

# 1. 作业提交规范说明

## 1.1 文件名(压缩文件名)

<组号-组长姓名>

但之后按照下面新的框架来的话其实没关系了，但最好还是按照这个来

## 1.2 新的作业框架

由于雨课堂上需要一个个下载压缩包然后解压很麻烦，麻烦同学用下面的方法把自己的项目挂载到github上，然后我这里可以直接网页端浏览。

这个方法同学们自己也可以对所有的作业进行一个有效的归类。

### 1.2.1 给作业创建 github 仓库

首先去自己的github主页创建一个新的仓库用于存放所有自己的代码（包括后续作业），确保为**public**仓库

The screenshot shows the GitHub repository creation interface. At the top, it displays the user's profile picture, 11 public repositories, and repository statistics (Pin 0, Watch 0, Fork 0, Star 0). Below this, there are two main sections: 'Start coding with Codespaces' and 'Add collaborators to this repository'. Under 'Start coding with Codespaces', there is a 'Create a codespace' button. Under 'Add collaborators to this repository', there is a 'Invite collaborators' button. A large blue box labeled 'Quick setup — if you've done this kind of thing before' contains instructions for setting up on desktop or via HTTPS/SSH, along with a link to the repository's URL (<https://github.com/Lemonade68/11.git>). It also suggests creating a new file or uploading an existing one, and recommends including README, LICENSE, and .gitignore files. Below this, another section titled '...or create a new repository on the command line' provides a command-line script:

```
echo "# 11" >> README.md  
git init  
git add README.md  
git commit -m "first commit"  
git branch -M main  
git remote add origin https://github.com/Lemonade68/11.git  
git push -u origin main
```

At the bottom, another section titled '...or push an existing repository from the command line' provides a command-line script:

```
git remote add origin https://github.com/Lemonade68/11.git  
git branch -M main  
git push -u origin main
```

下面为初始化仓库的流程（已经有仓库的同学可以跳转到下一步）：

1. 在自己的作业文件夹中进入cmd页面
2. 依次输入：

```

// 初始化本地仓库
git init
// 添加所有的文件
git add .
// 提交更改
git commit -m "init"

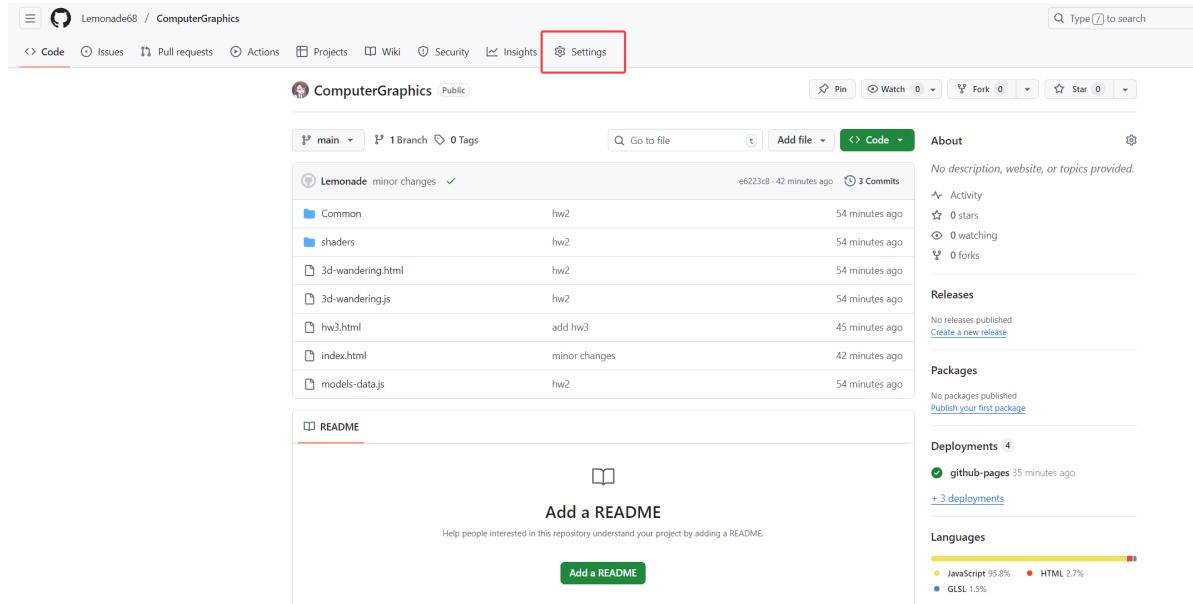
// 然后依次输入上面红框的部分
git remote add origin https://github.com/xxx/xxx.git
git branch -M main
git push -u origin main

```

3. 此时仓库应该已经初始化完成，并且上传了对应的代码

## 1.2.2 为仓库添加pages

前往settings页面：



然后按照下图的步骤依次进行：

The screenshot shows the GitHub Pages settings page. On the left, there's a sidebar with various options like General, Access, Collaborators, Moderation options, Code and automation, and Pages (which is highlighted with a red arrow labeled '1'). The main area is titled 'GitHub Pages' and contains the following steps:

2. 选择main: A red arrow points to the 'main' dropdown under 'Source'.
3. save: A red arrow points to the 'Save' button under 'Source'.
4. 输入custom的域名用于访问: A red arrow points to the 'Custom domain' input field.
5. save: A red arrow points to the 'Save' button under 'Custom domain'.
6. 过一段时间之后会出现对应的域名 (后续我看的网页链接): A red arrow points to the 'Your site is live at https://lemonade68.github.io/ComputerGraphics/' link.

Other visible sections include 'Build and deployment', 'Custom domain', 'Enforce HTTPS', and 'Visibility'.

此时进入你的域名时应该会报错，因为作业默认的框架里**没有index.html**，即**没有默认加载的页面**，因此前往文件夹下，添加 `index.html` 文件

📁 .git	2025/11/8 20:48	文件夹	
📁 Common	2025/10/19 10:59	文件夹	
📁 shaders	2025/10/19 10:59	文件夹	
📝 3d-wandering.html	2025/10/19 15:34	Chrome HTML D...	2 KB
📄 3d-wandering.js	2025/10/19 16:16	JSFile	12 KB
📝 hw3.html	2025/11/8 20:45	Chrome HTML D...	1 KB
📝 index.html	2025/11/8 20:47	Chrome HTML D...	1 KB
📄 models-data.js	2024/9/29 19:57	JSFile	8 KB

使用vscode或者其他编辑器进入页面，对index.html进行修改：

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <h1>Home</h1>
    <a href=". /3d-wandering.html">HW2</a>
    <a href=". /hw3.html">HW3</a>
</body>
</html>

```

中间对应的分别是链接的实际展示文件（如作业2的 `3d-wandering.html`），以及网页上显示的名字。

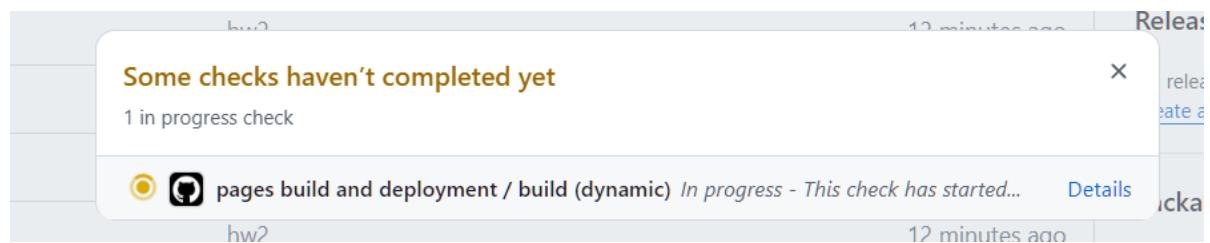
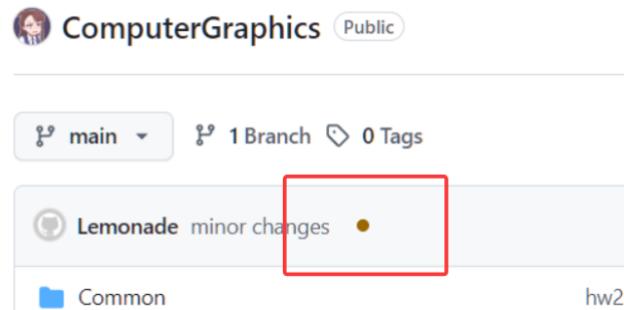
此时将变更进行更新：

```

git add .
git commit -m "..."
git push

```

注意推上去之后变更不会立刻进行，而是会进行pending：

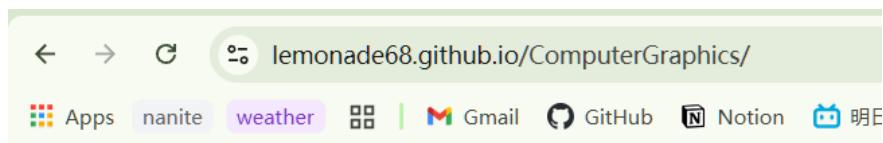


## ⌚ pages build and deployment #2

The screenshot shows the GitHub Actions interface. On the left, under 'Jobs', the 'build' job is highlighted with a green checkmark, indicating it has succeeded. Below it are 'report-build-status' and 'deploy' jobs, both also marked as successful with green checkmarks. To the right, the 'build' job's details are expanded, showing a list of steps: Set up job, Pull ghcr.io/actions/jekyll-build-pages:v1.0.13, Checkout, Build with Jekyll, Upload artifact, Post Checkout, and Complete job. All steps are marked with green checkmarks, indicating they were successful.

### 1.2.3 最终结果

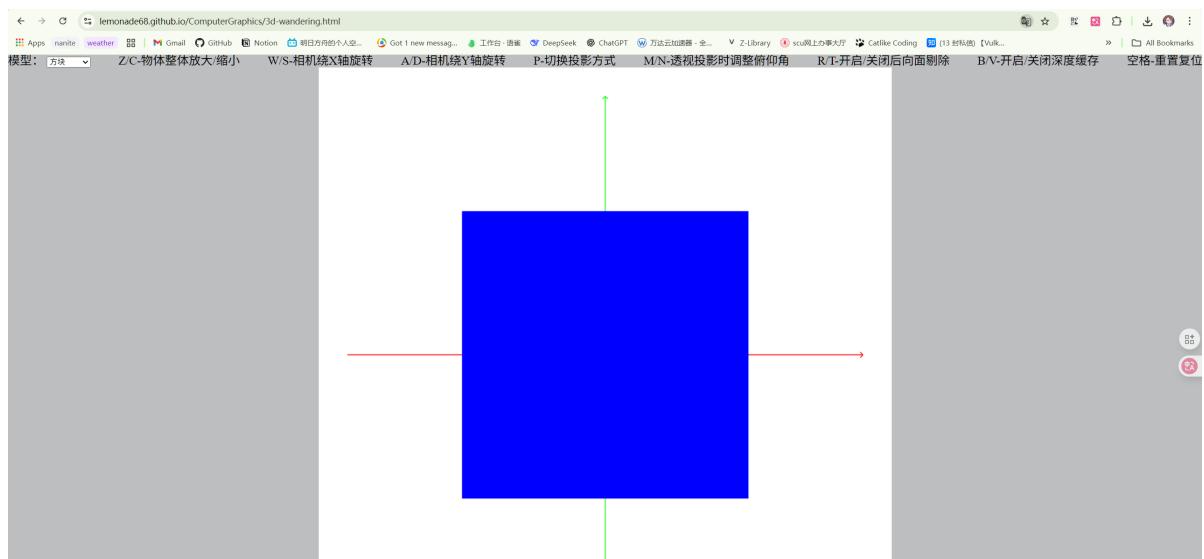
等待其结束后，重新打开你的域名，可以得到如下结果：



### Home

[HW2](#) [HW3](#)

点击HW2后，跳转到对应的实际页面：



下一次以及以后提交作业时，交上作业的源代码以及复制下你的域名，这样方便批改