1️⃣ Week 1

1. ROS2 Humble 설치

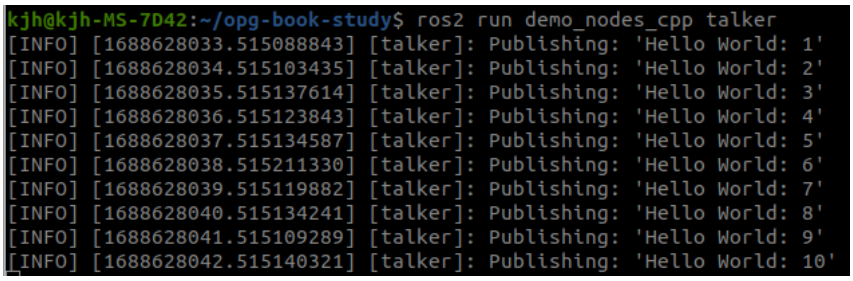
- ROS2 Humble Documentation

> ROS 2 Documentation — ROS 2 Documentation: Humble documentation
>
> :::2 https://docs.ros.org/en/humble/index.html

```
sudo apt install software-properties-common
sudo add-apt-repository universe

sudo apt update && sudo apt install curl -y
sudo curl -sSL https://raw.githubusercontent.com/ros/rosdistro/master/ros.key -o /usr/share/keyrings/ros-archive-keyring.gpg

echo "deb [arch=$(dpkg --print-architecture) signed-by=/usr/share/keyrings/ros-archive-keyring.gpg] http://packages.ros.org/ros2/ubunt

sudo apt update
sudo apt upgrade

sudo apt install ros-humble-desktop
# Replace ".bash" with your shell if you're not using bash
# Possible values are: setup.bash, setup.sh, setup.zsh
source /opt/ros/humble/setup.bash
```

간단한 예제를 실행해서 설치가 완료됐는지 확인할 수 있다.

```
source /opt/ros/humble/setup.bash
ros2 run demo_nodes_cpp talker

# New terminal
source /opt/ros/humble/setup.bash
ros2 run demo_nodes_py listener
```



```
kjh@kjh-MS-7D42:~/opg-book-study$ ros2 run demo_nodes_cpp talker
[INFO] [1688628033.515088843] [talker]: Publishing: 'Hello World: 1'
[INFO] [1688628034.515103435] [talker]: Publishing: 'Hello World: 2'
[INFO] [1688628035.515137614] [talker]: Publishing: 'Hello World: 3'
[INFO] [1688628036.515123843] [talker]: Publishing: 'Hello World: 4'
[INFO] [1688628037.515134587] [talker]: Publishing: 'Hello World: 5'
[INFO] [1688628038.515211330] [talker]: Publishing: 'Hello World: 6'
[INFO] [1688628039.515119882] [talker]: Publishing: 'Hello World: 7'
[INFO] [1688628040.515134241] [talker]: Publishing: 'Hello World: 8'
[INFO] [1688628041.515109289] [talker]: Publishing: 'Hello World: 9'
[INFO] [1688628042.515140321] [talker]: Publishing: 'Hello World: 10'
```

2. VSCode 개발환경 설정

- c_cpp_properties.json

```
{
  "configurations": [
    {
      "browse": {
        "databaseFilename": "${default}",
        "limitSymbolsToIncludedHeaders": false
      },
      "includePath": [
        "/opt/ros/humble/include/**",
        "/home/kjh/ros2_ws/src/my_box_bot_description/include/**",
        "/home/kjh/ros2_ws/src/my_box_bot_gazebo/include/**",
        "/usr/include/**"
      ],
      "name": "ROS",
      "intelliSenseMode": "gcc-x64",
      "compilerPath": "/usr/bin/gcc",
      "cStandard": "gnu11",
      "cppStandard": "c++14"
    }
  ],
  "version": 4
}
```

- tasks.json

```
{
    "version":"2.0.0",
    "tasks": [
        {
            "label":"colcon: build",
            "type": "shell",
            "command": "colcon build --cmake-args '-DCMAKE_BUILD_TYPE=Debug'",
            "problemMatcher": [],
            "group": {
                "kind": "build",
                "isDefault": true
            }
        },
        {
            "label": "colcon: test",
            "type": "shell",
            "command": "colcon test && colcon test-result"
        },
        {
            "label": "colcon: clean",
            "type": "shell",
            "command": "rm -rf build install log"
        }

    ]
}
```

- launch.json

```
{
    "version": "0.2.0",
    "configurations": [
        {
            "name": "Debug-rclpy(debugpy)",
            "type": "python",
            "request": "launch",
            "program": "${file}",
            "console": "integratedTerminal"
        },
        {
            "name": "Debug-rclcpp(gbd)",
            "type": "cppdbg",
            "request": "launch",
            "program": "${workspaceFolder}/install/${input:package}/lib/${input:package}/${input:node}",
            "args": [],
            "preLaunchTask": "colcon: build",
            "stopAtEntry": true,
            "cwd": "${workspaceFolder}",
            "externalConsole": false,
            "MIMode": "gdb",
            "setupCommands": [
                {
                    "description": "Enable pretty-printing for gdb",
                    "text": "-enable-pretty-printing",
                    "ignoreFailures": true
                }
            ]
        }
    ],
    "inputs": [
        {
            "id": "package",
            "type": "promptString",
            "description": "package name",
            "default": "topic_service_action_rclcpp_example"
        },
        {
            "id": "node",
            "type": "promptString",
            "description": "node name",
            "default": "argument"
        }
    ]
}
```

- launch.json 은 VSCode에서 디버거 구성을 담당한다. 각 구성은 디버깅할 프로그램, 명령줄 인주, 작업 directory 등을 지정한다.
- tasks.json은 VSCode에서 코드 빌드, 테스트 및 정리와 같은 작업을 구성하는 데 사용된다.
- c_cpp_properties.json은 VSCode가 C/C++ 내부 경로 (헤더 파일 등) 의 위치를 알 수 있도록 해준다.