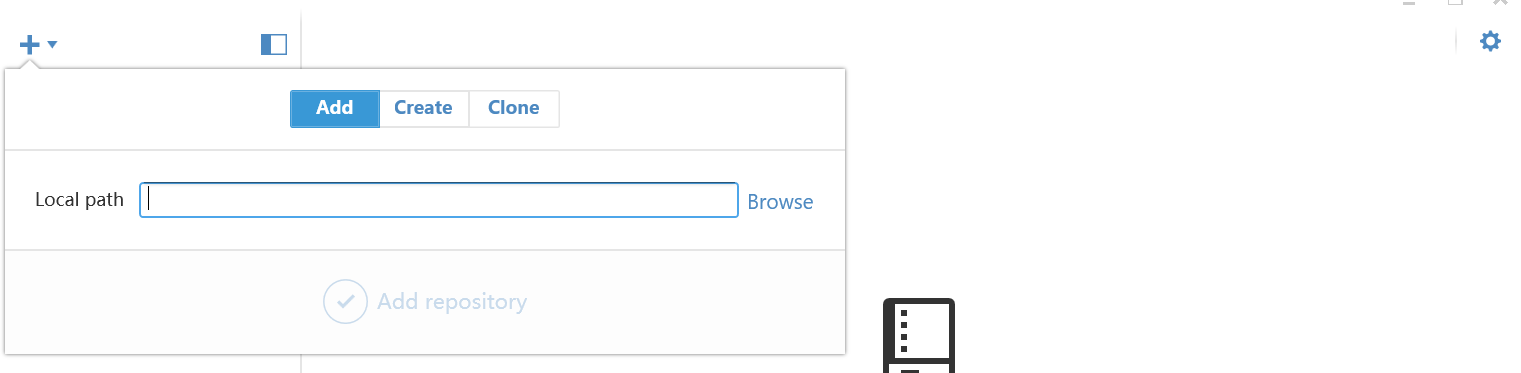
github for windows的界面非常清爽

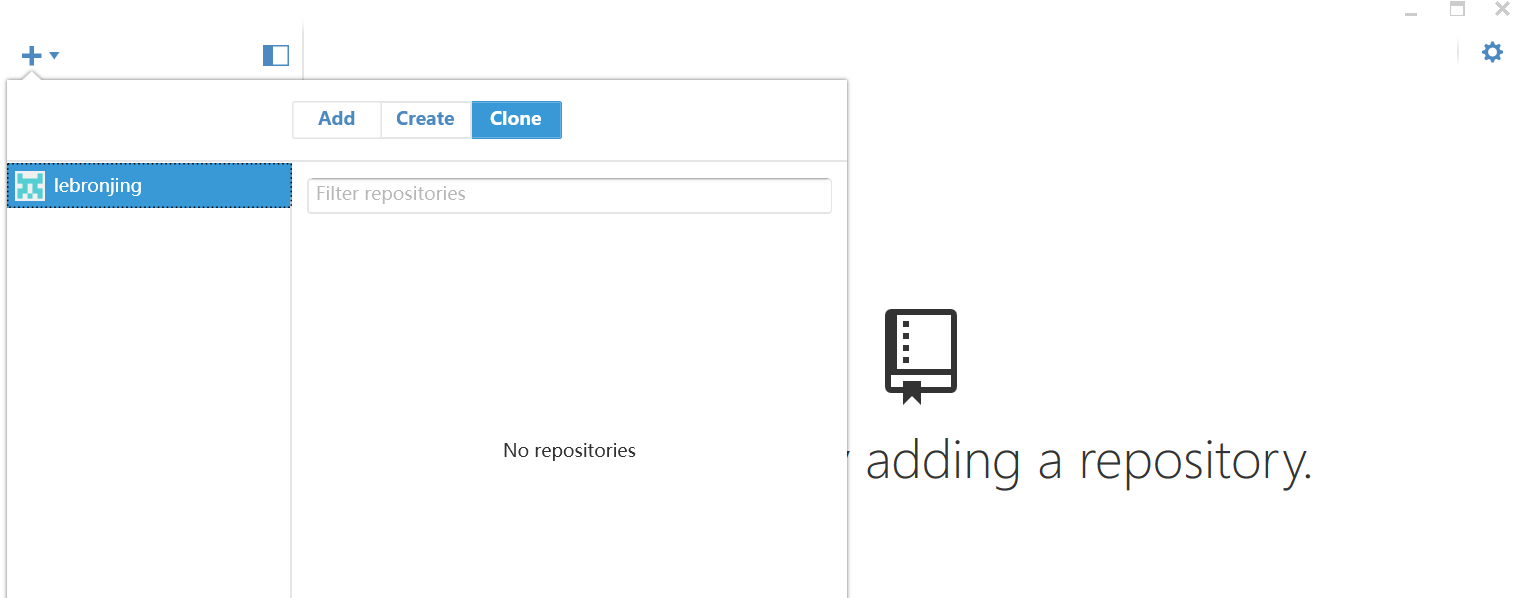
1. **客户端的使用**
2. 我们来建立第一个仓库，点击左上角+号，打开之后有三个选项，Add,Create,Clone



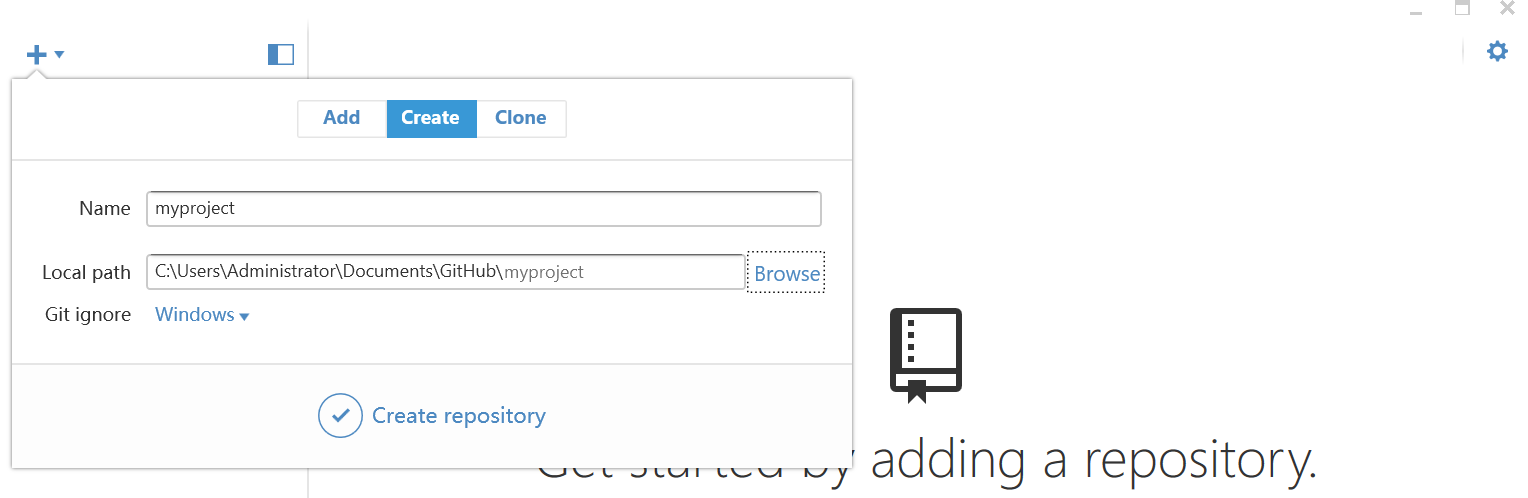
下面介绍一下这三个选项的功能：

Add:如果本地有工程，就可以使用Add添加

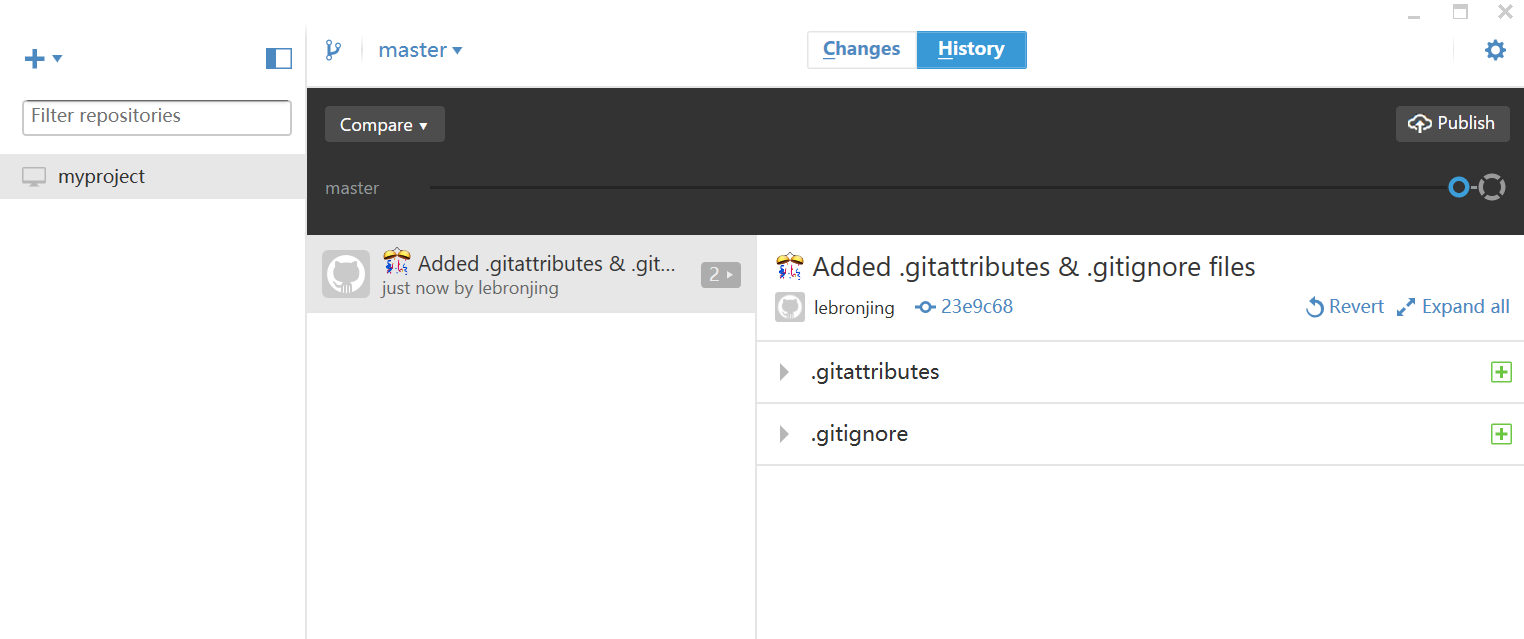


Clone：这个功能其实最好理解了，克隆这名字通俗易懂好理解。 就是将在浏览器上已经创建好的项目导入到本地，换句话说就是下载到本地。  


Create:创建一个代码库,Name填写你的仓库名称Local path写你仓库在本地的存放地址。我们主要从这个功能开始我们的github之旅



2、我们在这里填写myproject，选择本地存放地址，点击Create repaoitory来创建第一个我们自己的repaoitory。 如下图：



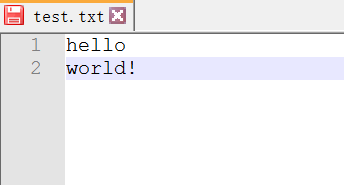
1. 开始使用第一个仓库

修改我们第一个代码库中的内容

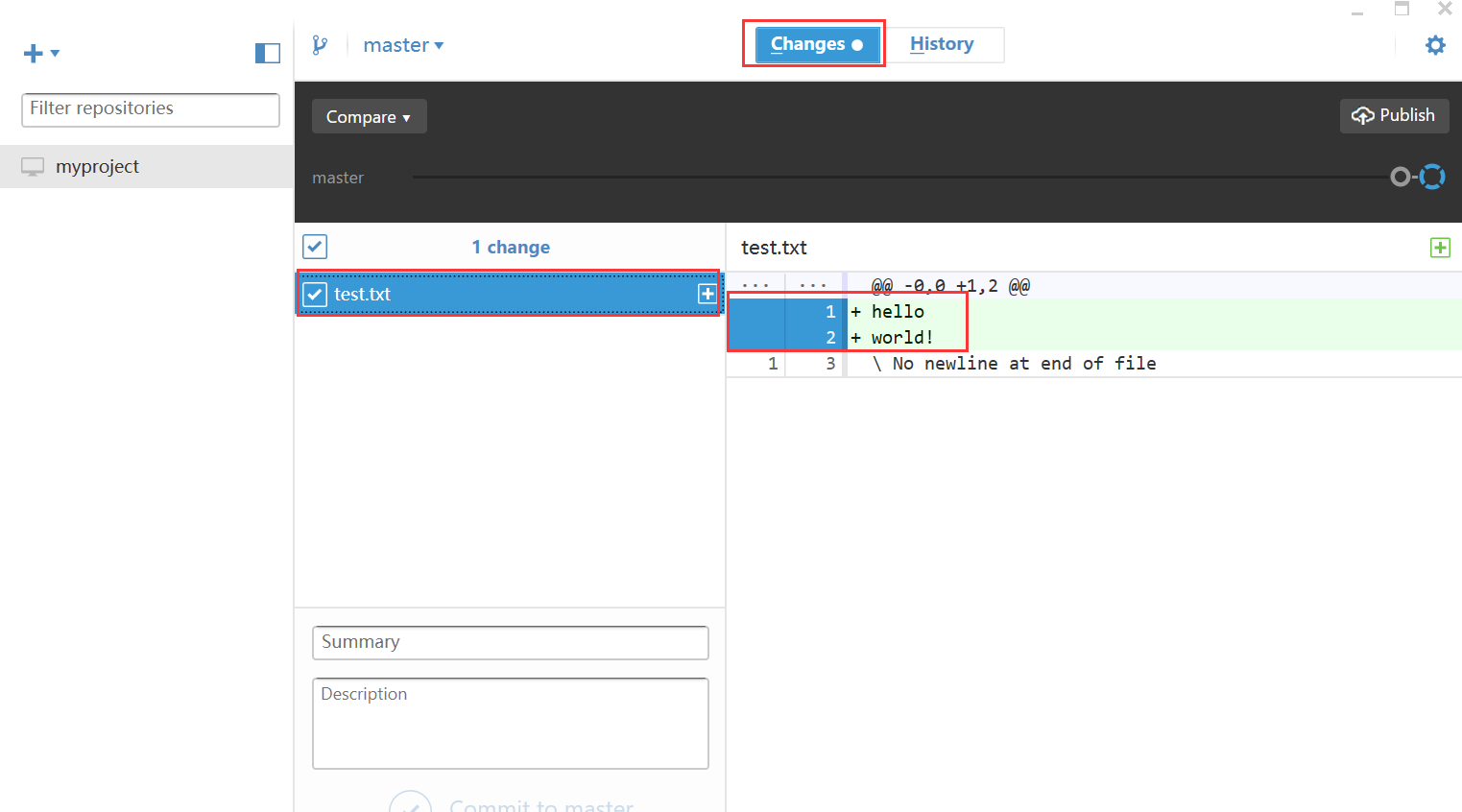
我们来找到刚刚创建的代码库在本地的位置。就是刚刚在local path的地址路径，当然如果你忘了，请右键点击myproject，选择Open in Explorer进入我们的myproject目录下。

接下来我们在myproject目录下新建一个test.txt的文本文档用于测试及演示。

输入hello world!。如下图：



此时github就会变成如下图这个样子



你会发现此时github会出现刚刚编辑的内容。

hello

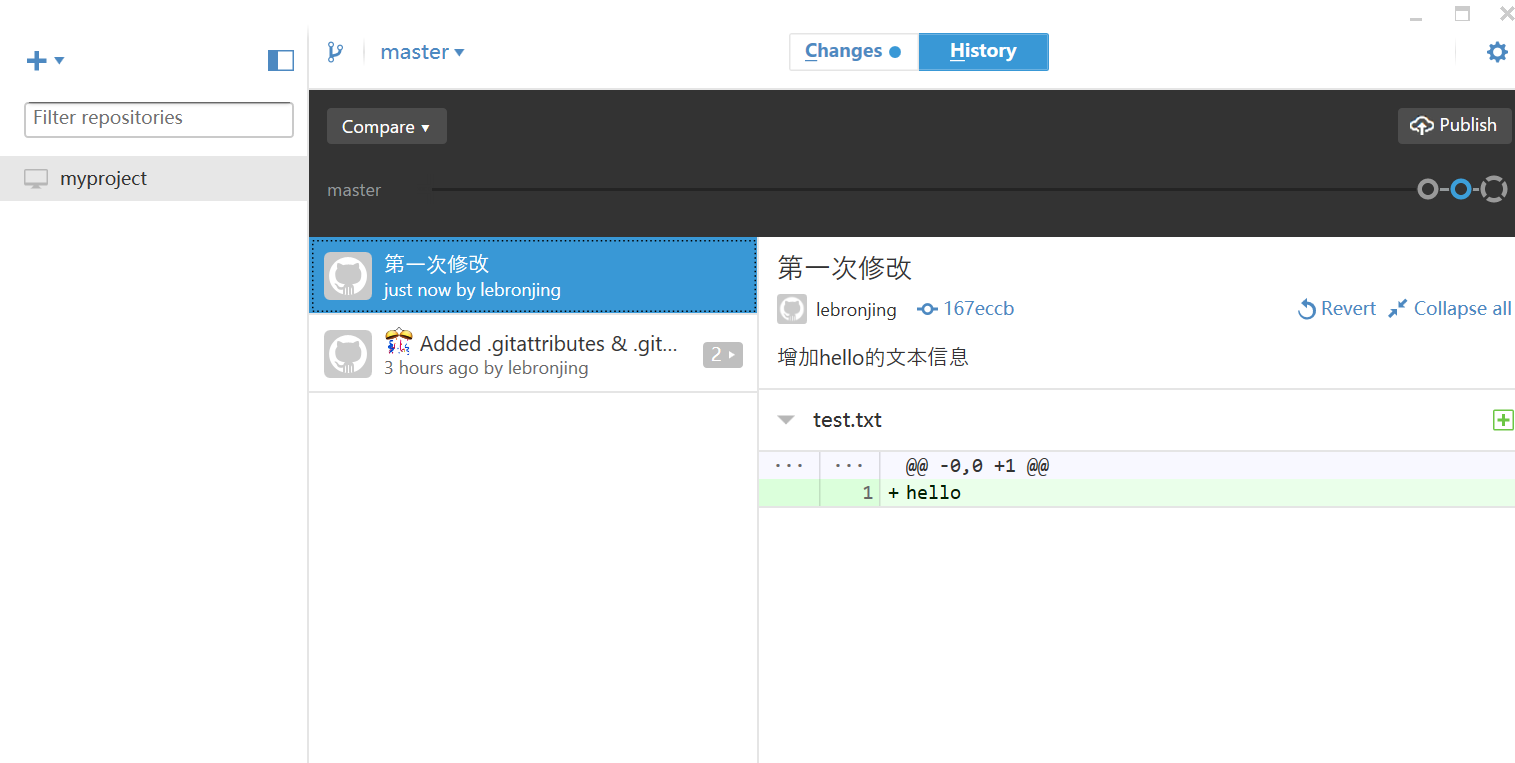
world!

并且前面会有蓝色标识，那么这个蓝色标识是什么用呢？   
其实这个蓝色标识是提示你改变了的文本。比如我第一次只想改变hello那我们就把world!前面的蓝色标识单击去掉。

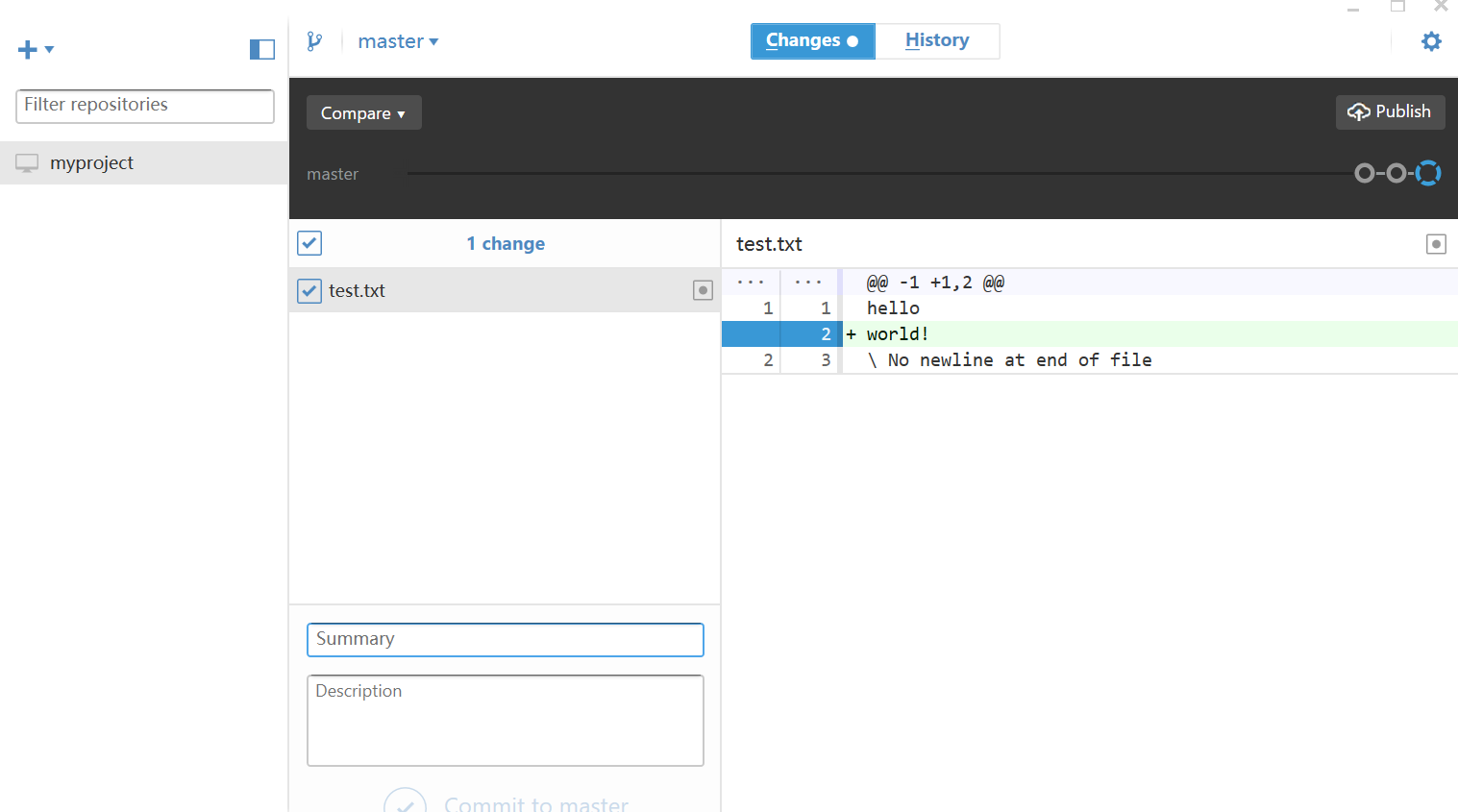
我们填写好Summer和Description   
Summer就是这次改动的总结，我们也可以理解为标题（必填），而Description可以理解为详细概况（选填）

我们这里只选择第一个修改对象，也就是hello。summer我们填写为第一次修改，Description我们填写为增加了hello的文本信息，之后点击Commit to master。

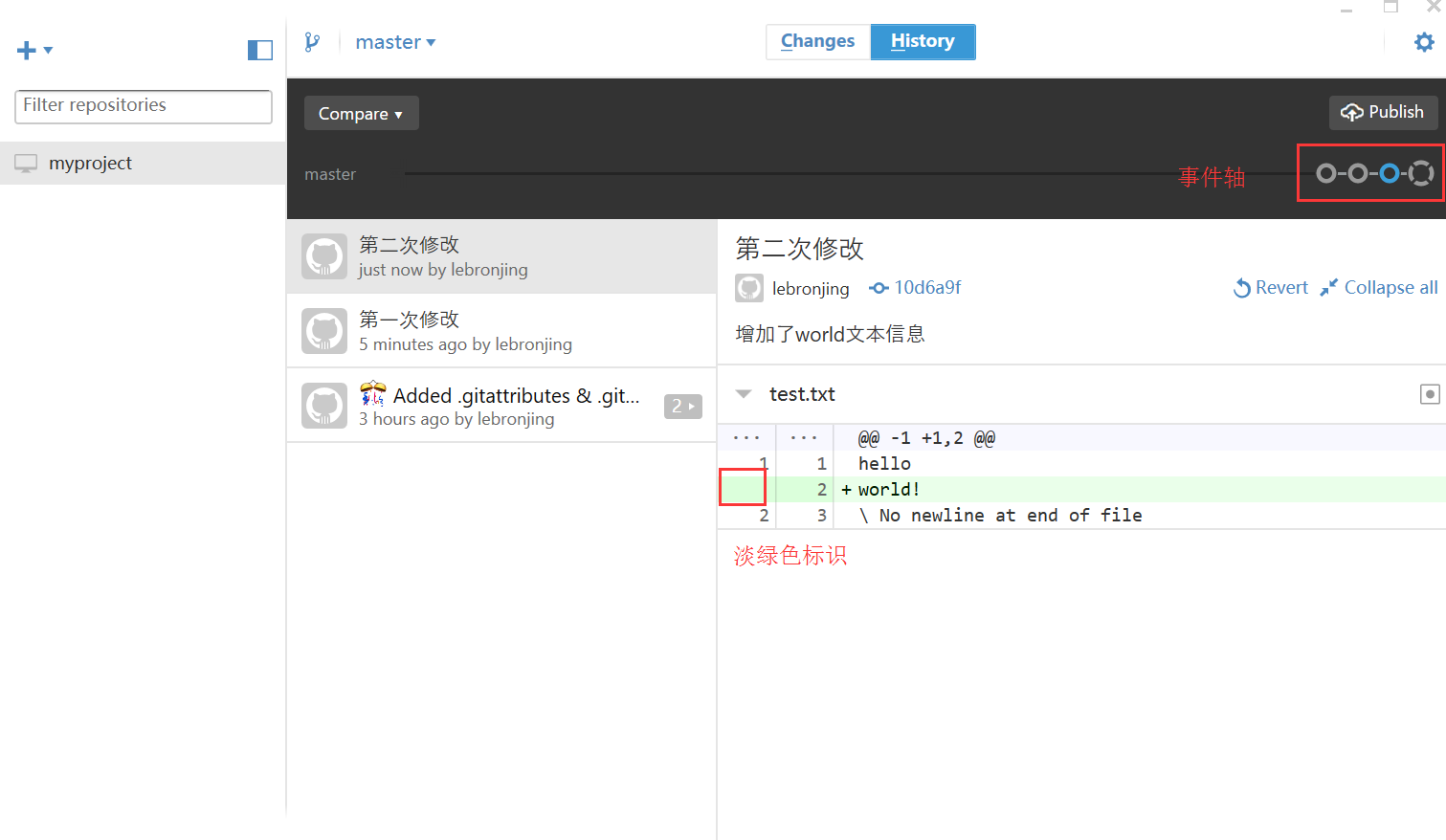
切换到History目录下



我们会发现他改变了。   
这次我们把world进行添加。点击changes,勾选test.txt，确保world前面有蓝色标识，在summer我们填写为第二次修改，Description我们填写为增加了world的文本信息，之后点击Commit to master。

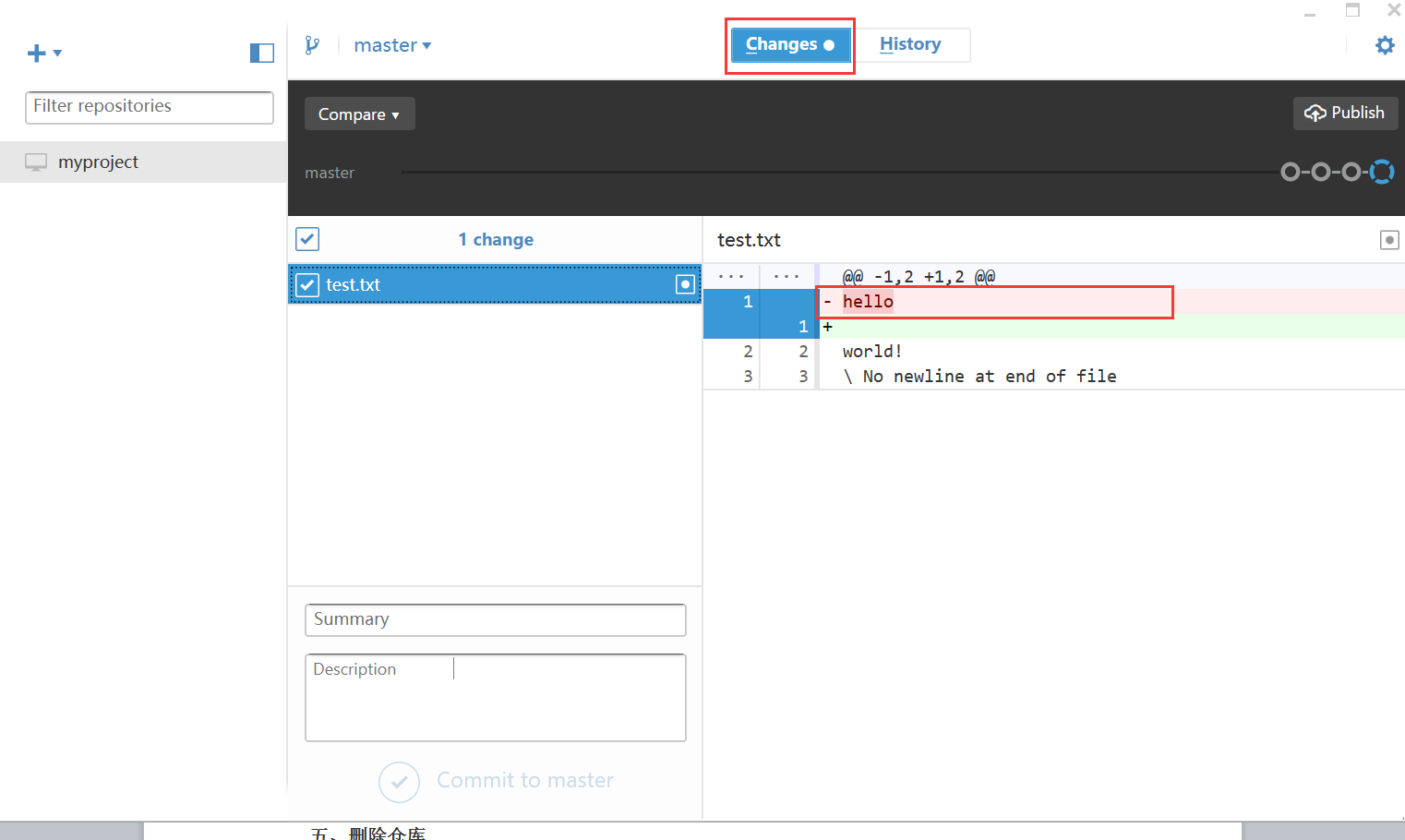


在History目录下发生了这样的改变。会在History目录下形成一条时间线，来指出每一次的修改标题和内容，同时会把修改的内容用绿色标识标出。



我们打开本地的文本，删除刚刚添加的第一行hello。 此时你就会发现github发生了变化。

此时的红色标识标识删除。我们写好Summer和Description并点击Commit to master。   
这样我们就删除了第一行。同时在History目录下又多了一条时间轴。

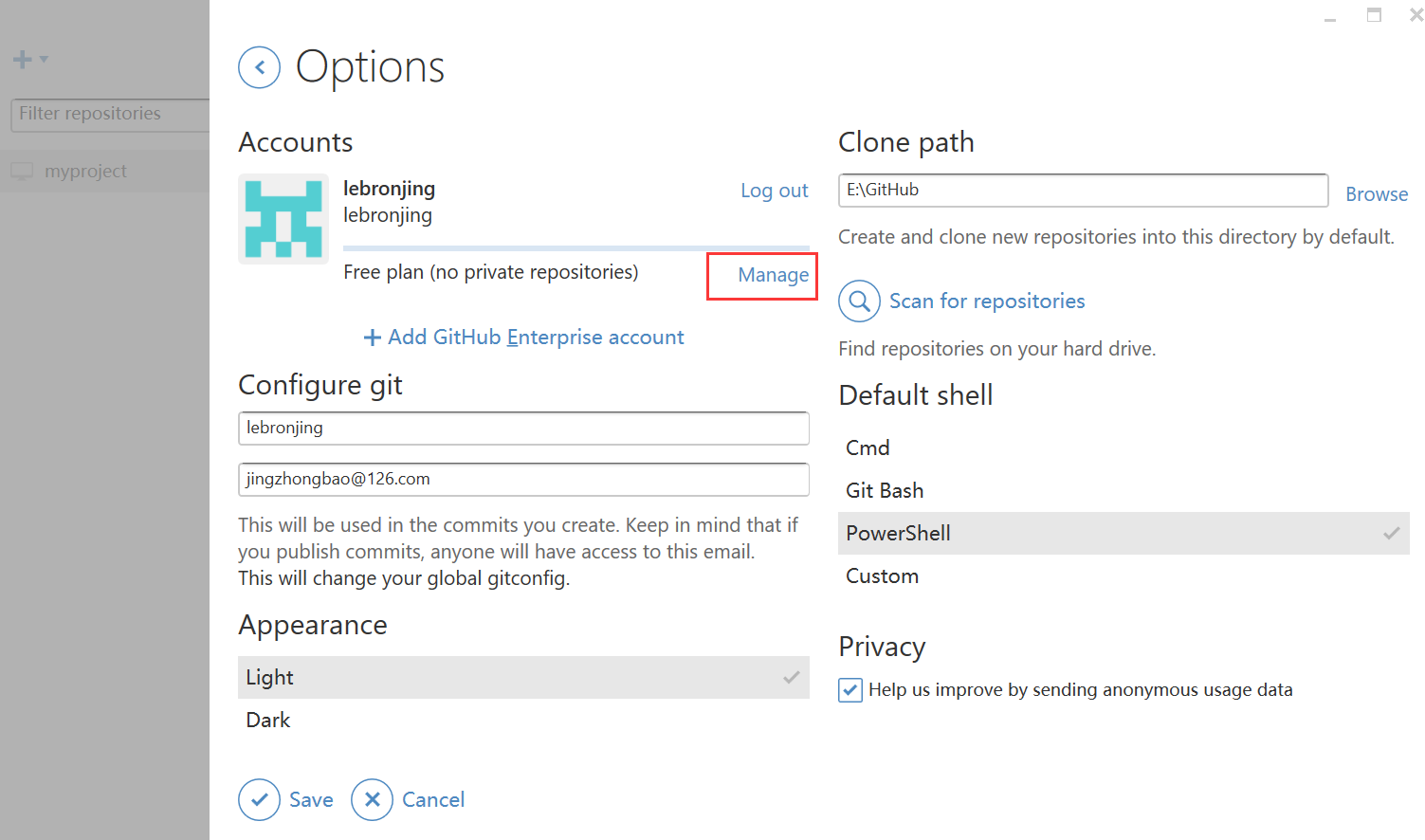


这样我们就完成了删除。

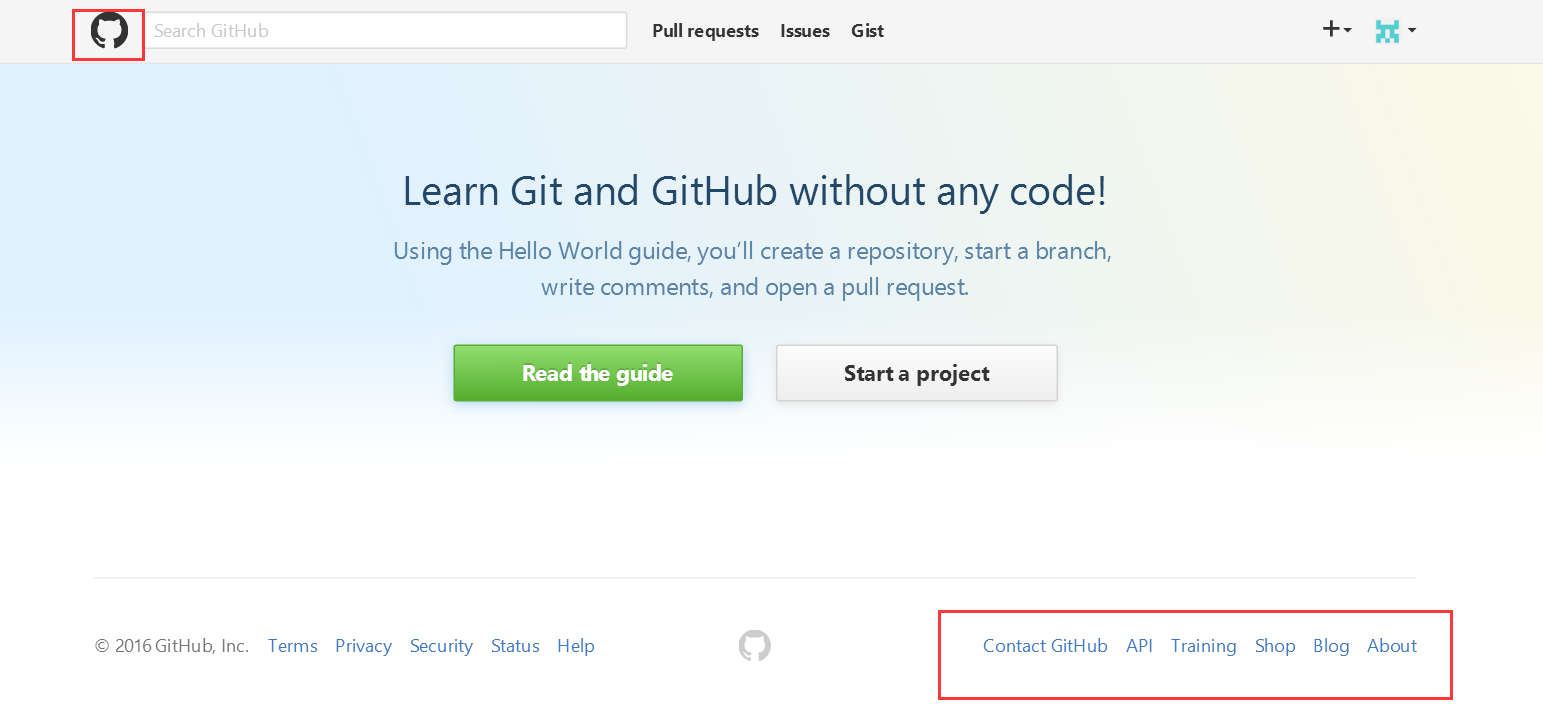
**上传与同步**

上传

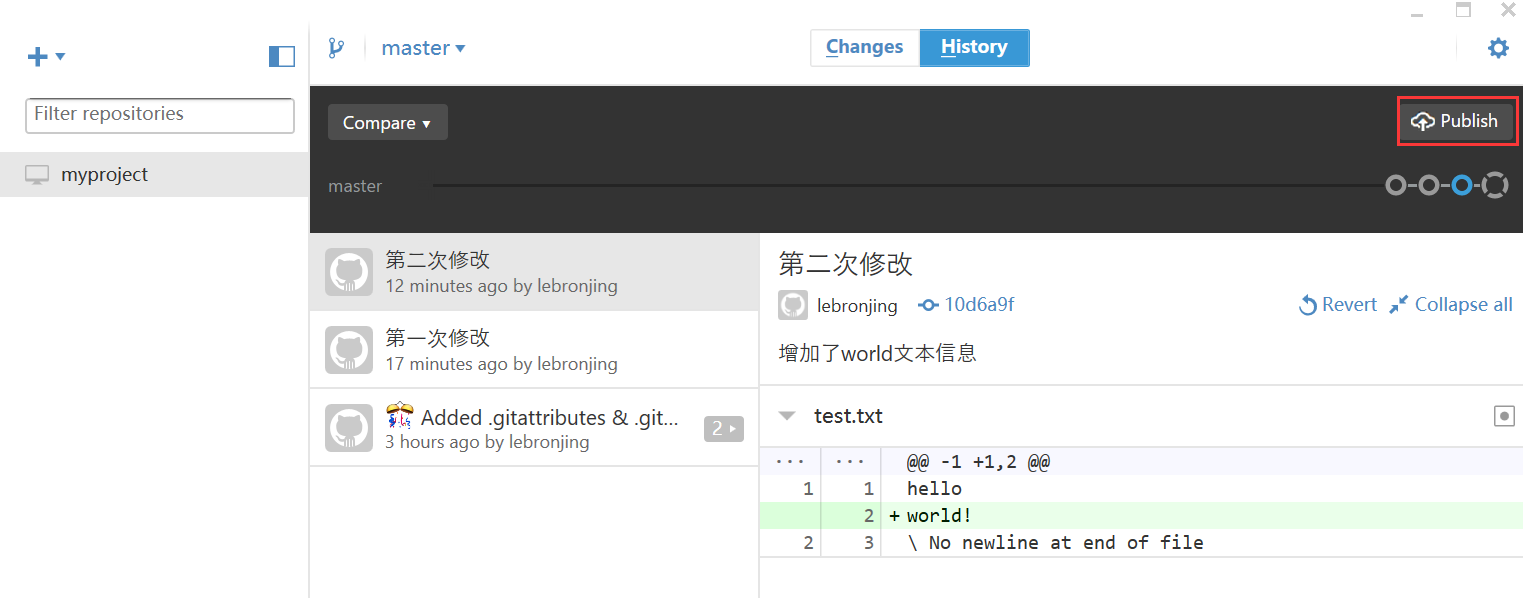
此时，当我们打开github网页版（可以从客户端直接进入，点击右上角设置的按钮选择Options进入Options选项页面点击manage进入网页），



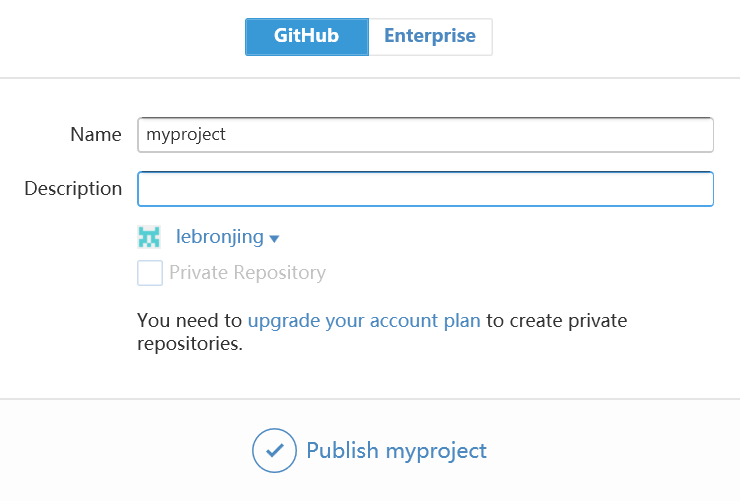
就会发现此时你的修改的内容并没有出现在这里。



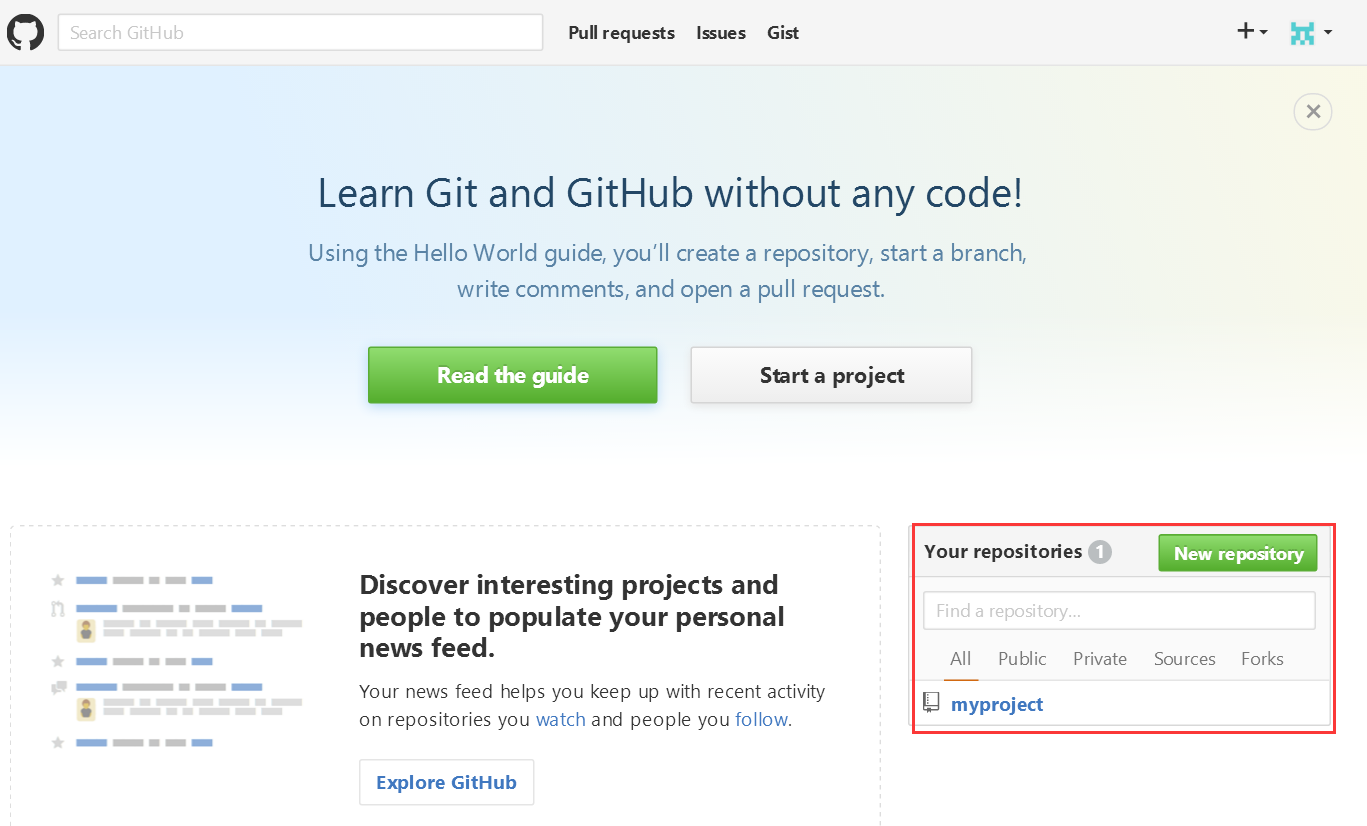
这是因为你没有进行同步，仅仅是在本地进行了修改。此时我们仅仅需要点击客户端右上角的Publish。



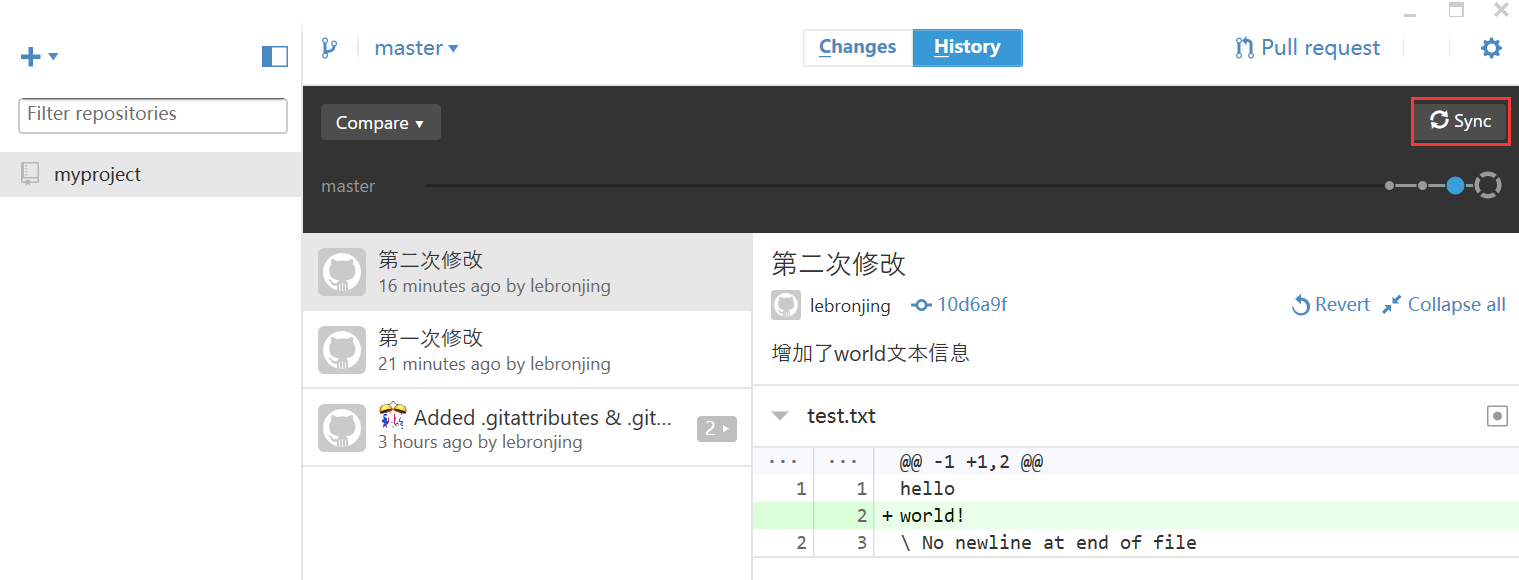
会弹出如下图的确认。点击Publish myproject。



我们再刷新网页，就会发现项目已同步成功。如下图：

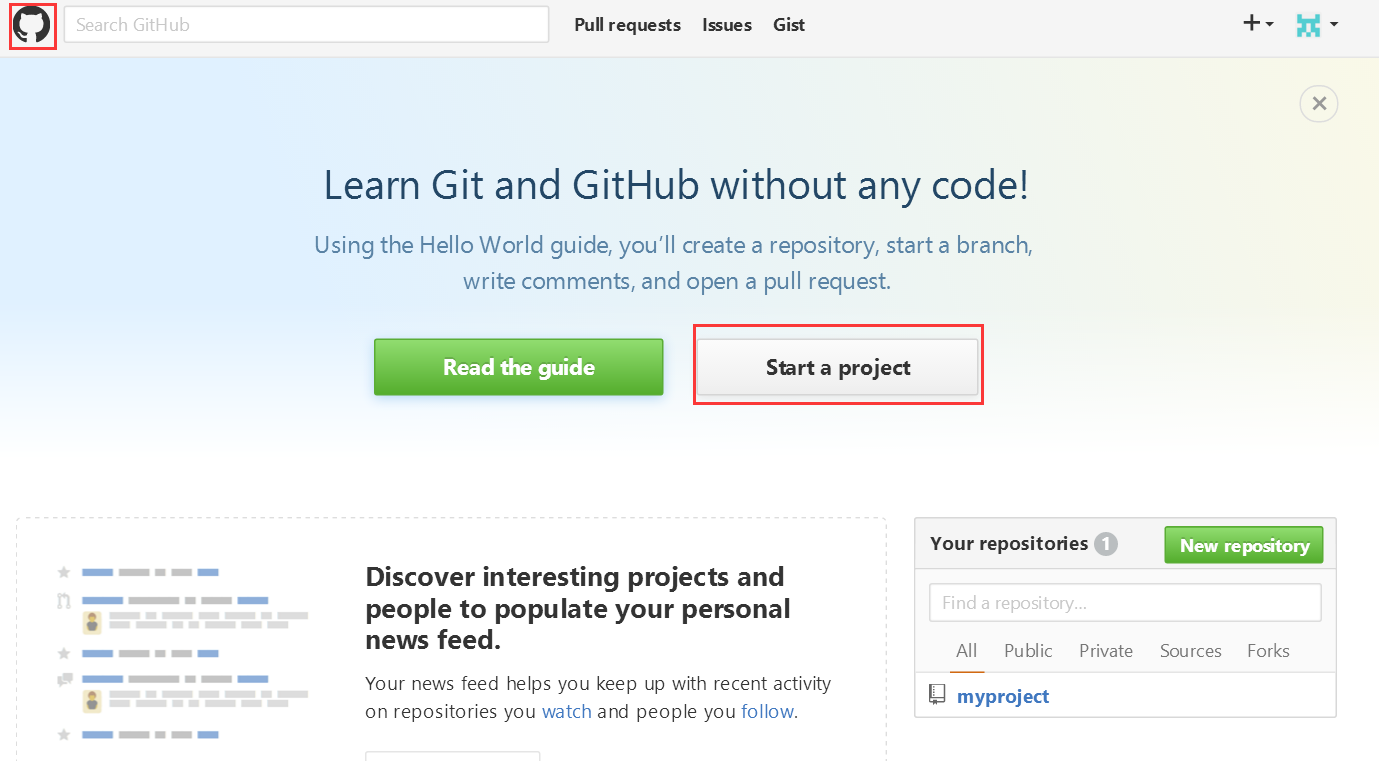


再回到客户端你会发现原来的publish已经变为了Sync。点击Sync同步代码库！



1. **网页上新建仓库**
2. 现在我们演示在网页上新建仓库。
3. 点击那个像小猫一样的图标进入如下图页面点击Start a project进入新建仓库页面。

也可以点击右上角那个+号点击弹出的 New repository进入新建仓库页面。



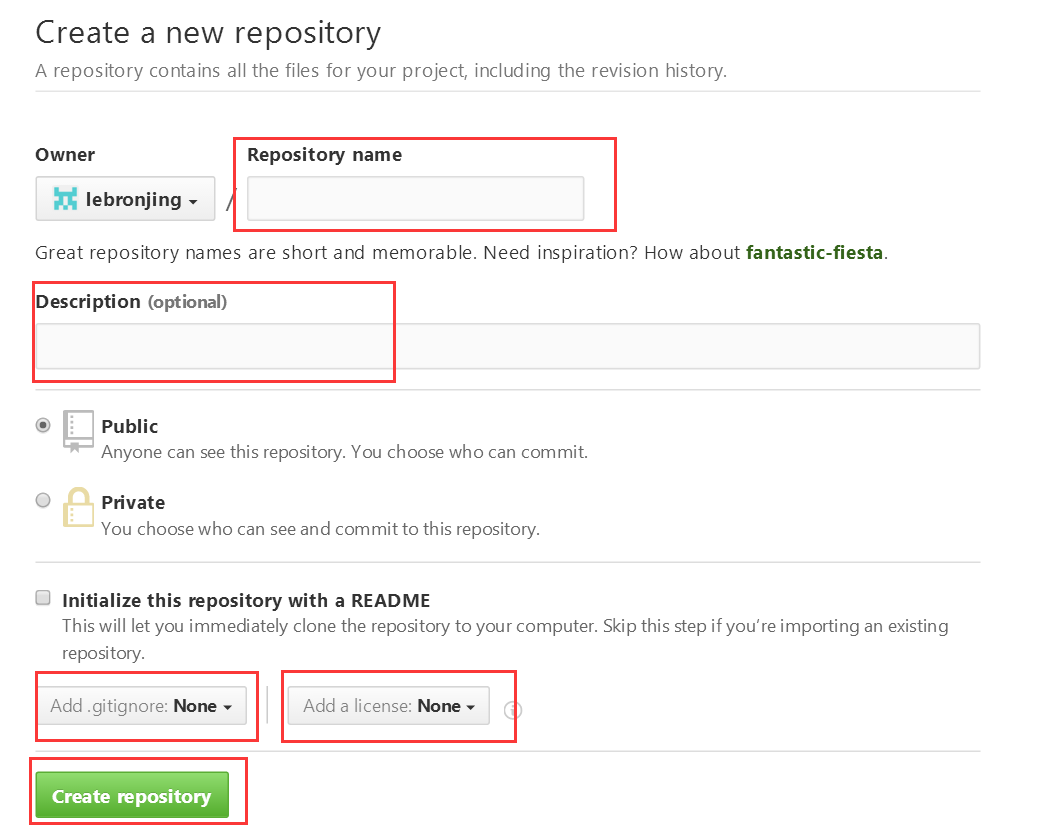
3、跳转到这个页面上Repository name 让你输入一个仓库的名字

Description (optional) 对这个仓库的描述

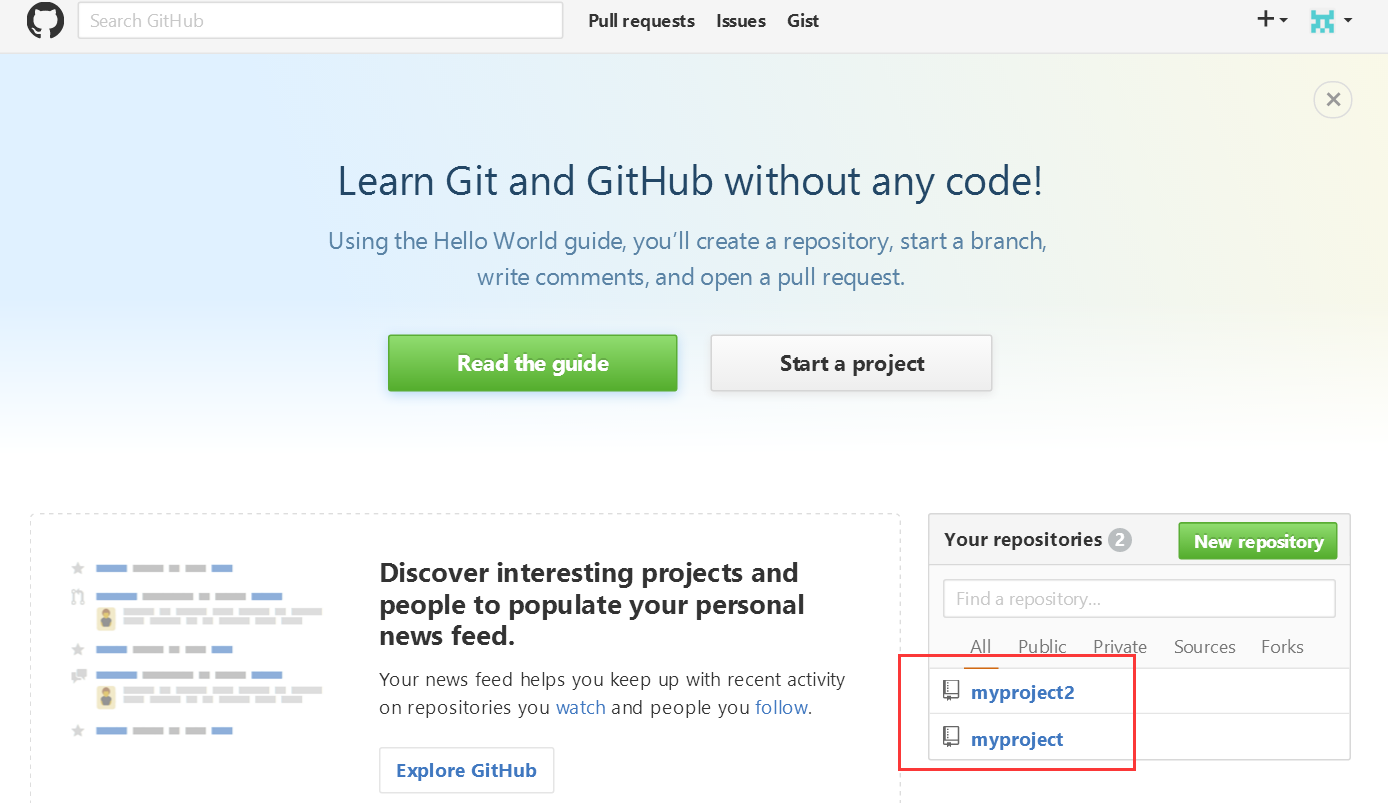
public和private是对这个仓库的开放权限

Initialize this repository with a README问你要不要帮你根据你输入的项目名字和描述初始化一个项目(生成一个README的md文件)

然后点击Create repository 就创建成功了.



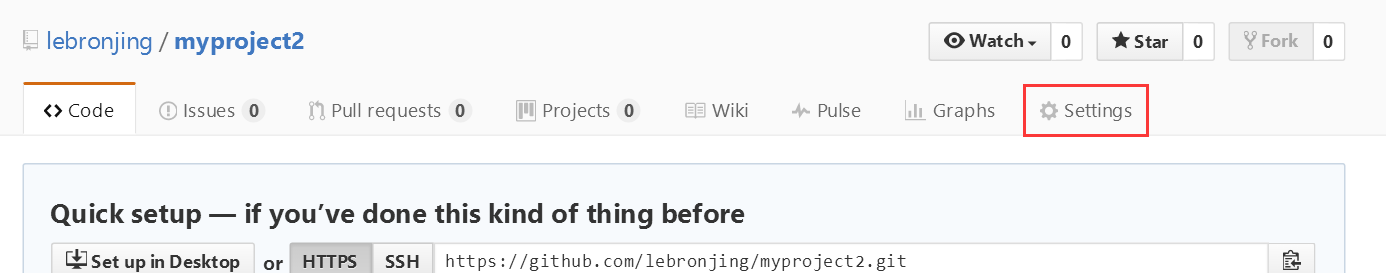
在网页首页可以看到二个仓库了，不过觉得还是客户端方便点，另外命令行形式的还没研究过，希望研究过的朋友也可以共享出来。

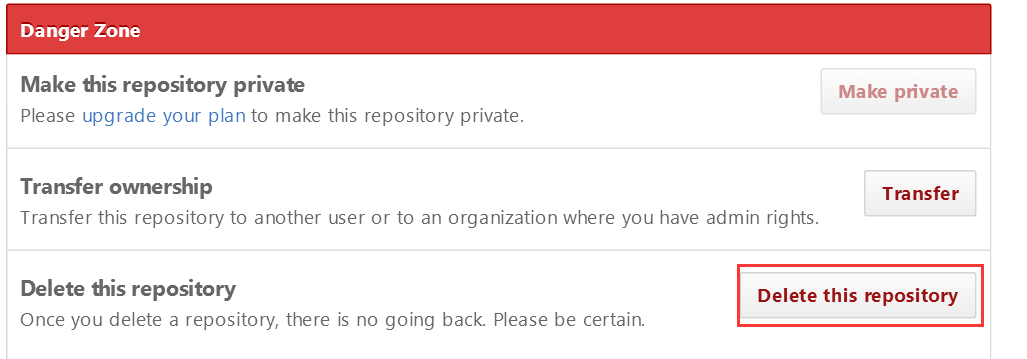


1. **删除仓库**

**删除仓库需要在网页端操作。**

点击需要删除的仓库进入仓库页面选择settings最下面有删除操作





点击delete this repository按钮会弹出确认框，输入需要删除的仓库名，然后确定仓库就删除了。

