

2022最新版github入门教程，教你如何一步步创建自己的github账号并初始化仓库，然后使用git工具配置个人工作环境。配合gitee仓库，作为github的镜像仓库使用。这篇文章很基础，对萌新来说是非常友好的，会持续更新优化。顺带一提，同样将最新的github pages服务的设置一并写入了文档中。

官方文档：

<https://docs.github.com/cn/get-started/quickstart>

本文将永久保留在个人的github仓库demo示例中：

<https://github.com/cnwangk/demo>

前言

用心作题图，用脚写文档。

其实这篇文章是对之前的git系列文章的一个补充，毕竟还是需要有人完善最新版的github创建教程，并且使用gitee作为镜像仓库。

当年也是萌新过来的，自己踩过不少坑，现如今很多教程老化了。为了照顾到新入坑的萌新，近段时间写了一个最新版的创建github账号以及登录的详细教程，并使用gitee作为github的镜像仓库。最后会存放到我新建的demo示例仓库中，并且使用gitee作为图床。



正文

如下所描述的教程需要事先创建好github和gitee账号。此处会详细讲解github如何创建账号，毕竟这货是纯英文的。对gitee则不会详细介绍如何创建账号，本身就是中文的，比较容易上手。

个人使用github搭建的hexo博客示例：

<https://cnwangk.github.io/>



一、创建github账号

1、创建github账号

1.1、github登录页面

登录账号：<https://github.com/login>

最下面有一行Create an account就是创建账号的链接，打开即可进入创建页面。

1.2、创建账号页面

创建账号：<https://github.com/signup>

github创建账号的页面，这个欢迎界面比起N年前确实漂亮不少哟，提示输入你的邮箱。**最好是填写自己常用的邮箱账号**，后续登录验证接收验证码之类的，都需要使用这个填写的邮箱。

1.3、创建账号需要的一些参数

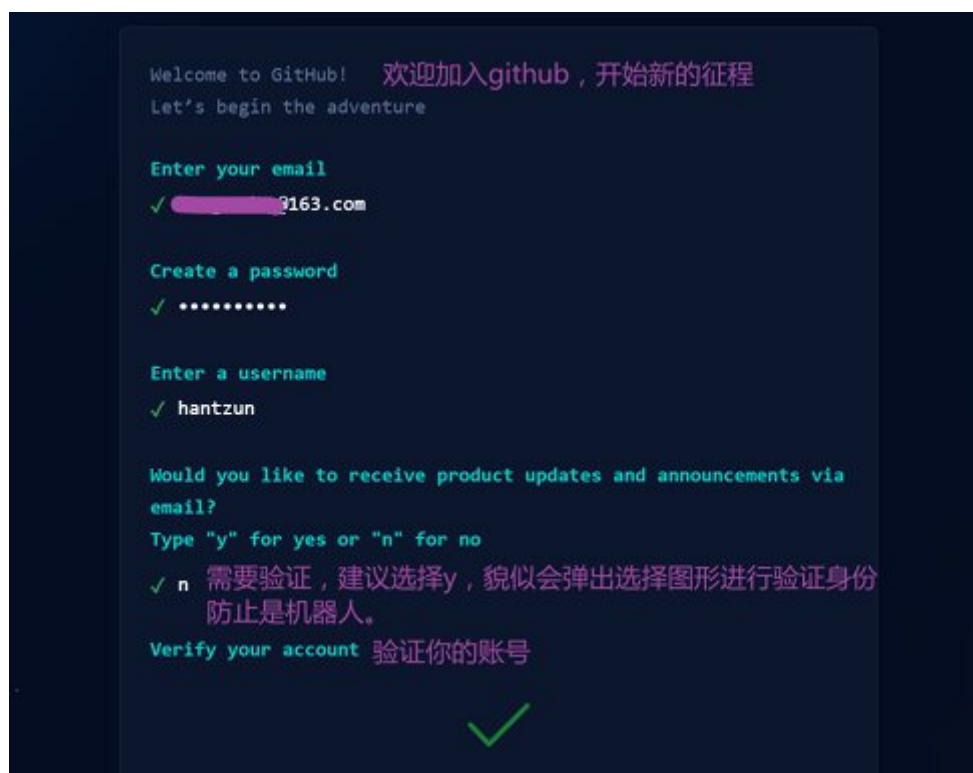
- enter your email：输入你的邮箱账号；
- create a password：输入密码；
- enter a username：输入用户名，注册完后可以用于登录；
- 最后一行提示y与n，**建议选n**，貌似有个防机器人验证。



1.4、二次验证

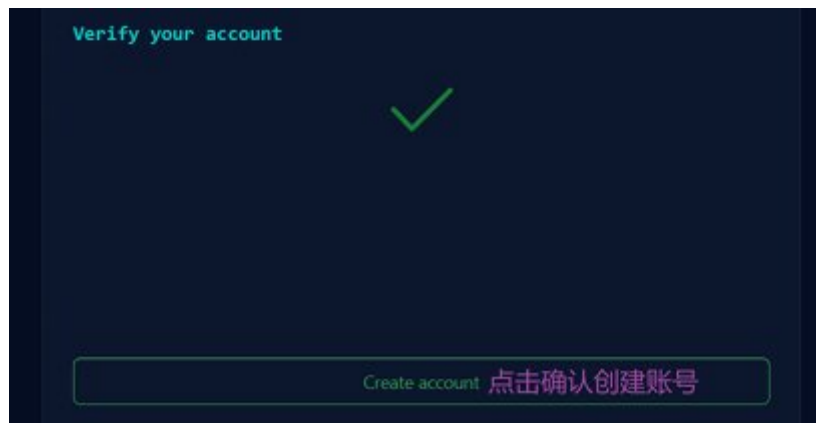
欢迎加入github，这一步确实验证了我的猜想，**选择y**验证自己新创建的账号。

确实弹出来一个页面，让你选择图片，进行真人认证。



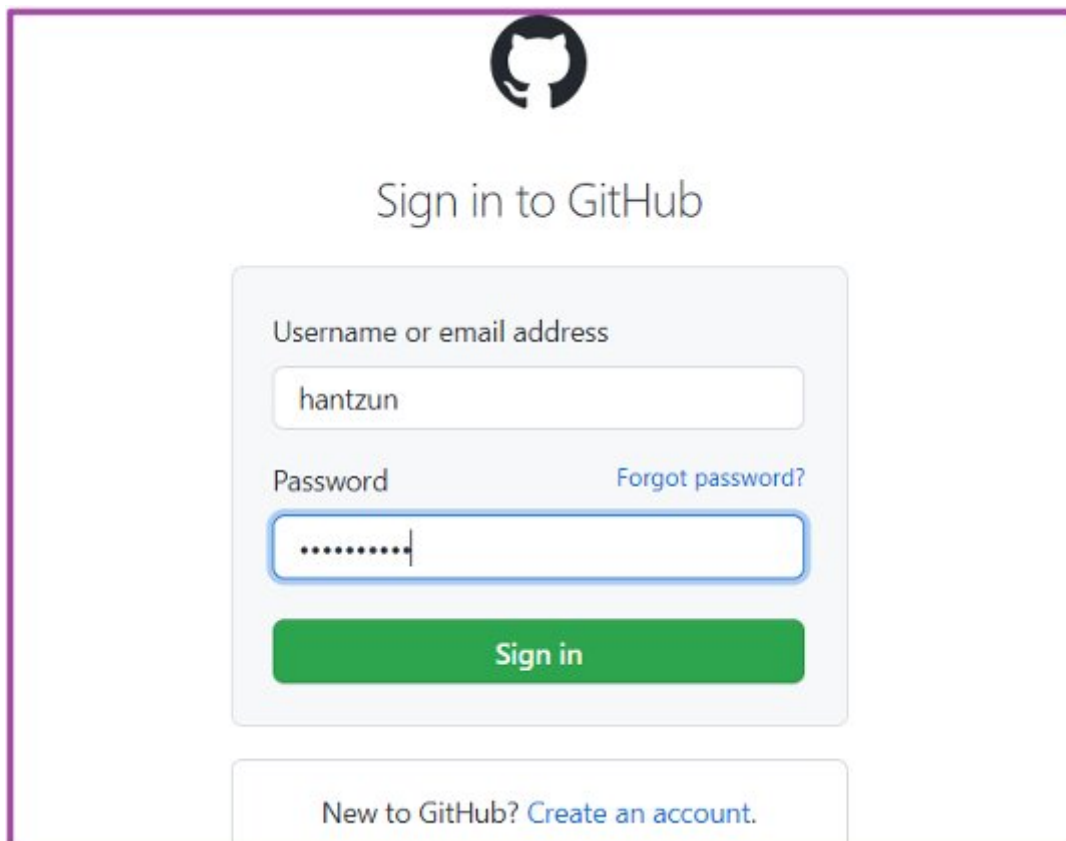
1.5、续上面的截图

点击确认创建github账号：



1.6、登录之前创建好的账号

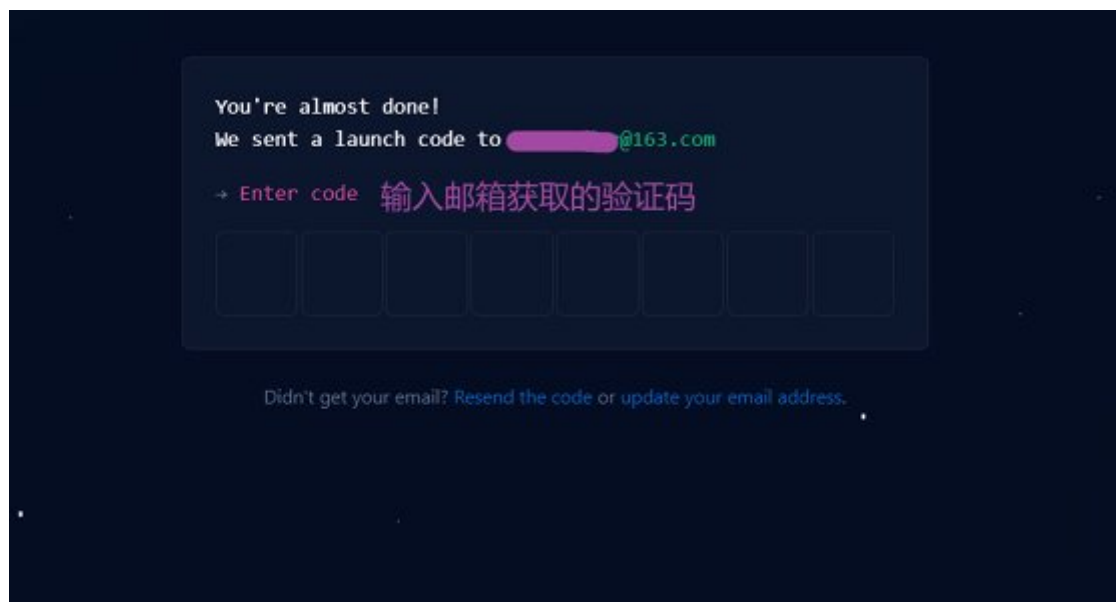
需要接收验证码，还记得上面说的要填写的那么邮箱吗？



1.7、邮箱验证码

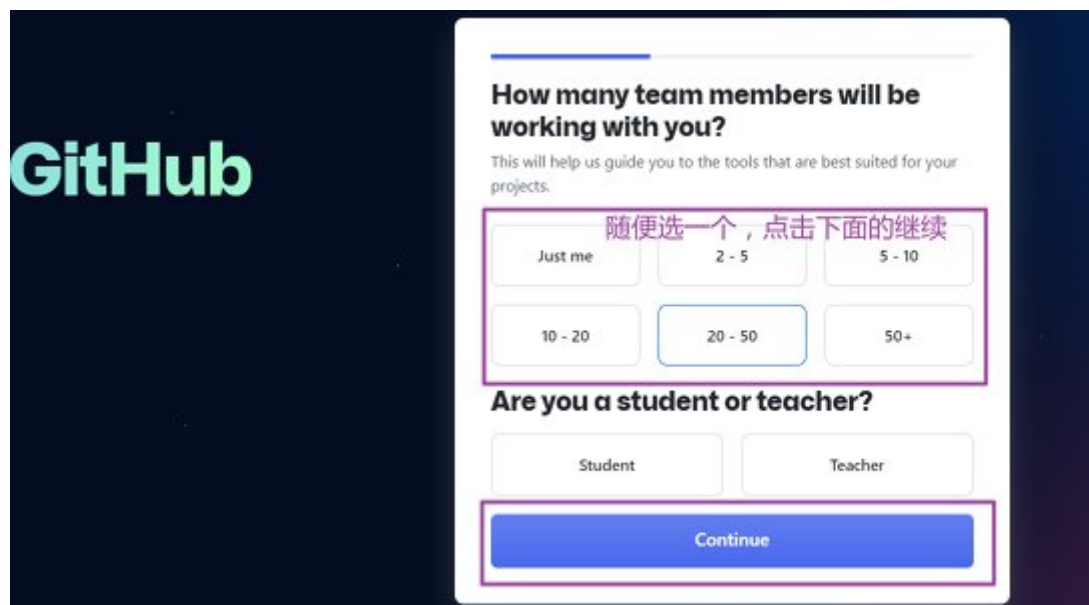
之前输入的邮箱账号，接收到验证码输入即可进行到下一步界面。

有可能网速原因，大家都懂得，有时会抽风访问不进去。



1.8、询问你是个人还是团队，当时没仔细看，所以写了随便选一个。

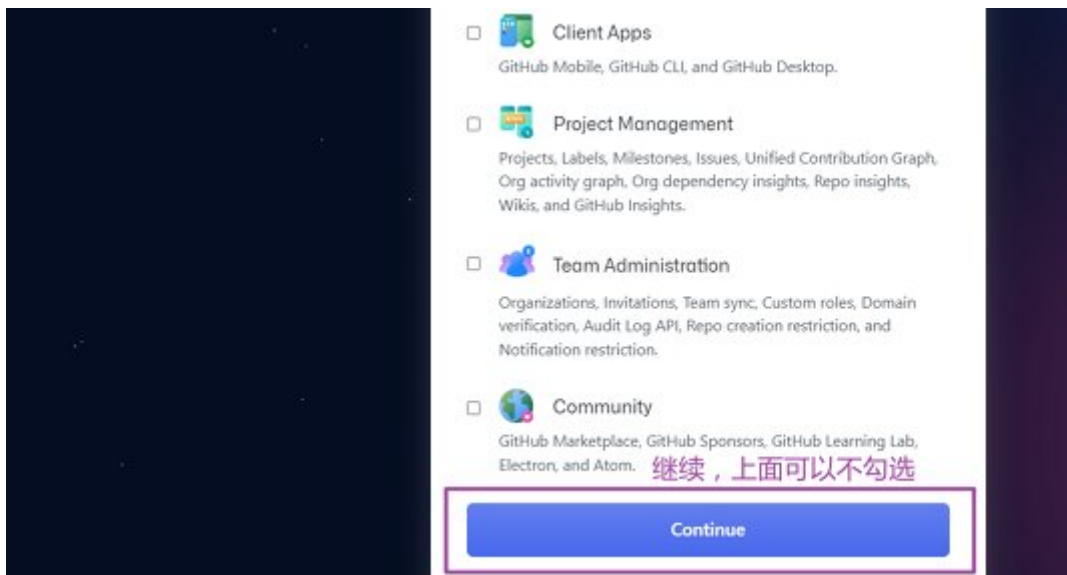
- just me：个人就选择just me；
- 如果是team，就选择后面的数字



1.9、参数的勾选

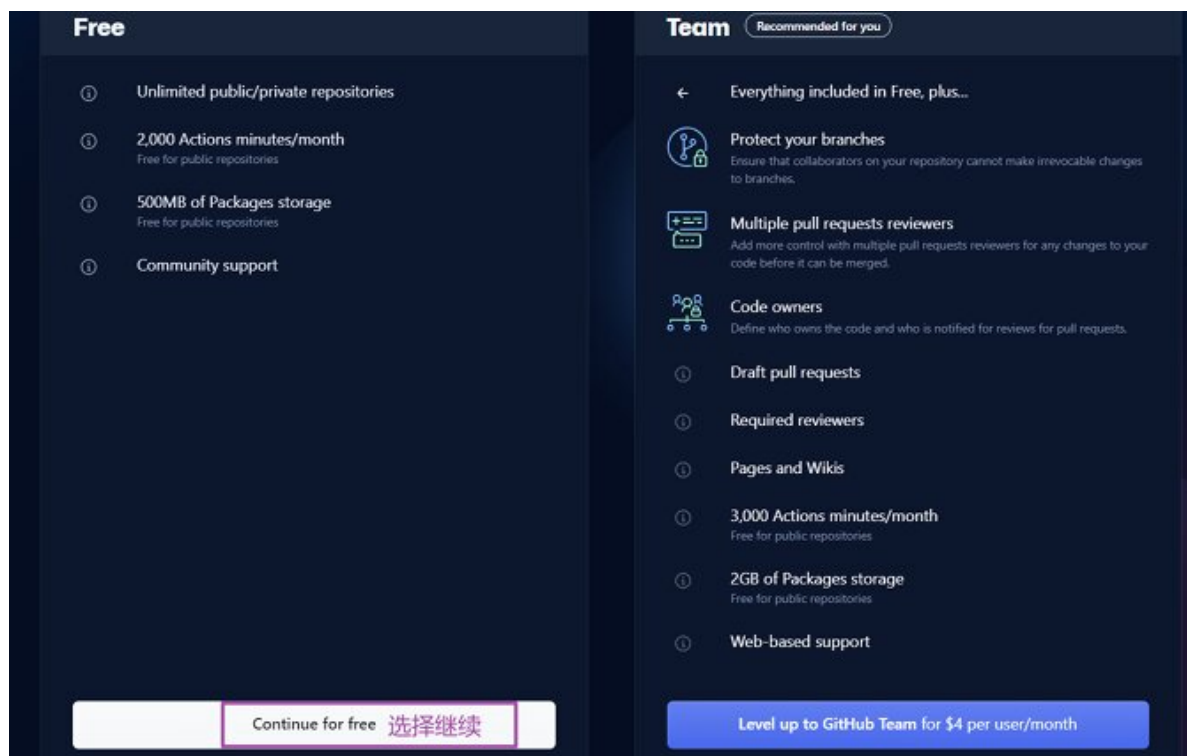
可以不选，点击continue继续。下图中部分参数：

- 客户端APP
- 项目管理
- 团队管理员
- 社区



1.10、选择免费或者团队

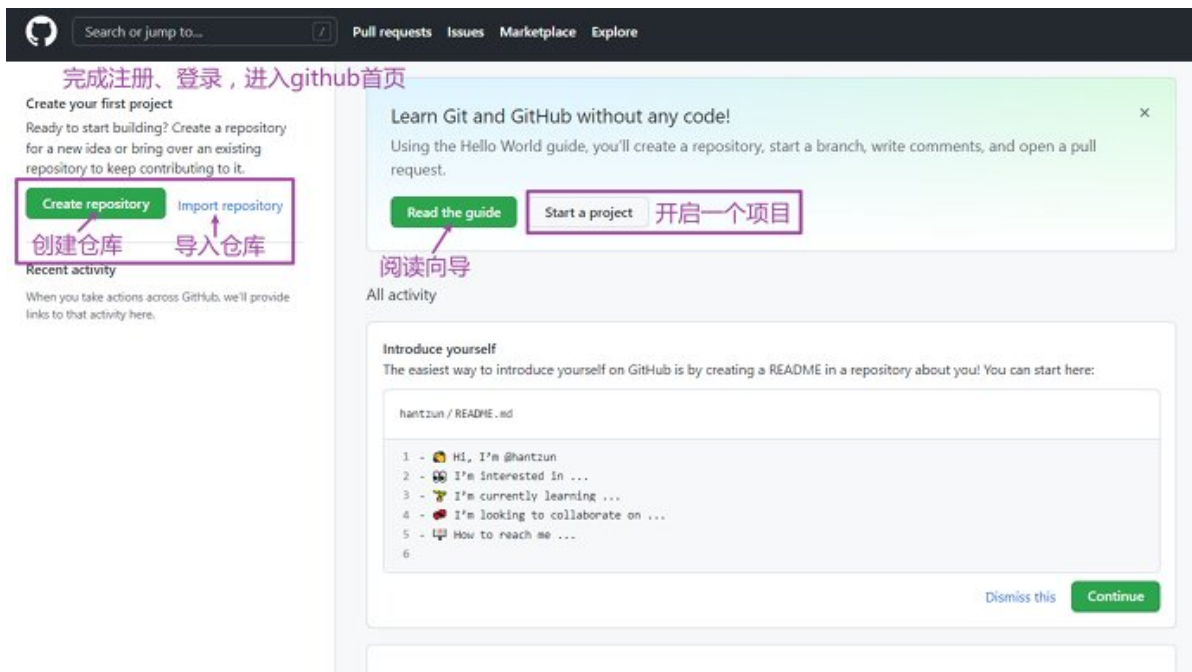
- free：个人版免费提供的，选择继续；
- team：这个是团队版，收费的，每月4美刀。



1.11、登录到github首页

至此终于创建完成，成功进入个人的github主页。介绍一些基本的使用：

- create repository：创建仓库，在当前页面右上角个人的图像展开一样可以新建仓库和导入仓库；
- import repository：导入远程仓库；
- read the guide：阅读向导。



到此创建账号过程就完成了，纯英文的确实对萌新不是很友好。但是习惯就好，毕竟是学习编程的好平台。

2、初始化仓库

2.1、创建仓库demo

点击创建仓库：<https://github.com/new>

新建的demo示例：<https://github.com/cnwangk/demo>

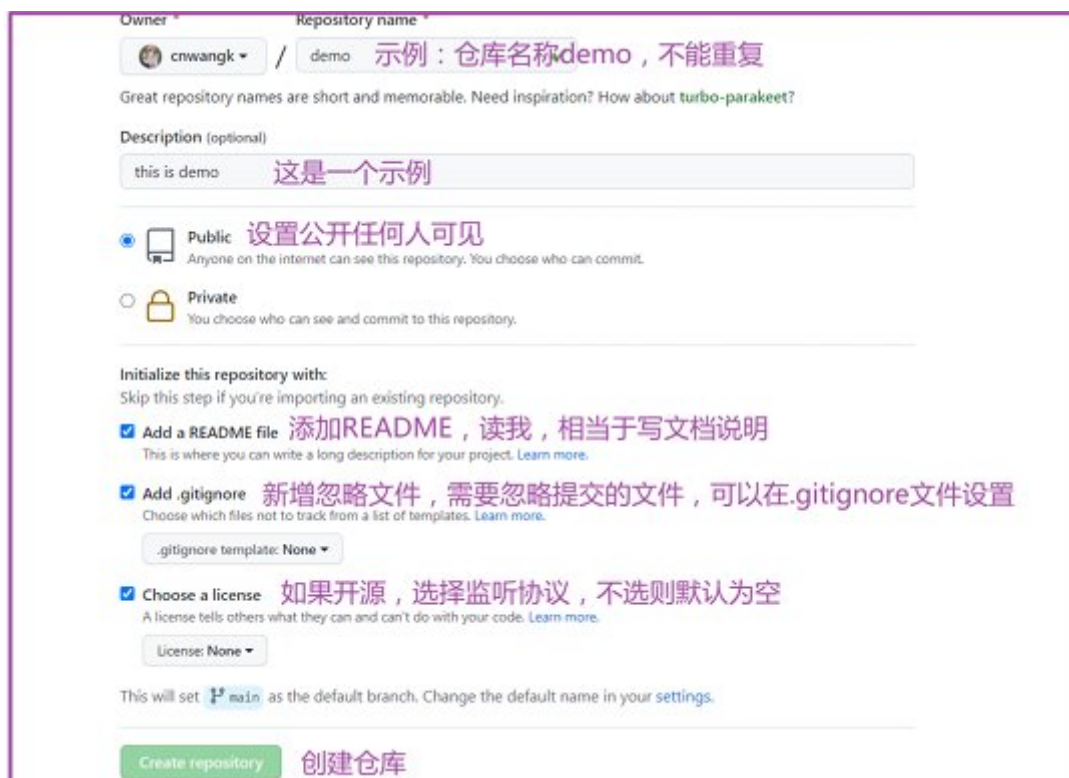
接着上面创建完账号，继续新建仓库，并且进行个人的第一个仓库初始化设置。

github新建仓库，初始化设置如果勾选了下列三个选项则需要设置模板，没有选择模板否则无法创建仓库。

- Add a README file：初始化设置可以不勾选，不选则不会创建。
- Add .gitignore：初始化新建忽略文件需要下拉设置模板，不选则不会创建。
- Choose a license：初始化设置可以不选开源协议，如果有开源需求则设置，例如：Apache License2.0。

2.2、设置仓库初始化参数

续上面的图，接着设置参数，选择了下面的三个参数则需要选择模板才能创建仓库：



The screenshot shows the GitHub repository creation interface. The 'Repository name' field is set to 'demo', with a note: '示例：仓库名称demo，不能重复' (Example: Repository name demo, cannot be repeated). The 'Description' field contains 'this is demo', with a note: '这是一个示例' (This is an example). The 'Public' option is selected, with a note: '设置公开任何人可见' (Set public, anyone can see). The 'Add a README file' option is checked, with a note: '添加README，读我，相当于写文档说明' (Add README, read me, equivalent to writing document description). The 'Add .gitignore' option is checked, with a note: '新增忽略文件，需要忽略提交的文件，可以在.gitignore文件设置' (Add new ignore file, need to ignore files to be committed, can be set in .gitignore file). The 'Choose a license' option is checked, with a note: '如果开源，选择监听协议，不选则默认为空' (If open source, choose license, if not selected, default is empty). The 'Create repository' button is visible at the bottom.

2.3、README.MD文件设置

初始化设置可以新增一个README.MD文件，对这个仓库的一个介绍，markdown格式。

说明比较简单，将仓库的名称以及简介记录到文件中，图中输错了，其实是this is demo，后面修正了，可在线编辑。



2.4、.gitignore模板设置

这里以近几十年比较火热的Java语言作为demo进行讲解。

默认选择了Java语言模板，里面设置了一些忽略文件：

- log日志；
- class编译文件；
- jar包、war包以及压缩包等等。

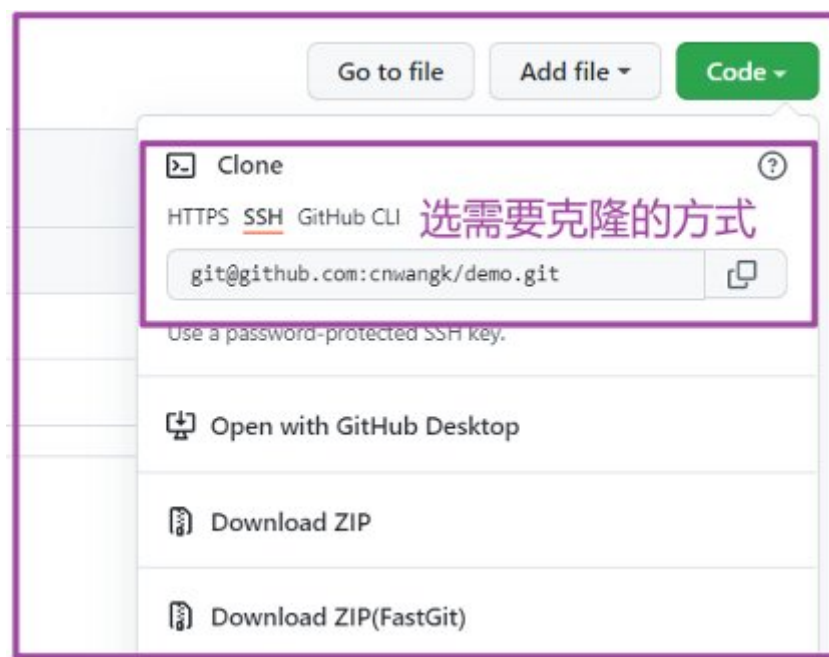


至此，我们的第一个github仓库demo就创建完成，并且预配置了readme以及忽略设置文件。

3、通过git配置工作空间

接着上面的初始化仓库进行讲解，首先需要安装好git环境，Windows下安装比较容易，就不再赘述。

然后将我们创建的demo仓库克隆到本地，选择则克隆方式为SSH，接下来配置的也是公钥SSH：



关于git工具的下载与详细使用请参考我之前的文章《献给初学者的git和github教程，使用hexo搭建个人博客》，地址如下：

github仓库地址：<https://github.com/cnwangk/SQL-study/tree/master/md/git>

备用地址：<https://cnwangk.github.io/tags/git/>

3.1、配置git环境

安装好了git工具，此时可以在桌面右键（Git Bash），输入以下命令进行设置用户以及email（邮箱）。

```
$ git config --global user.name "demo"
$ git config --global user.email "demo@example.com"
```

参数`--global`代表配置全局的，不加`--global`参数，则是配置当前仓库生效，当然设置了也会覆盖当前仓库的设置。可以通过`git config -l`命令列出整组配置文件共同查找的所有变量设置值，或者使用你熟悉的`cat`以及喜欢的`vim`命令查看配置文件：

```
$ git config -l
#或者使用你熟悉的cat以及喜欢的vim命令查看配置文件
$ cat .git/config
$ vim .git/config
```

配置ssh-key，生成ssh公钥。Windows下默认在系统盘的当前用的ssh目录下，可以配置ed25519或者是rsa方式都行，github官网的教程默认写的是rsa方式。如下图所示，默认回车生成ssh-key：

```
$ ssh-keygen -t ed25519 -C "demo@example.com"
Generating public/private ed25519 key pair.
Enter file in which to save the key (/c/Users/admin/.ssh/id_ed25519):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /c/Users/admin/.ssh/id_ed25519
Your public key has been saved in /c/Users/admin/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:co+BAyXti6CuZuJ+e/...
The key's randomart image is:
+--[ED25519 256]--+
|..B.ooO+|
|.B o Bo.|
|. . +.B .|
|. . =.% |
|. ...O.S E|
|. ..O.+ =|
|. . + . .|
|. + . o |
|. + . o |
|Oo..o |
+-----[SHA256]-----+
```

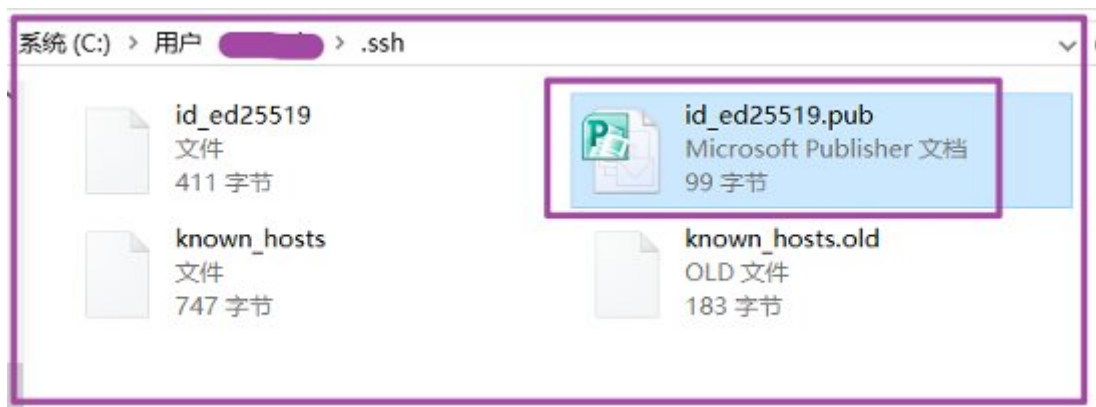
注意：ssh-keygen之间不能有空格，紫色框中的是你生成ssh key的目录，最后需要配置到github中。

```
#方式一
$ ssh-keygen -t ed25519 -C "demo@example.com"
#方式二
$ ssh-keygen -t rsa -b 4096 -C "demo@example.com"
```

检查你的系统目录是否有ssh公钥。默认情况ssh公钥，可能是以下几种文件形式：

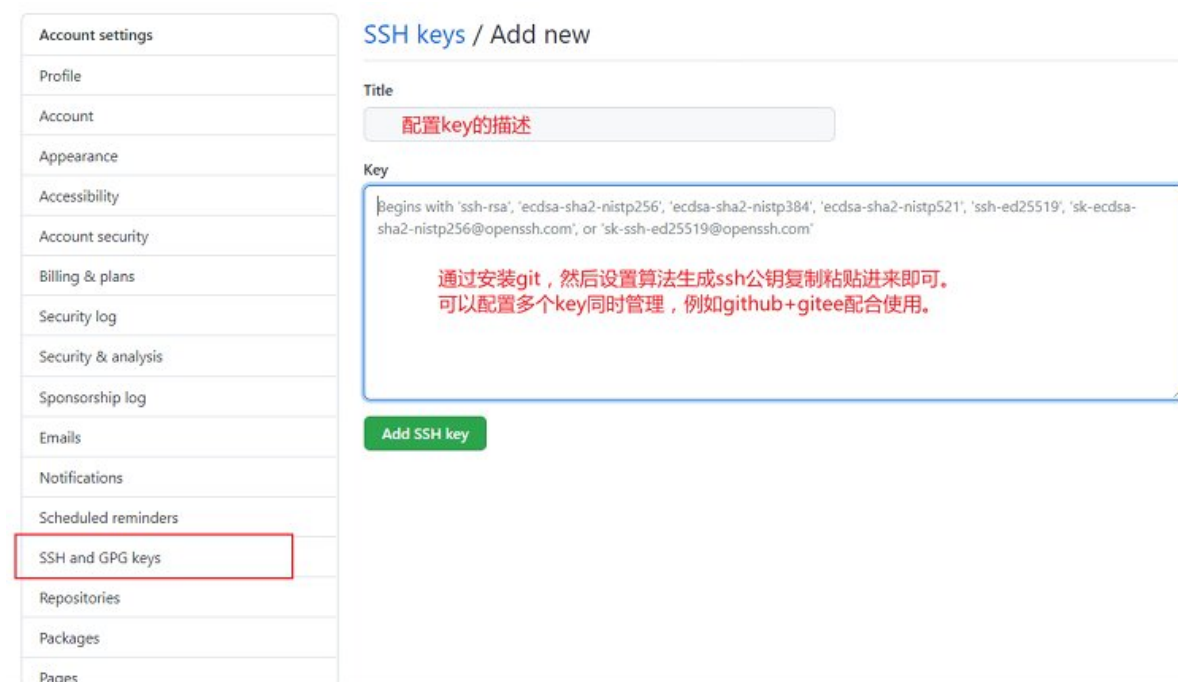
- id_ed25519.pub
- id_rsa.pub
- id_ecdsa.pub
- id_dsa.pub

个人在Windows系统下演示生成的ssh-key文件，使用的是ed25519模式。



将生成的pub文件的key值使用记事本、SublimeText或者VSCode打开，然后复制到github账号的ssh-key中。

<https://github.com/settings/ssh/new>



配置完ssh-key之后，Windows下右键打开 Git Bash，使用 ssh -T 命令测试验证。当前配置了github的ssh公钥，验证返回结果成功。未配置github的ssh公钥，则测试验证返回的结果是权限（permission denied）拒绝。

使用命令测试验证：

```
#验证github
ssh -T git@github.com
```

示例：验证成功，返回结果为successfully；验证失败，则返回permission deny权限拒绝。

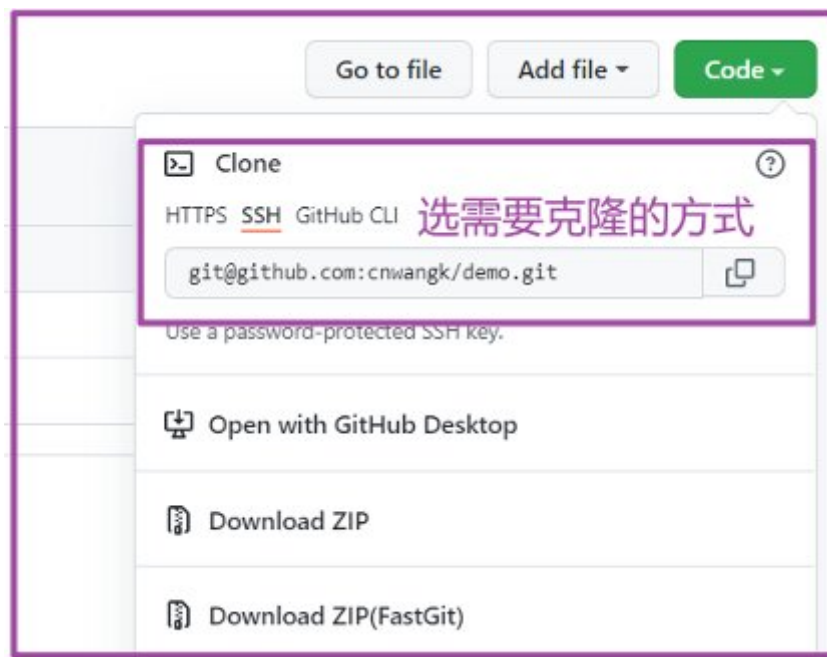
```
MINGW64 ~/github/SQL-study (master)
$ ssh -T git@github.com
git@github.com: Permission denied (publickey). 验证：权限拒绝

MINGW64 ~/github/SQL-study (master)
$ ssh -T git@github.com
Hi cnwangk! You've successfully authenticated, but GitHub does not provide shell
access. 验证：测试返回成功
```

3.2、克隆仓库

打开git bash命令窗口，使用git clone命令克隆远程仓库demo示例，可以选择多种方式：

- HTTPS方式
- SSH方式，个人最常用的方式
- github cli方式



```
$ git clone git@github.com:cnwangk/demo.git
```

远程仓库demo示例：<https://github.com/cnwangk/demo>

3.3、连接远程仓库完成初始化提交

接着上一步克隆demo仓库，然后**进入demo仓库（目录）进行如下操作**，空目录默认是不会提交的，需要写点内容进去：

```
$ git add --all #暂存所有未追踪的文件
$ git commit -a -m "初始化提交" #初始化提交
$ git push git@github.com:cnwangk/demo.git
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To github.com:cnwangk/demo.git
   c670e49..48b58fb  main -> main
```

至此git环境的配置与github克隆以及推送至远程仓库就讲解完毕，更多的git操作可以参考《git版本控制管理》这本书。github操作则可以参考官方文档，在偶然的一次浏览github文档时发现部分的官方文档已经汉化了，我将当时的截图上传到上面演示的demo仓库中了。

4、加速访问github

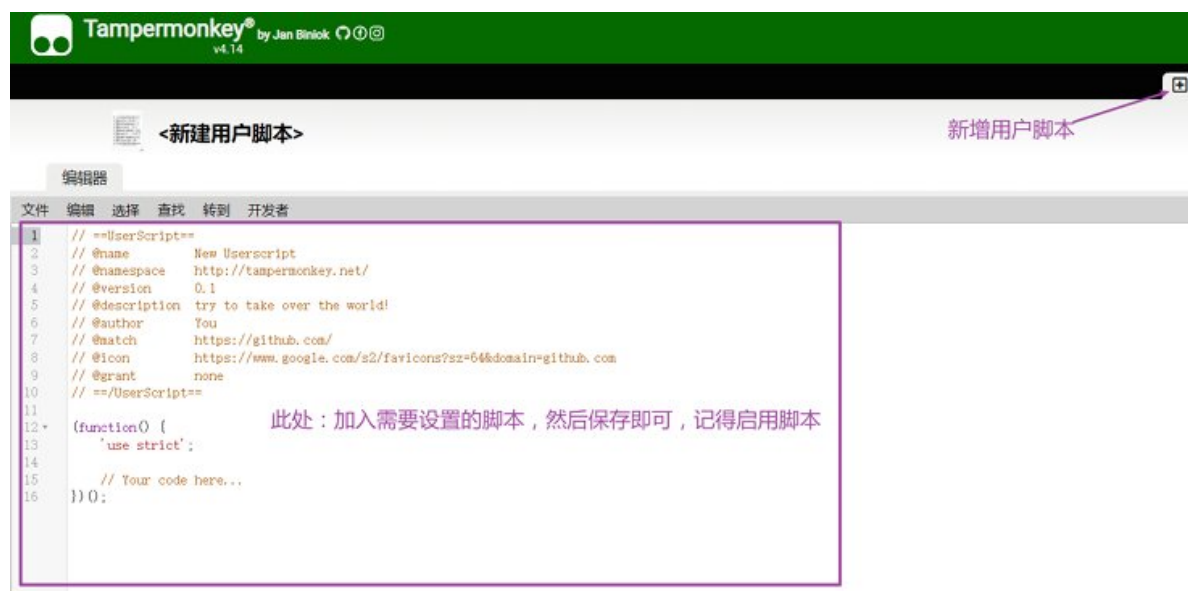
油猴插件，估计大家都不陌生吧！没错，就是油猴插件。

油猴插件下载：<https://www.tampermonkey.net/>

在Google Chrome管理油猴插件：



新增js脚本：



在我们的浏览器中启用新增的js脚本：



二、gitee初次使用

1、创建gitee账号

1.1、gitee创建账号，支持多种方式进行登录：

<https://gitee.com/signup>



2、初始化仓库

登录到个人的gitee账号，进行创建仓库。gitee新建仓库：

<https://gitee.com/projects/new>

三、使用gitee作github镜像仓库

要问为什么需要使用gitee作为镜像仓库使用，那就是大家都懂得。

gitee仓库管理界面，找到功能设置，然后滑至底部，进行设置需要强制同步的远程github仓库：



<https://gitee.com/dywangk/htz/settings#function>

gitee设置同步github做镜像仓库：



<https://github.com/cnwangk/SQL-study>

最后定期进行同步github仓库：



gitee同步github仓库就介绍到这里，上手比较容易，毕竟是中文界面。

tips： gitee的私人仓库是免费使用的哟，一般我不告诉别人，自己的私人工作空间可以使用gitee哟！还可以使用PicGo配置gitee图床，毕竟访问gitee比较快，访问github还得配置CDN加速。配置图床的教程，我写过一篇稀烂的文章，可以在我的博客或者公众号上找一找。

四、github遇上hexo

1、准备环境

当github遇上hexo、jekyll或者hugo，即将发生美妙的事情，那就是搭建个人博客小站。你只需要做几步简单的配置，就可以轻轻松松搭建属于自己的私人博客，再也不用担心乱七八糟的审核了。这里说的github指的是github pages服务，创建一个仓库命令为**用户名追加.github.io**，并开启github pages服务。

github pages仓库名配置例如：

sky.github.io

Windows下需要准备环境：

- 下载并安装nodejs环境；
- 通过nodejs安装hexo init blog；
- 安装git环境，便于推送至github。

通过hexo new “hello world”生成第一篇博客，使用hexo server启动服务，访问如下链接测试：

<http://localhost:4000>

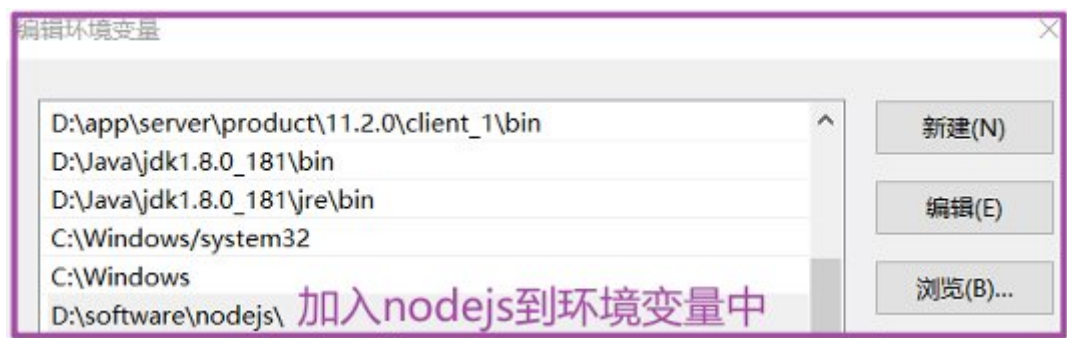
2、hexo配置

2.1、安装nodejs

下载安装比较容易，不做过多介绍，但会讲注意事项。

<https://nodejs.org/en/>

下载Windows版本的nodejs并安装，**建议将nodejs安装目录加入配置环境变量**便于操作：



个人安装的nodejs版本，安装nodejs后在开始菜单右键即可打开cmd命令窗口，**也可以使用node -v查看版本**：

```
D:\work\createSpace\hexo>node -v
v14.16.1
```

```
管理员: Node.js command prompt
Your environment has been set up for using Node.js 14.16.1 (x64) and npm.
C:\Windows\System32>color a
C:\Windows\System32>d:
D:\>cd \work\createSpace\
D:\work\createSpace>cd \work\createSpace\hexo
D:\work\createSpace\hexo>
```

个人安装的nodejs环境

hexo目录自己创建的，一些操作放在此目录了。切换到初始化的blog目录中进行管理，新建文件、生成、启动服务

2.2、安装hexo

然后使用npm再安装hexo模块。这里只介绍Windows下安装hexo环境：

```
D:\work\createSpace\hexo\blog> npm install hexo-cli -g
$ hexo init blog
$ cd blog
$ npm install
$ hexo server
```

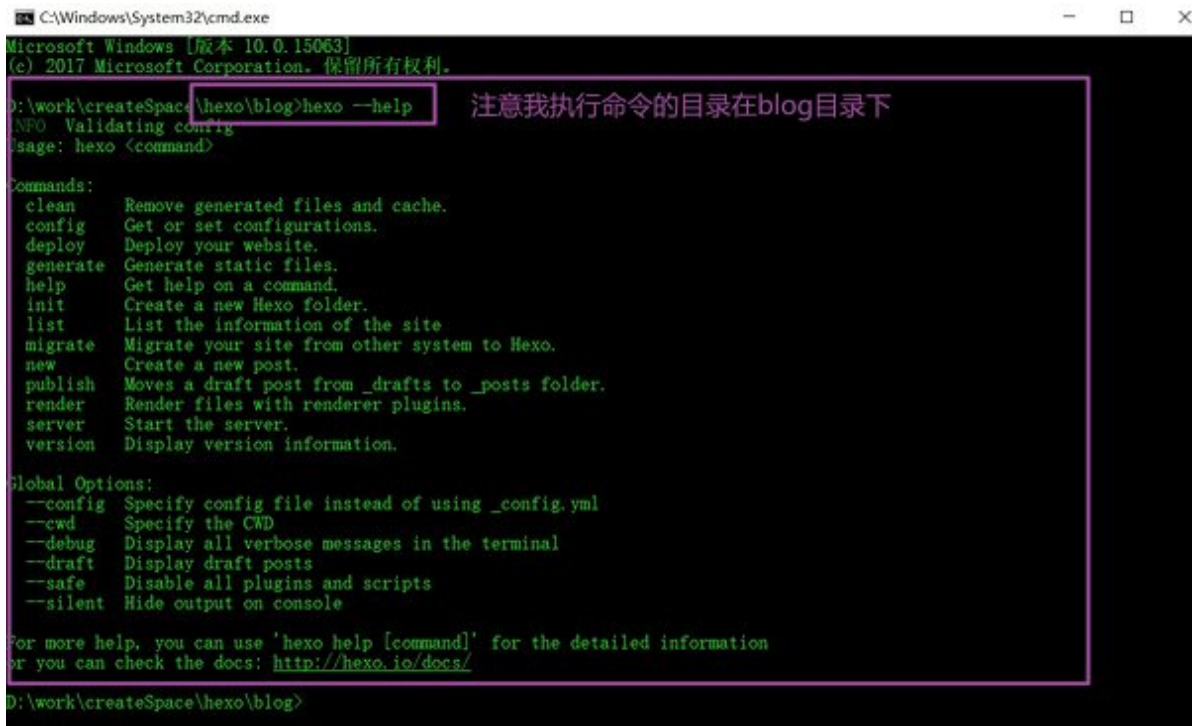
- 01、安装nodejs环境；
- 02、在node环境下安装hexo，打开cmd命令窗口执行：`npm install -g hexo-cli`
- 03、初始化blog：`hexo init blog`
- 04、进入blog目录：`cd blog`
- 05、继续在cmd窗口命令安装：`npm install hexo`
- 06、启动服务：`hexo server`
- 07、访问：<http://localhost:4000>

在node环境下安装hexo后生成的blog文件目录：



使用hexo命令，hexo new命令生成文件：

```
D:\work\createSpace\hexo\blog>hexo new "你要生成的md文件名"
hexo generate #生成静态文件
hexo server #启动服务
```



```
C:\Windows\System32\cmd.exe
Microsoft Windows [版本 10.0.15063]
(c) 2017 Microsoft Corporation. 保留所有权利。

D:\work\createSpace\hexo\blog>hexo --help
INFO Validating config
Usage: hexo <command>

Commands:
  clean      Remove generated files and cache.
  config     Get or set configurations.
  deploy     Deploy your website.
  generate   Generate static files.
  help       Get help on a command.
  init       Create a new Hexo folder.
  list       List the information of the site
  migrate    Migrate your site from other system to Hexo.
  new        Create a new post.
  publish    Moves a draft post from _drafts to _posts folder.
  render     Render files with renderer plugins.
  server     Start the server.
  version    Display version information.

Global Options:
  --config Specify config file instead of using _config.yml
  --cwd    Specify the CWD
  --debug  Display all verbose messages in the terminal
  --draft  Display draft posts
  --safe   Disable all plugins and scripts
  --silent Hide output on console

For more help, you can use 'hexo help [command]' for the detailed information
or you can check the docs: http://hexo.io/docs/

D:\work\createSpace\hexo\blog>
```

hexo server命令启动服务，通过hexo new “hello world”生成第一篇博客，使用hexo server启动服务，访问如下链接测试：

<http://localhost:4000>

个人搭建的示例：<https://cnwangk.github.io/>

hexo配合github还是挺方便的，毕竟在Windows平台我可以利用node.js安装hexo插件，进而配合一键生成模板然后提交到github。直接在hexo生成的blob模板中的markdown文件中写入你的文章。抛开通用性，markdown确实很方便。具体其它平台安装hexo可以参考官网中文文档，这里提供一下**hexo的网址**：

- hexo的github仓库：<https://github.com/hexojs/hexo>
- hexo的中文文档：<https://hexo.io/zh-cn/docs/>
- hexo中文github-pages教程：<https://hexo.io/zh-cn/docs/github-pages>
- hexo安装deploy实现一键发布：<https://github.com/hexojs/hexo-deployer-git>

3、配置github pages服务

github pages的配置页面

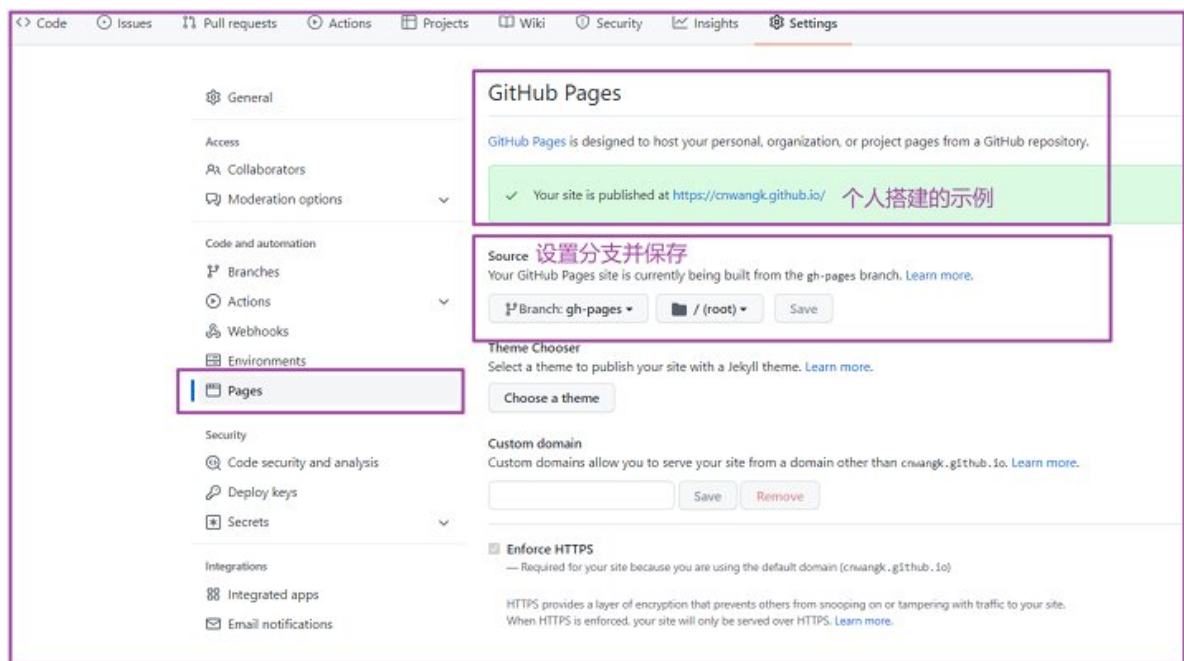
<https://github.com/cnwangk/test/settings/pages>

配置教程，纯英文的，可以用Google翻译一下哈

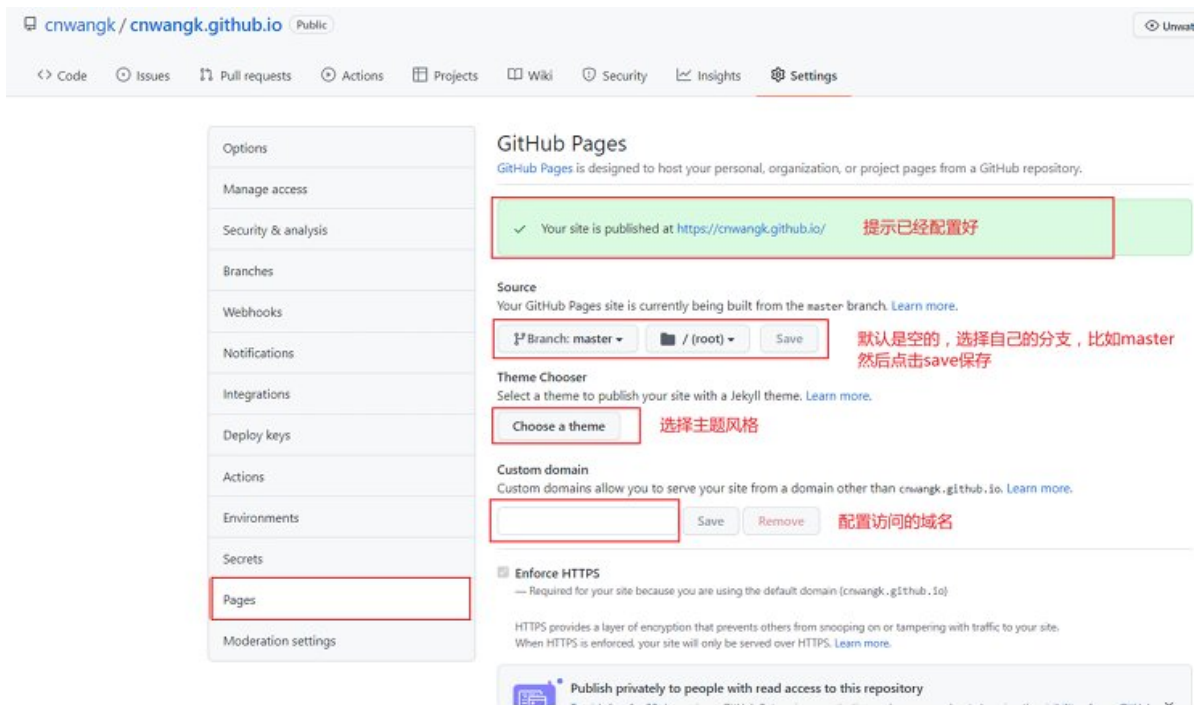
<https://pages.github.com/>

我测试配置了一个仓库

注意：仓库必须是公开的（public）、然后仓库命令可以命令为用户名加github.io。默认进入一个设置好的gh-pages分支的仓库这样显示内容的：



进入需要配置成github pages服务的仓库，找到Pages页面设置，选择自己新建的分支gh-pages，这也是github推荐你这样设置的：



简单的github pages 服务搭建示例，可以测试访问。

<https://cnwangk.github.io/>

总结

以上就是对github入门教程进了一个总结，很基础的一个教程，萌新一样可以看懂哟！希望能对你的工作与学习有所帮助。感觉写的好，就拿出你的一键三连。公众号上更新的可能要快一点，目前还在完善中。能看到这里的，都是帅哥靓妹。如果感觉总结的不到位，也希望留下您宝贵的意见，我会在文章中进行调整优化。



原创不易，转载也请标明出处和作者，尊重原创。不定期上传到github或者gitee。认准龙腾万里sky，如果看见其它平台不是这个ID发出我的文章，就是转载的。本文已经上传至github和gitee仓库SQL-study。个人github仓库地址，一般会先更新PDF文件，然后再上传markdown文件。如果访问github太慢，可以使用gitee进行克隆。