

EE211 24Fall Lab2



南方科技大学
SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

Content

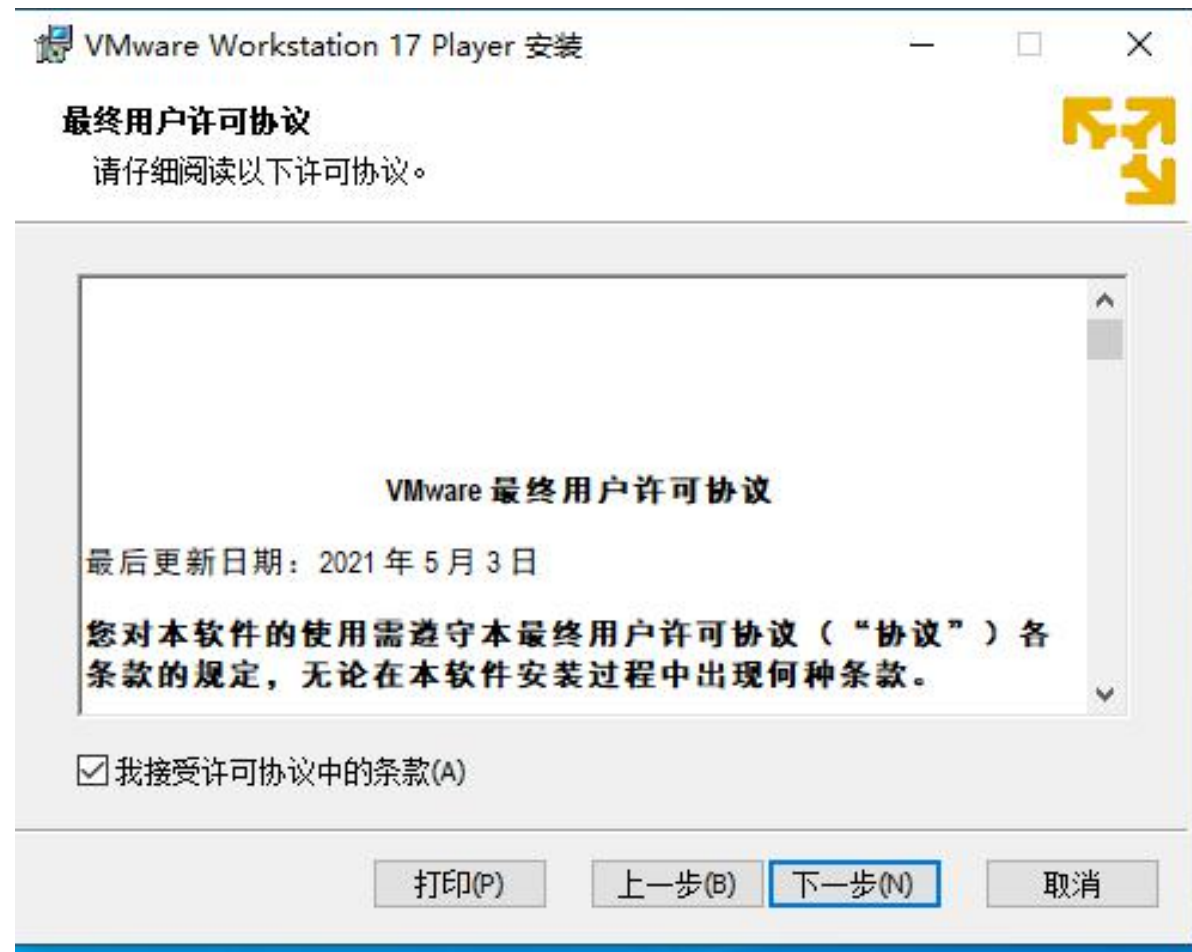
1 Prepare your OS

2 ROS2

For better development experience,
using a local ubuntu machine is recommended!

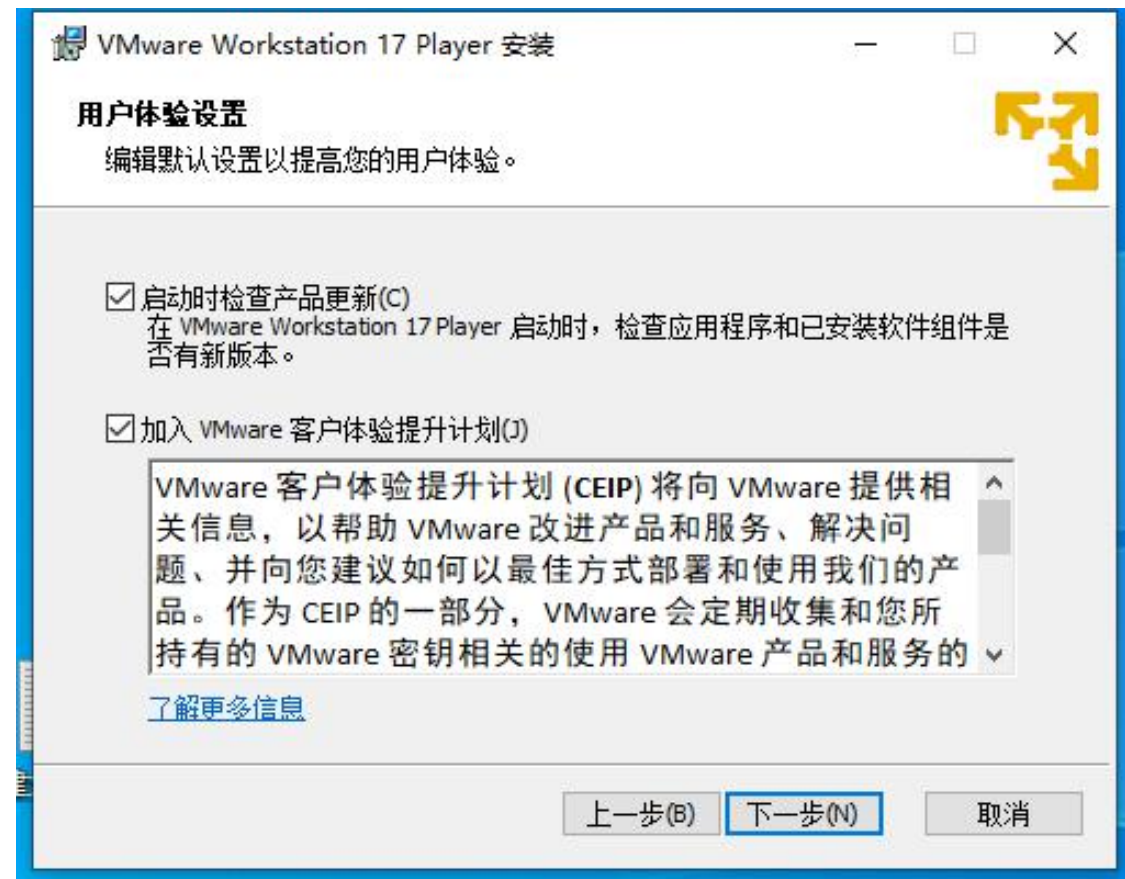
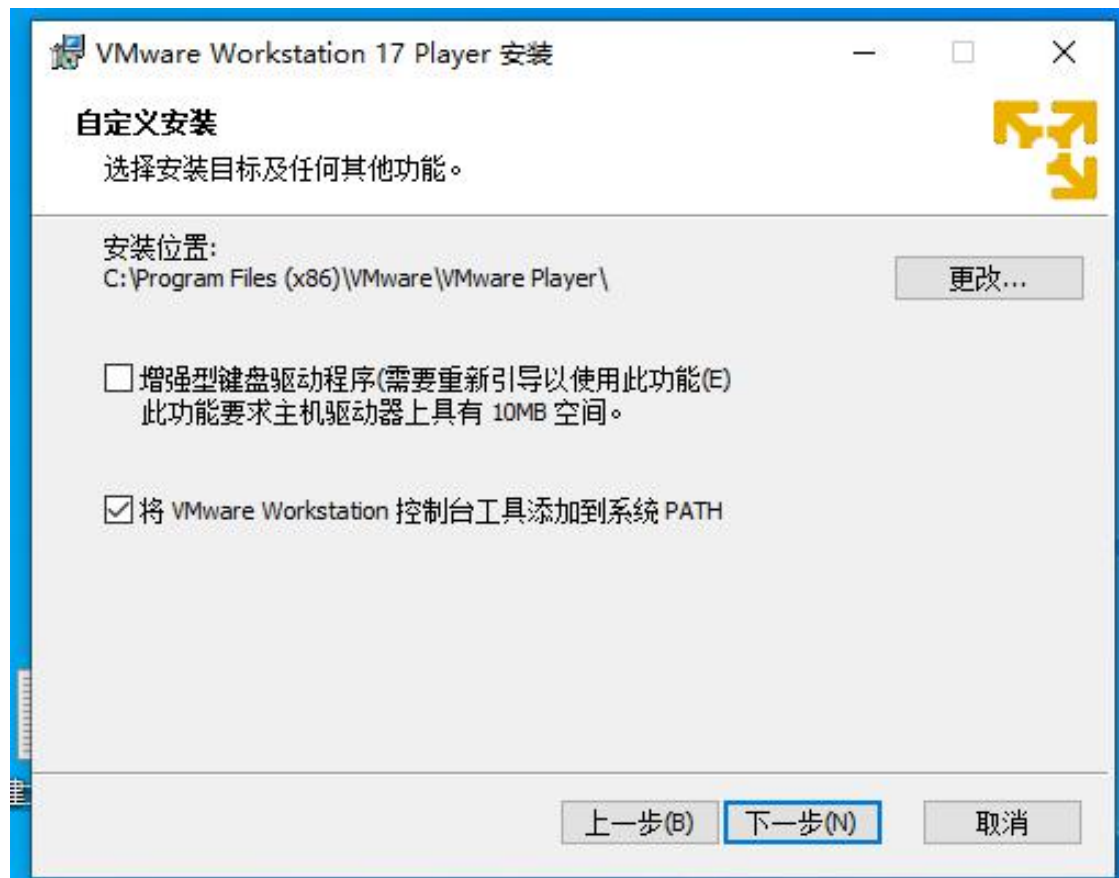
You may install a separate ubuntu os on your windows PC, so that you
can using win and ubuntu on the same computer!

1 Prepare your OS - Virtual Machine

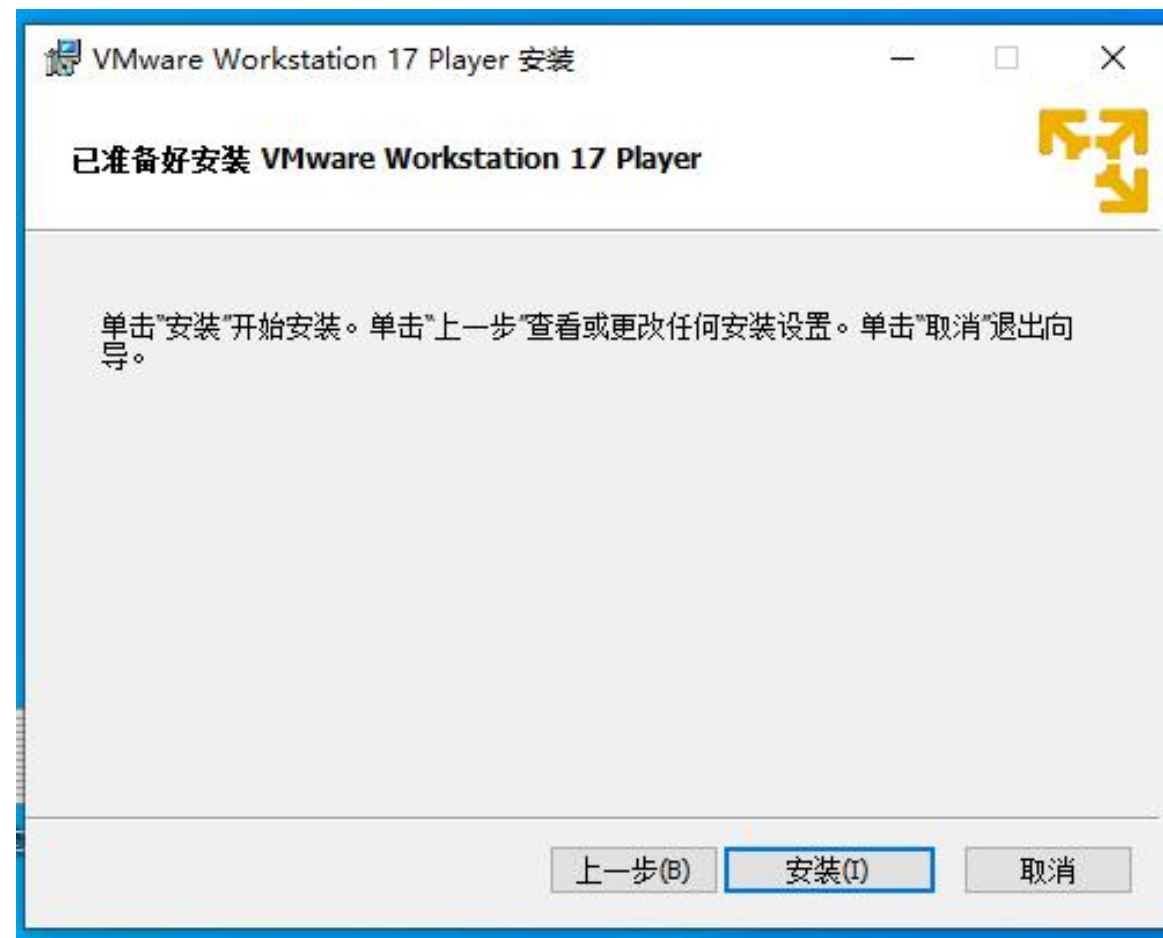
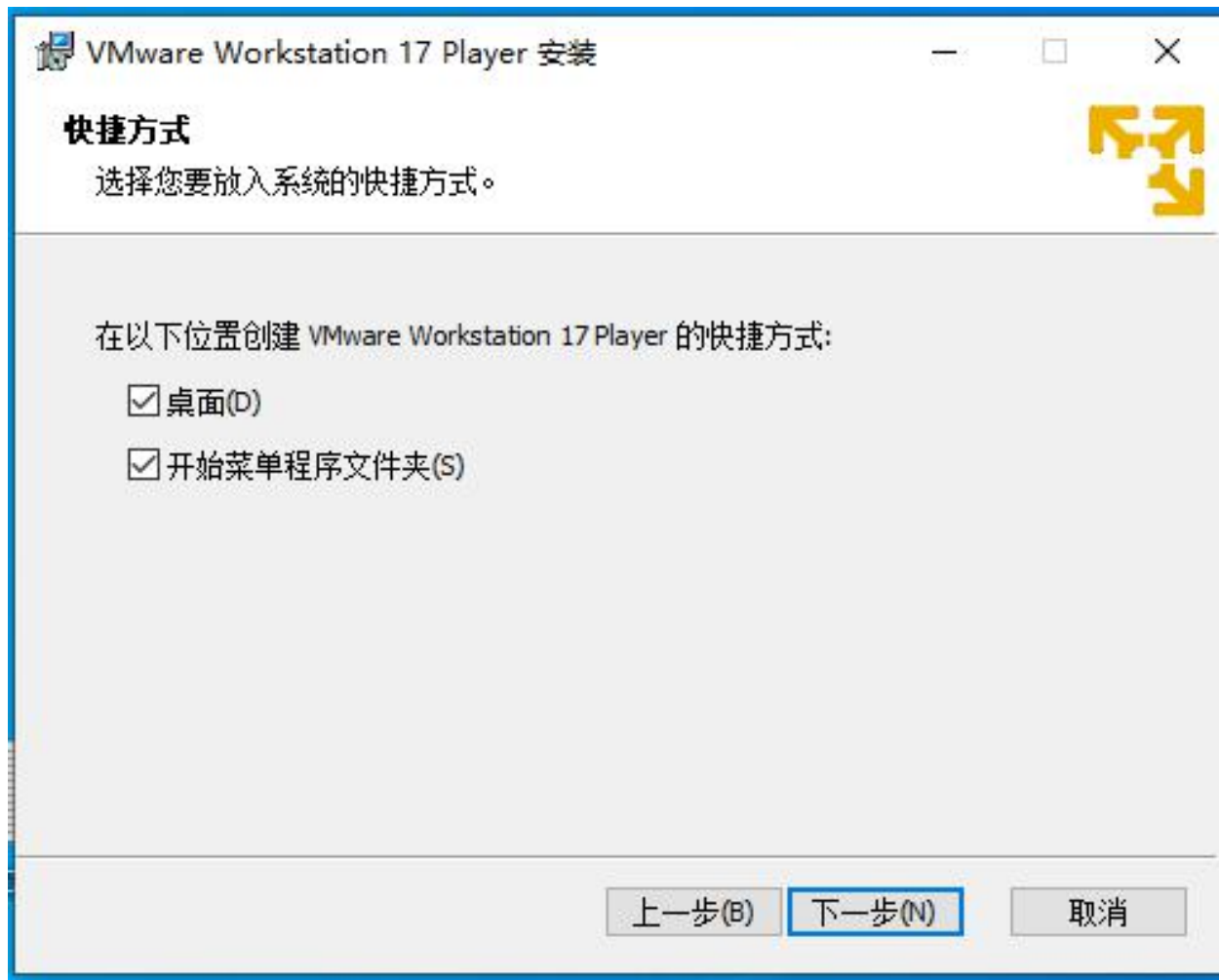


VMware Workstation Player - VMware Customer Connect

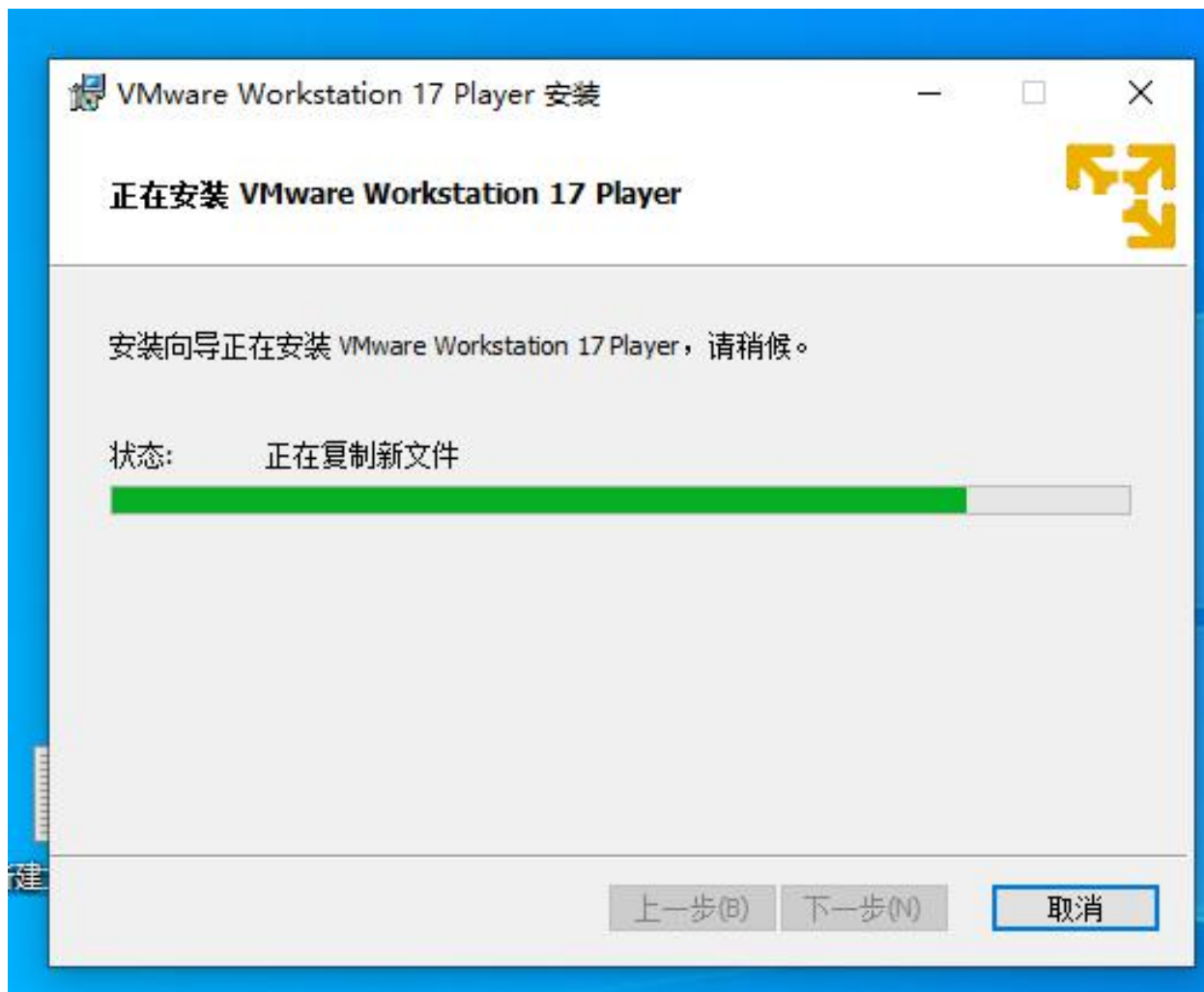
1 Prepare your OS - Virtual Machine



1 Prepare your OS - Virtual Machine



1 Prepare your OS - Virtual Machine



1 Prepare your OS - Virtual Machine



Prepare your OS - Virtual Machine



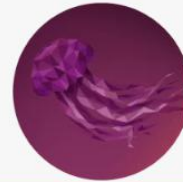
Ubuntu 22.04.3 LTS

下载专为桌面PC和笔记本精心打造的Ubuntu长期支持(LTS)版本。LTS意为“长期支持”，一般为5年。LTS版本将提供免费安全和维护更新至2027年4月。

[Ubuntu 22.04 LTS发布说明](#)

推荐的系统配置要求：

- ✓ 双核2 GHz处理器或更高
- ✓ 4 GB系统内存
- ✓ 25 GB磁盘存储空间
- ✓ 可访问的互联网
- ✓ 光驱或USB安装介质



下载 22.04.3

如需其他版本的Ubuntu桌面，包括BT下载，网络安装器，本地镜像站和历史版本，请访问 [其他下载](#)。

Ubuntu 23.04

专为桌面PC和笔记本精心打造的最新版Ubuntu操作系统，Ubuntu 23.04拥有9个月安全维护更新支持至2024年1月。

推荐的系统配置和要求与Ubuntu 22.04 LTS一致。

[Ubuntu 23.04 发布说明](#)

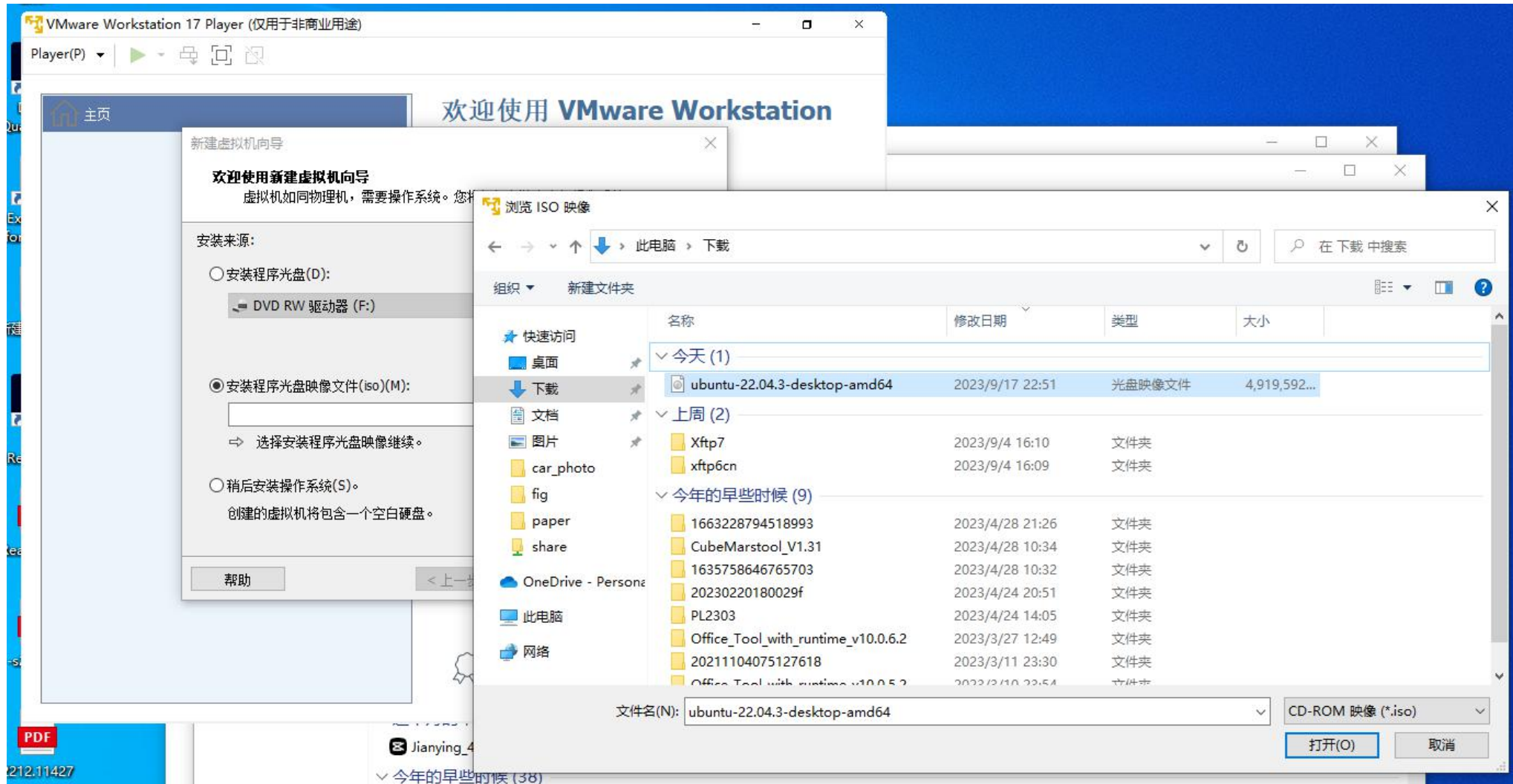


下载 23.04

下载 23.04 (旧版桌面安装器版本)

如需其他版本的Ubuntu桌面，包括BT下载，网络安装器，本地镜像站和历史版本，请访问 [其他下载](#)。

[下载Ubuntu桌面系统](#) | [Ubuntu](#)



新建虚拟机向导

欢迎使用新建虚拟机向导

虚拟机如同物理机，需要操作系统。您将如何安装客户机操作系统？

安装来源：

☐ 安装程序光盘(D):

DVD RW 驱动器 (F:)

☒ 安装程序光盘映像文件(iso)(M):

C:\Users\13120\Downloads\ubuntu-22.04.3-desktop-am

浏览(R)...

已检测到 Ubuntu 64 位 22.04.3。
该操作系统将使用简易安装。 [\(这是什么?\)](#)

☐ 稍后安装操作系统(S)。

创建的虚拟机将包含一个空白硬盘。

帮助

< 上一步(B)

下一步(N) >

取消

新建虚拟机向导

简易安装信息

这用于安装 Ubuntu 64 位。

个性化 Linux

全名(F): rpilab

用户名(U): rpi

密码(P): •

确认(C): •

帮助

< 上一步(B)

下一步(N) >

取消

1 Prepare your OS - Virtual Machine



新建虚拟机向导

命名虚拟机
您希望该虚拟机使用什么名称?

虚拟机名称(V):
Ubuntu 64 位

位置(L):
D:\Ubuntu 浏览(R)...

< 上一步(B) 下一步(N) > 取消

新建虚拟机向导

指定磁盘容量
磁盘大小为多少?

虚拟机的硬盘作为一个或多个文件存储在主机的物理磁盘中。这些文件最初很小，随着您向虚拟机中添加应用程序、文件和数据而逐渐变大。

最大磁盘大小 (GB)(S): 40.0

针对 Ubuntu 64 位的建议大小: 20 GB

☐ 将虚拟磁盘存储为单个文件(O)
☒ 将虚拟磁盘拆分成多个文件(M)
拆分磁盘后，可以更轻松地在计算机之间移动虚拟机，但可能会降低大容量磁盘的性能。

帮助 < 上一步(B) 下一步(N) > 取消

新建虚拟机向导

已准备好创建虚拟机
单击“完成”创建虚拟机，并开始安装 Ubuntu 64 位 和 VMware Tools。

将使用下列设置创建虚拟机：

| | |
|--------|--------------------------------------|
| 名称： | Ubuntu 64 位 |
| 位置： | D:\Ubuntu |
| 版本： | Workstation 17.x |
| 操作系统： | Ubuntu 64 位 |
| 硬盘： | 40 GB, 拆分 |
| 内存： | 4096 MB |
| 网络适配器： | NAT |
| 其他设备： | 2 个 CPU 内核, CD/DVD, USB 控制器, 打印机, 声卡 |

[自定义硬件\(C\)...](#)

☒ 创建后开启此虚拟机(P)

[< 上一步\(B\)](#) [完成](#) [取消](#)

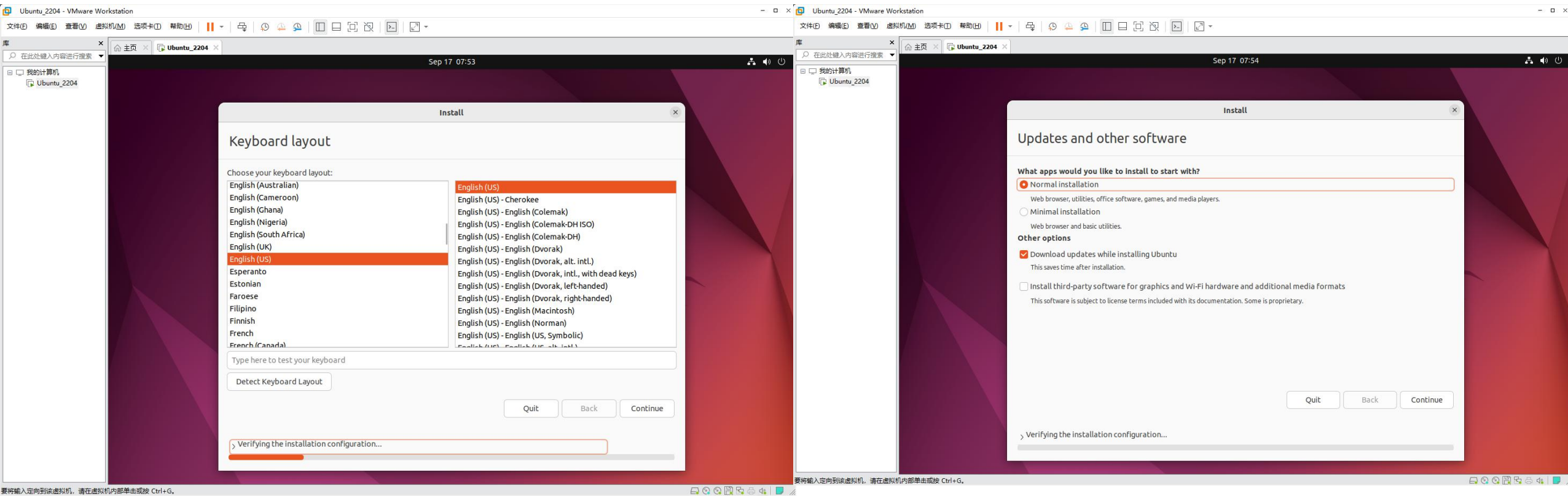
Ubuntu 64 位 - VMware Workstation 17 Player (仅用于非商业用途)

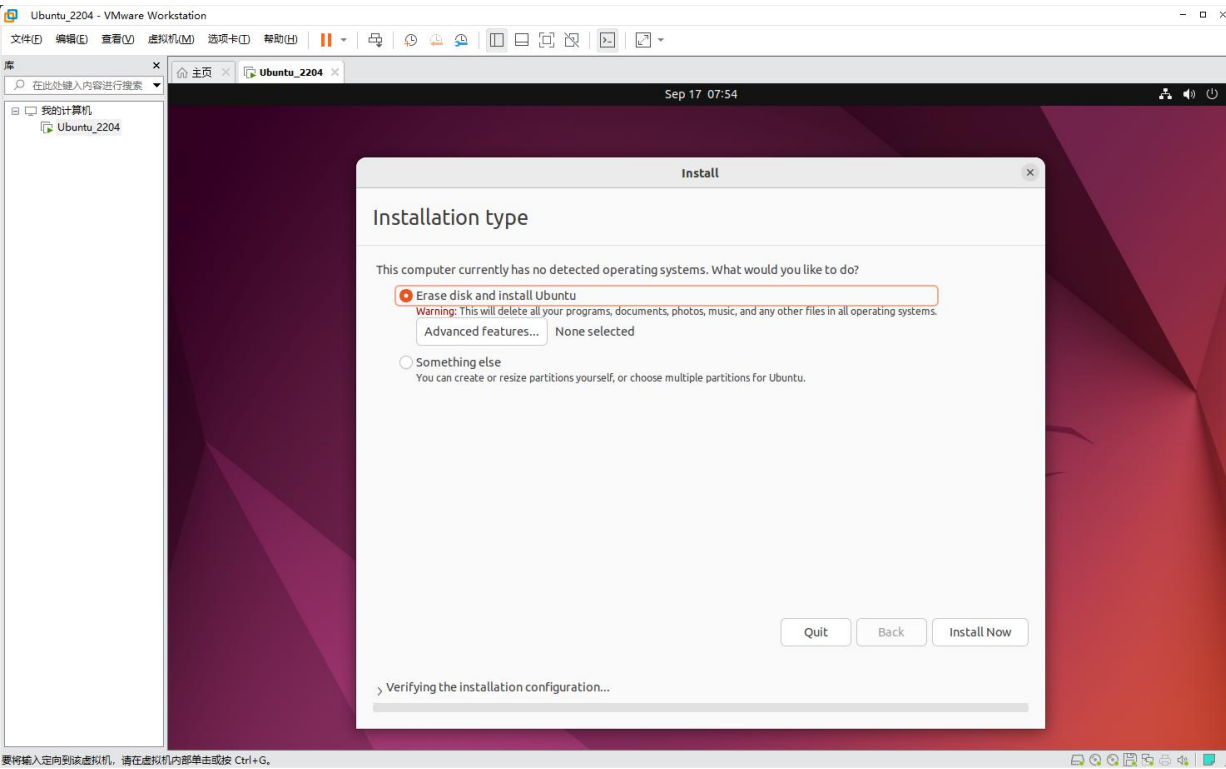
Player(P) | [Pause] [Full Screen] [Help]

单击虚拟屏幕可发送按键

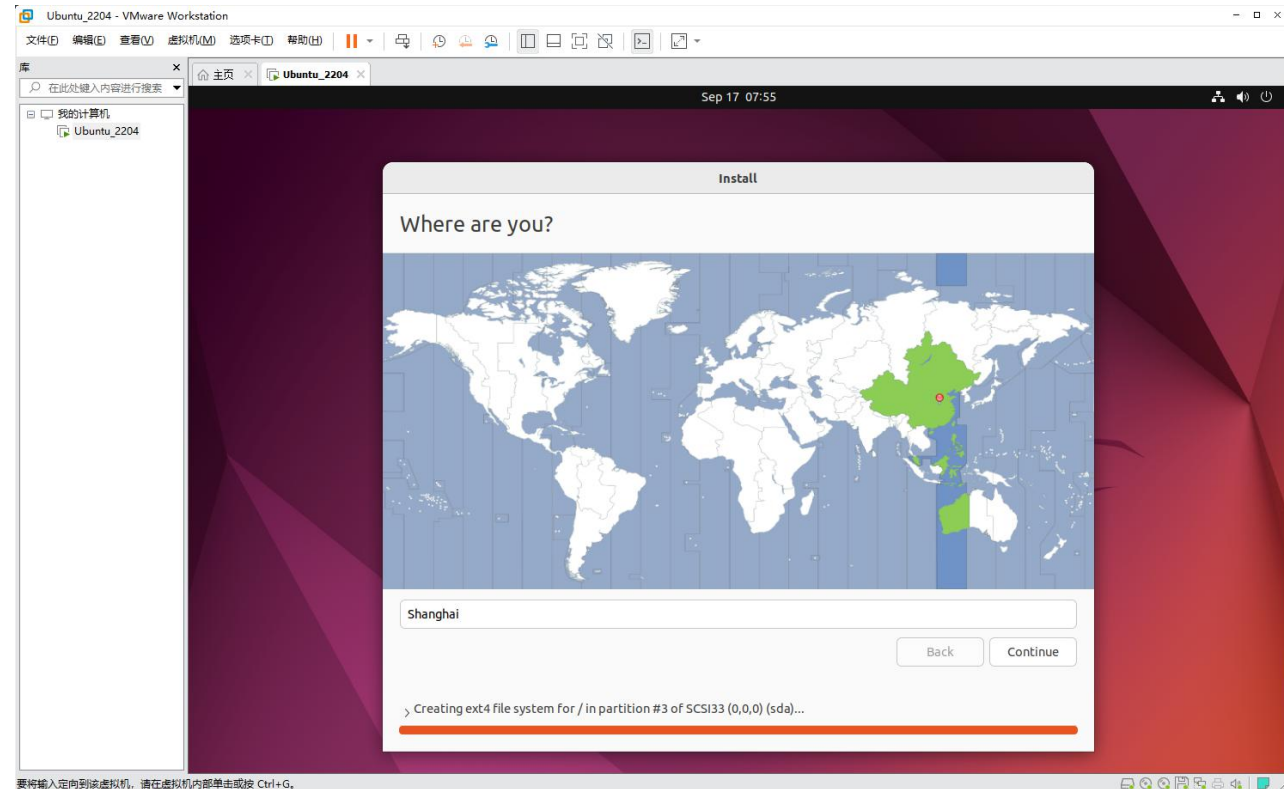
简易安装正在安装 Ubuntu 64 位。Ubuntu14.04 及更高版本需要 Internet 连接才能安装 open-vm-tools。

[帮助](#)



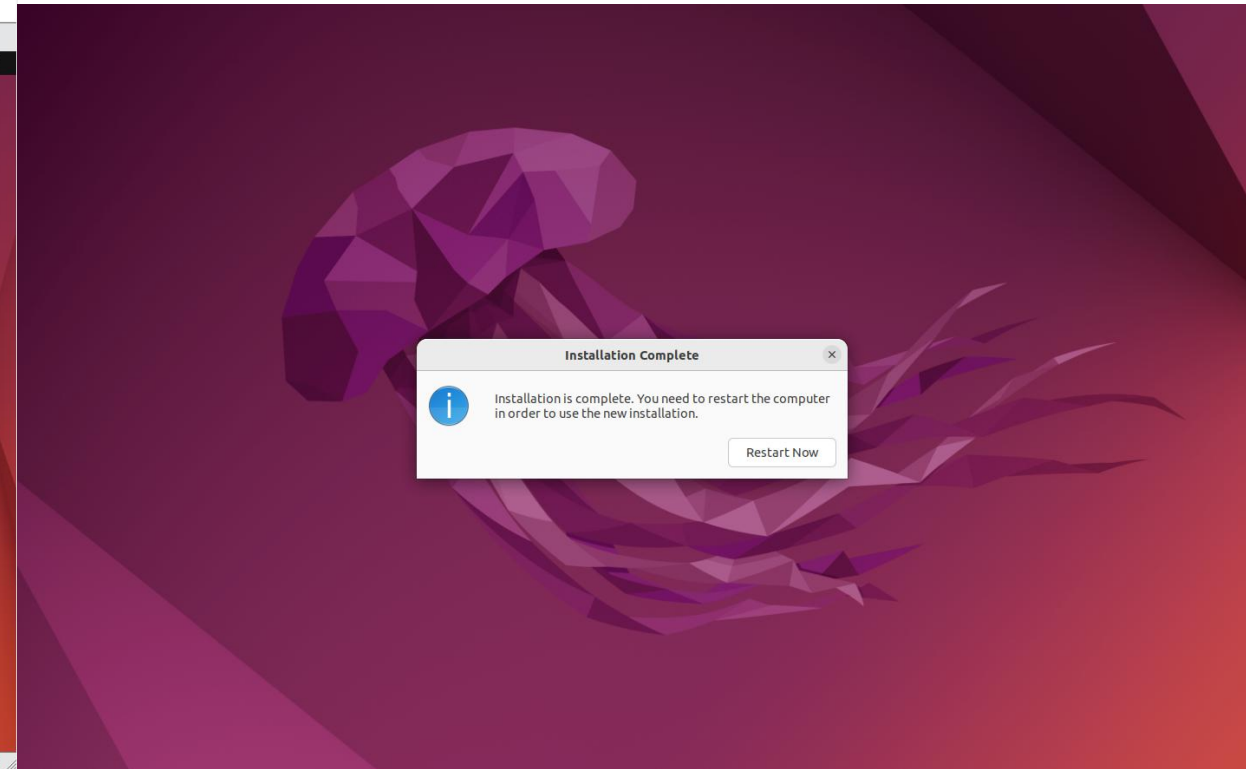
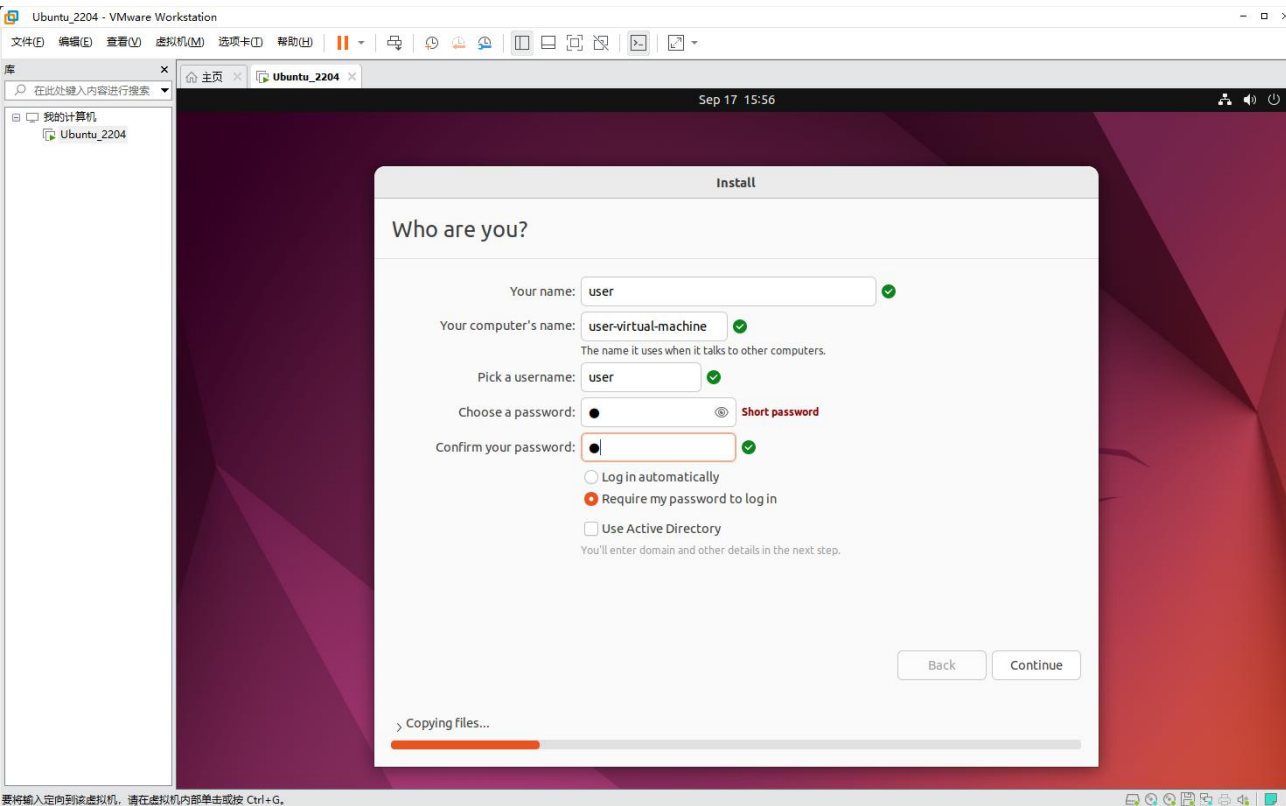


要将输入定向到该虚拟机，请在虚拟机内部单击或按 Ctrl+G。

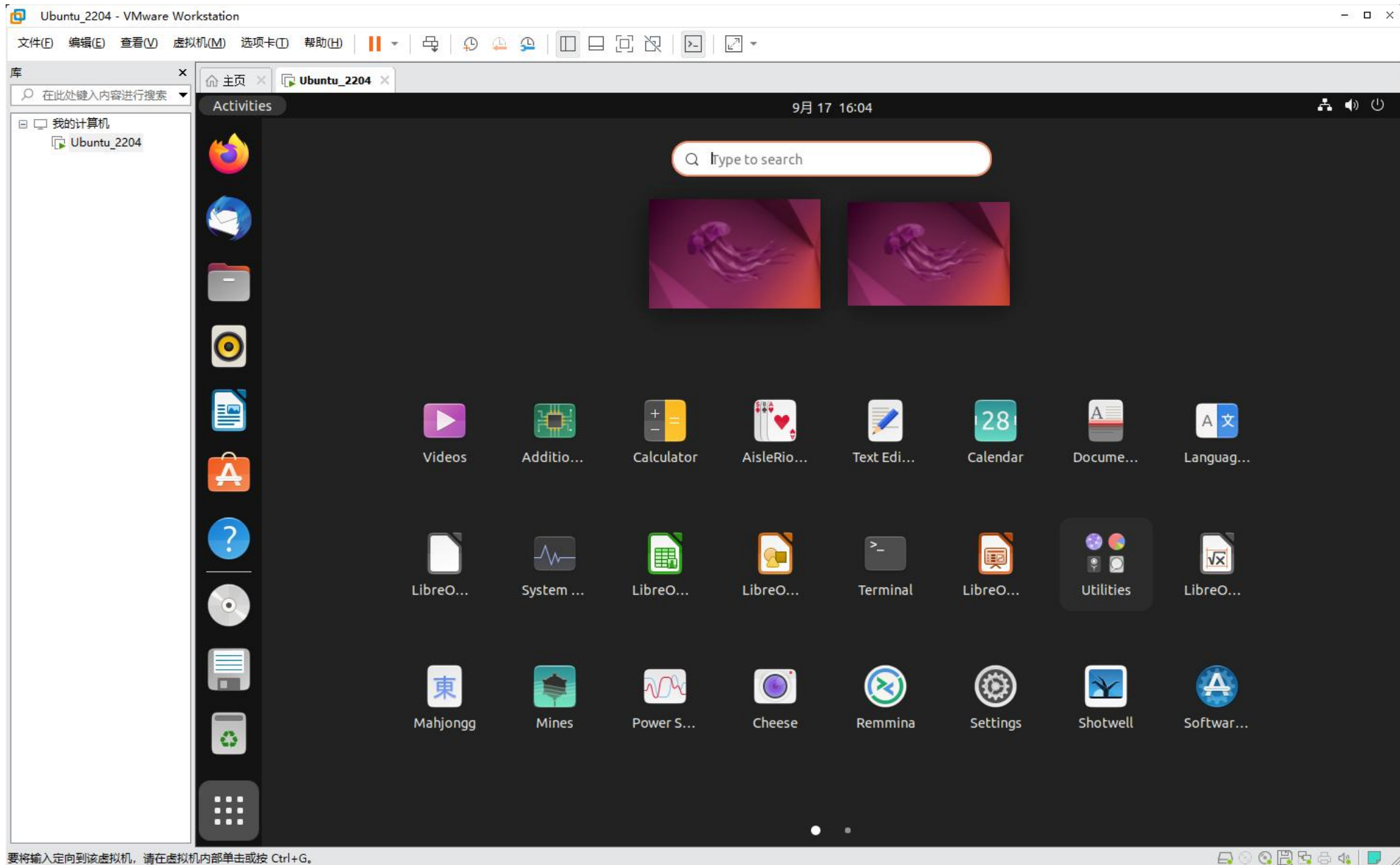


要将输入定向到该虚拟机，请在虚拟机内部单击或按 Ctrl+G。

Prepare your OS - Virtual Machine



Prepare your OS - Virtual Machine



1. Setting up Encoding

```
$ sudo apt update && sudo apt install locales  
$ sudo locale-gen en_US en_US.UTF-8  
$ sudo update-locale LC_ALL=en_US.UTF-8LANG=en_US.UTF-8  
$ export LANG=en_US.UTF-8
```

2. Setting apt source (<https://mirror.tuna.tsinghua.edu.cn/help/ros2/>)

```
$ sudo apt install curl gnupg2  
$ sudo curl -sSL https://raw.githubusercontent.com/ros/rosdistro/master/ros.key -o  
/usr/share/keyrings/ros-archive-keyring.gpg  
$ sudo echo "deb [arch=$(dpkg --print-architecture) signed-  
by=/usr/share/keyrings/ros-archive-keyring.gpg]  
https://mirrors.tuna.tsinghua.edu.cn/ros2/ubuntu jammy main" | sudo tee  
/etc/apt/sources.list.d/ros2.list > /dev/null  
$ sudo apt update  
$ sudo apt upgrade
```

1. Install

```
$ sudo apt install ros-humble-desktop  
$ sudo apt install python3-colcon-common-extensions python3-argcomplete python3-rosdep  
$ sudo apt install git  
$ echo " source /opt/ros/humble/setup.bash" >> ~/.bashrc
```

2. Uninstall

```
$ sudo apt remove ros-humble-*
```

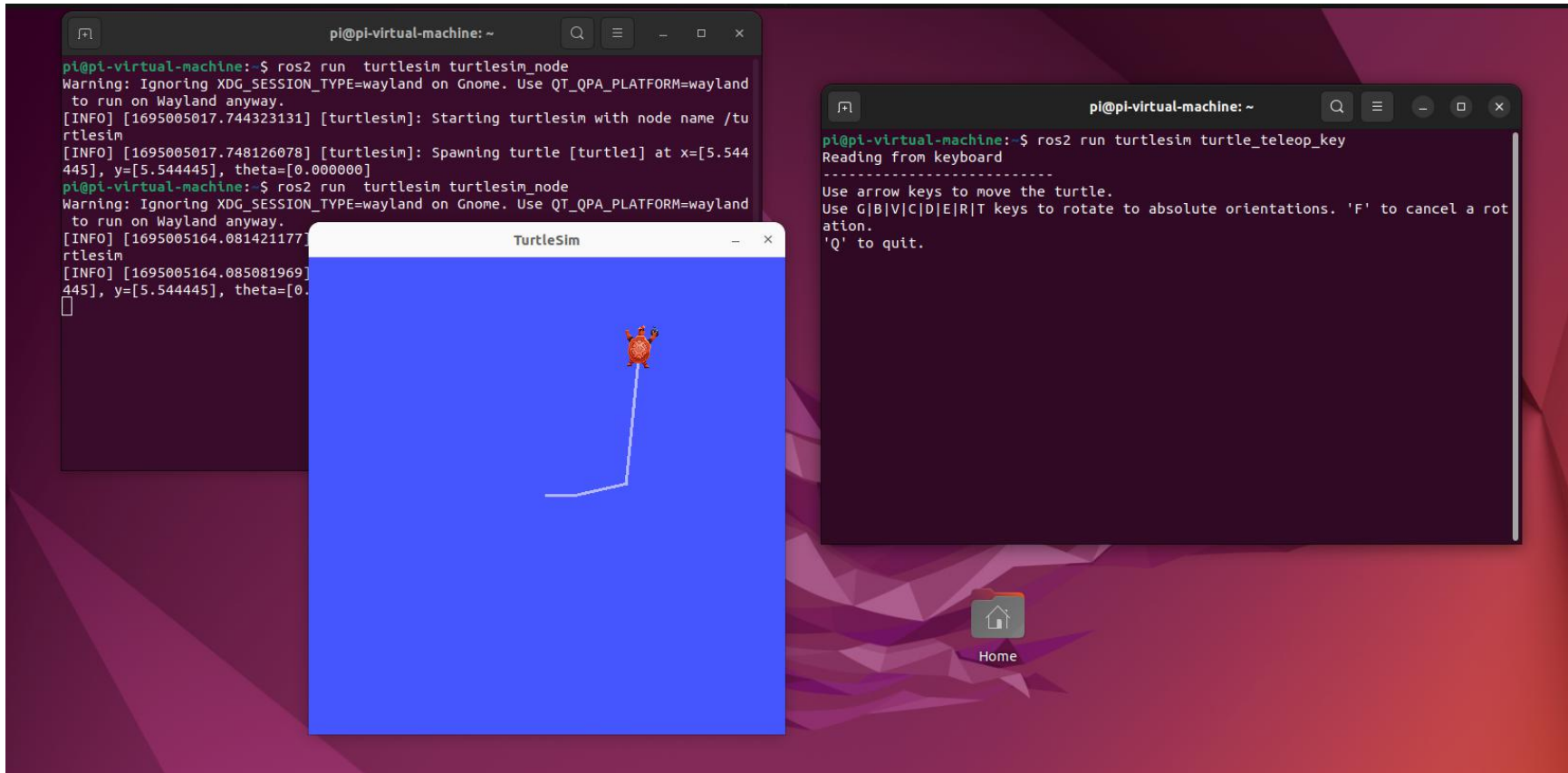
*

```
$ wget http://fishros.com/install -O fishros && . fishros
```

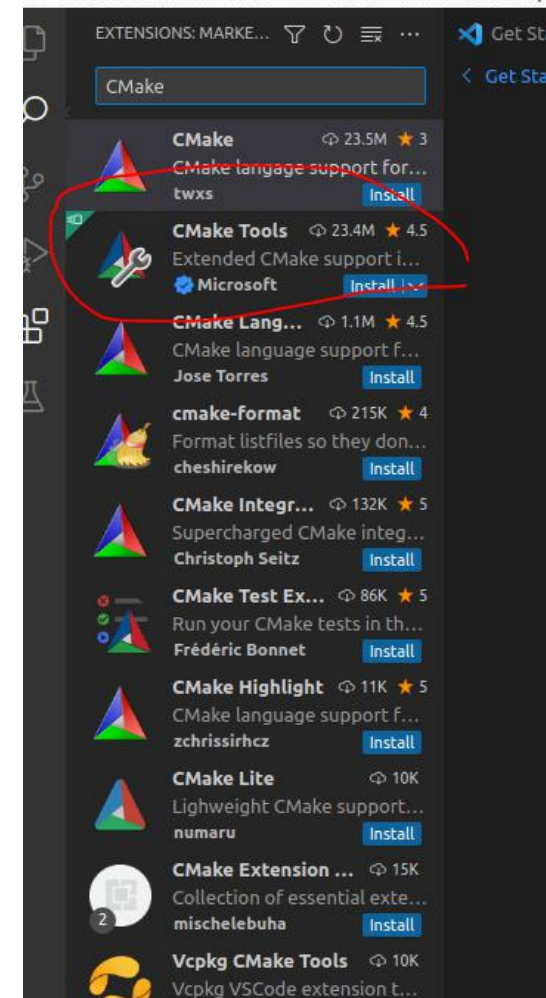
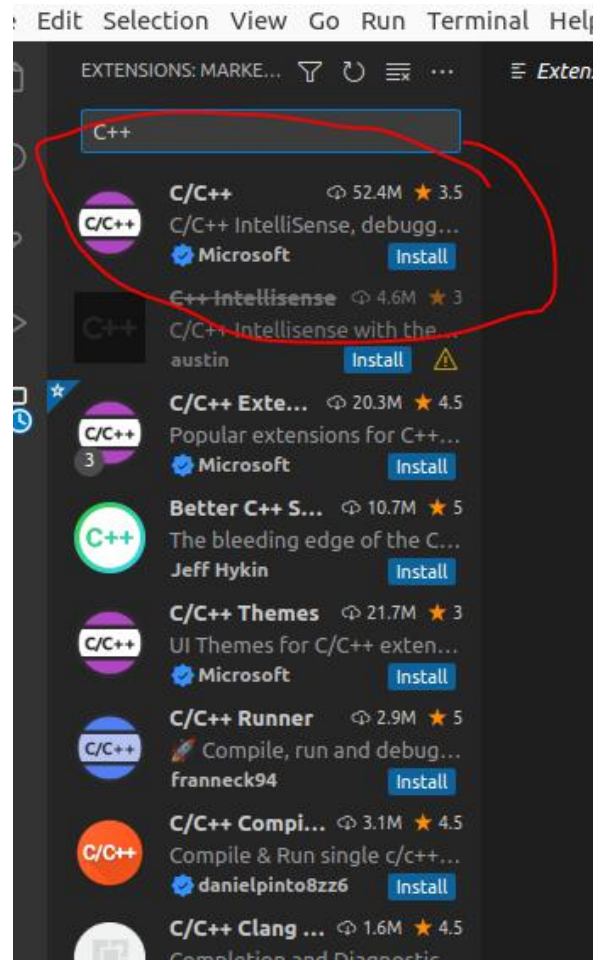
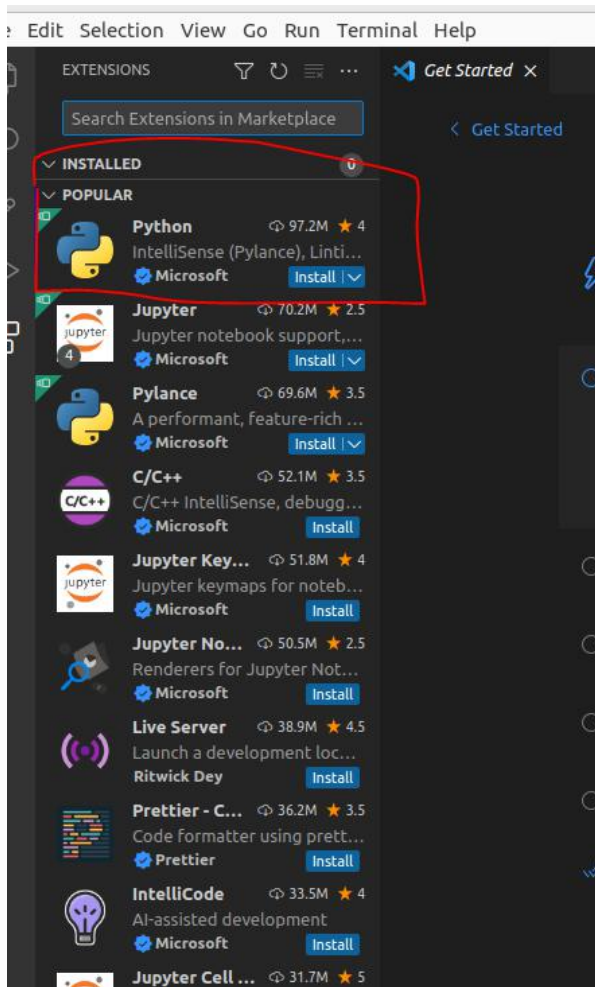
1. Run first ROS2 node

```
$ ros2 run turtlesim turtlesim_node
```

```
$ ros2 run turtlesim turtle_teleop_key
```



VScode Setting



Common command operations

`cd <dir>`

`mkdir [option] <dir>`

`cp [option] <source> <target>`

`gedit <file>`

`mv [option] <source> <target>`

`rm [option] <file/dir...>`

```
pi@pi-virtual-machine:~/Desktop$ ros2 --help
usage: ros2 [-h] [--use-python-default-buffering]
           Call 'ros2 <command> -h' for more detailed usage. ...

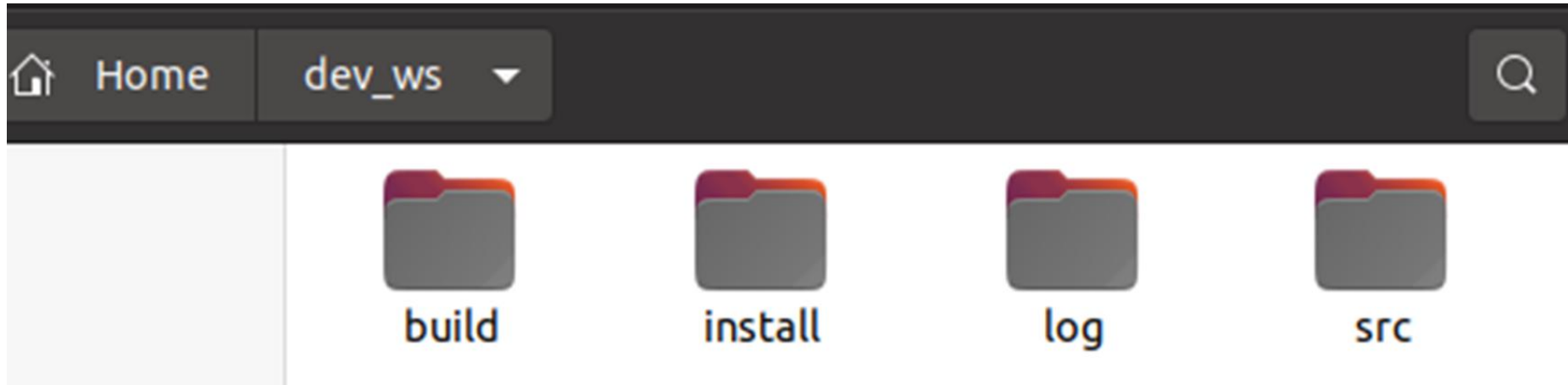
ros2 is an extensible command-line tool for ROS 2.

options:
  -h, --help            show this help message and exit
  --use-python-default-buffering
                        Do not force line buffering in stdout and instead use
                        the python default buffering, which might be affected
                        by PYTHONUNBUFFERED/-u and depends on whatever stdout
                        is interactive or not

Commands:
  action                Various action related sub-commands
  bag                   Various rosbag related sub-commands
  component             Various component related sub-commands
  daemon               Various daemon related sub-commands
  doctor                Check ROS setup and other potential issues
  interface             Show information about ROS interfaces
  launch                Run a launch file
  lifecycle             Various lifecycle related sub-commands
  multicast             Various multicast related sub-commands
  node                 Various node related sub-commands
  param                Various param related sub-commands
  pkg                   Various package related sub-commands
  run                  Run a package specific executable
  security              Various security related sub-commands
  service              Various service related sub-commands
  topic                Various topic related sub-commands
  wtf                   Use 'wtf' as alias to 'doctor'

Call 'ros2 <command> -h' for more detailed usage.
pi@pi-virtual-machine:~/Desktop$
```


Workspace



Create Workspace

```
$ mkdir -p ~/dev_ws/src  
$ cd ~/dev_ws/src  
$ git clone https://gitee.com/guyuehome/ros2_21_tutorials.git
```

1.Installing dependencies

```
$ sudo apt install -y python3-pip  
$ sudo pip3 install rosdepc  
$ sudo rosdepc init  
$ rosdepc update  
$ cd ..  
$ rosdepc install -i --from-path src --rostdistro humble -y
```

2.Building

```
$ sudo apt install python3-colcon-ros  
$ cd ~/dev_ws/  
$ colcon build
```

3.Source

```
$ source install/local_setup.sh
```

1. Create your first package

```
$ ros2 pkg create --build-type <build-type> <package_name>
```

<build-type>

C/C++: ament_cmake

Python: ament_python

```
$ cd ~/dev_ws/src
```

```
$ ros2 pkg create --build-type ament_cmake c_pkg --dependencies rclcpp std_msgs
```

```
$ ros2 pkg create --build-type ament_python pkg_p --dependencies rclpy std_msgs
```

```
pi@pi-virtual-machine: ~/ros2_ws

pi@pi-virtual-machine:~/ros2_ws$ tree
.
├── src
│   ├── c_pkg
│   │   ├── CMakeLists.txt
│   │   ├── include
│   │   │   └── c_pkg
│   │   ├── package.xml
│   │   └── src
│   └── pkg_p
│       ├── package.xml
│       ├── pkg_p
│       │   └── __init__.py
│       ├── resource
│       │   └── pkg_p
│       ├── setup.cfg
│       ├── setup.py
│       └── test
│           ├── test_copyright.py
│           ├── test_flake8.py
│           └── test_pep257.py
└── 9 directories, 10 files

pi@pi-virtual-machine:~/ros2_ws$
```

```

1 cmake_minimum_required(VERSION 3.8)
2 project(c_pkg)
3
4 if(CMAKE_COMPILER_IS_GNUCXX OR CMAKE_CXX_COMPILER_ID MATCHES "Clang")
5   add_compile_options(-Wall -Wextra -Wpedantic)
6 endif()
7
8 # find dependencies
9 find_package(ament_cmake REQUIRED)
10 find_package(rclcpp REQUIRED)
11 find_package(std_msgs REQUIRED)
12
13 if(BUILD_TESTING)
14   find_package(ament_lint_auto REQUIRED)
15   # the following line skips the linter which checks for copyrights
16   # comment the line when a copyright and license is added to all source files
17   set(ament_cmake_copyright_FOUND TRUE)
18   # the following line skips cpplint (only works in a git repo)
19   # comment the line when this package is in a git repo and when
20   # a copyright and license is added to all source files
21   set(ament_cmake_cpplint_FOUND TRUE)
22   ament_lint_auto_find_test_dependencies()
23 endif()
24
25 ament_package()

```

CMake Tab Width: 8 Ln 1, Col 1 INS

```

1 ?xml version="1.0"?
2 <?xml-model href="http://download.ros.org/schema/package_format3.xsd" schematypens="http://
  www.w3.org/2001/XMLSchema"?>
3 <package format="3">
4   <name>c_pkg</name>
5   <version>0.0.0</version>
6   <description>TODO: Package description</description>
7   <maintainer email="pi@todo.todo">pi</maintainer>
8   <license>TODO: License declaration</license>
9
10  <buildtool_depend>ament_cmake</buildtool_depend>
11
12  <depend>rclcpp</depend>
13  <depend>std_msgs</depend>
14
15  <test_depend>ament_lint_auto</test_depend>
16  <test_depend>ament_lint_common</test_depend>
17
18  <export>
19    <build_type>ament_cmake</build_type>
20  </export>
21 </package>

```

XML Tab Width: 8 Ln 1, Col 1 INS


```

1 from setuptools import find_packages, setup
2
3 package_name = 'pkg_p'
4
5 setup(
6     name=package_name,
7     version='0.0.0',
8     packages=find_packages(exclude=['test']),
9     data_files=[
10         ('share/ament_index/resource_index/packages',
11          ['resource/' + package_name]),
12         ('share/' + package_name, ['package.xml']),
13     ],
14     install_requires=['setuptools'],
15     zip_safe=True,
16     maintainer='pi',
17     maintainer_email='pi@todo.todo',
18     description='TODO: Package description',
19     license='TODO: License declaration',
20     tests_require=['pytest'],
21     entry_points={
22         'console_scripts': [
23         ],
24     },
25 )

```

Python 2 ▾ Tab Width: 8 ▾ Ln 1, Col 1 ▾ INS

```

1 <?xml version="1.0"?>
2 <?xml-model href="http://download.ros.org/schema/package_format3.xsd" schematypens="http://
  www.w3.org/2001/XMLSchema"?>
3 <package format="3">
4   <name>pkg_p</name>
5   <version>0.0.0</version>
6   <description>TODO: Package description</description>
7   <maintainer email="pi@todo.todo">pi</maintainer>
8   <license>TODO: License declaration</license>
9
10  <depend>rclpy</depend>
11  <depend>std_msgs</depend>
12
13  <test_depend>ament_copyright</test_depend>
14  <test_depend>ament_flake8</test_depend>
15  <test_depend>ament_pep257</test_depend>
16  <test_depend>python3-pytest</test_depend>
17
18  <export>
19    <build_type>ament_python</build_type>
20  </export>
21 </package>

```

XML ▾ Tab Width: 8 ▾ Ln 1, Col 1 ▾ INS

1.Create first node (C++)

```
$ cd ~/ros2_ws/src/c_pkg/src/  
$ touch talker.cpp
```

```
#include "rclcpp/rclcpp.hpp"
```

```
int main(int argc,char **argv)  
{  
    rclcpp::init(argc,argv);  
    auto node = std::make_shared<rclcpp::Node>("node1");  
    RCLCPP_INFO(node->get_logger(),"node1 is start!");  
    rclcpp::spin(node);  
    rclcpp::shutdown();  
    return 0;  
}
```

```
src > c_pkg > M CMakeLists.txt
1  cmake_minimum_required(VERSION 3.8)
2  project(c_pkg)
3
4  if(CMAKE_COMPILER_IS_GNUCXX OR CMAKE_CXX_COMPILER_ID MATCHES "Clang")
5    add_compile_options(-Wall -Wextra -Wpedantic)
6  endif()
7
8  # find dependencies
9  find_package(ament_cmake REQUIRED)
10 find_package(rclcpp REQUIRED)
11 find_package(std_msgs REQUIRED)
12
13
14 add_executable(talker src/talker.cpp)
15 ament_target_dependencies(talker rclcpp)
16
17
18 install(TARGETS
19   talker
20   DESTINATION lib/${PROJECT_NAME}
21 )
22
23 if(BUILD_TESTING)
24   find_package(ament_lint_auto REQUIRED)
25   # the following line skips the linter which checks for copyrights
26   # comment the line when a copyright and license is added to all source files
27   set(ament_cmake_copyright_FOUND TRUE)
28   # the following line skips cpplint (only works in a git repo)
29   # comment the line when this package is in a git repo and when
30   # a copyright and license is added to all source files
31   set(ament_cmake_cpplint_FOUND TRUE)
32   ament_lint_auto_find_test_dependencies()
33 endif()
34
35 ament_package()
36
```

```
add_executable(talker src/talker.cpp)
ament_target_dependencies(talker rclcpp)
```

```
install(TARGETS
  talker
  DESTINATION lib/${PROJECT_NAME}
)
```

```
$ cd ~/ros2_ws/  
$ colcon build  
$ source install/setup.bash
```

```
$ros2 run c_pkg talker
```

```
pi@pi-virtual-machine:~/ros2_ws$ ros2 run c_pkg talker  
[INFO] [1695017603.523663677] [node1]: node1 is start!  
█
```

1. Create first package(python)

```
$ cd ~/ros2_ws/src/pkg_p/pkg_p/  
$ touch listener.py
```

```
import rclpy  
from rclpy.node import Node  
import sys  
def main(args=None):  
    rclpy.init(args=sys.argv)  
    node = Node("node2")  
    node.get_logger().info("node2 is start!")  
    rclpy.spin(node=node)  
    rclpy.shutdown()
```



```
> pkg_p > setup.py > ...
1 from setuptools import find_packages, setup
2
3 package_name = 'pkg_p'
4
5 setup(
6     name=package_name,
7     version='0.0.0',
8     packages=find_packages(exclude=['test']),
9     data_files=[
10         ('share/ament_index/resource_index/packages',
11          ['resource/' + package_name]),
12         ('share/' + package_name, ['package.xml']),
13     ],
14     install_requires=['setuptools'],
15     zip_safe=True,
16     maintainer='pi',
17     maintainer_email='pi@todo.todo',
18     description='TODO: Package description',
19     license='TODO: License declaration',
20     tests_require=['pytest'],
21     entry_points={
22         'console_scripts': [
23             "listener = pkg_p.listener:main"
24         ],
25     },
26 )
27
```

"listener = pkg_p.listener:main"

```
pi@pi-virtual-machine:~/ros2_ws$ ros2 run pkg_p listener  
[INFO] [1695018532.918658796] [node2]: node2 is start!
```



南方科技大学
SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

Thanks for Listening