

前言

灵感来自K神的krahets/hello-algo: 《Hello 算法》：动画图解、一键运行的数据结构与算法教程。支持 Python, Java, C++, C, C#, JS, Go, Swift, Rust, Ruby, Kotlin, TS, Dart 代码。简体版和繁体版同步更新, English version ongoing

于是想着能不能把github仓库存储的文件作为博客文章阅读

演示站点

📖 BAOER の BLOG 📖

效果展示



部署准备

1、本地下载typora或者其他markdown编辑器

[Typora for windows — 测试版发布](#) --- [Typora for windows — beta version release](#)

2、一个github账户

3、本地安装git，用于管理仓库

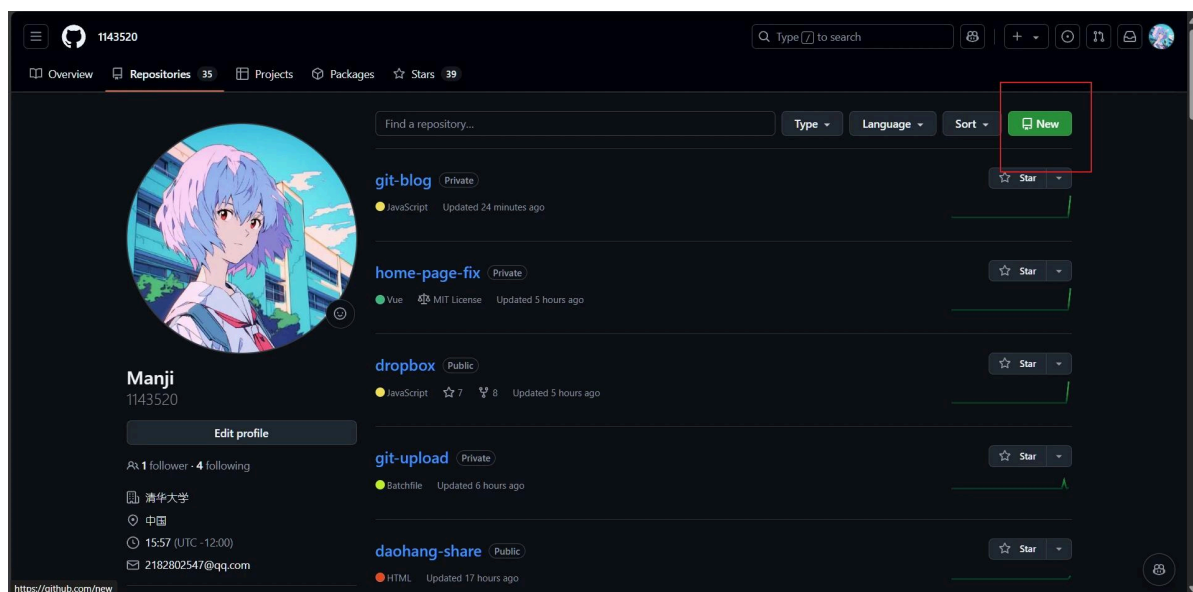
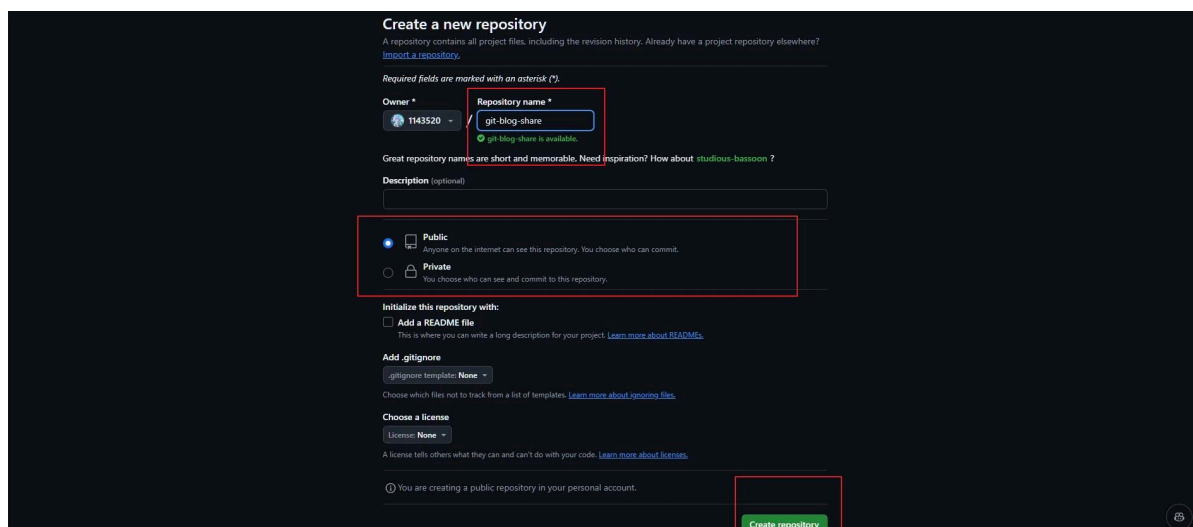
3、一个cloudflare免费账号

4、一个B站账号

保姆教程

1、新建github仓库

访问 <https://github.com/你的github用户名?tab=repositories>

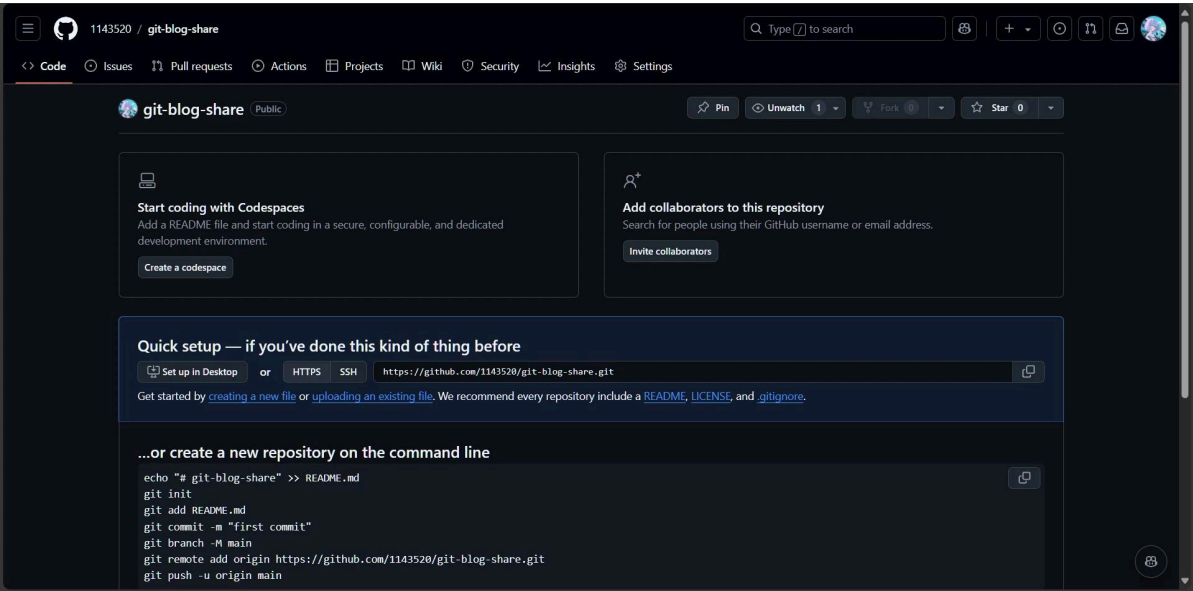


填写仓库名字

public还是Private看个人喜好，自己看就使用Private，后面加密码验证，分享给别人就public

点击创建

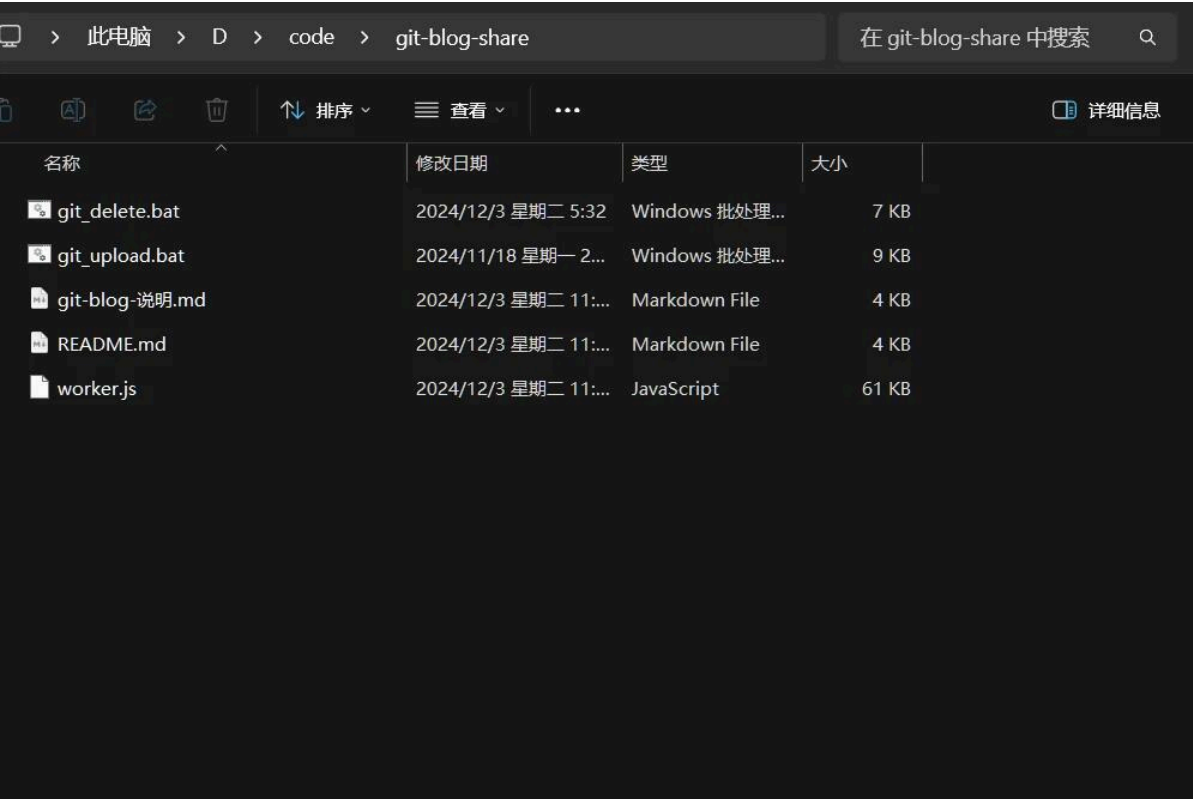
停留在这个界面



2、本地创建文件夹

随便找个全英文路径文件夹作为上传文件夹，不要使用你的typora的笔记文件夹

把你要上传的.md文件放到这个文件夹



从[1143520/git-blog-share](https://github.com/1143520/git-blog-share)下载 git_upload.bat 和 git_delete.bat

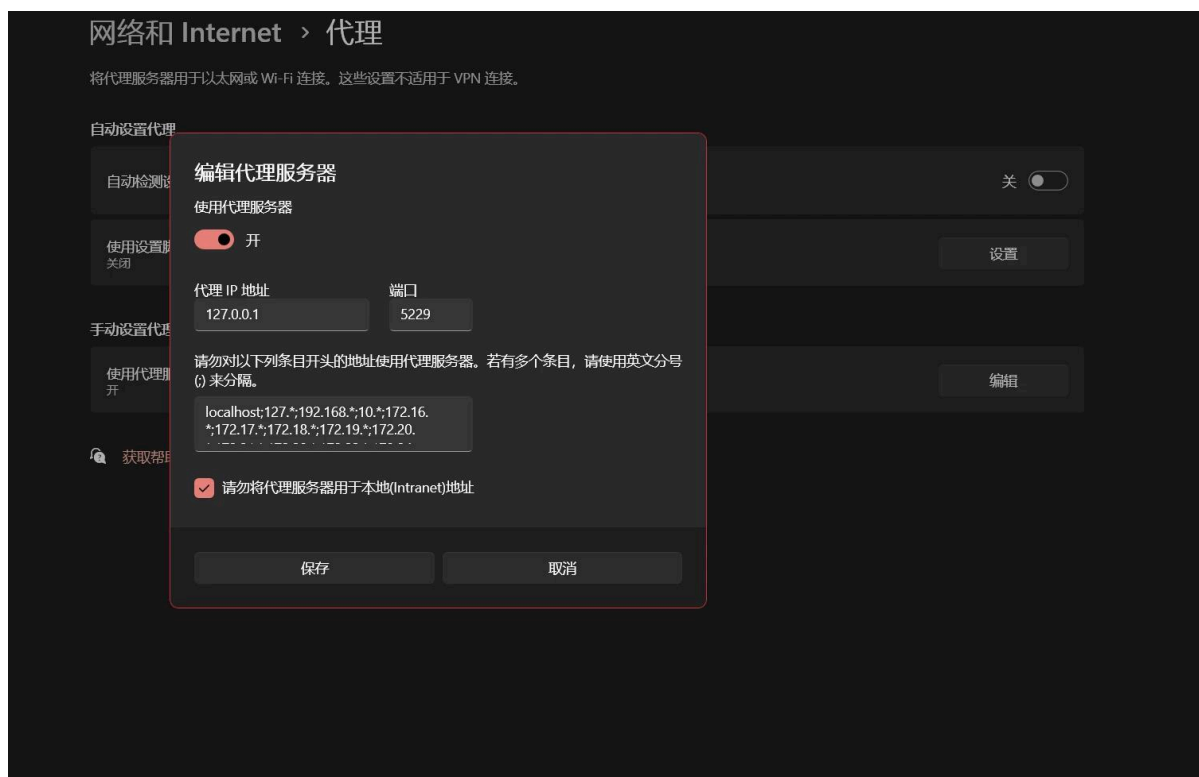
用于管理仓库文件

双击运行git_upload.bat

```
GitHub Auto Upload Tool
[INFO] Starting GitHub Auto Upload Tool...
[INFO] Current directory: D:\code\git-blog-share
请按任意键继续. . .
[INFO] Checking Git installation...
[INFO] Creating config folder...
[INFO] Creating .gitignore file...
[INFO] Configuring git exclude file...
[INFO] First time proxy setup...
Enter proxy port (default 7890):
```

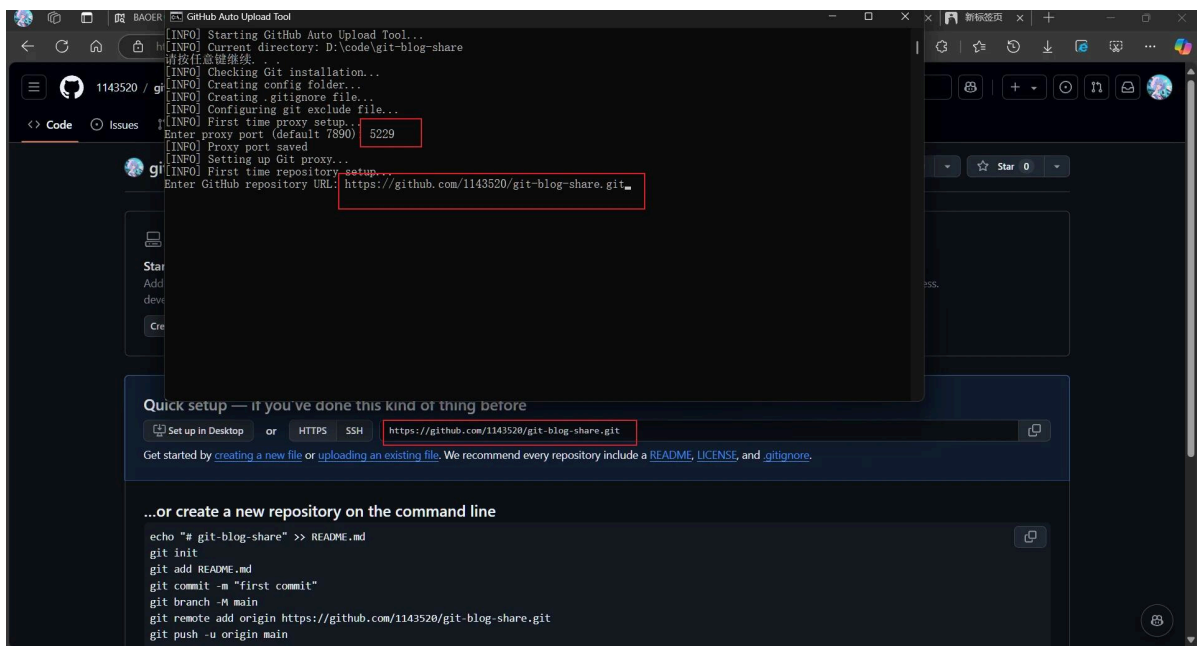
因为 github 国内访问不畅，你知道的呀，需要开启代理

开启代理确保能够正常访问后,在设置中找到自己的代理端口，比如我的是5229



在git_upload.bat 填入5229之后回车

复制仓库的git地址填入之后回车

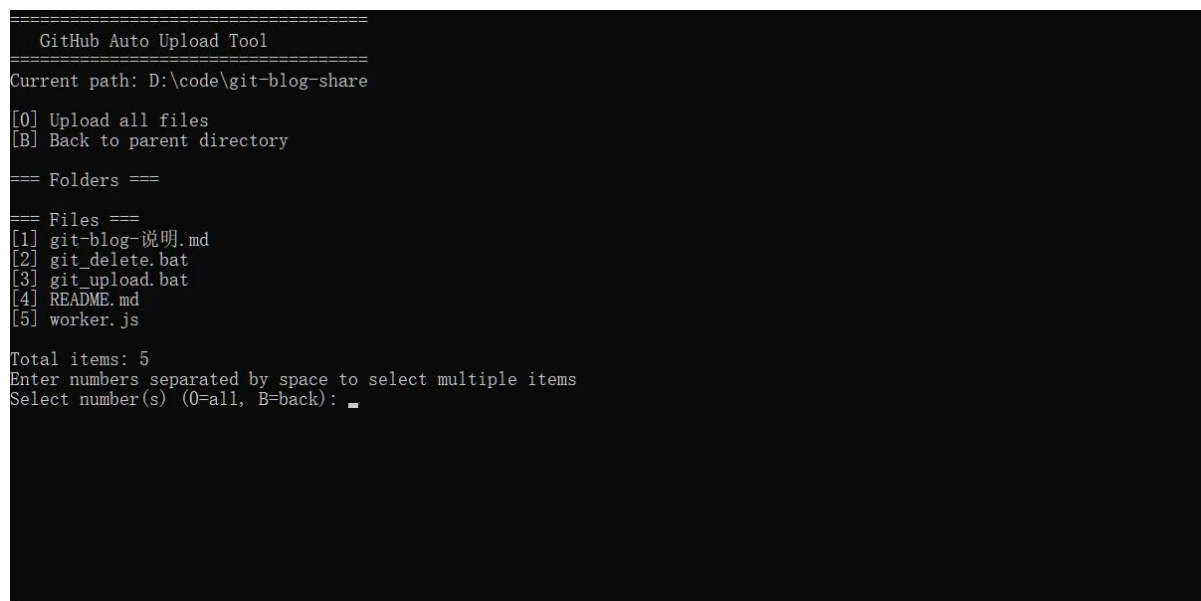


选择需要上传的文件编号，0为全部上传，和git是一样的，只是增量上传

比如1 3 5就是上传1、3、5这几个文件

回车上传，自动main分支，确保本地安装了git，而且登陆了github账户，

详细请参考[手把手教你用git上传项目到GitHub（图文并茂，这一篇就够了）](#)，相信你一定能成功！！ - 知乎



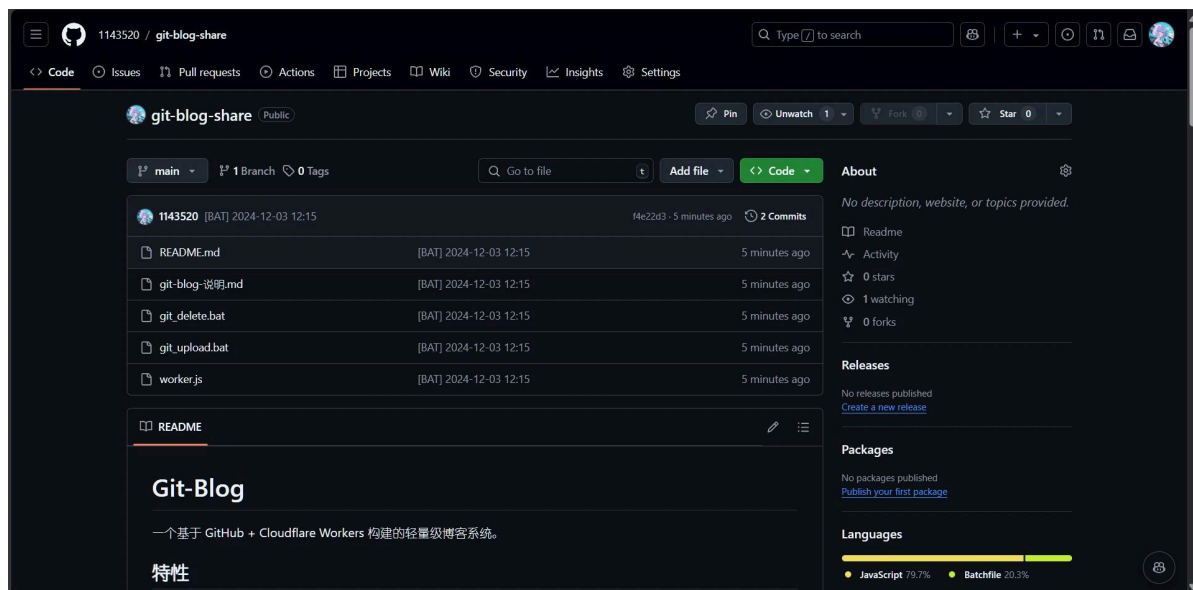
```
Total items: 5
Enter numbers separated by space to select multiple items
Select number(s) (0=all, B=back): 3 4 5
[INFO] Adding selected files...
[INFO] Added: git_upload.bat
[INFO] Added: README.md
[INFO] Added: worker.js
[INFO] Committing changes...
[main f4e22d3] [BAT] 2024-12-03 12:15
3 files changed, 2412 insertions(+)
create mode 100644 README.md
create mode 100644 git_upload.bat
create mode 100644 worker.js
[INFO] Pushing to GitHub...
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 12 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 15.05 KiB | 2.15 MiB/s, done.
Total 4 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/1143520/git-blog-share.git
   9990c65..f4e22d3  main -> main
branch 'main' set up to track 'origin/main'.

=====
[SUCCESS] Upload completed
=====
Repository: https://github.com/1143520/git-blog-share.git
请按任意键继续. . .
```

SUCCESS即为成功

回到仓库[1143520/git-blog-share](https://github.com/1143520/git-blog-share)刷新

可以看到已经有了文件



本地生成了git的默认配置和bat的配置文件夹，注意不要误删

| 名称 | 修改日期 | 类型 | 大小 |
|--------------------|----------------------|----------------|-------|
| git_delete.bat | 2024/12/3 星期二 5:32 | Windows 批处理... | 7 KB |
| git_upload.bat | 2024/11/18 星期一 2... | Windows 批处理... | 9 KB |
| git-blog-说明.md | 2024/12/3 星期二 11:... | Markdown File | 4 KB |
| README.md | 2024/12/3 星期二 11:... | Markdown File | 4 KB |
| worker.js | 2024/12/3 星期二 11:... | JavaScript | 61 KB |
| .git-upload-config | 2024/12/3 星期二 12:... | 文件夹 | |
| .git | 2024/12/3 星期二 12:... | 文件夹 | |

git_delete.bat 是用于删除仓库某个文件，本地不变，使用方法同理，大伙可以用GPT把二者合成一个bat

3、配置cloudflare的workers

进入CF点击workers and pages进行创建

Workers and Pages

概述

KV

Durable Objects

工作流 Beta

Queues

D1 SQL 数据库

Hyperdrive

浏览器呈现

概述

使用 Workers 和 Pages 构建和部署无服务器功能、站点和全栈应用程序。阅读 [Workers 文档](#) 和 [Pages 文档](#)，了解更多信息。

搜索应用程序

搜索

筛选方式

全部显示

排序方式

上次修改时间

git-blog

访问

过去 24 小时

请求

错误数量

中值 CPU 时间

中值挂钟时间

2.2k

846

2.3 ms

639.2 ms

1' main

c3987cc

13 小时前

查看详细信息

dropbox-test

访问

1143520/dropbox

dropbox-test-88y.pages.dev, liuyan.1143520.xyz

制作

1' main

16a8f9f

6 小时前

查看详细信息

NEW VIDEO TUTORIAL

Workers 201 - 完整课程

了解如何使用 Cloudflare Workers (API, Workers KV、D1、SQLite、Prisma)构建有状态应用。

观看视频

帐户详细信息

免费

升级计划

12:00凌晨 周二 (UTC) - 4:26凌晨 周二 (UTC)

今天的请求 830 / 100,000

今天的可观测性事件 0 / 200,000

帐户 ID 29x345b954481170a233c2a7a5759227

管理 API 令牌

子域 2182802547lx.workers.dev

计算设置

[← 概述](#)

创建应用程序

使用 Workers 构建无服务器功能。使用 Pages 部署网站和全栈应用程序。阅读 [Workers 文档](#) 和 [Pages 文档](#)，了解更多信息。

Workers Pages

[使用 CLI 创建](#)



创建一个“Hello World”Worker 并在全球范围内部署

创建 Worker

从模板开始

全部 新手 存储 AI

A/B Test

Try different variations of your website across your visitors.

Common Worker Examples

Build from a collection of simple Worker use cases.

Worker + D1 Database

Cloudflare's native serverless SQL database.

Image Classification App

Identify and label objects found in images.

自定义名称然后部署

[← 创建应用程序](#)

"Hello World" Worker

创建“Hello World”Worker

git-blog-share

您的 Worker 将被部署到: <https://git-blog-share.2182802547h.workers.dev>

```
worker.js

/**
 * Welcome to Cloudflare Workers! This is your first worker.
 *
 * - Run "npm run dev" in your terminal to start a development server
 * - Open a browser tab at http://localhost:8787/ to see your worker in action
 * - Run "npm run deploy" to publish your worker
 *
 * Learn more at https://developers.cloudflare.com/workers/
 */

export default {
  async fetch(request, env, ctx) {
    return new Response('Hello World!');
  },
};
```

取消 部署

继续处理项目



编辑添加以下变量

| 变量名 | 必填 | 说明 |
|----------------|----|--|
| GITHUB_TOKEN | 是 | GitHub Personal Access Token，用于访问仓库内容 |
| GITHUB_OWNER | 是 | GitHub 用户名或组织名 |
| GITHUB_REPO | 是 | 博客内容所在的仓库名称 |
| ADMIN_PASSWORD | 否 | 博客管理员密码，仅在 enablePasswordProtection 为 true 时需要 |

GITHUB_TOKEN在登录github后访问

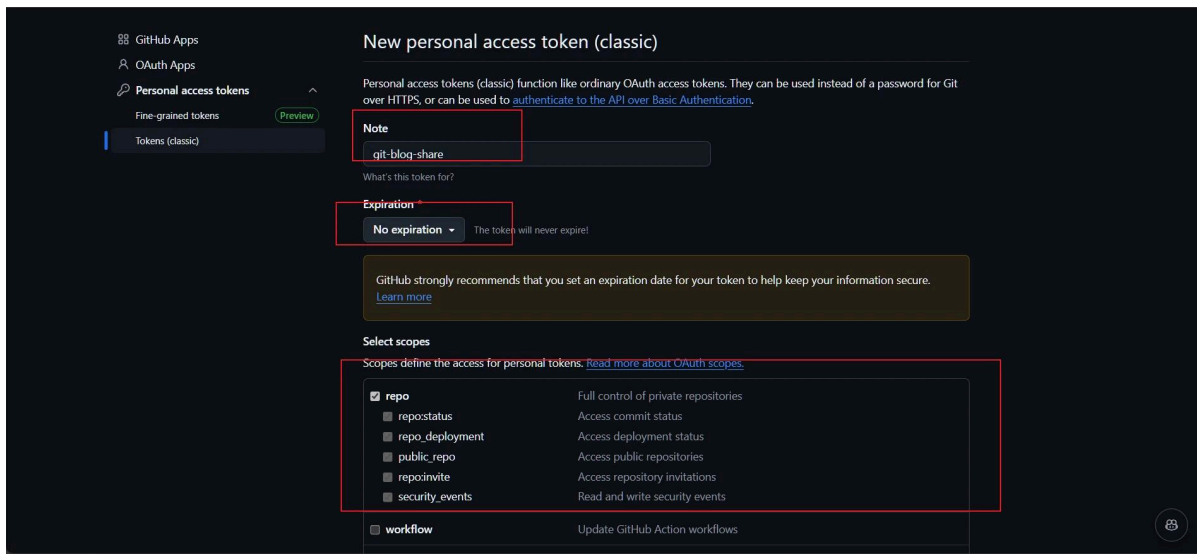
[Personal Access Tokens \(Classic\)](#) 手动生成，需要repo也就是仓库查看的权限（公开仓库限额太少，使用token可以增加限额，也可以访问私库）

点击 **Generate new token**

选择Generate new token (classic)

完成验证（如果有）

名称随意，过期时间自己把握，repo一定要打勾



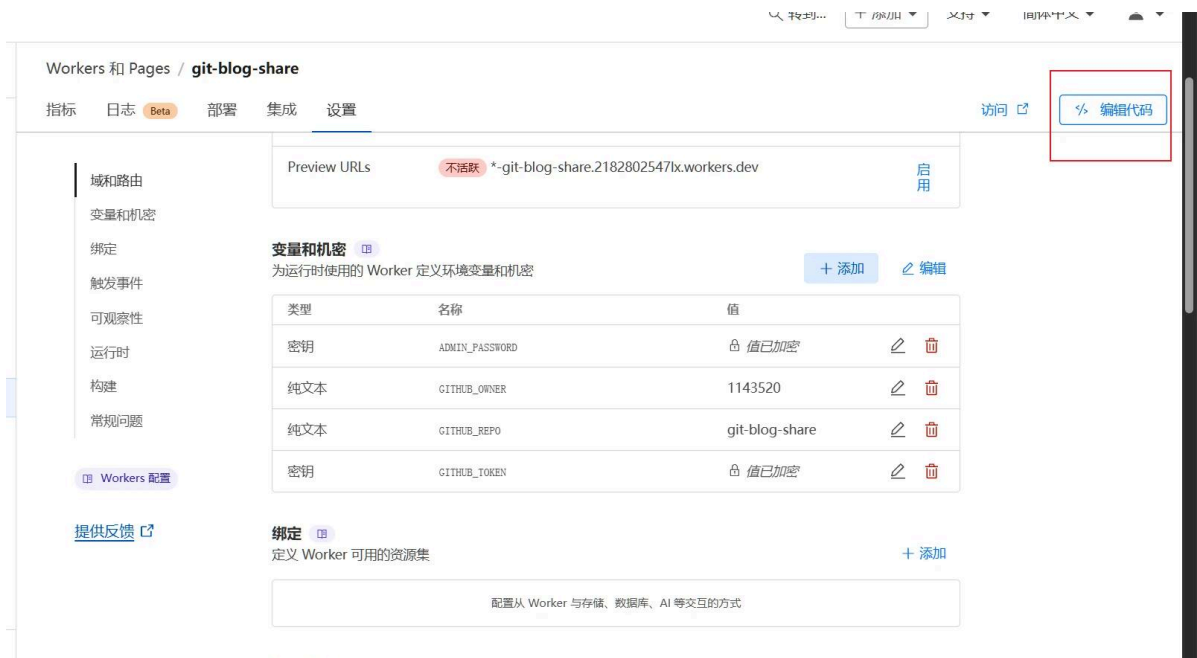
点击底下的 **Generate token**

复制token（只显示一次）填入密钥类型的变量，变量名为GITHUB_TOKEN

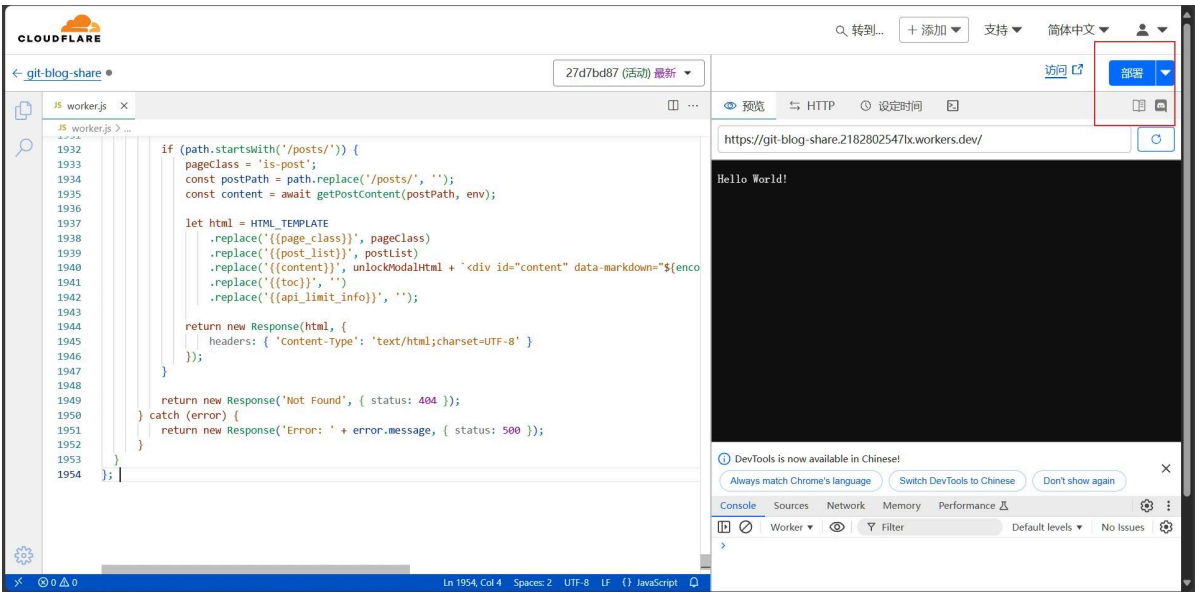
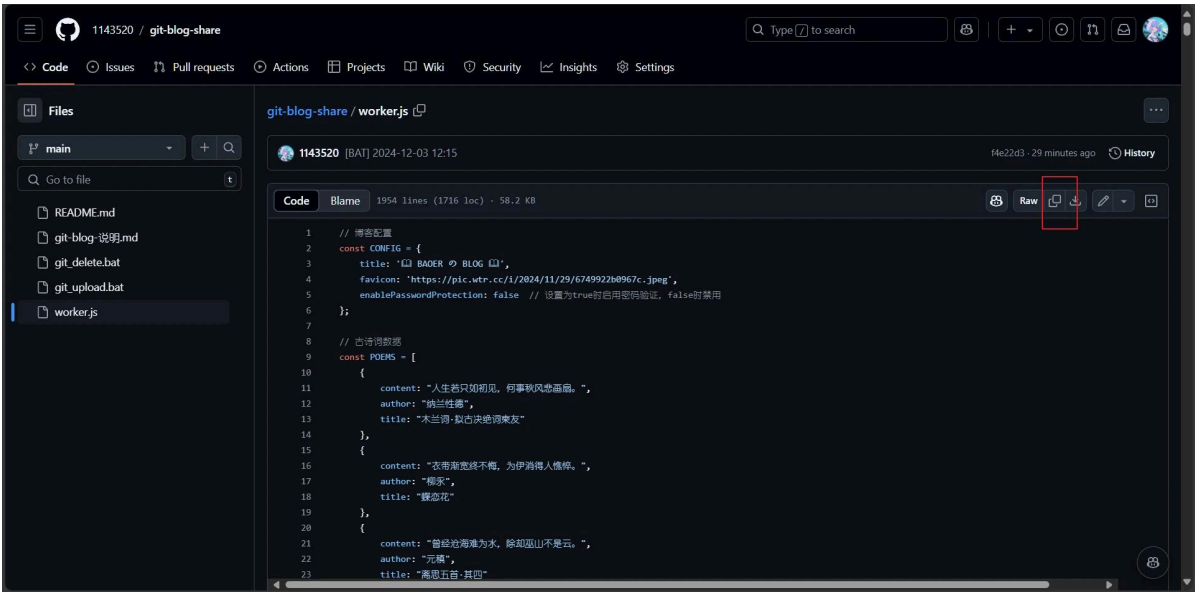
相对应的填入其他变量



然后点击编辑代码



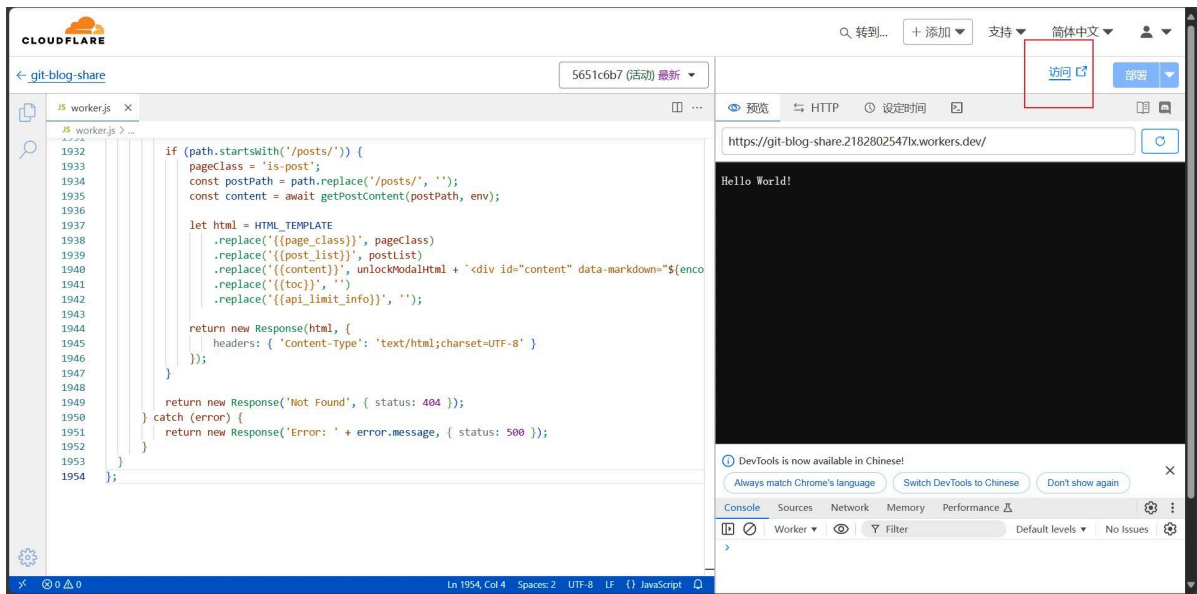
将[git-blog-share/worker.js at main · 1143520/git-blog-share](#)的内容复制后粘贴到workers全选替换之后部署



选择是否启用密码验证，和设置站点标题和logo



之后点击访问大功告成



补充

1、自定义域名



2、填入变量注意前后不要有空格

3、推荐搭配typora+bili图床使用

[xlzy520/typora-plugin-bilibili](https://github.com/xlzy520/typora-plugin-bilibili): Typora粘贴图片自动上传到Bilibili图床，也可以自定义修改成任意其他图床接口。使用教程：<https://b23.tv/urxCc3>

4、更多内容请阅读仓库文档或者阅读代码后自行修改