

Hexo blog

#Yangsu

#Notes

#Datawhale

#Hexo

1. 引言

在开始教程之前，先回答几个小问题。

1.1 如何进行笔记管理

当个人学习的笔记非常多时，笔记的检索和分享就变得颇为繁琐，这时候需要一个工具来对笔记进行管理，个人在进行选择时主要考虑如下因素：

- 界面美观。至少在打开笔记的时候，能够有一个美观简洁的界面。
- 便于检索。一方面，在单个笔记中能够快速定位到想查找的内容，另一方面，能够在不同文件中进行全局检索。这项功能在许多笔记软件中也是能实现的，如 Obsidian。
- 易于分享。很多时候，需要对自己的笔记进行分享，尤其是对于不使用 `markdown` 的朋友来说，常需要导出成 `pdf` 进行分享，由于格式和渲染的问题，常常会出现一些小问题。
- 跨终端。部分场景下，我们想通过移动端进行分享和查看笔记，此时本地的笔记项目存储显得有点捉襟见肘。

综上所述，最终，我问选择了使用基于 `GitHub Pages` 的静态博客方案来管理我的笔记。

1.2 为什么选择 GitHub Pages

很多人会选择 Wordpress，为什么选择 GitHub Pages 来搭建？

- GitHub Pages 有 300M 的免费空间，可以自己管理资料，本地编辑，远程管理和部署，安全可靠，资料不容易丢失。
- Github 是世界上最大的代码托管平台，全球用户已经破亿。对于不熟悉的用户而言，可以借助此次机会，了解和学习使用 GitHub 平台。
- 顺着学习 Git 的工作原理，对文件进行版本控制。更进一步地，可以熟悉和学习团队协作流程。
- Github 上有许多 AI 领域内前沿的解决方案，可以开拓视野。

1.3 Github Pages 是什么？

[GitHub Pages](#) 最初用于介绍托管 GitHub 的项目。由于其空间免费稳定，非常适合用于搭建博客，因此现在逐渐被用于搭建个人博客。

1.4 为什么要搭建个人网站博客

- 个人学习和积累的输入和输出，不进行记录的话不久之后就会忘记。
- 个人成果需要更多的展示平台，借助个人博客可以将自己的成果进行展示，当发现自己博客内容越来越充实时，未尝不可不是一种满足感。
- 写作和分享是很好的学习方式。通过记笔记的形式倒逼自己学习，想要深入理解知识，记下感想和笔记是较好的方式。

2. 博客搭建

本章节主要介绍基于 [Hexo](#) 搭建个人博客网站，主要用途是记录学习笔记，具体呈现效果可以参考我的个人博客：[独孤诗人的学习驿站](#)。

首先，简单介绍一下本章搭建博客用到的框架。[Hexo](#) 是高效的静态网站生成框架，它基于 `Node.js`，简单易使用且功能强大，是搭建博客的首选方案之一。

[Hexo](#) 的具体使用方式可以查看其官网：hexo.io/zh-cn/。通过 [Hexo](#) 我们搭建个人博客网站的方式就变的异常简单，只需要直接使用 `Markdown` 语法来记录笔记，然后使用两三条命令行语句就可以将文件编译为网页文件，并上传到 [GitHub](#) 或码云等代码托管平台，这样别人就可以浏览博客内的网页了。[Hexo](#) 帮我们省去了网页源代码生成的具体细节，让我们只需要关注记笔记本身，回归到学习本身的乐趣。

总结： [Hexo](#) 产品成熟，使用简单，功能强大，有丰富的插件资源。

本章的教程大致分为三个部分：

1. [Hexo](#) 的初级搭建和部署到 [GitHub Pages](#) 上；
2. [Hexo](#) 的基本配置，更换主题，实现多终端工作；
3. [Hexo](#) 添加各种功能，如阅读量统计，访问量统计等。

2.1 Hexo 博客搭建步骤

本小节主要参考 [Hexo+Github: 个人博客网站搭建完全教程](#)。具体搭建包含如下步骤：

- 安装 `Git`
- 安装 `Node.js`
- 安装 `Hexo`
- [GitHub](#) 创建个人仓库
- 生成 `SSH` 添加到 [GitHub](#)

- 将 Hexo 部署到 GitHub
- 设置个人域名（可选）
- 发布笔记文章

2.1.1 安装 Git

为了把本地生成的网页文件上传到 GitHub 上，需要用到 Git。因此首先我们需要下载 Git 工具，[Git 官网](#)。Git 是世界上最先进的分布式版本控制系统，可以高效地对项目版本进行管理和控制。

版本控制（Revision control）是一种在开发过程中用于管理我们的文件、目录或工程等内容的修改历史，方便查看更改历史记录，以及备份和恢复以前版本的软件工程技术。

为什么要进行版本控制？

相信大家在编辑 word 文档时，常常有这样的经历。想要删除一个段落，又怕将来想恢复找不回来怎么办？一个解决办法是，把当前文件“另存为.....”一个新的 Word 文件，再接着改，改到一定程度，再“另存为.....”一个新文件，这样一直改下去，最后就出现了许多独立的 word 文档版本。

版本控制的优势：

- 实现跨区域多人协同开发
- 追踪和记载一个或多个文件的历史记录
- 组织和保护源代码和文档
- 并行开发、提高开发效率
- 追踪记录整个软件的开发过程
- 减轻开发人员的负担，节省时间，同时降低认为错误。。

这里，我们主要借助 Git 进行文件的上传和下载操作。关于 Git 的应用教程，可以查看廖雪峰老师的 Git 教程，[链接](#)

下载 Git

不同平台上的 Git 下载和使用大同小异，安装方式如下：

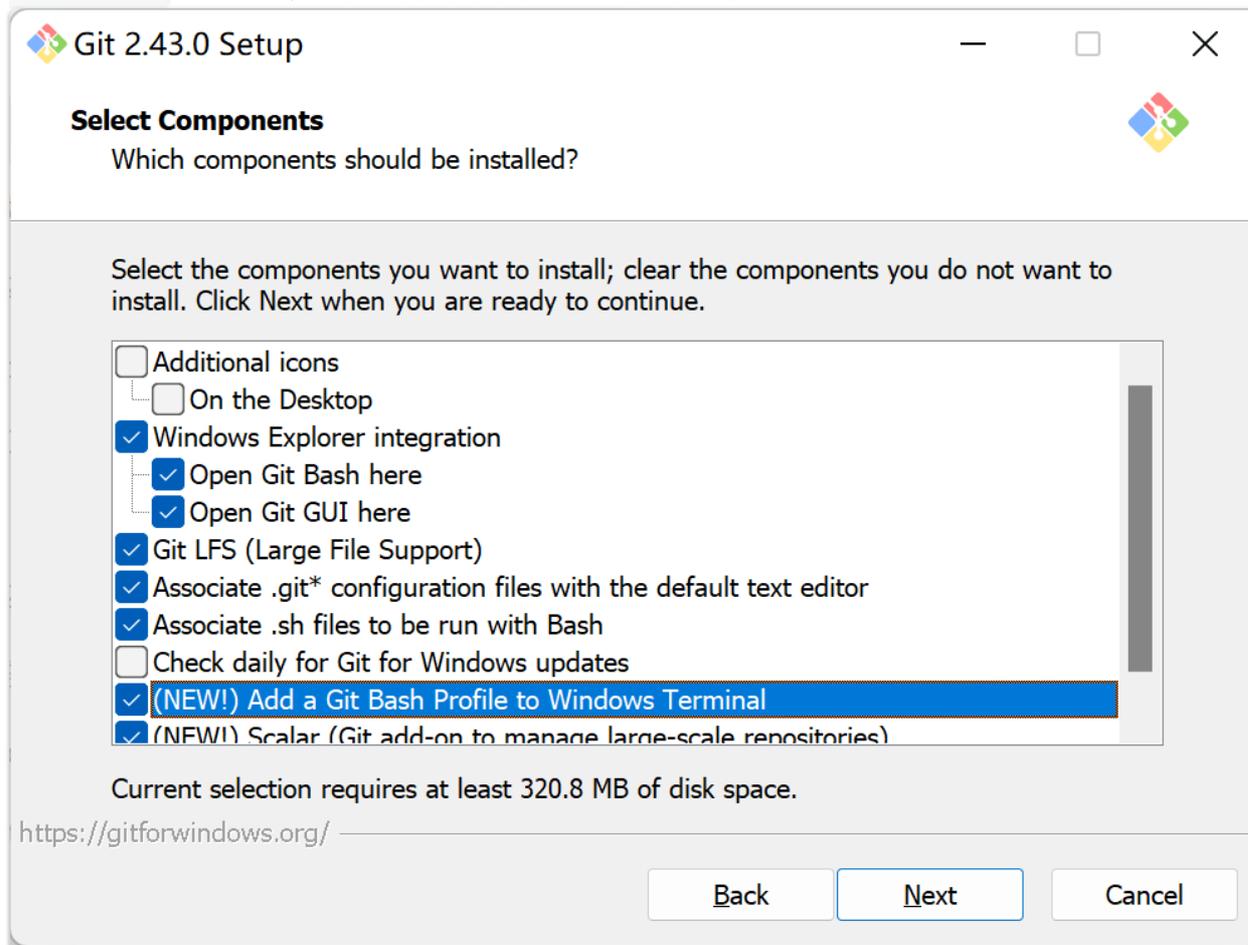
- Windows：下载并安装 [git](#)。
- Mac：使用 [Homebrew](#), [MacPorts](#) 或者下载 [安装程序](#)。
- Linux (Ubuntu, Debian)： `sudo apt-get install git-core`
- Linux (Fedora, Red Hat, CentOS)： `sudo yum install git-core`

Mac 用户

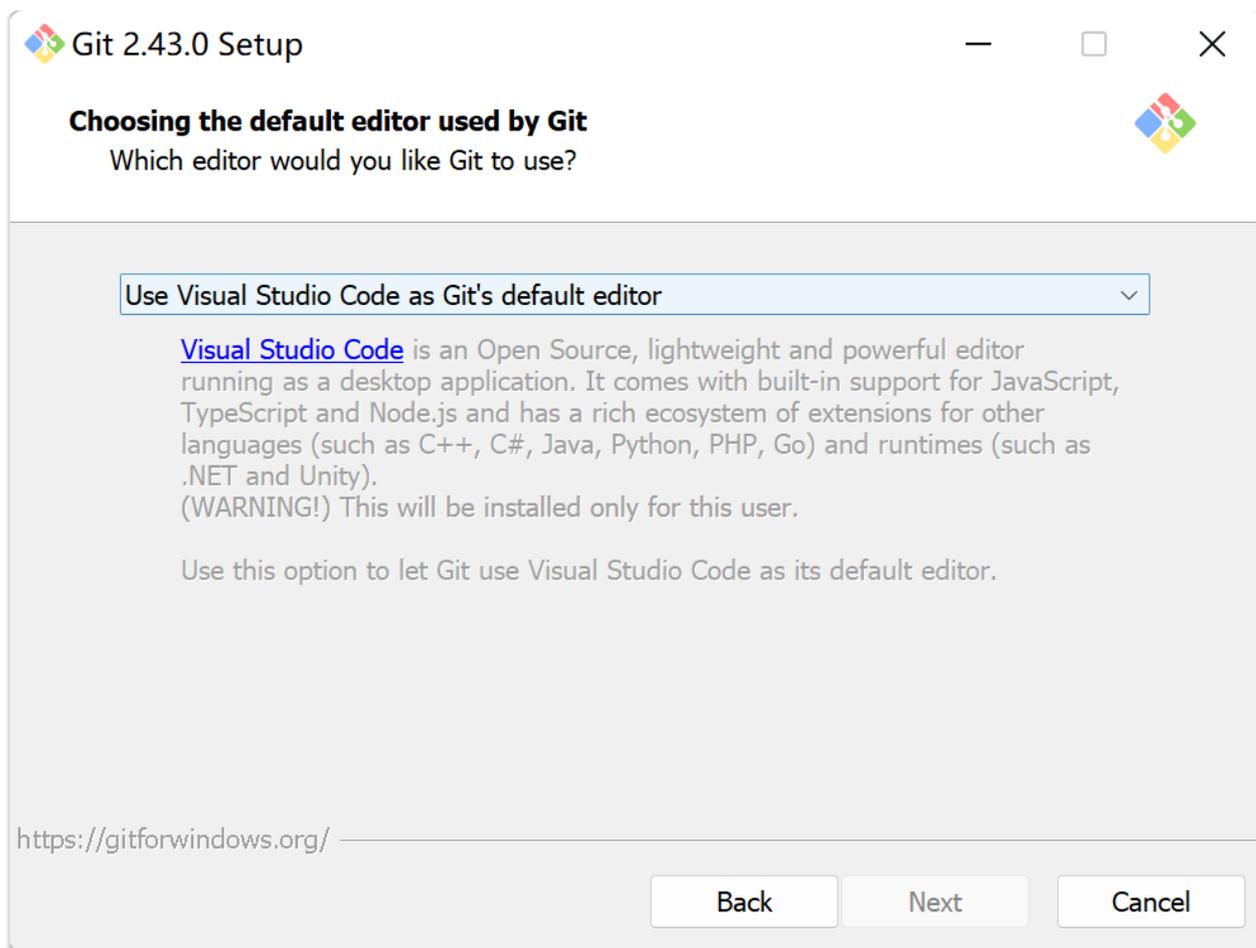
如果在编译时可能会遇到问题，请先到 App Store 安装 Xcode，Xcode 完成后，启动并进入 **Preferences -> Download -> Command Line Tools -> Install** 安装命令行工具。

此处仅以 Windows 平台的下载和安装为例，介绍 Git 的安装。安装流程如下：

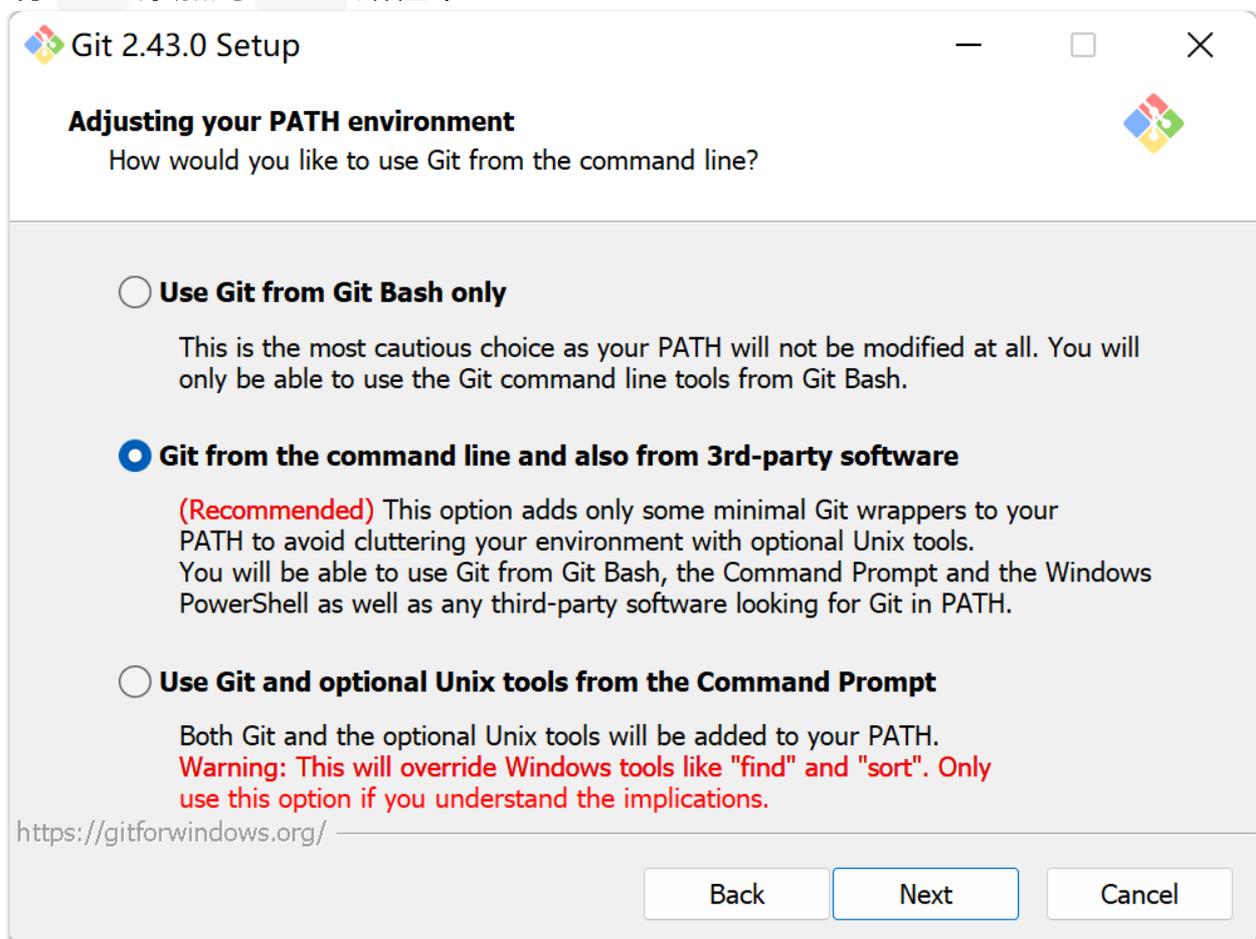
- 首先，从官网下载自己电脑对应的 64 或 32 位 Git 版本。
- 然后单击安装包进行安装，选择默认的组件进行安装，对于使用 Windows Terminal 的用户，可以将下图中的选中勾选。



- 选择 VS code 作为默认的编辑器，当然也可以选择自己有的其他编辑器作为默认选项。



- 将 Git 添加到 Path 路径中



- 其他选项选择默认的配置，一直点击 Next 就可以进行安装了。

安装完成后在命令提示符中输入 `git --version` 来查看一下版本验证是否安装成功。

```
>>> git --version  
  
git version 2.43.0.windows.1
```

2.1.2 安装 Node.js

Hexo 是基于 Node.js 编写的，所以需要安装一下这个工具和里面的 npm 工具。Node.js 为大多数平台提供了官方的 [安装程序](#)。对于 Windows 平台，下载稳定版或者最新版都可以，安装选项全部默认，一路点击 Next。最后安装好之后，按 Win+R 打开命令提示符，输入 `node -v` 和 `npm -v`，如果出现版本号，那么就安装成功了。

```
>>> node -v  
v18.18.0  
  
>>> npm -v  
9.8.1
```

注：使用 Node.js 官方安装程序时，请确保勾选 **Add to PATH** 选项（默认已勾选）。

其他平台的安装方法：

- Mac：使用 [Homebrew](#) 或 [MacPorts](#) 安装。
- Linux (DEB/RPM-based)：从 [NodeSource](#) 安装。
- 其它：使用相应的软件包管理器进行安装，可以参考由 Node.js 提供的 [指导](#)。

添加国内镜像源：

由于 npm 默认的下载源为国外网站，因此国内下载可能会较慢，这时可以使用国内的镜像网站进行加速。常用的镜像站有：

```
npm 官方原始镜像网址是：https://registry.npmjs.org/  
淘宝 NPM 镜像：https://registry.npm.taobao.org  
阿里云 NPM 镜像：https://npm.aliyun.com  
腾讯云 NPM 镜像：https://mirrors.cloud.tencent.com/npm/  
华为云 NPM 镜像：https://mirrors.huaweicloud.com/repository/npm/  
网易 NPM 镜像：https://mirrors.163.com/npm/
```

相对而言，腾讯、华为和阿里郎的镜像站基本比较全。以华为云镜像站为例，我们设置华为云镜像源加速 `NPM`。

- 查看当前的镜像：

```
>>> npm config get registry
https://registry.npmjs.org/
```

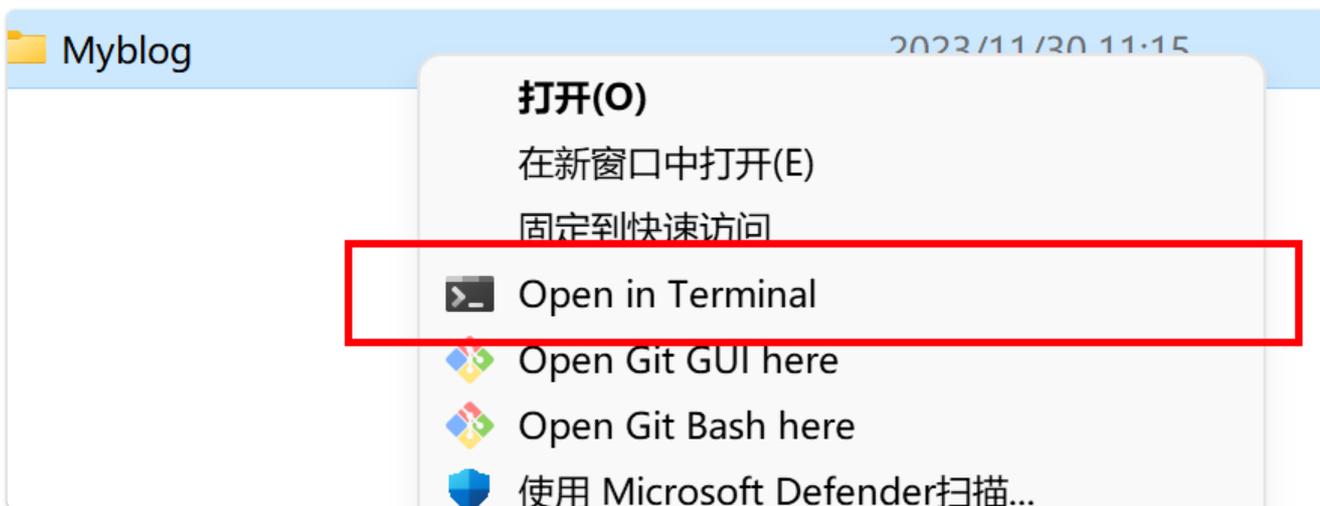
- 设置华为云镜像站加速 `NPM`：

```
>>> npm config set registry
https://mirrors.huaweicloud.com/repository/npm/

>>> npm config get registry # 查看当前镜像
https://mirrors.huaweicloud.com/repository/npm/
```

2.1.3 安装 Hexo

安装完 `Git` 和 `Node.js` 后，就可以安装 `Hexo` 了。在此之前，可以先创建一个博客文件夹，如 `Myblog`，用于存放自己的博客文件，例如我的项目路径为 `Z:\Demo\Python\team-learning-program\WhaleBlog\Myblog`。然后在此文件夹下打开命令行：



如果右键中没有该选项，也可以选择直接打开 `Terminal` 或 `Git bash`，然后 `cd` 到该路径：

```
cd "Z:\Demo\Python\team-learning-program\WhaleBlog\Myblog"
```

Note: 由于在安装 `Git` 时，我们已经将其添加进了 `Terminal` 和环境 `PATH` 中，因此我们可以直接在 `Terminal` 中直接调用 `Git`。如果读者电脑中没有安装 `Terminal`，也可以直接使用 `Git bash` 运行本教程中提及的命令行语句。

安装和搭建博客步骤

- 安装 `Hexo`。从 `Terminal` 中定位到项目路径后，输入 `npm install -g hexo-cli` 安装 `Hexo`。可能会有几个报错提示，直接无视就好。

```
>>> npm install -g hexo-cli

added 54 packages in 21s

14 packages are looking for funding
  run `npm fund` for details
npm notice
npm notice New major version of npm available! 9.8.1 → 10.2.4
npm notice Changelog:
https://github.com/npm/cli/releases/tag/v10.2.4
npm notice Run npm install -g npm@10.2.4 to update!
npm notice
```

- 查看是否安装成功。输入 `hexo -v` 查看是否安装成功。

```
>>> hexo -v

hexo-cli: 4.3.1
os: win32 10.0.22000
node: 18.18.0
acorn: 8.10.0
ada: 2.6.0
ares: 1.19.1
brotli: 1.0.9
cldr: 43.1
icu: 73.2
llhttp: 6.0.11
modules: 108
napi: 9
nghttp2: 1.55.0
nghttp3: 0.7.0
ngtcp2: 0.8.1
openssl: 3.0.10+quic
```

```
simdutf: 3.2.14
tz: 2023c
undici: 5.22.1
unicode: 15.0
uv: 1.46.0
uvwasi: 0.0.18
v8: 10.2.154.26-node.26
zlib: 1.2.13.1-motley
```

- 初始化。接下来在项目文件夹下输入 `hexo init` 进行初始化。

```
>>> hexo init
INFO Cloning hexo-starter https://github.com/hexojs/hexo-
starter.git
INFO Install dependencies
INFO Start blogging with Hexo!
```

- 安装必要的组件。接下来输入 `npm install` 安装必要的组件。

```
>>> npm install
added 1 package in 2s

28 packages are looking for funding
run `npm fund` for details
```

至此，`Hexo` 和博客的基本框架就算搭建成功了。此时，项目文件夹 `Myblog` 中包含如下文件：

- `node_modules`：依赖包
- `public`：存放生成的页面，可以借助 `hexo s` 在本地直接查看
- `scaffolds`：生成文章的一些模板
- `source`：用来存放笔记的 `md` 文件
- `themes`：主题文件夹
- `_config.yml`：博客的配置文件

这样本地的网站文件就配置好了，接下来可以先输入 `hexo g` 生成静态网页：

```
>>> hexo g

INFO Validating config
```

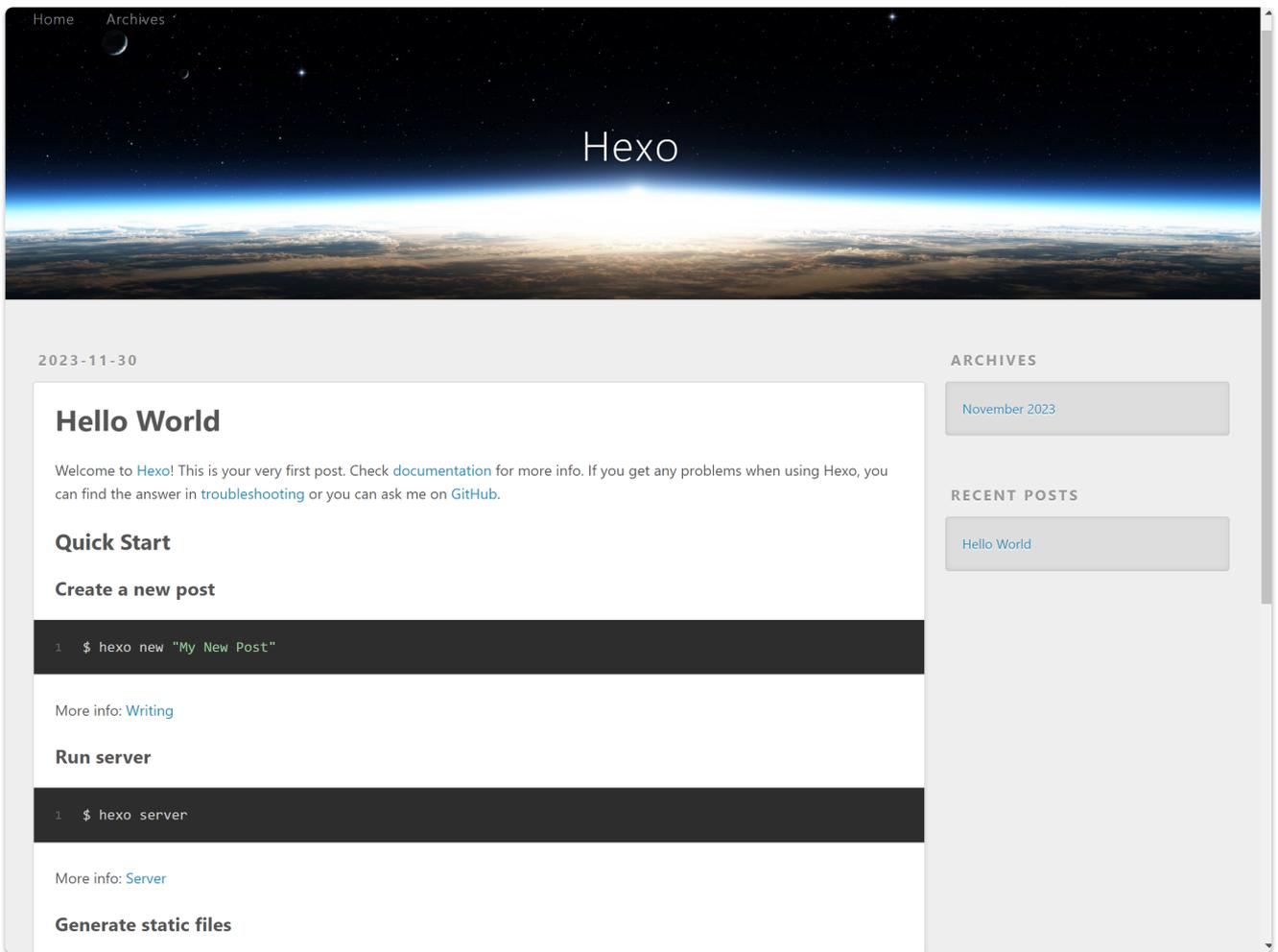
```
INFO Start processing
INFO Files loaded in 444 ms
INFO Generated: archives/index.html
INFO Generated: archives/2023/index.html
INFO Generated: archives/2023/11/index.html
INFO Generated: index.html
INFO Generated: fancybox/jquery.fancybox.min.css
INFO Generated: css/style.css
INFO Generated: js/script.js
INFO Generated: fancybox/jquery.fancybox.min.js
INFO Generated: js/jquery-3.6.4.min.js
INFO Generated: css/images/banner.jpg
INFO Generated: 2023/11/30/hello-world/index.html
INFO 11 files generated in 942 ms
PS Z:\Demo\Python\team-learning-prog
```

然后输入 `hexo s` 打开本地服务器：

```
>>> hexo s # hexo server

INFO Validating config
INFO Start processing
INFO Hexo is running at http://localhost:4000/. Press Ctrl+C to
stop.
```

用浏览器打开 <http://localhost:4000/>，就可以得到如下的博客网页效果啦：

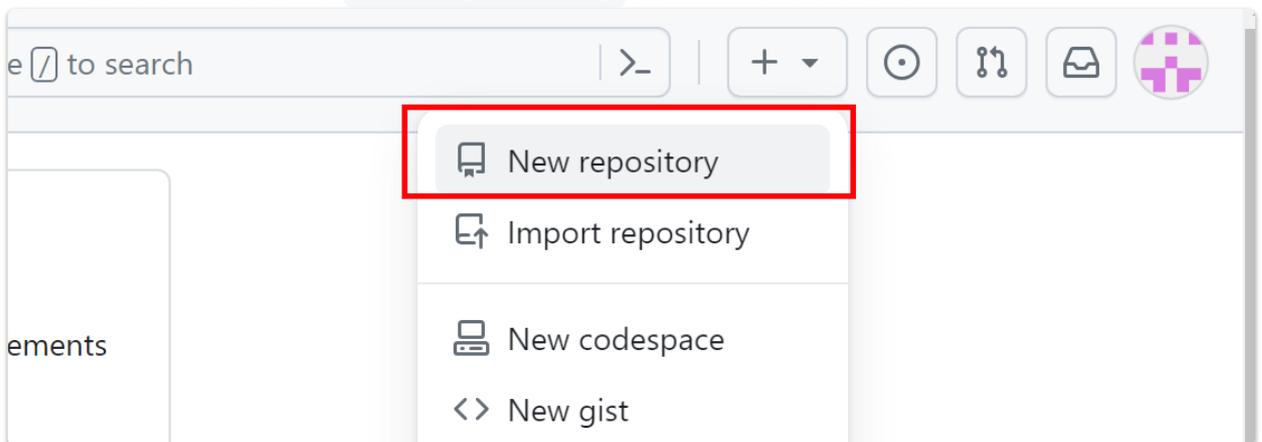


然后在 terminal 中按 `ctrl+c` 关闭本地服务器。

2.1.4 注册 GitHub 账号创建个人仓库

接下来，为了能让我们的博客可以在不同终端上通过浏览器访问，我们需要将其搭载在 GitHub 上。

- 申请 GitHub 账号。首先在 [GitHub 官网](#) 创建一个账号。
- 创建仓库。之后点击 `New repository` 创建一个项目仓库：



- 项目初始化。仓库项目命名需要跟用户名保持一致，并添加后缀 `.github.io`，同时记得勾选 `Add README` 选项。如下图所示：

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Required fields are marked with an asterisk (*).

Owner *

 blogtest2023

Repository name *

blogtest2023.github.io

✔ blogtest2023.github.io is available.

Great repository names are short and memorable. Need inspiration? How about [scaling-happiness](#) ?

Description (optional)



Public

Anyone on the internet can see this repository. You choose who can commit.



Private

You choose who can see and commit to this repository.

Initialize this repository with:

Add a README file

This is where you can write a long description for your project. [Learn more about READMEs.](#)

Add .gitignore

.gitignore template: None

Choose which files not to track from a list of templates. [Learn more about ignoring files.](#)

Choose a license

License: None

A license tells others what they can and can't do with your code. [Learn more about licenses.](#)

This will set  main as the default branch. Change the default name in your [settings](#).

 You are creating a public repository in your personal account.

Create repository

注：需要创建一个与用户名同名的仓库，并且注意后缀格式 `github.io`。只有这样，在进行部署 `Github Page` 时才会被正确识别。初始的访问链接是 `http://username.github.io`（后续也可以绑定自己的个人域名）。例如此处用于展示的连接就是：<http://blogtest2023.github.io>。

2.1.5 生成 SSH 添加到 GitHub

接下来，我们需要生成 SSH 并将它添加到 GitHub，连接 Github 与本地。这样就可以借助 `Git` 或 `Hexo` 将本地的文件自动上传到个人的 `GitHub` 仓库中。

- 配置用户名和邮箱。右键单击项目文件夹打开 `Terminal`，然后输入下面命令：

```
git config --global user.name "yourname"
git config --global user.email "youremail"
```

这里的 `yourname` 和 `youremail` 表示刚刚创建的 GitHub 用户名和邮箱，比如本例中：

```
>>> git config --global user.name 'blogtest2023'
>>> git config --global user.email 'keqinzan@163.com'
```

- 查看配置。可以使用 `git config`，检查一下是否配置成功

```
>>> git config user.name
blogtest2023
>>> git config user.email
keqinzan@163.com
```

- 配置 SSH。一直点击回车就行。

```
>>> ssh-keygen -t rsa -C "youremail"
```

```
PS Z:\Demo\Python\team-learning-program\WhaleBlog\Myblog> ssh-keygen -t
rsa -C "keqinzan@163.com"
Generating public/private rsa key pair.
Enter file in which to save the key (C:\Users\yangsu\.ssh/id_rsa):
Created directory 'C:\Users\yangsu\.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in C:\Users\yangsu\.ssh/id_rsa.
Your public key has been saved in C:\Users\yangsu\.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:aMqaBLPocoSOT5ZC5LbeKnx4Ec5bCgl5bvZIDp0BRUk keqinzan@163.com
The key's randomart image is:
+---[RSA 3072]-----+
| oE.
| o
|. +
|B .. .
|+X+ . o S
|O*B* +
|BXBoB
|*O=B.
|o*B.
+----[SHA256]-----+
```

代码会生成一个 `.ssh`，如笔者的生成文件夹保存在 `C:/Users/yangsu/.ssh/` 中。

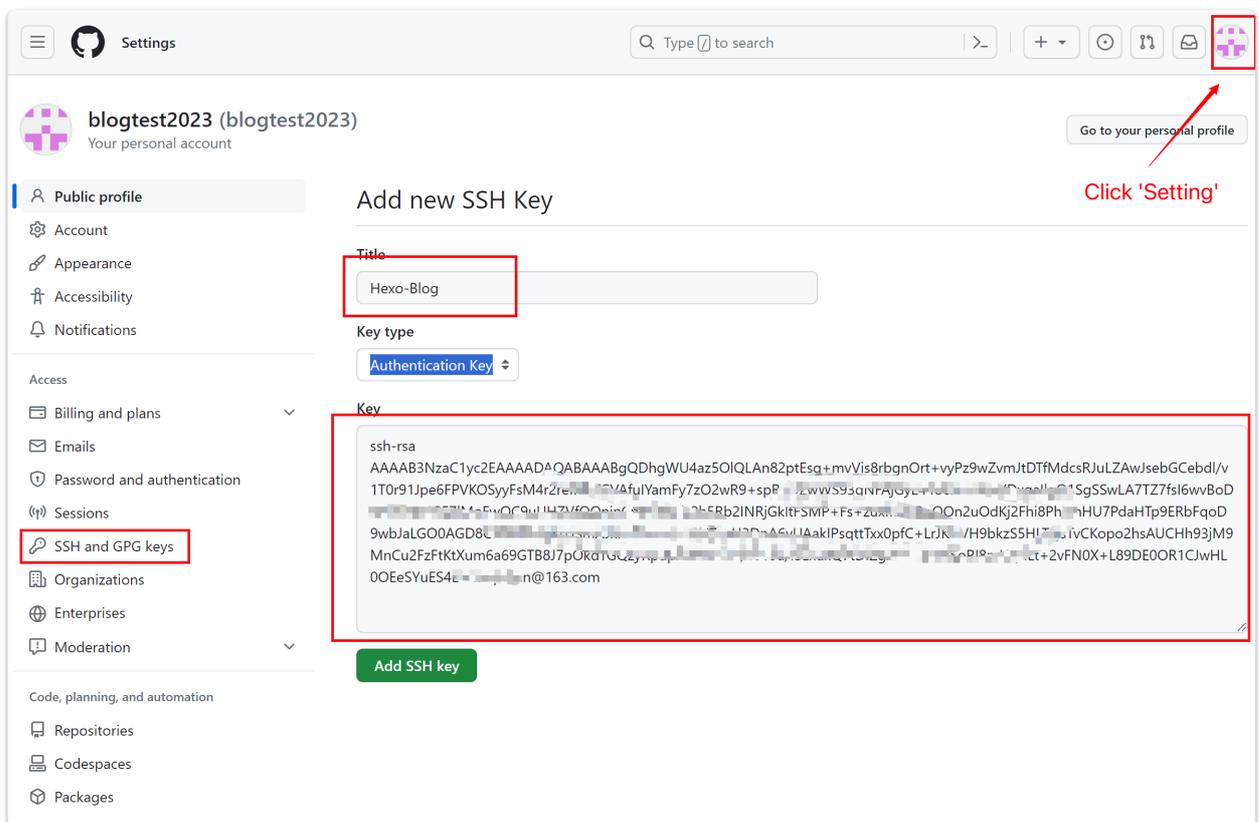
注：SSH 指密钥。`id_rsa` 是本台电脑的私人密钥，需要保存好，而 `id_rsa.pub` 是公共密钥，存放在 `GitHub` 中，这样当链接 `GitHub` 自己的账户时，会根据公钥和私钥进行匹配。当匹配成功时，才能顺利地通过 `git` 将本地文件上传到 `GitHub` 中。

- 查看 SSH 密钥。

```
>>> cat ~/.ssh/id_rsa.pub
```

```
PS Z:\Demo\Python\team-learning-program\WhaleBlog\Myblog> cat ~/.ssh/id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQGDgE=501QAn82ptEsg+mvVis8rbgnOrt+vyPz9wZvmJdTfMdcRJuLZAwJsebGCEbd/v1T0r91Jpe6FPVKOSyFsm4r2reIMxTGM4fulYamFy7zO2wR9+spR2wwS93qrFayGyca+100n2uOdKj2Fhi8PhHU7PdaHTp9ERbfqoD9wbJalG00AGD8L89DE0OR1CJWHL00EeSYuES4E= ke@163.com
```

- 创建 SSH key。将输出的内容复制，然后打开 `GitHub` 的设置，再点击 `SSH and GPG keys`，新建一个 `SSH`，`Title` 可以随便取，然后将刚刚复制的 `id_rsa.pub` 的信息复制到 `Key` 中：



- 验证是否链接成功。在 `Terminal` 中输入 `ssh -T git@github.com`。

```
>>> ssh -T git@github.com
Hi blogtest2023! You've successfully authenticated, but GitHub
does not provide shell access.
```

出现的是 GitHub 的用户名，则说明配置成功。

2.1.6 将 Hexo 部署到 GitHub

在做好上面的准备工作之后，可以开始将本地的 Hexo 博客项目文件上传到 GitHub 中，即将 Hexo 部署到 GitHub 中。

- 修改配置文件。首先打开博客根目录下的 `_config.yml` 配置文件，修改最后一行 `deploy` 下的配置。

```
deploy:
  type: git
  repository:
  https://github.com/blogtest2023/blogtest2023.github.io
  branch: master
```

此处，`repository` 修改为自己的 `github` 项目地址即可，`type` 为 `git`。此处设置的目的是告诉工具，将生成的本地静态网页文件通过 `git` 上传到给定链接仓库的 `master` 分支下。

- 安装 `deploy-git`。在进行部署时，需要用到 `git` 工具将文件上传到仓库中，需要用到 `deploy-git` 工具，也就是部署的命令。可以使用 `npm` 进行安装

```
>>> npm install hexo-deployer-git --save
added 9 packages in 6s
28 packages are looking for funding
run `npm fund` for details
```

- 部署。接下来，我们就可以使用 `Hexo` 指令轻松地将本地的博客文件部署到 GitHub 上了。

```
hexo clean
hexo generate # hexo server
hexo deploy
```

其中，`hexo clean` 用于清理本地的缓存，即之前生成的静态网页文件。`hexo generate` 或 `hexo g` 则是生成本地的静态网页文件，在生成了本地的静态文件之后，也可以使用 `hexo server` 或 `hexo s` 在本地进行直接查看和预览。`hexo deploy` 或 `hexo d` 则是进行部署，将生成的文件上传到 `GitHub` 中。

关于更多 `Hexo` 的指令，可以借助 `hexo -h` 命令查看帮助文档。

```
>>> hexo -h
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/
```

注：首次部署可能会需要输入 `GitHub` 的账号和密码进行验证，或者在浏览器登录验证。

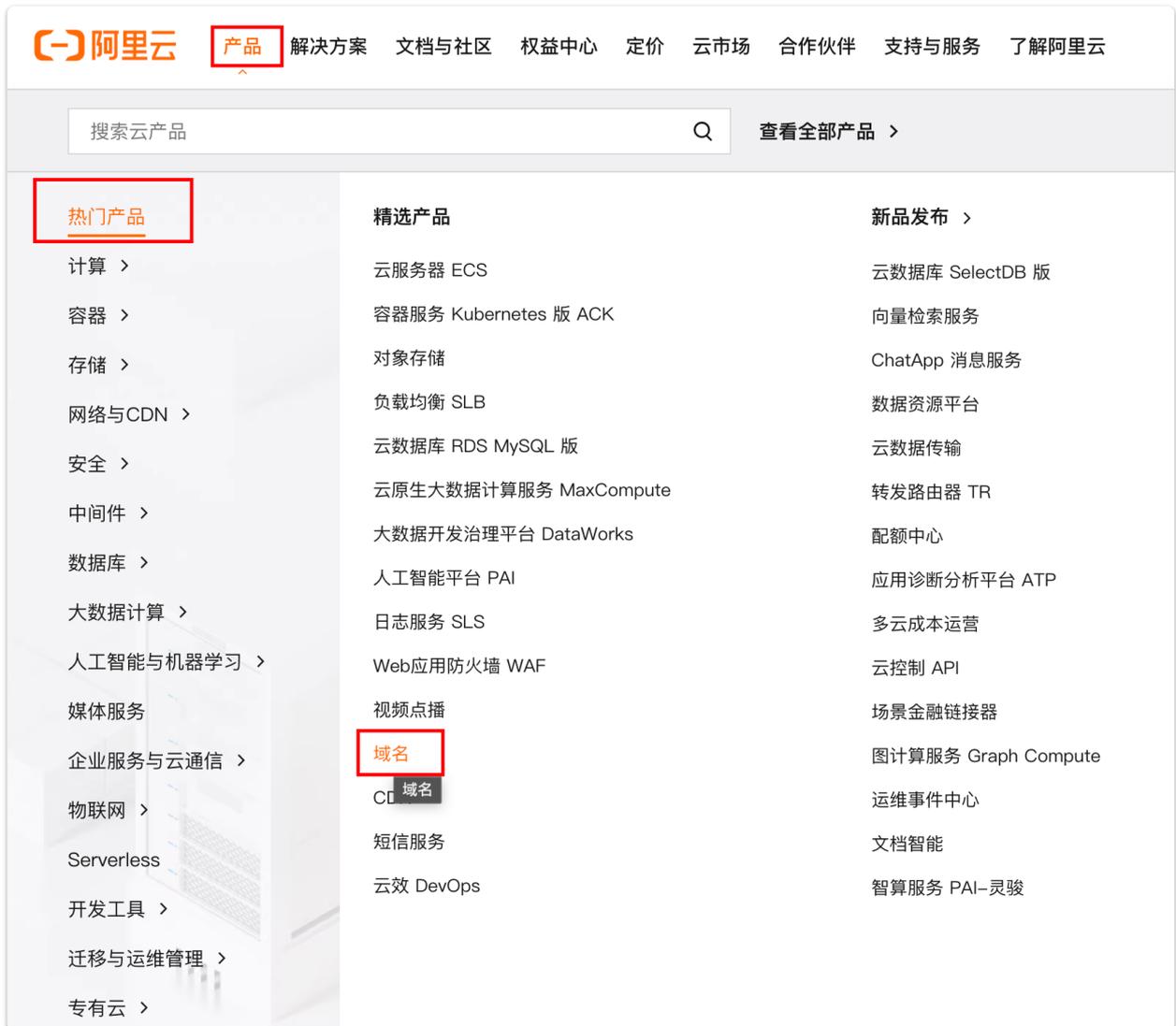
至此，基于 Hexo 和 GitHub 的个人博客网站就搭建好了。此时可以访问专属的域名 `https://<username>.github.io/` 进行访问，比如教程中的域名：<https://blogtest2023.github.io/>。

2.1.7 配置个人域名 (Optional)

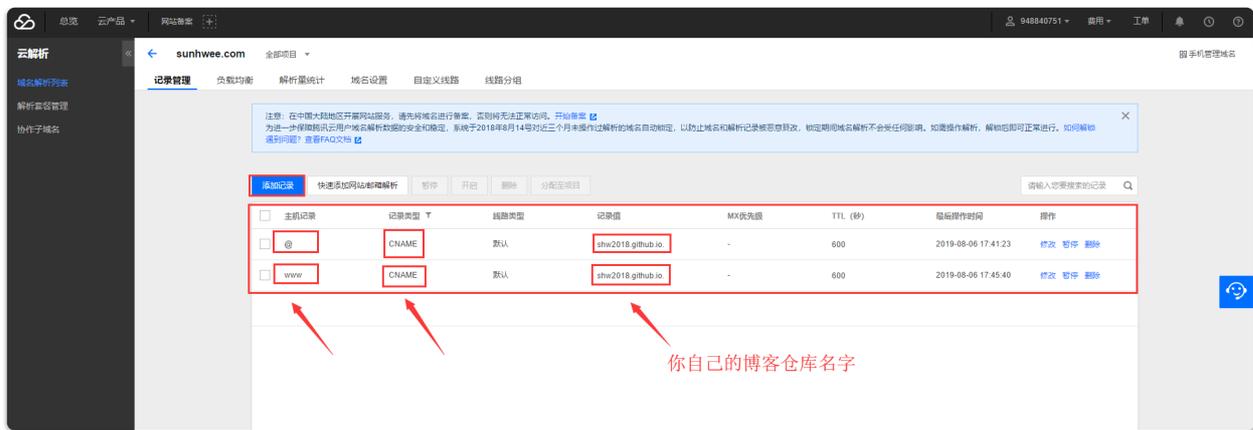
通过上述步骤，我们配置完成了个人网站，地址为 `https://<username>.github.io/`。除了官方给定的网址之外，我们也可以根据自己喜好设置个人域名。

注：域名购买是需要付费的，对于初学者而言没有必要在这上面进行花销。此处只做普及说明如何为博客网站配置个人域名。没有域名配置需求的读者可以先跳过此部分。

- 购买域名。首先需要购买一个专属域名，国内主流的运营商都可以进行购买，如腾讯云、百度云、阿里云和华为云。此处以阿里云为例。



- 配置域名信息。购买域名之后进入域名控制台，找到刚购买的域名进行云解析，添加两条解析记录。



(图源自 <https://www.cnblogs.com/shwee/p/11421156.html>)

- 接下来打开 GitHub 博客项目，点击 setting，拉到下面的 Custom domain 填写自己的域名，保存，就可以使用购买的域名进行博客访问了，比如笔者的博客 <https://www.yangsuoly.com/>。