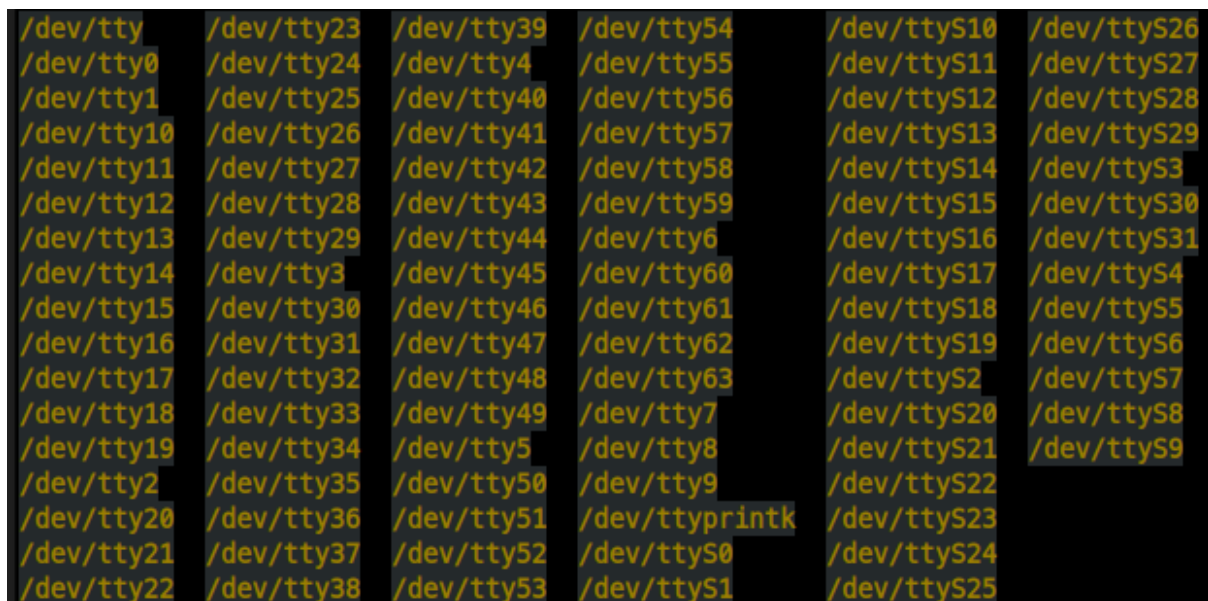


노드를 이용해서 로봇 팔 제어하기

로봇팔과 PC간의 연결 확인하기

1. pc터미널에서 현재 직렬포트에 연결된 목록을 확인한다

```
ls /dev/tty*
```



```
/dev/tty      /dev/tty23   /dev/tty39   /dev/tty54   /dev/ttyS10  /dev/ttyS26
/dev/tty0     /dev/tty24   /dev/tty4     /dev/tty55   /dev/ttyS11  /dev/ttyS27
/dev/tty1     /dev/tty25   /dev/tty40   /dev/tty56   /dev/ttyS12  /dev/ttyS28
/dev/tty10    /dev/tty26   /dev/tty41   /dev/tty57   /dev/ttyS13  /dev/ttyS29
/dev/tty11    /dev/tty27   /dev/tty42   /dev/tty58   /dev/ttyS14  /dev/ttyS3
/dev/tty12    /dev/tty28   /dev/tty43   /dev/tty59   /dev/ttyS15  /dev/ttyS30
/dev/tty13    /dev/tty29   /dev/tty44   /dev/tty6     /dev/ttyS16  /dev/ttyS31
/dev/tty14    /dev/tty3    /dev/tty45   /dev/tty60   /dev/ttyS17  /dev/ttyS4
/dev/tty15    /dev/tty30   /dev/tty46   /dev/tty61   /dev/ttyS18  /dev/ttyS5
/dev/tty16    /dev/tty31   /dev/tty47   /dev/tty62   /dev/ttyS19  /dev/ttyS6
/dev/tty17    /dev/tty32   /dev/tty48   /dev/tty63   /dev/ttyS2   /dev/ttyS7
/dev/tty18    /dev/tty33   /dev/tty49   /dev/tty7     /dev/ttyS20  /dev/ttyS8
/dev/tty19    /dev/tty34   /dev/tty5    /dev/tty8     /dev/ttyS21  /dev/ttyS9
/dev/tty2     /dev/tty35   /dev/tty50   /dev/tty9     /dev/ttyS22
/dev/tty20    /dev/tty36   /dev/tty51   /dev/ttyprintk /dev/ttyS23
/dev/tty21    /dev/tty37   /dev/tty52   /dev/ttyS0    /dev/ttyS24
/dev/tty22    /dev/tty38   /dev/tty53   /dev/ttyS1    /dev/ttyS25
```

2. 로봇팔을 PC의 USB 포트에 연결한 후 다시 한번 목록을 확인

```
> ls /dev/tty*
/dev/tty      /dev/tty23  /dev/tty39  /dev/tty54  /dev/ttyS10 /dev/ttyS26
/dev/tty0     /dev/tty24  /dev/tty4    /dev/tty55  /dev/ttyS11 /dev/ttyS27
/dev/tty1     /dev/tty25  /dev/tty40  /dev/tty56  /dev/ttyS12 /dev/ttyS28
/dev/tty10    /dev/tty26  /dev/tty41  /dev/tty57  /dev/ttyS13 /dev/ttyS29
/dev/tty11    /dev/tty27  /dev/tty42  /dev/tty58  /dev/ttyS14 /dev/ttyS3
/dev/tty12    /dev/tty28  /dev/tty43  /dev/tty59  /dev/ttyS15 /dev/ttyS30
/dev/tty13    /dev/tty29  /dev/tty44  /dev/tty6    /dev/ttyS16 /dev/ttyS31
/dev/tty14    /dev/tty3   /dev/tty45  /dev/tty60  /dev/ttyS17 /dev/ttyS4
/dev/tty15    /dev/tty30  /dev/tty46  /dev/tty61  /dev/ttyS18 /dev/ttyS5
/dev/tty16    /dev/tty31  /dev/tty47  /dev/tty62  /dev/ttyS19 /dev/ttyS6
/dev/tty17    /dev/tty32  /dev/tty48  /dev/tty63  /dev/ttyS2  /dev/ttyS7
/dev/tty18    /dev/tty33  /dev/tty49  /dev/tty7   /dev/ttyS20 /dev/ttyS8
/dev/tty19    /dev/tty34  /dev/tty5   /dev/tty8   /dev/ttyS21 /dev/ttyS9
/dev/tty2     /dev/tty35  /dev/tty50  /dev/tty9   /dev/ttyS22 /dev/ttyUSB0
/dev/tty20    /dev/tty36  /dev/tty51  /dev/ttyprintk /dev/ttyS23
/dev/tty21    /dev/tty37  /dev/tty52  /dev/ttyS0  /dev/ttyS24
/dev/tty22    /dev/tty38  /dev/tty53  /dev/ttyS1  /dev/ttyS25
```

- /dev/ttyUSB0라는 항목이 생긴 것을 알 수 있다
 - 연결된 장치의 이름은 PC환경마다 다를 수 있다
 - 윈도우에서 사용할 경우, 제조사에서 명시한 장치명은 CP2102N USB 이다

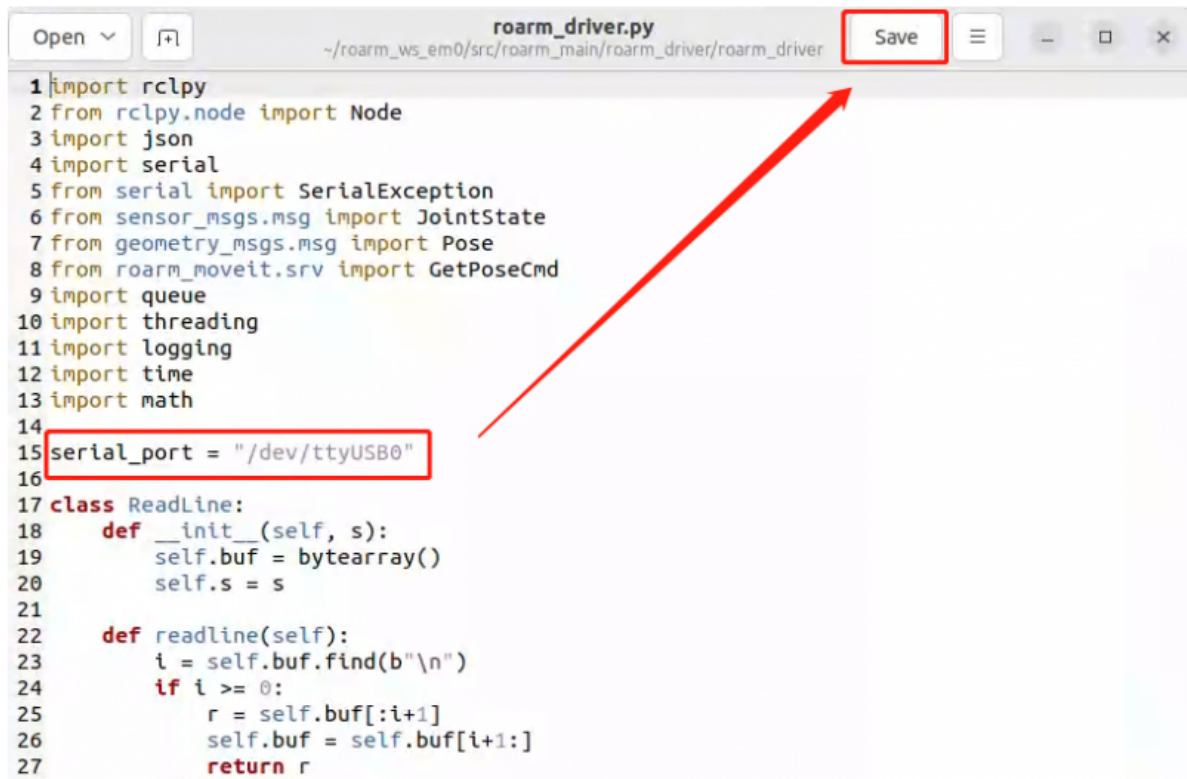
직렬 장치 수정하기

: 만약 로봇과 PC를 직렬연결 했을 때, 장치 이름이 ttyUSB0가 아닐경우 장치의 이름을 수정해주어야 한다

로봇에 설치된 패키지들이 장치의 포트명을 ttyUSB0로 지정하여 소스 코드를 작성했기 때문이다

다음의 경로에서 파일을 열고 /dev/ttyUSB0를 자신의 PC에서 확인한 장치명으로 바꾸어 주면 된다

```
~/roarm_ws_em0/src/roarm_main/roarm_driver/roarm_driver/roarm_driver.py
```



```
1 import rclpy
2 from rclpy.node import Node
3 import json
4 import serial
5 from serial import SerialException
6 from sensor_msgs.msg import JointState
7 from geometry_msgs.msg import Pose
8 from roarm_moveit.srv import GetPoseCmd
9 import queue
10 import threading
11 import logging
12 import time
13 import math
14
15 serial_port = "/dev/ttyUSB0"
16
17 class ReadLine:
18     def __init__(self, s):
19         self.buf = bytearray()
20         self.s = s
21
22     def readline(self):
23         i = self.buf.find(b"\n")
24         if i >= 0:
25             r = self.buf[:i+1]
26             self.buf = self.buf[i+1:]
27             return r
```

로봇팔 드라이브 노드 컴파일 및 실행

1. Ubuntu에서 Python 스크립트를 사용하여 직렬 장치와 통신하기 위해서 장치에 읽기 및 쓰기 권한을 부여해야 한다.

```
sudo chmod 666 /dev/ttyUSB0
```

ls -l /dev/ttyUSB0 제대로 권한이 부여되었는지 확인

첫번째 rw : 소유자가 읽기 쓰기 권한이 있음

두번째 rw : 특정 그룹이 읽기 쓰기 권한이 있음

세번째 rw : 기타 사용자가 읽고 쓰기 권한이 있음

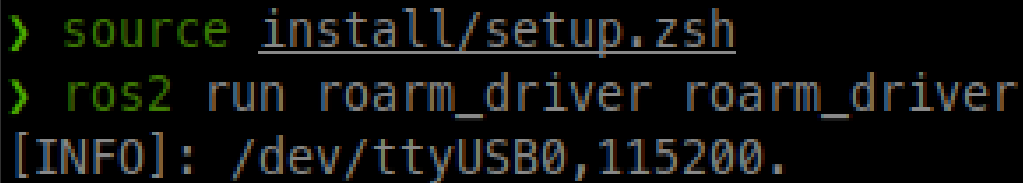
즉 모든 사용자들이 읽고 쓰기 권한이 있다는 뜻이다

2. 로봇팔의 기능 패키지를 컴파일 한다.

```
cd ~/roarm_ws_em0/
colcon build
source install/setup.bash
```

3. 로봇팔의 제어 노드를 실행한다.

```
cd ~/roarm_ws_em0/  
ros2 run roarm_driver roarm_driver
```



```
> source install/setup.zsh  
> ros2 run roarm_driver roarm_driver  
[INFO]: /dev/ttyUSB0,115200.  
█
```

- 정상작동 된 모습
- 현재는 조인트 인터페이스에 대한 노드를 실행시키지 않고 전체 드라이버에 대한 노드만 실행시킨 상태라 별다른 피드백은 없다
- 이 노드를 실행시킨 채로 조인트 노드를 실행시켜야 한다

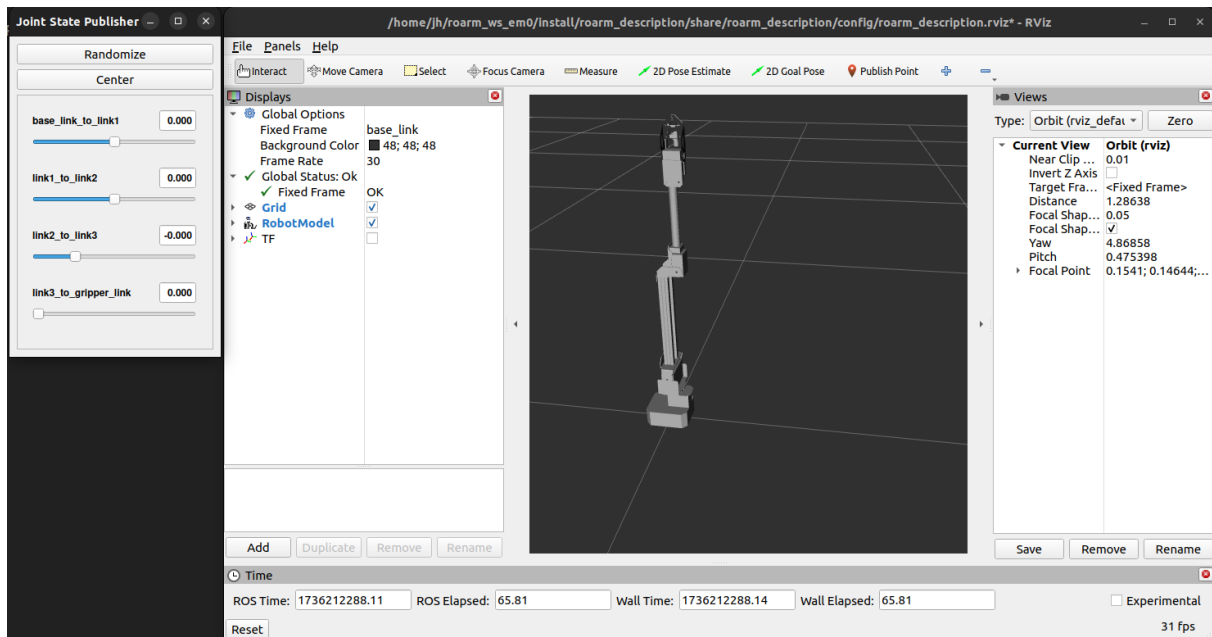
로봇팔 모델 조인트 보기

: Rviz2를 이용하여 로봇 팔의 모델과 조인트 각도 제어 패널을 시연한다

1. 로봇팔을 Rviz2에서 시각화하면서 제어할 수 있는 launch를 실행한다

```
cd ~/roarm_ws_em0/  
ros2 launch roarm_description display.launch.py
```

- 처음에는 다음과 같이 하늘을 향한 상태로 세팅되기 때문에 로봇의 가동범위를 고려하여 주변을 비워두는 것이 좋다



- Rviz2에서 로봇의 URDF를 통해 모델을 나타내고 각 조인트의 각도를 드라이버 노드에 전송한다
- 드라이버 노드에서는 전송받은 조인트의 각도를 JSON 형식의 제어 명령으로 변환하여 로봇팔에 전송하고 로봇팔이 동작한다
- Joint State Publisher 라는 제어 패널을 이용해 실시간으로 로봇을 제어할 수 있다
 - Center : 로봇의 자세를 초기화한다
 - Randomize : 각 조인트의 값을 랜덤으로 하여 동작시킬 수 있다

```
ros2 run roarm_driver roarm_driver
dev_ws llvm.sh nav2_ws Ro-Arm24 ROS2_FAST_CAMPUS Templates
> cd roarm_ws_em0
> ls
build build_roarm_web_app.sh log RoArm-M2-S_python.zip
build_common.sh images README.md src
build_first.sh install requirements.txt
> colcon build
Starting >>> ros2web_interfaces
Starting >>> roarm_description
Starting >>> roarm_moveit
Starting >>> moveit_servo
Starting >>> launch_api
Starting >>> roarm_driver
Starting >>> roarm_moveit_ikfast_plugins
Finished <<< roarm_description [0.67s]
Starting >>> roarm_moveit
Finished <<< moveit_servo [0.68s]
Finished <<< ros2web_interfaces [0.73s]
Starting >>> ros2web
Finished <<< roarm_moveit_ikfast_plugins [0.98s]
Finished <<< roarm_moveit [0.46s]
Starting >>> roarm_moveit_cmd
Finished <<< roarm_moveit_cmd [0.14s]
Finished <<< roarm_driver [1.29s]
Finished <<< launch_api [1.74s]
Finished <<< ros2web [1.36s]
Starting >>> ros2web_app
Starting >>> ros2web_example_py
Finished <<< ros2web_example_py [1.54s]
Finished <<< ros2web_app [1.56s]
Starting >>> roarm_web_app
Starting >>> ros2web_widgets
Finished <<< ros2web_widgets [1.52s]
Finished <<< roarm_web_app [1.53s]

Summary: 13 packages finished [5.71s]
> source install/setup.zsh
> ros2 run roarm_driver roarm_driver
[INFO]: /dev/ttyUSB0,115200.
```

```
ros2 launch roarm_description display.launch.py
[INFO] [launch]: All log files can be found below /home/jh/.ros/log/2025-0
1-07-10-10-21-696187-jh-16836
[INFO] [launch]: Default logging verbosity is set to INFO
[INFO] [robot_state_publisher-1]: process started with pid [16837]
[INFO] [joint_state_publisher_gui-2]: process started with pid [16839]
[INFO] [rviz2-3]: process started with pid [16841]
[robot_state_publisher-1] [WARN]: No robot_description parameter, but comm
and-line argument available. Assuming argument is name of URDF file. Thi
s backwards compatibility fallback will be removed in the future.
[robot_state_publisher-1] [WARN]: The root link base_link has an inertia s
pecified in the URDF, but KDL does not support a root link with an inertia
. As a workaround, you can add an extra dummy link to your URDF.
[robot_state_publisher-1] [INFO]: got segment base_link
[robot_state_publisher-1] [INFO]: got segment gripper_link
[robot_state_publisher-1] [INFO]: got segment hand_tcp
[robot_state_publisher-1] [INFO]: got segment link1
[robot_state_publisher-1] [INFO]: got segment link2
[robot_state_publisher-1] [INFO]: got segment link3
[rviz2-3] [INFO]: Stereo is NOT SUPPORTED
[rviz2-3] [INFO]: OpenGL version: 4.6 (GLSL 4.6)
[rviz2-3] [INFO]: Stereo is NOT SUPPORTED
[joint_state_publisher_gui-2] [INFO]: Centering
[joint_state_publisher_gui-2] [INFO]: Centering
[joint_state_publisher_gui-2] [INFO]: Centering
[joint_state_publisher_gui-2] [INFO]: Randomizing
[joint_state_publisher_gui-2] [INFO]: Centering
[joint_state_publisher_gui-2] [INFO]: Randomizing
[joint_state_publisher_gui-2] [INFO]: Centering
```