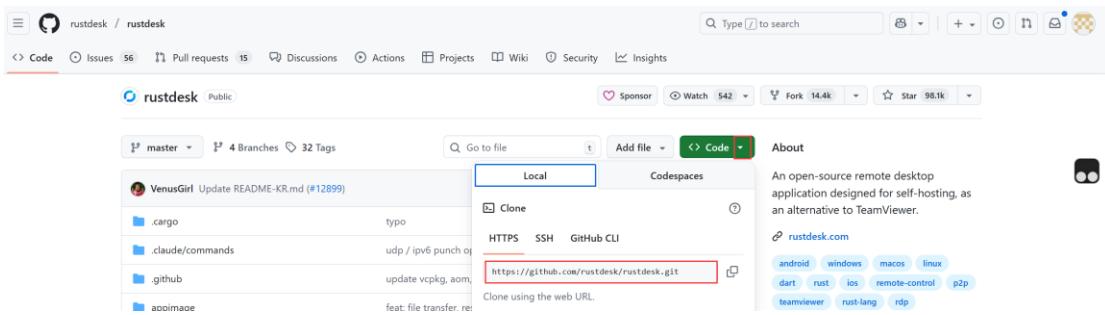


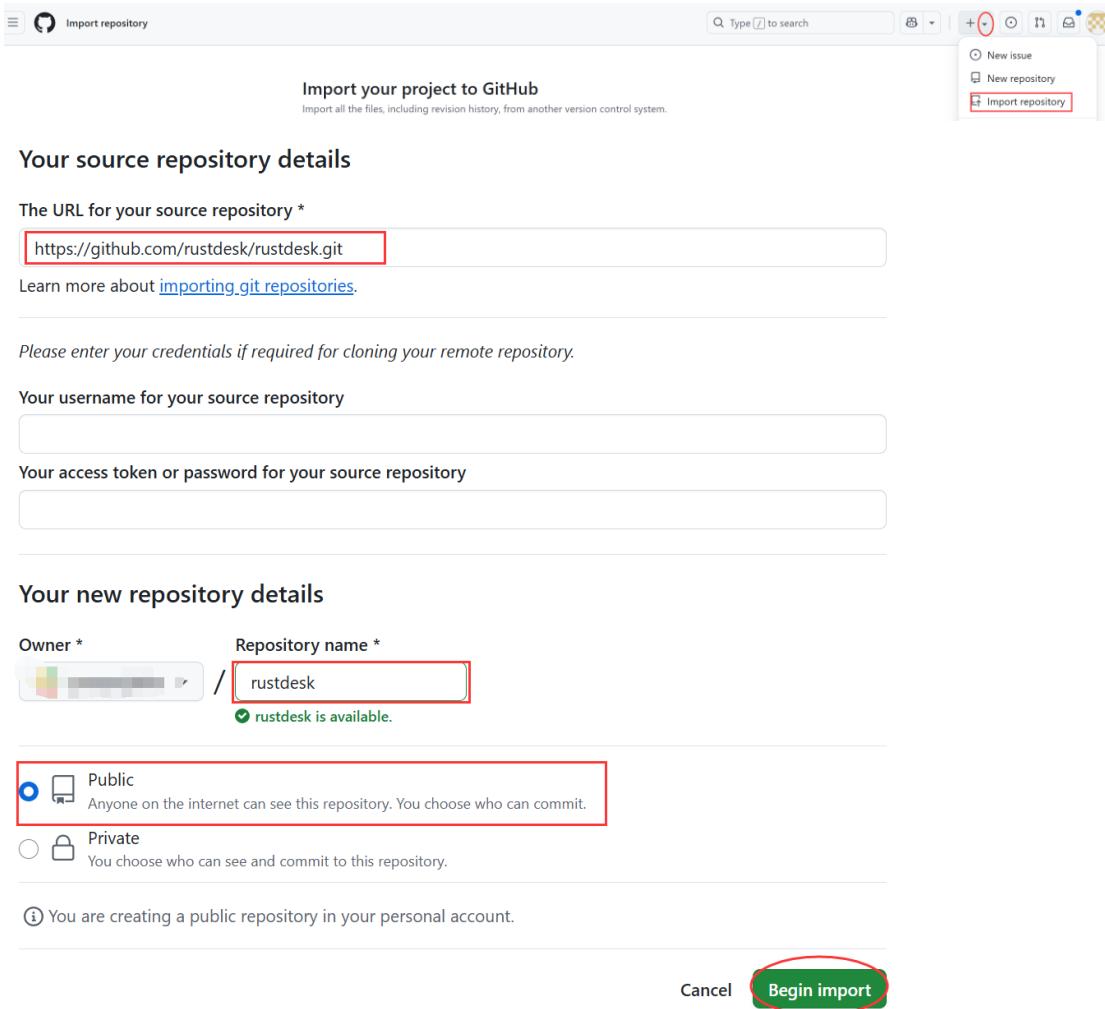
RustDesk 子模块本地化

导入 Rustdesk 官方仓库

- 一、如果没有 GitHub 账号，先注册，<https://github.com>
- 二、看要导入的仓库地址，比如要导入 rustdesk 官方仓库地址，先访问他的页面
<https://github.com/rustdesk/rustdesk>，然后点击右侧绿色 Code 下拉箭头，显示对应的仓库地址为：<https://github.com/rustdesk/rustdesk.git>



- 三、点击右上角的+号下拉箭头，选导入仓库(Import repository)



The URL for your source repository *

Learn more about [importing git repositories](#).

Please enter your credentials if required for cloning your remote repository.

Your username for your source repository

Your access token or password for your source repository

Your new repository details

Owner * / Repository name *

Personal account / rustdesk / rustdesk is available.

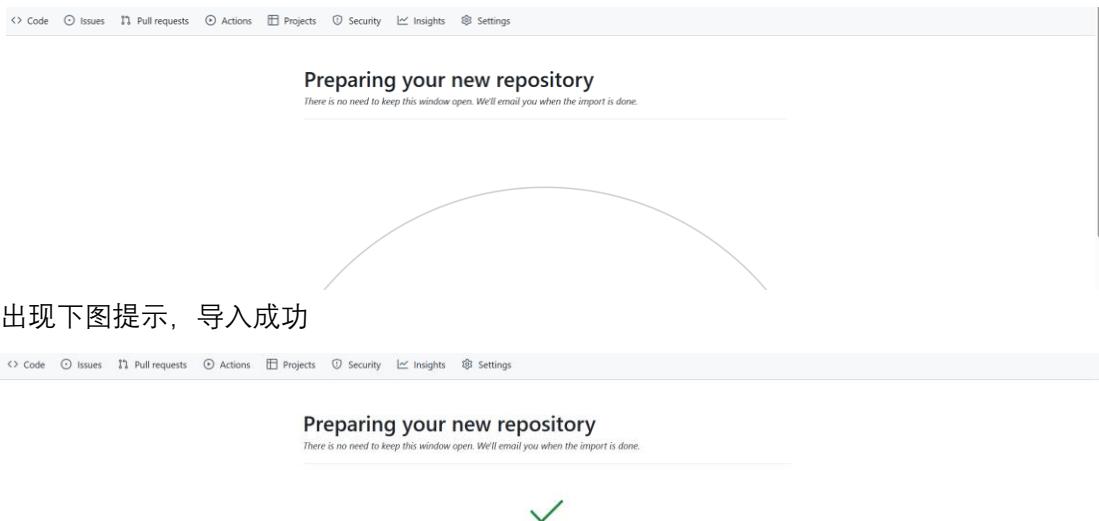
Public
Anyone on the internet can see this repository. You choose who can commit.

Private
You choose who can see and commit to this repository.

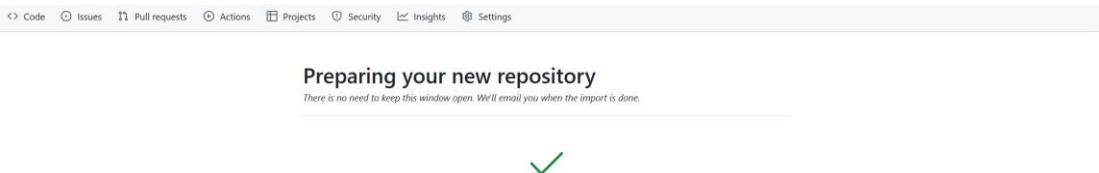
(i) You are creating a public repository in your personal account.

Cancel Begin import

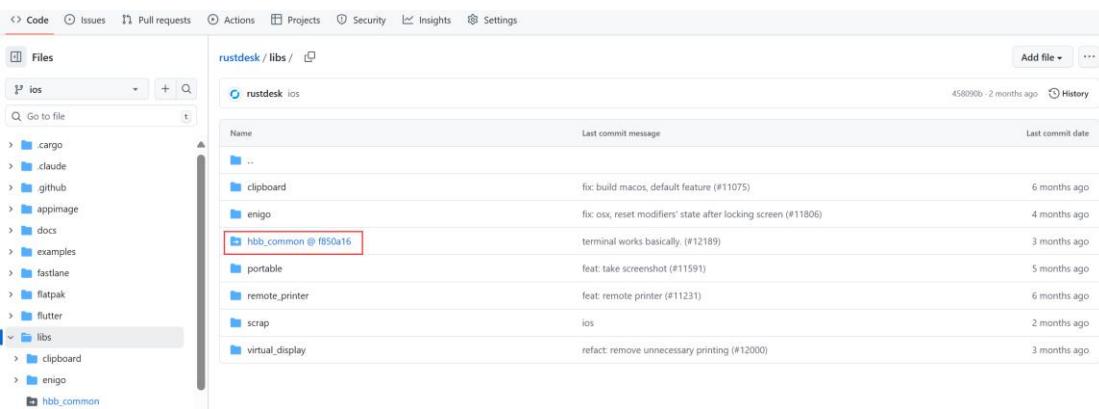
四、开始导入，耐心等待



出现下图提示，导入成功

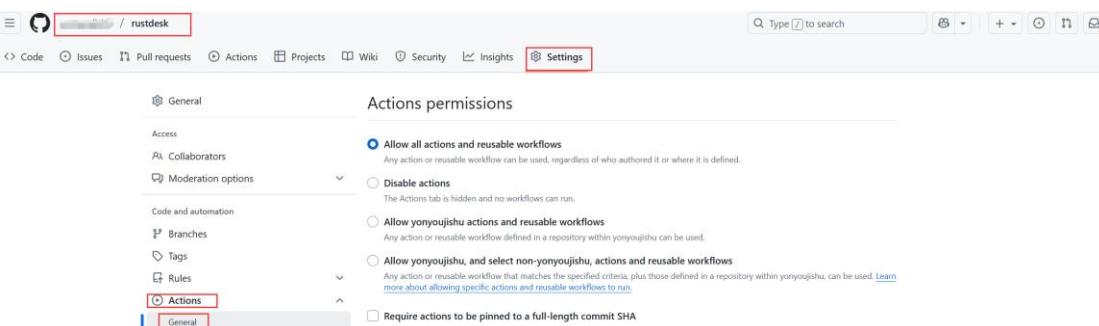


五、进入到我们导入的仓库，定位到 libs 文件夹下，可以看到 hbb_common 前面有个箭头，这是作者将其设置为了子模块，点击它，会跳转到作者对应的子模块仓库（也就是说这个子模块仓库不是我们自己的）



六、修改 GitHub 相应权限

进入到你 GitHub 的 rustdesk 仓库，点击 Settings，然后左侧点击 Actions 下的 General



右侧找到 Workflow permissions 部分，按照下图设置，最后点击 save

Workflow permissions

Choose the default permissions granted to the GITHUB_TOKEN when running workflows in this repository. You can specify more granular permissions in the workflow using YAML. [Learn more about managing permissions.](#)

Read and write permissions

Workflows have read and write permissions in the repository for all scopes.

Read repository contents and packages permissions

Workflows have read permissions in the repository for the contents and packages scopes only.

Choose whether GitHub Actions can create pull requests or submit approving pull request reviews.

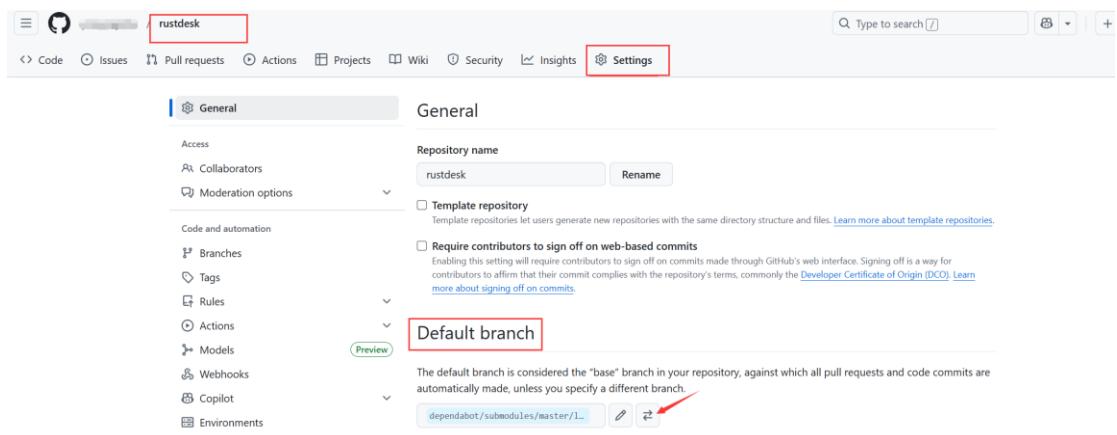
Allow GitHub Actions to create and approve pull requests

[Save](#)

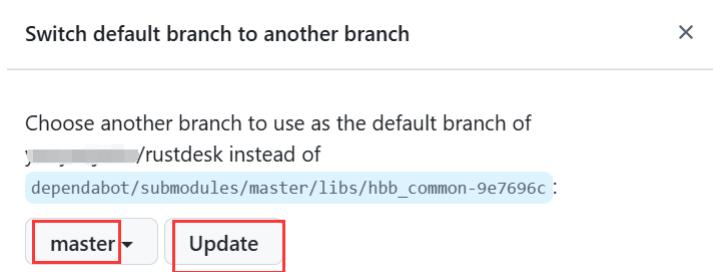
修改默认分支

导入的 rustdesk 默认分支可能不是 master，设置默认分支为 master

进入你导入的 rustdesk 仓库，点击 settings，右侧找到 Default branch，点击下图按钮



选择 master 分支，点 Update

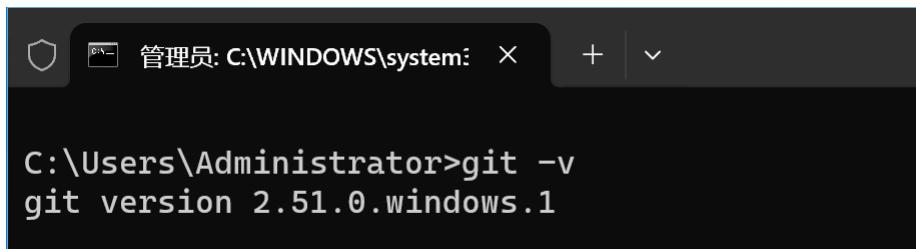


安装 Git，配置 GitHub SSHkey，测试连接

一、本地电脑安装 Git（安装的时候选项都默认，一直下一步）

官网地址：<https://git-scm.com/downloads>

安装完毕打开 cmd，输入 git -v 回车，显示版本号则安装成功。



```
C:\Users\Administrator>git -v
git version 2.51.0.windows.1
```

安装成功只是第一步,想要正常克隆和推送github源码,我们需要配置github的sshkey,才可以将本地git和github对接。

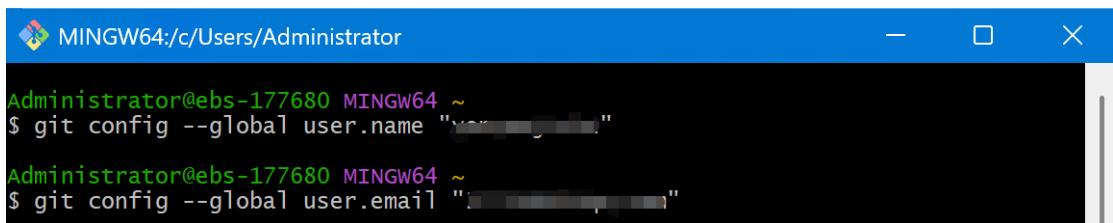
二、从开始菜单中找到 Git Bash, 运行它



```
Administrator@ebs-177680 MINGW64 ~
$ git config --global user.name "yourname"
```

Git Bash 默认的路径是当前登录 windows 账号的目录, 例如 C:\Users\Administrator

三、如下配置账号和邮箱



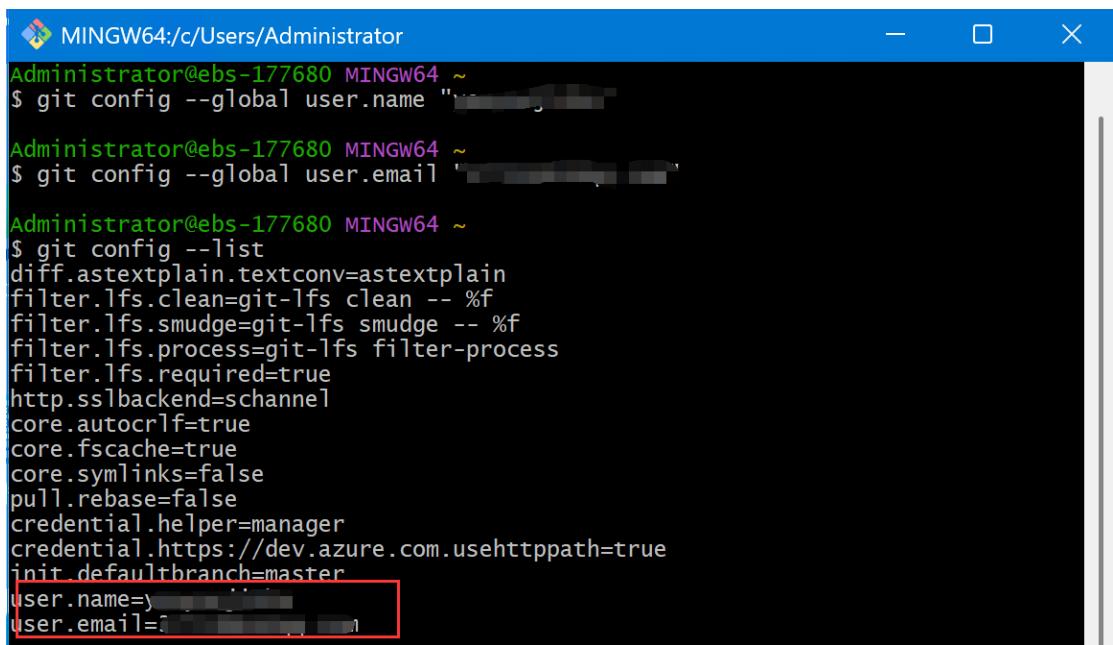
```
Administrator@ebs-177680 MINGW64 ~
$ git config --global user.name "yourname"
Administrator@ebs-177680 MINGW64 ~
$ git config --global user.email "youremail"
```

git config --global user.name "替换为你的 github 用户名"

git config --global user.email "替换为你的 github 注册邮箱"

上面两条命令执行完毕后, 输入以下命令查看是否生效

git config --list



```
Administrator@ebs-177680 MINGW64 ~
$ git config --global user.name "yourname"
Administrator@ebs-177680 MINGW64 ~
$ git config --global user.email "youremail"

Administrator@ebs-177680 MINGW64 ~
$ git config --list
diff.astextplain.textconv=astextplain
filter.lfs.clean=git-lfs clean -- %f
filter.lfs.smudge=git-lfs smudge -- %f
filter.lfs.process=git-lfs filter-process
filter.lfs.required=true
http.sslbackend=schannel
core.autocrlf=true
core.fscache=true
core.symlinks=false
pull.rebase=false
credential.helper=manager
credential.https://dev.azure.com.usehttppath=true
init.defaultbranch=master
user.name=yourname
user.email=youremail
```

四、生成密钥

在 GitBash 输入如下命令

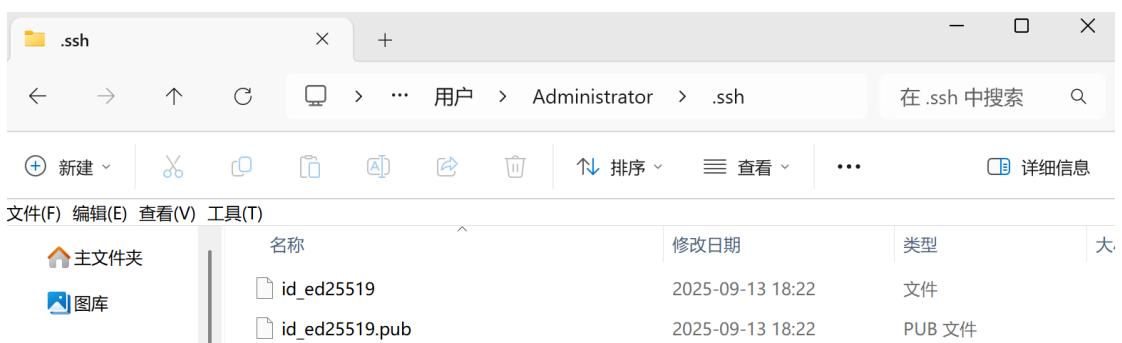
```
ssh-keygen -t ed25519 -C "你的 github 注册邮箱"
```

如果系统不支持 ed25519 算法，可以使用下面的命令

```
ssh-keygen -t rsa -C "替换为你的 github 注册邮箱"
```

回车后提示正在创建公钥/私钥，然后让你输入密钥保存路径，直接回车保存到默认路径，即：C:\Users\Administrator\.ssh，提示 enter passphrase for 是让你输入 SSH 私钥的密码，密码是用来保护你的 SSH 私钥的，如果有人获取了你的私钥文件，没有这个密码也无法使用它，从而提高安全性。想设置密码，就输入要设置的私钥密码，回车，再确认输入一次密码，回车。不想设置私钥密码，直接两次回车，如下图，生成完毕。

打开存储密钥的路径，有两个文件，其中 pub 格式的是公钥，另外一个是私钥。



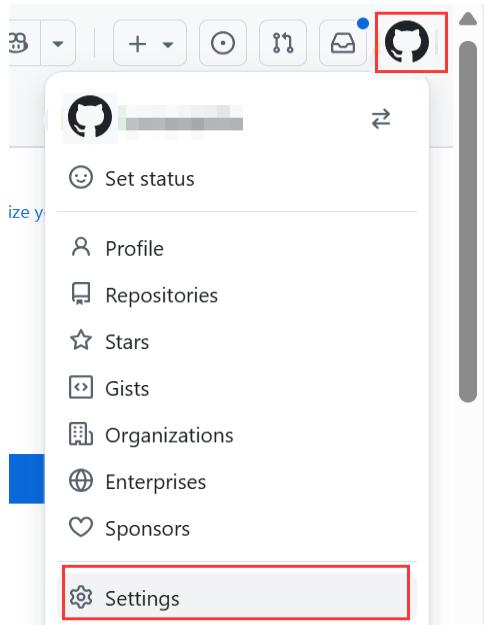
可以用记事本打开 id_ed25519.pub 查看公钥，也可以通过如下命令查看

```
cat ~/.ssh/id_ed25519.pub
```

```
Administrator@ebs-177680 MINGW64 ~
$ cat ~/.ssh/id_ed25519.pub
ssh-ed25519 A...  
[REDACTED]
```

接下来配置 Github 端

五、登录github账号，点击右上角个人头像，弹出的菜单中选 Settings



左侧导航点击 SSH and GPG keys, 再点击 New SSH key

The screenshot shows the GitHub Settings interface. On the left, there's a sidebar with various account settings like Public profile, Account, Appearance, etc. The 'SSH and GPG keys' section is currently selected and highlighted with a blue box. At the top right, there's a green 'New SSH key' button, which is also highlighted with a red box.

标题>Title)随便填写, 密钥类型(Key type)选 Authentication Key, 将上面得到的公钥(那个pub后缀的文件内容)复制到 key 这里, 最后点击 Add SSH key 按钮添加保存。

The screenshot shows the 'Add new SSH Key' form. The 'Title' field is filled with 'key1'. The 'Key type' dropdown is set to 'Authentication Key'. The 'Key' text area contains a long string of characters, which is a public key. A red arrow points from this area to the green 'Add SSH key' button at the bottom.

可能需要验证你的 Github 密码, 输入密码后点击 Confirm, 添加成功后如下图

The screenshot shows the 'SSH keys' list. A new key named 'key1' has been added. The details for this key are shown: SHA256:..., Added on Sep 13, 2025, and Never used — Read/write. A 'Delete' button is located to the right of the key entry.

六、回到本地 git bash，输入如下命令，测试是否可以成功链接 GitHub

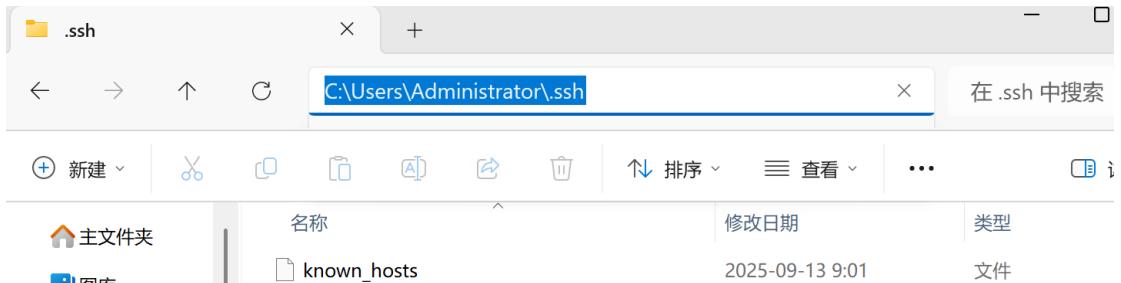
注意：命令中的 git@github.com 是固定的，不要改为自己邮箱

ssh -T git@github.com

上述命令回车有下面的提示，是因为本地电脑从未与 GitHub 建立过 SSH 信任关系，本地电脑 ~/.ssh/known_hosts 文件未生成。输入 yes 再按回车

```
Administrator@ebs-177680 MINGW64 ~
$ ssh -T git@github.com
The authenticity of host 'github.com (20.205.243.166)' can't be established.
ED25519 key fingerprint is SHA256:+D...[REDACTED]...
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'github.com' (ED25519) to the list of known hosts.
Hi [REDACTED] You've successfully authenticated, but GitHub does not provide shell access.
```

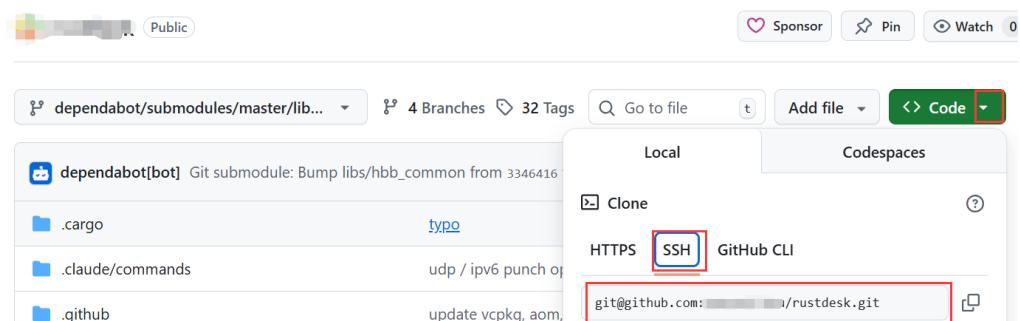
在对应路径下生成 known_hosts 文件



经过上述操作，就将本地电脑与 GitHub 建立了连接，接下来就可以克隆 GitHub 仓库到本地了。

通过 Git 克隆 GitHub 中的仓库到本地

一、登录 Github，查看要克隆仓库对应的 SSH 地址



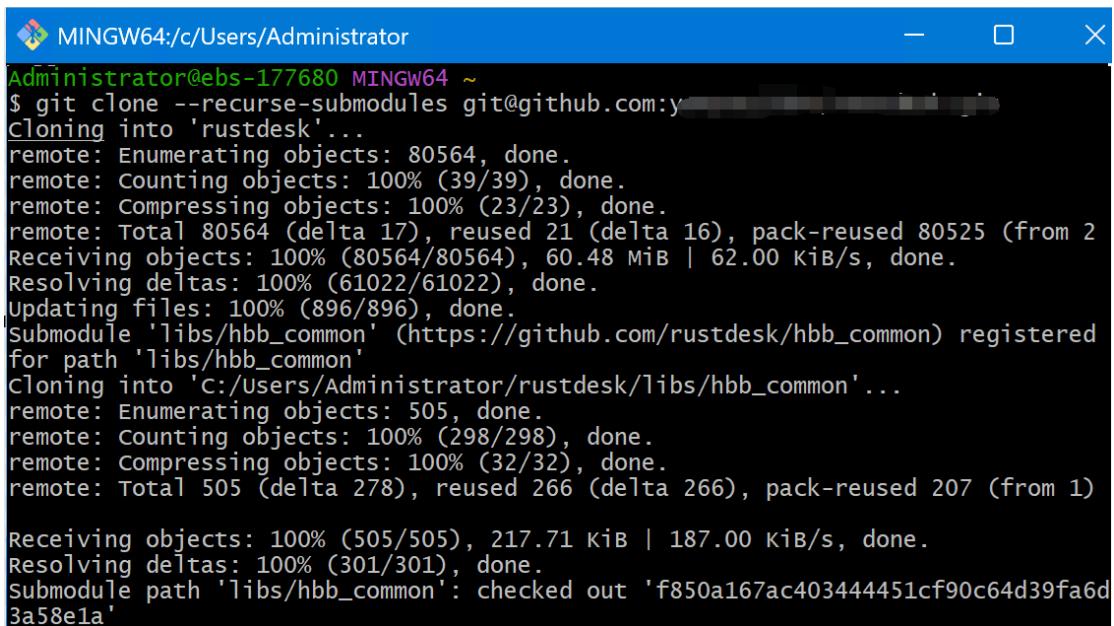
二、在本地的 Git Bash 输入如下命令进行克隆

git clone --recurse-submodules <替换为你的仓库 SSH 地址>

解释：--recurse-submodules 参数，递归克隆子模块，会把 hbb_common 这个子模块一并克隆。回车后开始克隆仓库。

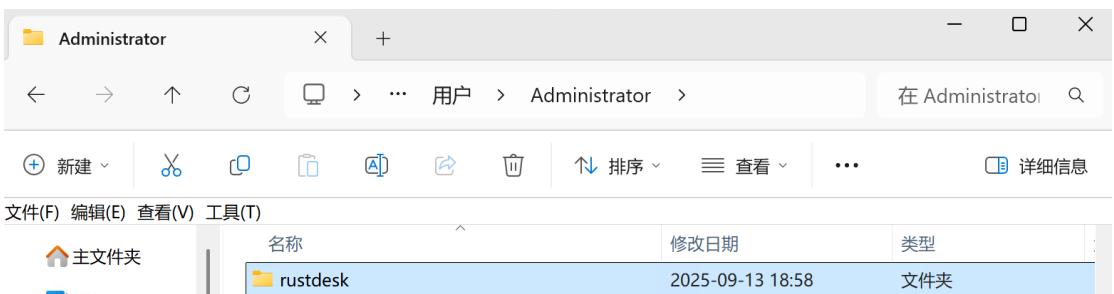
```
Administrator@ebs-177680 MINGW64 /d/GitHub
$ git clone --recurse-submodules git@github.com:[REDACTED]/rustdesk.git
Cloning into 'rustdesk'...
remote: Enumerating objects: 80564, done.
remote: Counting objects: 100% (50/50), done.
remote: Compressing objects: 100% (21/21), done.
Receiving objects: 20% (16113/80564), 3.61 MiB | 1.73 MiB/s
```

耐心等待，直到克隆完毕



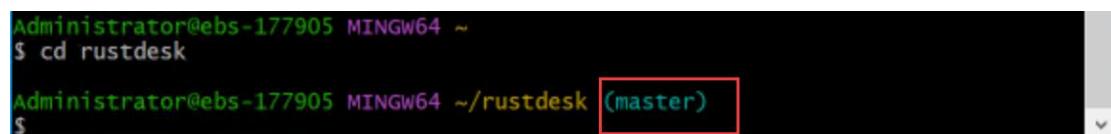
```
MINGW64:/c/Users/Administrator
Administrator@ebs-177680 MINGW64 ~
$ git clone --recurse-submodules git@github.com:y...[REDACTED]
Cloning into 'rustdesk'...
remote: Enumerating objects: 80564, done.
remote: Counting objects: 100% (39/39), done.
remote: Compressing objects: 100% (23/23), done.
remote: Total 80564 (delta 17), reused 21 (delta 16), pack-reused 80525 (from 2
Receiving objects: 100% (80564/80564), 60.48 MiB | 62.00 KiB/s, done.
Resolving deltas: 100% (61022/61022), done.
Updating files: 100% (896/896), done.
Submodule 'libs/hbb_common' (https://github.com/rustdesk/hbb_common) registered
for path 'libs/hbb_common'
Cloning into 'c:/users/Administrator/rustdesk/libs/hbb_common'...
remote: Enumerating objects: 505, done.
remote: Counting objects: 100% (298/298), done.
remote: Compressing objects: 100% (32/32), done.
remote: Total 505 (delta 278), reused 266 (delta 266), pack-reused 207 (from 1)
Receiving objects: 100% (505/505), 217.71 KiB | 187.00 KiB/s, done.
Resolving deltas: 100% (301/301), done.
Submodule path 'libs/hbb_common': checked out 'f850a167ac403444451cf90c64d39fa6d
3a58e1a'
```

克隆完成，会在本地目录看到一个 rustdesk 文件夹（默认在 C:\Users\Administrator）
如下图



将 hbb_common 子模块改为普通文件夹

在 gitbas 中输入以下命令，进入到 rustdesk 主目录，确认分支是 master，如下图
cd rustdesk



```
Administrator@ebs-177905 MINGW64 ~
$ cd rustdesk
Administrator@ebs-177905 MINGW64 ~/rustdesk (master)
```

如果分支不是 master，输入如下命令切换到 master 分支
git checkout master



```
Administrator@ebs-177905 MINGW64 ~/rustdesk (dependabot/submodules/master/libs/h
bb_common-9e7696c)
$ git checkout master
M     libs/hbb_common
branch 'master' set up to track 'origin/master'.
Switched to a new branch 'master'

Administrator@ebs-177905 MINGW64 ~/rustdesk (master)
$ |
```

确认子模块路径

```
git config --file .gitmodules --get-regexp path
```

或者使用如下命令：

```
git submodule status
```

```
Administrator@ebs-177905 MINGW64 ~/rustdesk (master)
$ git config --file .gitmodules --get-regexp path
submodule.libs/hbb_common.path libs/hbb_common
```

如上图，libs/hbb_common 就是子模块的路径，下面的语句要用到这个路径

移除对 hbb_common 子模块的跟踪（保留文件）

```
git rm --cached libs/hbb_common
```

```
Administrator@ebs-177680 MINGW64 ~/rustdesk (master)
$ git rm --cached libs/hbb_common
rm 'libs/hbb_common'
```

执行后，.gitmodules 文件中 hbb_common 的配置会被自动删除

接下来删除 hbb_common 自身的 Git 仓库数据（使其失去独立仓库属性）

```
rm -rf libs/hbb_common/.git
```

```
Administrator@ebs-177680 MINGW64 ~/rustdesk (master)
$ rm -rf libs/hbb_common/.git
```

删除主仓库中存储的 hbb_common 子模块配置

```
rm -rf .git/modules/libs/hbb_common
```

```
Administrator@ebs-177680 MINGW64 ~/rustdesk (master)
$ rm -rf .git/modules/libs/hbb_common
```

将转换后的 libs/hbb_common 作为普通目录添加到主仓库

```
git add libs/hbb_common
```

```
Administrator@ebs-177680 MINGW64 ~/rustdesk (master)
$ git add libs/hbb_common
```

提交修改

```
git commit -m "将子模块 libs/hbb_common 转换为普通目录"
```

```
Administrator@ebs-177680 MINGW64 ~/rustdesk (master)
$ git commit -m "将子模块 libs/hbb_common 转换为普通目录"
[master 982880627] 将子模块 libs/hbb_common 转换为普通目录
 28 files changed, 9706 insertions(+), 1 deletion(-)
 delete mode 160000 libs/hbb_common
 create mode 100644 libs/hbb_common/.gitignore
 create mode 100644 libs/hbb_common/Cargo.toml
 create mode 100644 libs/hbb_common/build.rs
```

推送更改到 Github 仓库的 master 分支

```
git push origin master
```

```
Administrator@ebs-177680 MINGW64 ~/rustdesk (master)
$ git push origin master
Enumerating objects: 38, done.
Counting objects: 100% (38/38), done.
Delta compression using up to 32 threads
Compressing objects: 100% (34/34), done.
Writing objects: 100% (36/36), 72.77 KiB | 642.00 KiB/s, done.
Total 36 (delta 2), reused 14 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To github.com: [REDACTED]/rustdesk.git
 527730094..982880627 master -> master
```

推送完成，查看 GitHub 相应的仓库，子模块的箭头标识已经没有了，`hbb_common` 已经变成了普通文件夹。(注意如果 GitHub 默认不是 master 分支，要切换到 master 分支查看)

The screenshot shows a GitHub repository interface for the 'rustdesk' project. The left sidebar shows a tree view of the repository structure under the 'master' branch. A red box highlights the 'libs' folder, which contains submodules like 'clipboard', 'enigo', and 'hbb_common'. The 'hbb_common' submodule is specifically highlighted with a red border. The main pane displays the commit history for the 'libs/hbb_common' directory. The commits are as follows:

Name	Last commit message	Date
..	feat: clipboard files, audit (#12730)	last month
clipboard	fix: osx, reset modifiers' state after locking screen (#11806)	4 months ago
enigo	将子模块 libs/hbb_common 转换为普通目录	1 minute ago
hbb_common	feat: file transfer, resume (#12626)	last month
portable	feat: remote printer (#11231)	6 months ago
remote_printer	Remove needless macros format! (#11456)	2 weeks ago
scrap	refact: remove unnecessary printing (#12000)	3 months ago
virtual_display		