# Turtlebot 3 Linux Software Commands

| Ubuntu | ISO download link: https://www.ubuntu-fr.org/download/ <br><br> • **Software update:** *sudo apt update* <br> • **Software upgrade**: *sudo apt upgrade* <br> • **Install specific package:** *sudo apt-get install package_name* <br> • **Uninstall specific program:** *sudo apt-get purge program_name\** |
|---|---|
| Requirements | **Install Pyton 3 Colcon and Curl :** <br> *sudo apt install python3-colcon-common-extensions* <br> *sudo apt install curl* |
| ROS 2 Humble | **Link:** https://docs.ros.org/en/humble/Installation/Ubuntu-Install-Debians.html <br><br> **Installation:** <br><br> • **Add the ROS 2 apt repository to your system:** <br><br> *sudo apt install software-properties-common* <br> *sudo add-apt-repository universe* <br><br><br> • **Add the ROS 2 GPG key with apt:** <br><br> *sudo apt update && sudo apt install curl -y* <br> *sudo curl -sSL* <br> *https://raw.githubusercontent.com/ros/rosdistro/master/ros.key -o* <br> */usr/share/keyrings/ros-archive-keyring.gpg* <br><br> • **Add the repository to your sources list:** <br><br> *echo "deb [arch=$(dpkg --print-architecture) signed-* <br> *by=/usr/share/keyrings/ros-archive-keyring.gpg]* <br> *http://packages.ros.org/ros2/ubuntu $(. /etc/os-release && echo* <br> *$UBUNTU_CODENAME) main" | sudo tee* <br> */etc/apt/sources.list.d/ros2.list > /dev/null* <br><br> • **Install ROS 2 packages (ROS, Rviz, Demos, Tutorials):** <br><br> *sudo apt update* <br> *sudo apt upgrade* <br> *sudo apt install ros-humble-desktop* <br><br> • **ROS-Base install: Communication libraries, message packages, command line tools (No GUI tools):** <br><br> *sudo apt install ros-humble-ros-base* |

| | |
|---|---|
| | • **Development tools: Compilers and other tools to build ROS packages**<br><br>    *sudo apt install ros-dev-tools*<br><br>• **Environment setup**<br><br>    *source /opt/ros/humble/setup.bash*<br><br>**Check installation:**<br>• **In one terminal, source the setup file and then run a C++ talker:**<br><br>    *source /opt/ros/humble/setup.bash*<br>    *ros2 run demo_nodes_cpp talker*<br><br>• **In another terminal source the setup file and then run a Python listener:**<br><br>    *source /opt/ros/humble/setup.bash*<br>    *ros2 run demo_nodes_py listener*<br><br>**Uninstall ROS 2 Humble:**<br><br>    *sudo apt remove ~nros-humble-\* && sudo apt autoremove*<br>    *sudo rm /etc/apt/sources.list.d/ros2.list*<br>    *sudo apt update*<br>    *sudo apt autoremove*<br>    *sudo apt upgrade* |
| **Gazebo =** | **Link:** https://gazebosim.org/docs/garden/install_ubuntu<br><br>https://installati.one/install-gazebo-ubuntu-22-04/<br><br>**Installation:**<br><br>• **Install some necessary tools:**<br><br>    *sudo apt-get update*<br>    *sudo apt-get install lsb-release wget gnupg*<br><br>• **Install Gazebo for Ubuntu 22.04**<br><br>    *sudo apt install gazebo*<br>    *sudo apt install libgazebo-dev*<br><br>**Uninstall Gazebo:**<br><br>    *sudo apt-get remove gazebo*<br>    *sudo apt-get -y autoremove gazebo*<br>    *sudo apt-get -y purge gazebo*<br>    *sudo apt-get -y autoremove* |

| Turtlebot3 Humble | **Link:**<br>https://emanual.robotis.com/docs/en/platform/turtlebot3/simulation/<br><br>**Installation:**<br><br>• **Install Simulation package:**<br><br>    *sudo apt install ros-humble-gazebo-\**<br>    *sudo apt install ros-humble-cartographer*<br>    *sudo apt install ros-humble-cartographer-ros*<br>    *sudo apt install ros-humble-navigation2*<br>    *sudo apt install ros-humble-nav2-bringup*<br><br>    *sudo apt remove ros-humble-turtlebot3-msgs*<br>    *sudo apt remove ros-humble-turtlebot3*<br>    *mkdir -p ~/turtlebot3_ws/src*<br>    *cd ~/turtlebot3_ws/src/*<br>    *git clone -b humble-devel https://github.com/ROBOTIS-GIT/DynamixelSDK.git*<br>    *git clone -b humble-devel https://github.com/ROBOTIS-GIT/turtlebot3_msgs.git*<br>    *git clone -b humble-devel https://github.com/ROBOTIS-GIT/turtlebot3.git*<br>    *cd ~/turtlebot3_ws*<br>    *colcon build*<br>    *echo 'source ~/turtlebot3_ws/install/setup.bash' >> ~/.bashrc*<br>    *source ~/.bashrc*<br>    *echo 'export TURTLEBOT3_MODEL=burger' >> ~/.bashrc*<br>    *source ~/.bashrc*<br><br>    *cd ~/turtlebot3_ws/src/*<br>    *git clone -b humble-devel https://github.com/ROBOTIS-GIT/turtlebot3_simulations.git*<br>    *cd ~/turtlebot3_ws && colcon build*<br><br>• **Launch Simulation World**<br>    o **Empty World**<br><br>    *ros2 launch turtlebot3_gazebo empty_world.launch.py*<br><br>    o **Turtlebot3 World**<br><br>    *ros2 launch turtlebot3_gazebo turtlebot3_world.launch.py*<br><br>• **Operate Turtlebot3**<br><br>    *ros2 run turtlebot3_teleop teleop_keyboard* |

- **Cartographer SLAM**

  *ros2 launch turtlebot3_cartographer cartographer.launch.py*

- **Save Map**
  *ros2 run nav2_map_server map_saver_cli -f ~/map*

- **TurtleSim :** *ros2 run turtlesim turtlesim_node*

- **Teleop keyboard for TurtleSim :** *ros2 run turtlesim turtle_teleop_key*

## SSH Connection:

- **Configure a Wifi hotspot from a computer:**

  *nmcli con add type wifi ifname <ip of hotspot wireless interface) con-name <name> autoconnect yes ssid <name SSID> ap ipv4.method shared*

**In our case:** *nmcli con add type wifi ifname wlp2s0 con-name Co4Sys autoconnect yes ssid Co4Sys 802-11-wireless.mode ap ipv4.method shared*

→ *Normaly, the hotspot is configured, you just have to check if the Wifi network is On and in hotspot mode*

- **Connection to the hotspot created before:**

  *nmcli con up <name of the wireless network>*

**In our case** *: nmcli con up Co4Sys*

- **Configure the SSH settings of the RaspberryPi to connect it to the Wifi hotspot :**

  Login: ubuntu
  Password: turtlebot

- **Get the Turtlebot3 address:** *ip n (must be reachable)*

- **Connect to Turtlebot3:**

  *ssh ubuntu@ip_address*

**In our case:** *ssh ubuntu@10.42.0.216*

- **Launch the Turtlebot3 and then the keyboard on another terminal:**

  *ros2 launch turtlebot3_bringup robot.launch.py*

| | |
|---|---|
| | *ros2 run turtlebot3_teleop teleop_keyboard* |
| **Others commands** | **IP Information:** *ifconfig*<br>**List of wireless interfaces:** *ip a*<br>**Wireless scan:** *sudo arp-scan-localnet -I<wireless interface with ip a>*<br>**Display your own IP Address:** *hostname -I*<br>**Display all the hotspot connection:** *nmcli connection show* |
| **ROS2 Node** | **Create a ROS2 Workspace and code your own node in Python:**<br><br>• **Install colcon extensions:** *sudo apt install python3-colcon-common-extensions*<br><br>• **Autocompletion:** *gedit ~/.bashrc* **and add the following line at the end:** *source /usr/share/colcon_argcomplete/hook/ colcon-argcomplete.bash*<br><br>• **Make a workspace directory:** *mkdir ros2_ws*<br><br>• **Make a source directory inside the workspace:** *cd ros2_ws && mkdir src*<br><br>• **Build the workspace:** *colcon build*<br><br>• **Source the workspace:** *cd && source ~/ros2_ws/install/setup.bash && gedit ~/.bashrc* **and add in the file the following line at the end :** *source ~/ros2_ws/install/setup.bash*<br><br>• **Create a ROS2 package into the workspace:** *cd ros2_ws/src/ && ros2 pkg create name_package –build-type ament_python –dependencies rclpy*<br><br>• **Make sure Visual Studio Code is installed with Python and ROS2 extensions**<br><br>• **Check installation:** *sudo snap install code --classic*<br><br>• **Launch the workspace from the terminal:** *code .*<br><br>• **Build the package with colcon:** *cd ros2_ws/ && colcon build*<br><br>• **Fix "SetuptoolsDeprecationWarning: setup.py install is deprecated" in colcon build:** *sudo apt install python3-pip && pip3 install setuptools==58.2.0* |

- **Create a python file to write a node and make it executable:** *cd ros2_ws/src/name_package/name_package/touch name_file_node.py && chmod +x name_file_node.py*

- **Execute the python node file created:** *cd ros2_ws/src/name_package/name_package/ && ./name_file_node.py*

**Example of a Python node code:**

**→ my_first_node.py:**

```python
#!/usr/bin/env python3
import rclpy
from rclpy.node import Node

class MyNode(Node):

    def __init__(self):
        super().__init__("first_node") #Initialization of
the Node
        self.get_logger().info("Hello from ROS2") #Display
logger into the terminal

def main(args=None):
    rclpy.init(args=args) #Initialization ROS2
communication
    node = MyNode() #Create a node from the MyNode class
    rclpy.spin(node) #Keep alive the node


    rclpy.shutdown() #Close the ROS2 communication

if __name__ == '__main__':
    main()
```

**→ setup.py**

```python
entry_points={
        'console_scripts': [
            "test_node =
my_robot_controller.my_first_node:main"
        ],
    },
```

- **Auto-compilation without colcon build every time:**
  *cd ros2_ws/*
  *colcon build –symlink-install*
  *source ~/.bashrc*
  *ros2 run my_robot_controller test_node*

- **Useful commands:**
  *ros2 node list*
  *ros2 node info /name_of_the_node*

| | |
|---|---|
| | *ros2 topic list*<br>*ros2 topic info /name_of_the_topic*<br>*rqt*<br>*rqt_graph*<br>*ros2 interface show <type of ros2 topic info >* |