

下载docker安装应用

<https://www.docker.com/products/docker-desktop/>

Docker Desktop

Install Docker Desktop – the fastest way to containerize applications.

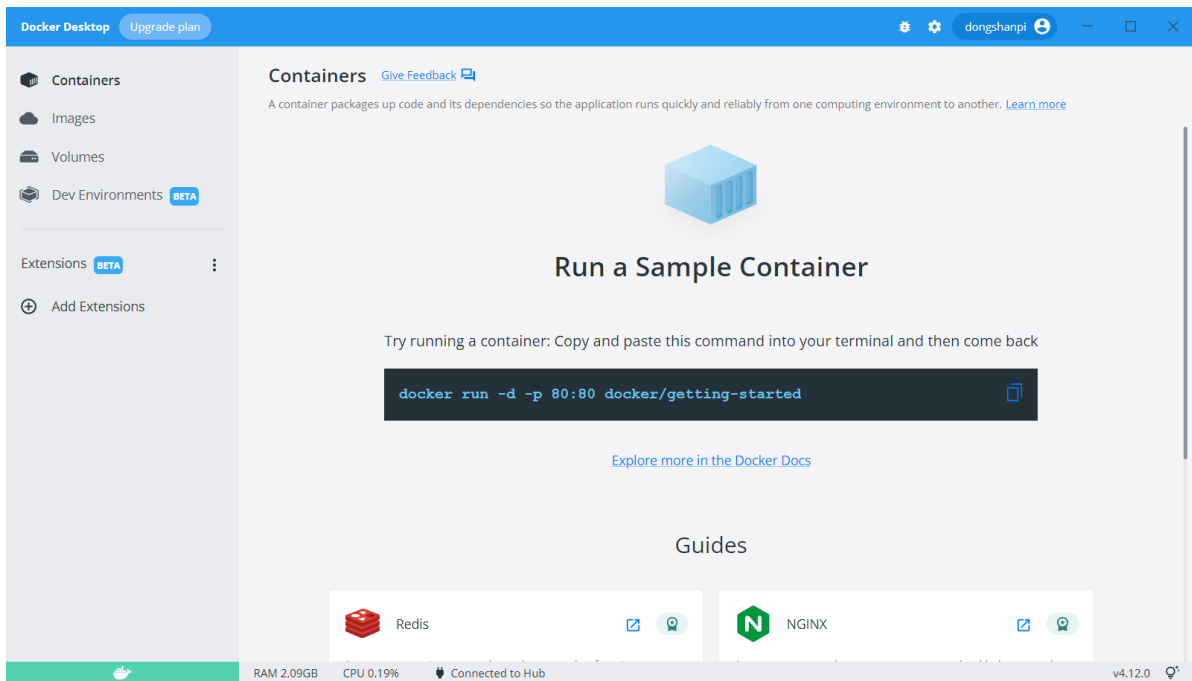
Mac with Intel Chip

Mac with Apple Chip

MOST COMMON

Also available for [Windows](#) and [Linux](#)

安装完成后界面



安装配置VSCode

下载安装vscode

<https://code.visualstudio.com/Download>

Download Visual Studio Code

Free and built on open source. Integrated Git, debugging and extensions.

The download section is divided into three main categories: Windows, Linux, and Mac. A red arrow points from the top right towards the Windows section.

- Windows** (Windows 8, 10, 11):
 - User Installer: 64 bit, 32 bit, ARM
 - System Installer: 64 bit, 32 bit, ARM
 - .zip: 64 bit, 32 bit, ARM
- Linux** (Debian, Ubuntu, Red Hat, Fedora, SUSE):
 - .deb: 64 bit, ARM, ARM 64
 - .rpm: 64 bit, ARM, ARM 64
 - .tar.gz: 64 bit, ARM, ARM 64
 - Snap Store
- Mac** (macOS 10.11+):
 - .zip: Universal, Intel Chip, Apple Silicon

安装Docker插件

The screenshot shows the Visual Studio Code interface with the extension marketplace open. The search bar contains '在应用商店中搜索扩展'. The '已安装' (Installed) section is expanded, showing several extensions. The 'Docker' extension by Microsoft is highlighted with a red box. To the right, a large, faint VS Code logo is visible. Below the logo, there are keyboard shortcuts for various actions:

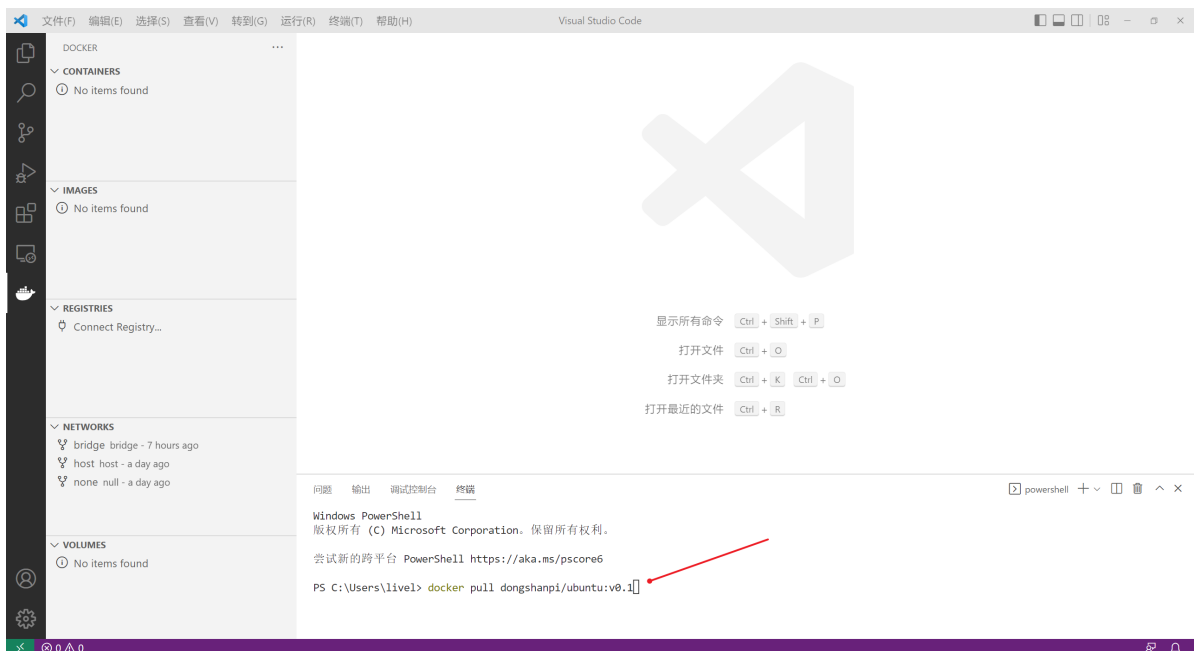
- 显示所有命令: `Ctrl + Shift + P`
- 打开文件: `Ctrl + O`
- 打开文件夹: `Ctrl + K` `Ctrl + O`
- 打开最近的文件: `Ctrl + R`

加载配套的docker镜像

在vscode主界面，按下键盘上的 `ctrl + `` 键，即可唤出终端界面。



在终端界面下输入 `docker pull dongshanpi/ubuntu:v0.1` 命令来获取镜像



等待拉取完成

版权所有 (C) Microsoft Corporation。保留所有权利。

尝试新的跨平台 PowerShell <https://aka.ms/pscore6>

```
PS C:\Users\livel> docker pull dongshanpi/ubuntu:v0.1
v0.1: Pulling from dongshanpi/ubuntu
726b8a513d66: Pull complete
e269db5d475a: Extracting [=====>
] 515.8MB/1.082GB
□
```

拉取完成示意图

```
PS C:\Users\livel> docker pull dongshanpi/ubuntu:v0.1
v0.1: Pulling from dongshanpi/ubuntu
726b8a513d66: Pull complete
e269db5d475a: Pull complete
Digest: sha256:ce15eed60dfc52f25424c496e1735ca93704bc163769c2e2b970c0ec6130d8fb
Status: Downloaded newer image for dongshanpi/ubuntu:v0.1
docker.io/dongshanpi/ubuntu:v0.1
PS C:\Users\livel> □
```

此时点击vscode左侧 Docker鲸鱼图标，可以看到已经出来一个镜像

DOCKER ...

▼ **CONTAINERS**

ⓘ No items found

▼ **IMAGES**

> 📄 dongshanpi/ubuntu

▼ **REGISTRIES**

🔑 Connect Registry...

▼ **NETWORKS** + ☰ ⚙️ ↻ ?

- 🔗 bridge bridge - 7 hours ago
- 🔗 host host - a day ago
- 🔗 none null - a day ago



▼ **CONTAINERS**

ⓘ No items found

▼ **IMAGES**

> 📄 dongshanpi/ubuntu

▼ **REGISTRIES**

🔑 Connect Registry...

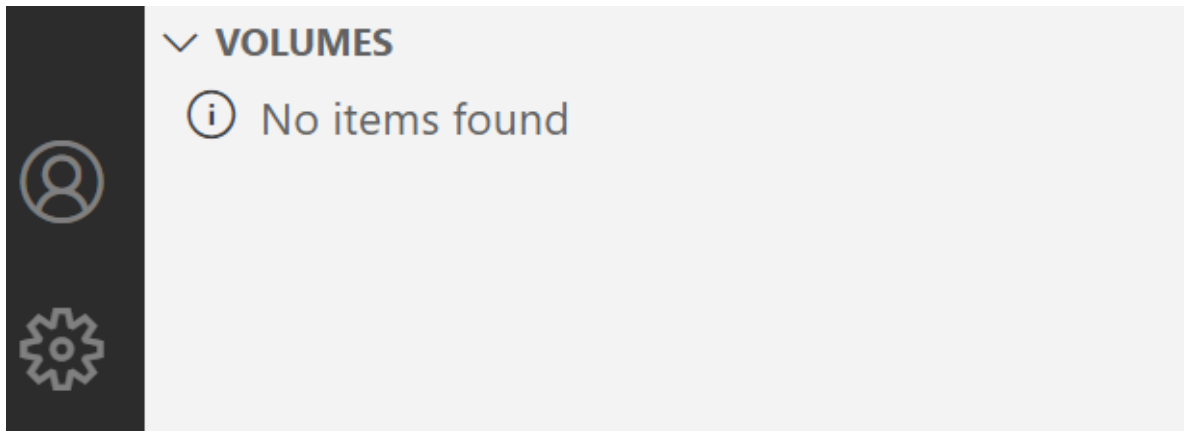
▼ **NETWORKS**



🔗 bridge bridge - 7 hours ago

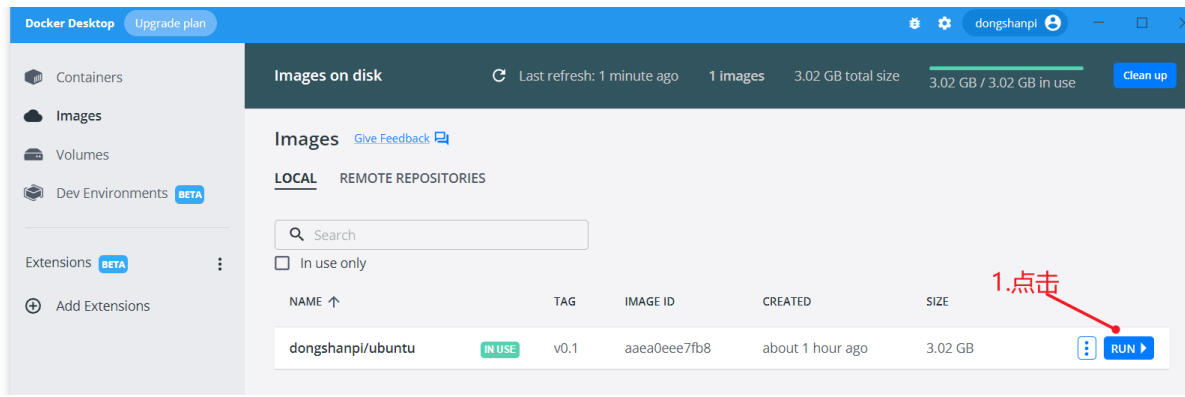
🔗 host host - a day ago

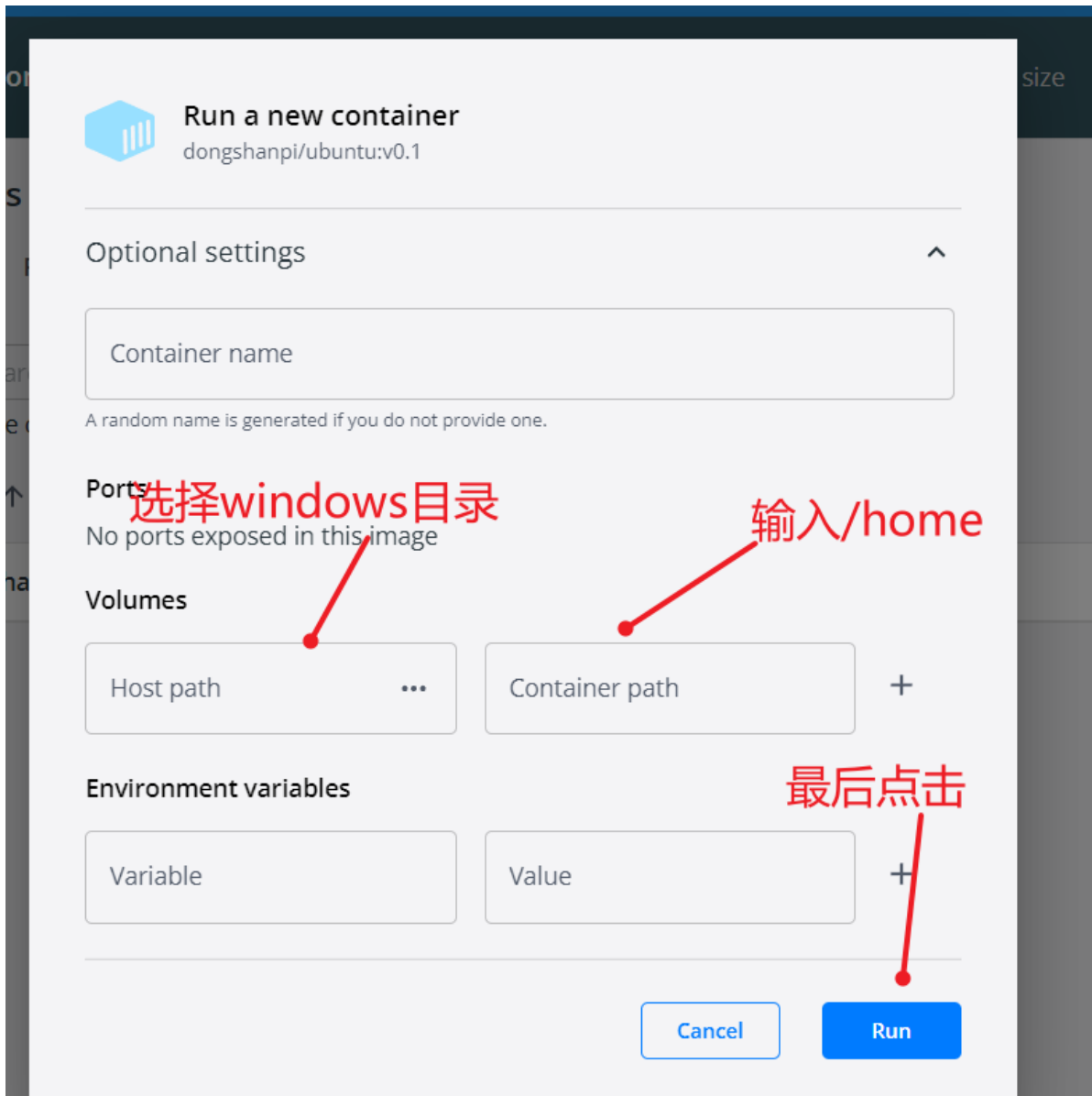
🔗 none null - a day ago



映射windows目录到Docker镜像内

切换到Docker Desktop软件界面，点击您已经获取到的 镜像文件

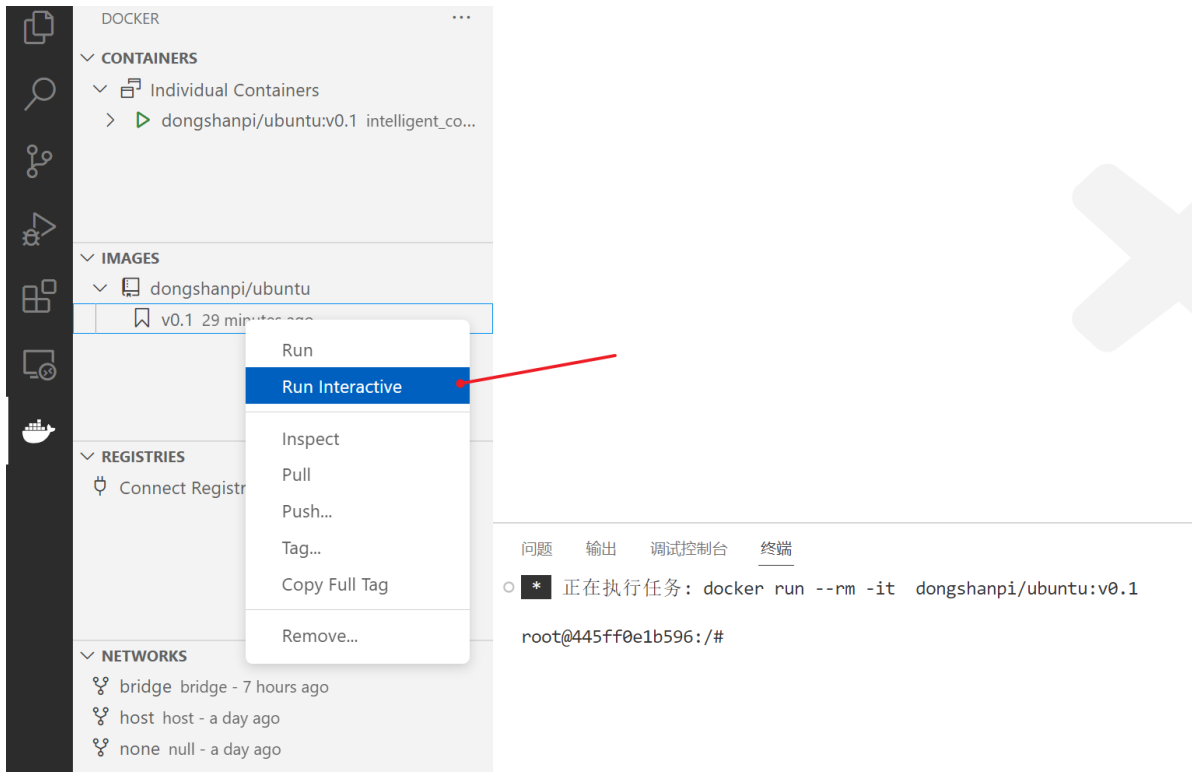




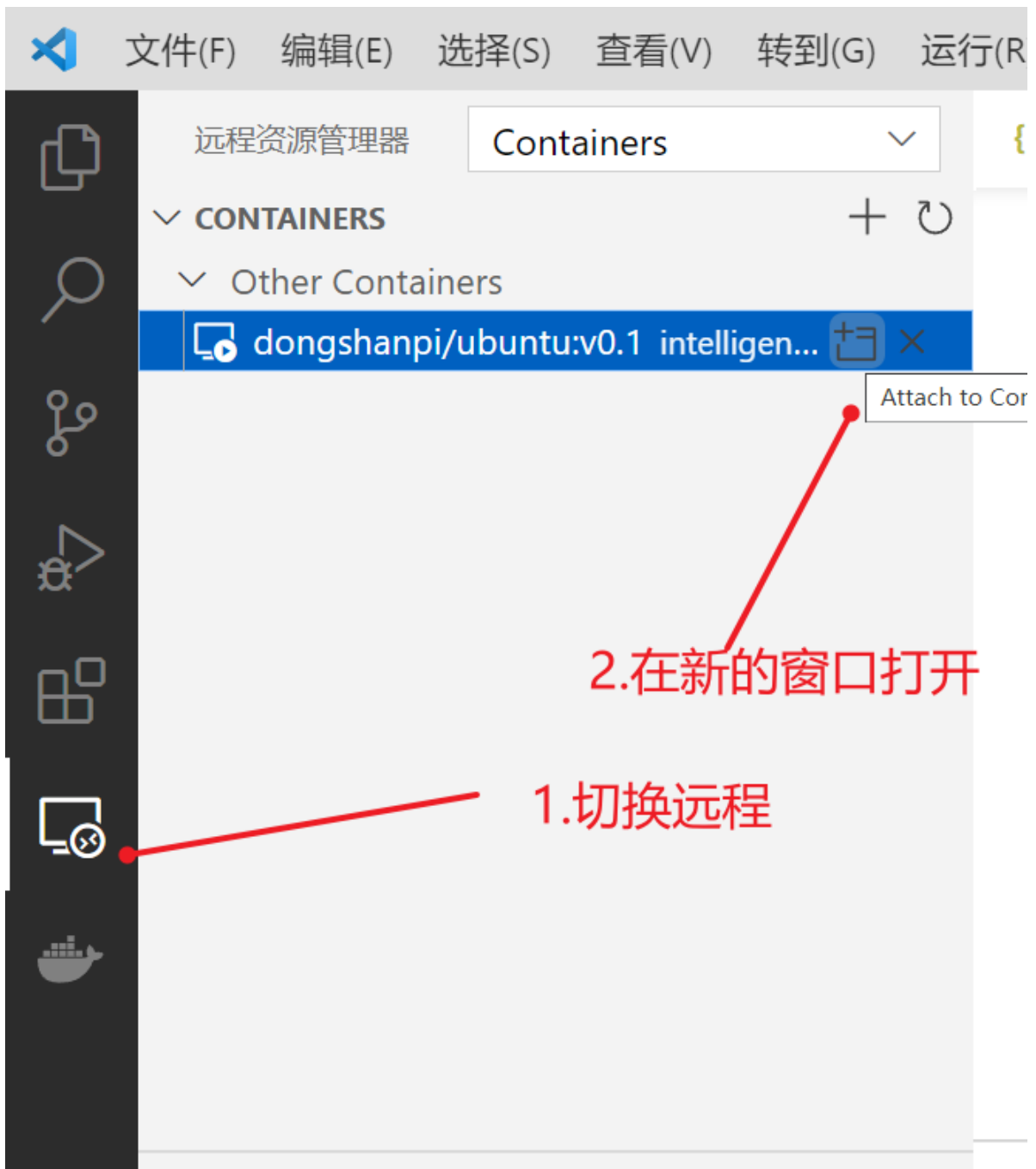
当然 也可以设置你要映射的目录，注意我的 windows 把D盘Dokcer目录 映射到了 ubuntu镜像内的 /home目录下。建议大家也在D盘目录下创建一个 Docker文件夹，用于存放工程文件。

启动Dokcer镜像

接下来我们要启动此镜像，把它在容器中运行起来



然后切换到左侧的 远程主机图标位置，首先切换到 Containers 也就是容器窗口，然后点击在新的窗口打开



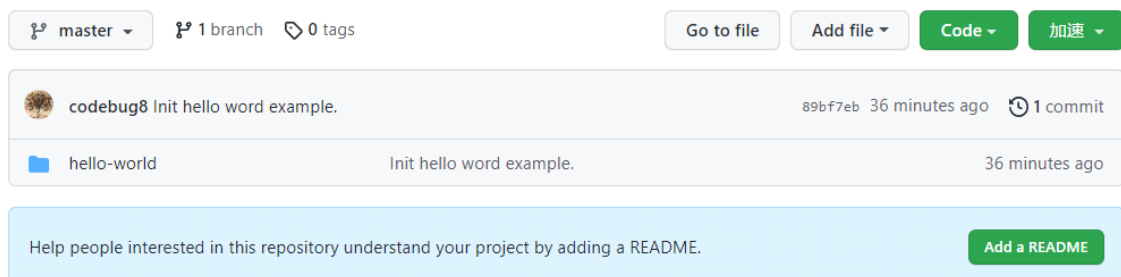
打开以后 就可以弹出一个打开对话框，我们就可以在这里面找到我们的工程 并打开使用了。



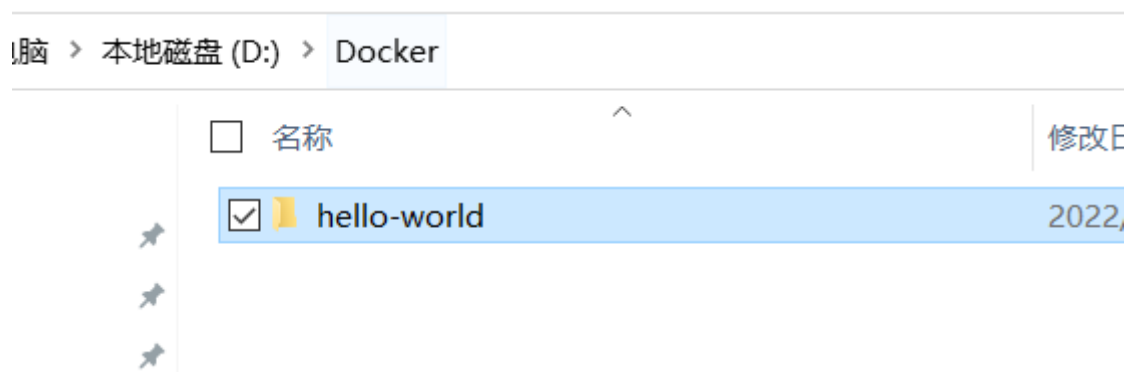
使用VScode编译hello word工程

在加载工程之前 我们需要先创建一个工程, 这里使用我们提供的 helloworld 模板。

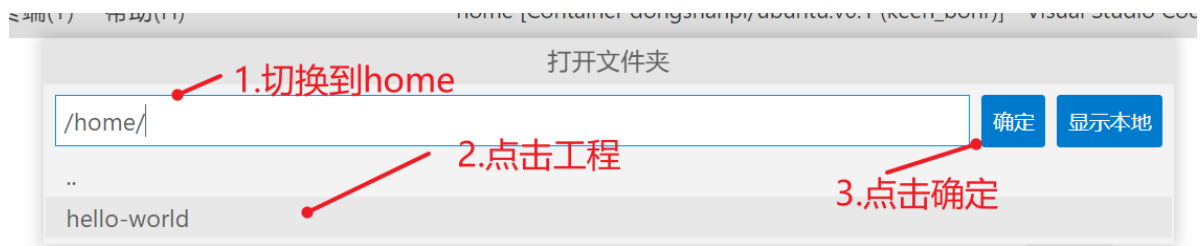
把这个示例工程整个下载下来 <https://github.com/100askTeam/Stage3-Components> 可以放在刚才映射的D盘Docker目录下



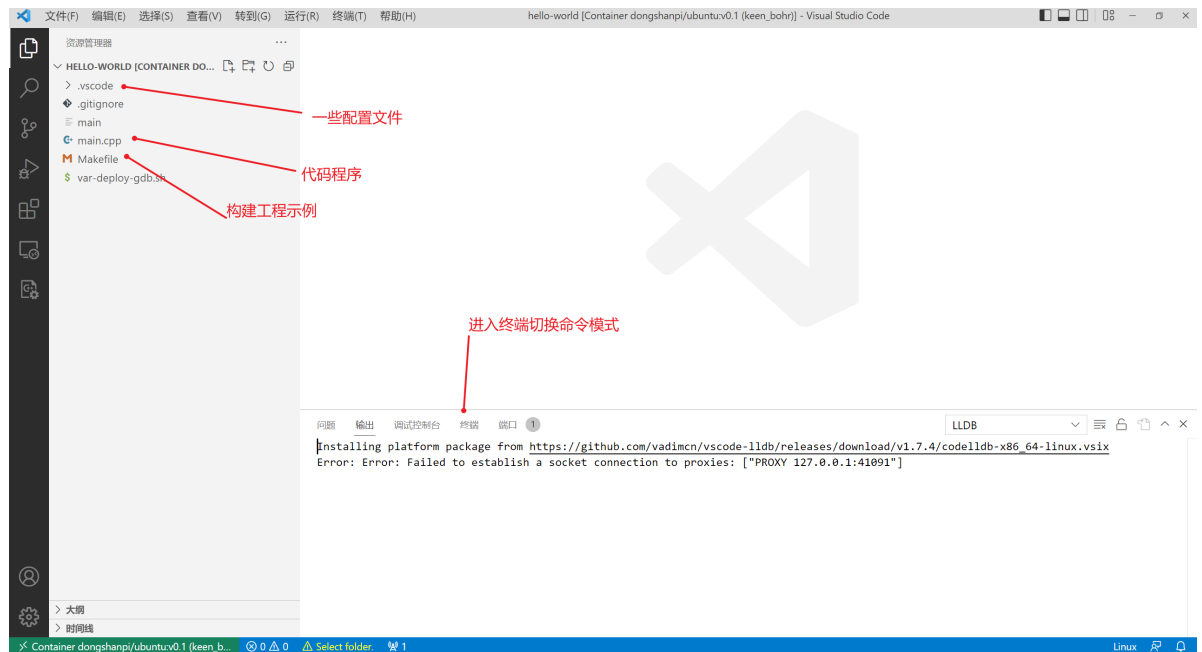
要确保你的工程在你映射的windows目录内



之后回到vscode界面，在打开文件夹哪里 找到/home目录，然后就可以看到你的工程了。点击确定就可以打开了



工程打开以后的主界面，我们可以切换到 终端模式。



接下来编译工程



如下图所示为编译工程输出的打印信息。

```
问题 输出 调试控制台 终端 端口 1 build - Task ✓ + ▢ 窗 ^ ×
● 正在执行任务: make clean; make -j$(nproc)

rm -f hello.bin
/opt/arm-buildroot-linux-gnueabihf_sdk-buildroot/bin/arm-buildroot-linux-gnueabihf-g++ --sysroot=/opt/arm-buildroot-linux-gnueabihf_sdk-buildro
ot/arm-buildroot-linux-gnueabihf/sysroot/ -Og main.cpp -g -o hello.bin
终端将被任务重用, 按任意键关闭。
```

开发板执行

设置开发板地址

因为我们的程序是通过网络ssh传输, 所以首先开发板上要可以联网 并且能使用ssh登录, 开发板需要登录的用户名设置为root登录密码为空。

如下示例 我得开发板 IP地址为 192.168.1.74 可以和我们的vscode windows主机通信

```
[root@100ask:~]# udhcpc
udhcpc: started, v1.31.1
udhcpc: sending discover
udhcpc: sending select for 192.168.1.74
udhcpc: lease of 192.168.1.74 obtained, lease time 86400
deleting routers
adding dns 192.168.1.1
[root@100ask:~]#
```

修改项目工程里面 .vscode/settings.json TARGET_IP为自己开发板获取到的IP地址。

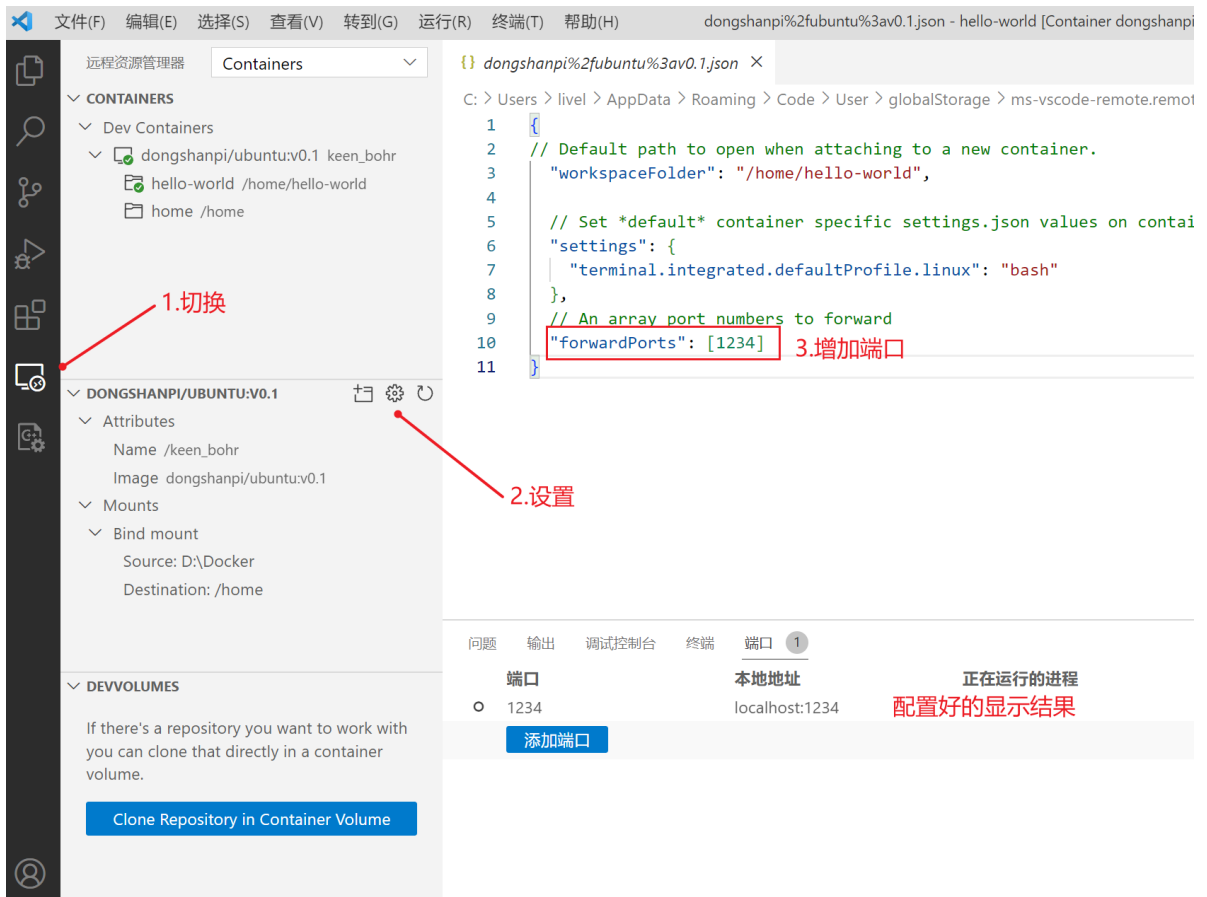
```
HELLO-WORLD [CONTAINER DO... | ↵ ⇄ 🔍 🗑
.vscode
  {} c_cpp_properties.json
  {} launch.json
  {} settings.json
  {} tasks.json
.gitignore
hello.bin
main
main.cpp
Makefile
$ var-deploy-gdb.sh

.vscode > {} settings.json > ...
1 {
2   "DONGSHANPI": {
3     /* Target Device Settings */
4     "TARGET_IP": "192.168.1.74", ← 修改为你的开发板 IP地址
5
6     /* Project Settings */
7     "PROGRAM": "hello.bin",
8
9     "ARCH": "arm-buildroot-linux-gnueabihf",
10    "SDKTARGETSYSROOT": "/opt/arm-buildroot-linux-gnueabihf_sdk-buildro
11
12    /* Yocto SDK Constants */
13    "CC_PREFIX": "/opt/arm-buildroot-linux-gnueabihf_sdk-buildroot/bin/
14    "CXX": "${config:DONGSHANPI.CC_PREFIX}g++ --sysroot=${config:DONGSH
15    "CC": "${config:DONGSHANPI.CC_PREFIX}gcc --sysroot=${config:DONGSHAN
16
17  }
```

远程调试开发板应用

使用GDB调试都是通过网络方式, 因为我们使用的容器下的镜像, 所以对外访问的端口需要重新映射一下才可以正常使用GDB调试功能, 这里需要参考下图 在我们运行的镜像内 增加一个网络端口

"forwardPorts": [1234]



增加完成后 使用 `ctrl + s`保存并退出，之后切换到工程页面，点击启动调试



VSCODE就会自动把程序上传到我们的开发板内，并执行调试了。

文件(F) 编辑(E) 选择(S) 查看(V) 转到(G) 运行(R) 终端(T) 帮助(H) main.cpp - hello-world [Container dongshanpi/ubuntu:0.1 (keen_bohr)] - Visual Studio Code

资源管理器

- HELLO-WORLD [CONTAINER DO...]
- vscode
- c_cpp_properties.json
- launch.json
- settings.json
- tasks.json
- .gitignore
- hello.bin
- main
- main.cpp
- Makefile
- var-deploy-gdb.sh

```
main.cpp > main(int, char * [])
1 #include <stdio.h>
2
3 int main(int argc, char *argv[]) {
4     printf("Hello, World!\n");
5     return 0;
6 }
7
```

问题 输出 调试控制台 终端 窗口 1

Hello, World!
Connection to 192.168.1.74 closed.
* 终端将被任务重用, 按任意键关闭。

正在执行任务: sh var-deploy-gdb.sh 192.168.1.74 hello.bin

Deploying to target
hello.bin 100% 11KB 1.7MB/s 00:00
Starting GDB Server on Target
Process /root/hello.bin created; pid = 483
Listening on port 1234
Remote debugging from host 192.168.1.8
[]

大纲
附属线