

# Lab0 -- GitLab

---

## 实验介绍

- 你需要学会使用一些基本的 git 功能。
- 这个实验非强制性要求，但学会使用 git 是大有裨益的

## To-Do List

- git 与 github 基础相关：
  - 在 wsl（或你使用的其他虚拟机）中安装 git
  - 在 wsl 中配置 ssh key
  - 创建 github 账号
  - 为 github 账号添加 wsl 的 ssh key
- github classroom 相关（也是今后 lab 的发布与提交流程）：
  - 在 github classroom 中接受作业
  - 将作业内容 clone 至本地（虚拟机）
  - 修改本地仓库，上传一个任意文件，并推送至远程仓库
- （非必需，但推荐）简单学习一下 git 的常用指令

## 实验指引

- git 的安装
  - 参考[这个网站](#)
- 你需要注册一个 [github](#) 账号
- github 配置 **ssh key**

**NOTE:** 由于我们的实验都在虚拟机 linux 环境下进行，所以无需为本地主机配置 ssh key。如果你以后希望将本地的仓库推送到远程，只要在本地的终端进行下面的配置。

- 1.打开 wsl
- 2.检查是否已经存在 ssh key。终端运行：

```
ls ~/.ssh  
# 列出 ~/.ssh 目录下的所有内容，相当于查看 ~/.ssh 目录是否存在
```

如果输出如下，则跳到第 4 步

```
root@tristin:~/.ssh# cd ~/.ssh
root@tristin:~/.ssh# ls
id_rsa  id_rsa.pub  known_hosts  known_hosts.old
```

若显示 `No such file or directory`, 则继续进行第 3 步

- 3.生成 ssh key。终端运行

```
ssh-keygen -t rsa -C "xxx@xx.com"
# 引号以及里面的内容替换为你的邮箱
# 执行后一直回车使用默认值即可（没必要设置密码）
```

- 4.获取 ssh key 公钥内容 (id\_rsa.pub)

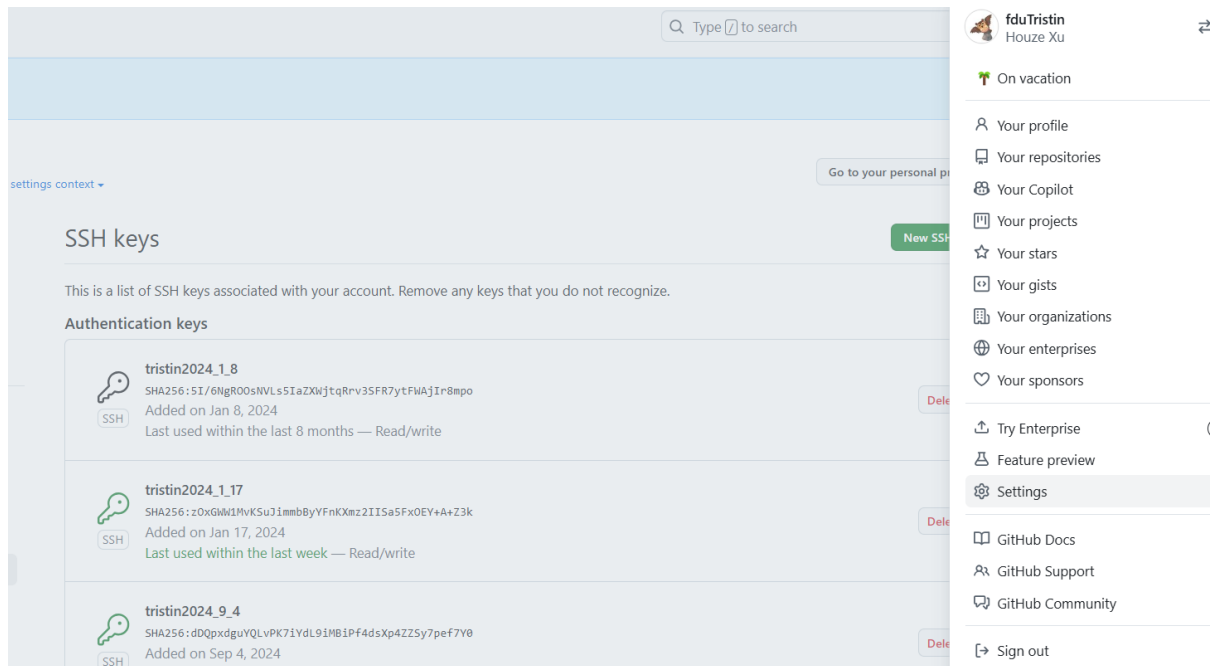
```
cat ~/.ssh/id_rsa.pub
# cat 命令用于连接文件并打印到标准输出设备
```

如下图所示:

```
root@tristin:/# cat ~/.ssh/id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQGCY
HqN7a9rHQzd3Gw0aMh6fPpVK+KUoojSE3lEIj0yp
800vsgglRodkjwvxdUqK7aMRT7vSHLK9WB6TuzJ9
QxvmLCq8bumXtt1wuov8RXc0Us92EuTmSmIXGPq8
qKmInCKSJcpqbq6lsE062rY13CTRFDF97twRb00S
```

复制该内容（从 ssh-rsa 开始）

- 5.github 账号上添加公钥
  - 点击进入 settings - SSH and GPG keys



- 点击 "New SSH keys", 将刚刚复制的公钥粘贴, 并给它起个名字, 例如 wsl-key
- 6.验证是否成功

```
root@tristin:~/ssh# ssh -T git@github.com
Hi fduTristin! You've successfully authenticated, but GitHub does not provide shell access.
root@tristin:~/ssh#
```

- 进入课堂
  - 点击加入 [github classroom](#)

- 你将显示如下页面：

Join the classroom:

ICS-FDU-24Fall-classroom

To join the GitHub Classroom for this course, please select yourself from the list below to associate your GitHub account with your school’s identifier (i.e., your name, ID, or email).

Can’t find your name? [Skip to the next step →](#)

Identifiers	
FYY	>
KLY	>
XHZ	>

- 选择和你对应的学号/姓名


- 完成后:



## You're ready to go!

You accepted the assignment, **Lab0**.

Your assignment repository has been created:

 <https://github.com/ICS-FDU-24Fall/lab0-fduTristin>

We've configured the repository associated with this assignment.



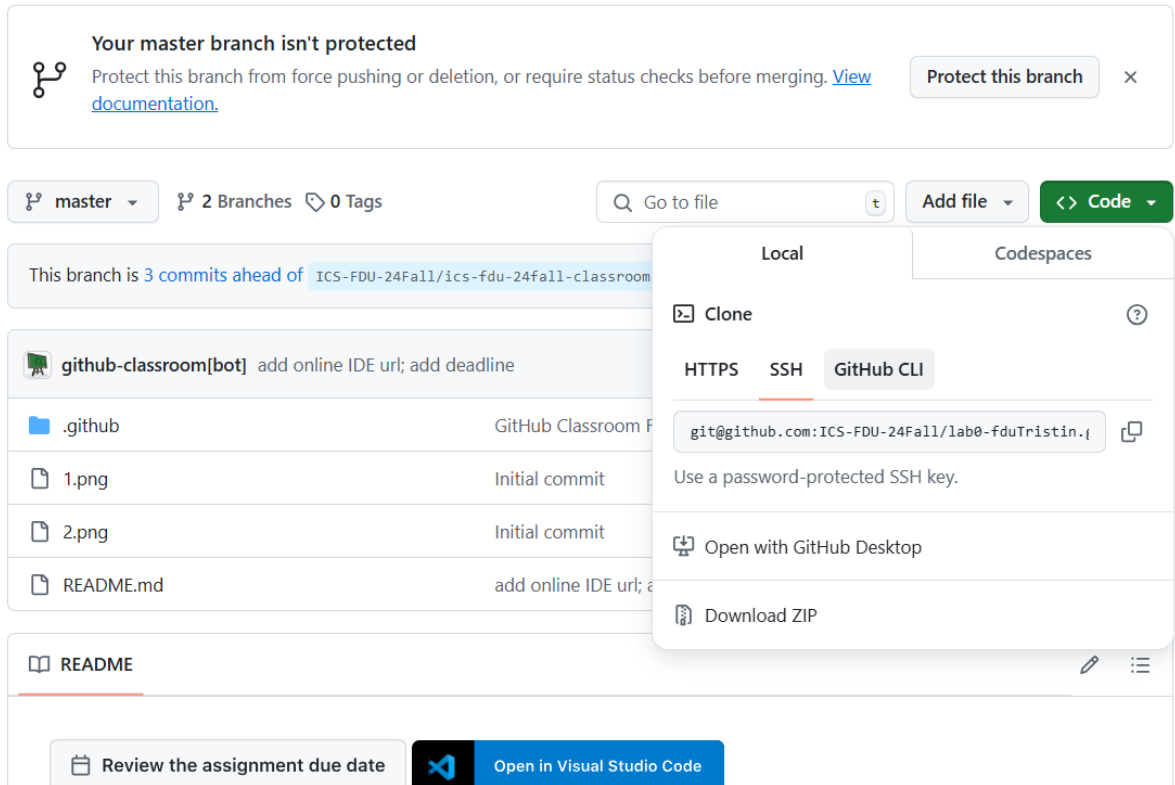
 Your assignment is due by **Sep 22, 2024, 15:59 UTC**

Note: You may receive an email invitation to join [ICS-FDU-24Fall](#) on your behalf. No further action is necessary.

- 这个链接是你的远程仓库（你只能访问自己的，需要保持 github 的登录状态）
- 创建本地仓库（你也可以直接点进链接，直接更新远程仓库，但是推荐使用本地仓库）
  - 打开 wsl，新建一个 lab 文件夹

```
cd ~  
# 进入默认文件夹  
mkdir lab0  
# 创建名为 lab0 的文件夹（你也可以在此之前创建一个总的课程文件夹，进入之后创建  
每个 lab 的文件夹）  
cd lab0  
# 进入 lab0 文件夹
```

- 在网页中访问刚刚获取的链接，点击 "Code" 下的 "SSH"，复制



## 运行

```
git clone xxxx
# (将 xxxx 替换为刚刚获取的 SSH)
# git clone 指令用于将远程仓库克隆到本地
```

```
root@tristin:~/ICS-24fall/lab0# git clone git@github.com:ICS-FDU-24Fall/lab0-fduTristin.git
Cloning into 'lab0-fduTristin'...
remote: Enumerating objects: 13, done.
remote: Counting objects: 100% (13/13), done.
remote: Compressing objects: 100% (11/11), done.
remote: Total 13 (delta 2), reused 3 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (13/13), 106.07 KiB | 288.00 KiB/s, done.
Resolving deltas: 100% (2/2), done.
root@tristin:~/ICS-24fall/lab0#
```

你会发现当前目录下有一个文件夹

```
root@tristin:~/ICS-24fall/lab0# ls
lab0-fduTristin
root@tristin:~/ICS-24fall/lab0#
```

- 修改本地仓库 (vscode 打开)



- 你只需要成功上传一个文件，可以是任何内容（例如一个简单的 helloworld.c，或是对本课程的期待？）（但最好不要是 .doc/.docx 文件，文本文件推荐上传 .md 文件）

- 上传!

- 你需要先在终端进入克隆下来的文件夹

```
● root@tristin:~/ICS-24fall/lab0# cd lab0-fduTristin/  
○ root@tristin:~/ICS-24fall/lab0/lab0-fduTristin#
```

- 在终端运行以下指令：

```
git add -A  
# 提交当前文件夹下的所有更改到暂存区  
git commit -m "xxx(可以是你的提交注释)"  
# 将暂存区的所有更改提交到本地仓库  
git push  
# 将本地仓库推送到远程
```

- 你也可以使用 vscode 自带的提交功能
- 在这里简单介绍一下 git 的功能：
  - git 是一种**分布式版本控制系统**，可以记录文件的不同版本，方便阶段性保存、回滚等操作，也能确保团队中的多人可以同时在同一项目上工作而不冲突。
  - 在 git 中，工作区、缓存区和本地仓库是三个核心概念，它们共同构成了 git 的版本控制流程。
    - **工作区**是实际操作项目文件的地方，可以在这里编辑、删除或添加文件。当你对文件进行修改后，这些更改首先出现在工作区中。此时，文件处于“未跟踪”或“已修改”的状态，git 并未正式记录这些修改。
    - **缓存区**是一个临时存储区，记录你希望包含在下一个提交（commit）中的更改。也就是说，文件从工作区进入缓存区后，git 会认为这些更改已经准备好被提交。
    - **本地仓库**是你项目的完整历史记录库。每次提交（commit）都会将缓存区的内容永久保存到本地仓库中。提交后，这些修改就会成为项目历史的一部分。
    - 另外还有**远程仓库**（如 github）。本地仓库的提交可以推送（push）到远程仓库中，与其他开发者共享。

## #参考资料

- [github classroom 的使用](#)
- 更多的 git 操作可以参考 lab 文档下的 [Git 基本使用](#) 或 [这个链接](#)