ROS Installation

ROS

- An open-source, meta-operating system for your robot.
- Provides the services you would expect from an operating system, including hardware abstraction, low-level device control, implementation of commonly-used functionality, message-passing between processes, and package management.
- Provides tools and libraries for obtaining, building, writing, and running code across multiple computers.

Installation

- It is convenient to use ROS in Linux environments.
- In a Windows environment, we can work inside a Linux virtual environment using VirtualBox.

VirtualBox Installation

- Windows: https://download.virtualbox.org/virtualbox/7.0.12/VirtualBox-7.0.12-159484-Win.exe
- MacOS: https://download.virtualbox.org/virtualbox/7.0.12/VirtualBox-7.0.12-159484-OSX.dmg
- Linux: Natively install the ROS (skip to ROS installation)

VirtualBox Extension Pack

- For using shared folder with the VM and the host machine.
- Download link: https://download.virtualbox.org/virtualbox/7.0.12/Oracle_VM_VirtualBox_Extension_Pack-7.0.12.vbox-extpack
- Double on it to install the extension pack

Ubuntu Virtual Machine in VirtualBox

- Download the virtual machine (Ubuntu_20.04.4_VB.7z) from this folder: https://drive.google.com/drive/folders/1q1ezlDOpsP3aUgI2H_ByrszPPaRlH9IM?usp=sharing

- Unzip the Ubuntu_20.04.4_VB.7z file
- Keep at least **20 GB** free space in the VM installation folder
- In VirtualBox, click “Add” button and select the .vbox file
- Log into the VM:
  - **Username: ubuntu**
  - **Password: ubuntu**

Shared Folder/Clipboard In VirtualBox (optional)

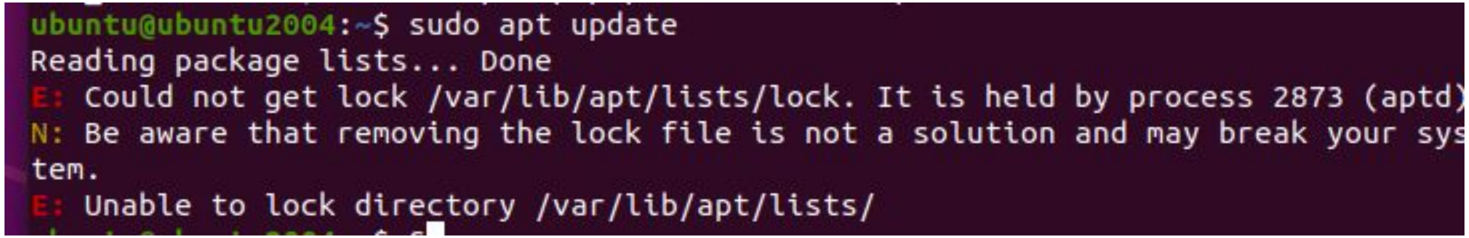
- For the convenience of code editing
  - In the VM options, click Setting > Shared Folders
  - Then, run this command in VM and **restart the VM**
    - `sudo adduser $USER vboxsf`
  - **Use Notepad++ to edit your code in Windows:**
    - **Open move.py (source file) in Notepad++**
    - **Then, Menu > Edit > EOL Conversion > Unix (LF)**

- For clipboard sharing:
  - Click Setting > Shared Clipboards > Bidirectional

ROS Installation

```
sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu
$(lsb_release -sc) main" > /etc/apt/sources.list.d/ros-latest.list'

sudo apt update

sudo apt install curl

curl -s
https://raw.githubusercontent.com/ros/rosdistro/master/ros.asc |
sudo apt-key add -

sudo apt update

sudo apt install ros-noetic-desktop-full

echo "source /opt/ros/noetic/setup.bash" >> ~/.bashrc
```

Error

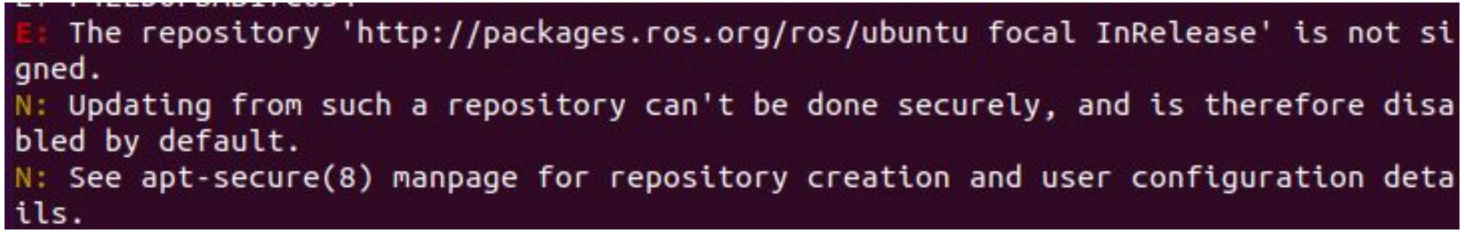


```
ubuntu@ubuntu2004:~$ sudo apt update
Reading package lists... Done
E: Could not get lock /var/lib/apt/lists/lock. It is held by process 2873 (aptd
N: Be aware that removing the lock file is not a solution and may break your sys
tem.
E: Unable to lock directory /var/lib/apt/lists/
```

Solution

```
sudo rm /var/lib/dpkg/lock

sudo dpkg --configure -a
```

Error



```
E: The repository 'http://packages.ros.org/ros/ubuntu focal InRelease' is not si
gned.
N: Updating from such a repository can't be done securely, and is therefore disa
bled by default.
N: See apt-secure(8) manpage for repository creation and user configuration deta
ils.
```

Solution:

```
sudo apt-key adv --keyserver keyserver.ubuntu.com --recv-keys F42ED6FBAB17C654

sudo rm /var/lib/dpkg/lock

sudo dpkg --configure -a
```

Create backup

- After ROS is installed, create a clone of your VM by clicking Clone option
- If the VM is corrupted somehow, use the cloned version to avoid reinstalling everything

ROS Basics

- Package (The package manager is Catkin)
  - Create
  - Build
  - Run
- Node
  - Master
  - Turtle
  - Controller
-

Turtle Bot

```
# Start the master node
roscore

# Start the turtle code in a new terminal (turtle window will pop-up)
rosrun turtlesim turtlesim_node


# Write the controller package in a new terminal, you can find the folder in /media
cd /media && ls
cd /media/sf_shared

# Create a workspace
mkdir -p catkin_ws/src
cd catkin_ws/src

# Create a controller package called turtlesim_cleaner with dependencies: geometry_msgs, rospy
catkin_create_pkg turtlesim_cleaner geometry_msgs rospy

# Goto the base directory of the workspace and build the package
cd catkin_ws
catkin_make

# Setup env register package
source ./devel/setup.bash

# Create the controller file
touch catkin_ws/src/turtlesim_cleaner/src/move.py

# Run the controller, the code for move.py is provided in the next slides
cd /media/sf_shared/catkin_ws
source ./devel/setup.bash
rosrun turtlesim_cleaner move.py
```

Avoiding EOL Error

- Edit the move.py file in Notepad++
- Goto Edit > EOL Conversion > Unix (LF)
- Then, paste the code given in the next slides

The controller file: move.py

```python
#!/usr/bin/python3
import rospy
from geometry_msgs.msg import Twist

def move():
    # Starts a new node
    rospy.init_node('robot_cleaner', anonymous=True)
    velocity_publisher = rospy.Publisher('/turtle1/cmd_vel', Twist, queue_size=10)
    vel_msg = Twist()

    #Receiveing the user's input
    print("Let's move your robot")
    speed = input("Input your speed:")
    distance = input("Type your distance:")
    isForward = input("Foward?: ")#True or False
    speed = float(speed)
    distance = float(distance)
    isForward = int(isForward)
```

The controller file: move.py (continued)

```
#Checking if the movement is forward or backwards
 if(isForward):
     vel_msg.linear.x = abs(speed)
 else:
     vel_msg.linear.x = -abs(speed)
 #Since we are moving just in x-axis
 vel_msg.linear.y = 0
 vel_msg.linear.z = 0
 vel_msg.angular.x = 0
 vel_msg.angular.y = 0
 vel_msg.angular.z = 0
```

The controller file: move.py (continued)

```
    while not rospy.is_shutdown():

        #Setting the current time for distance calculus
        t0 = rospy.Time.now().to_sec()
        current_distance =0

        #Loop to move the turtle in an specified distance
        while(current_distance < distance):
            #Publish the velocity
            velocity_publisher.publish(vel_msg)
            #Takes actual time to velocity calculus
            t1=rospy.Time.now().to_sec()
            #Calculates distancePoseStamped
            current_distance= speed*(t1-t0)
        #After the loop, stops the robot
        vel_msg.linear.x =0
        #Force the robot to stop
        velocity_publisher.publish(vel_msg)

if __name__ == '__main__':
    try:
        #Testing our function
        move()
    except rospy.ROSInterruptException:pass
```