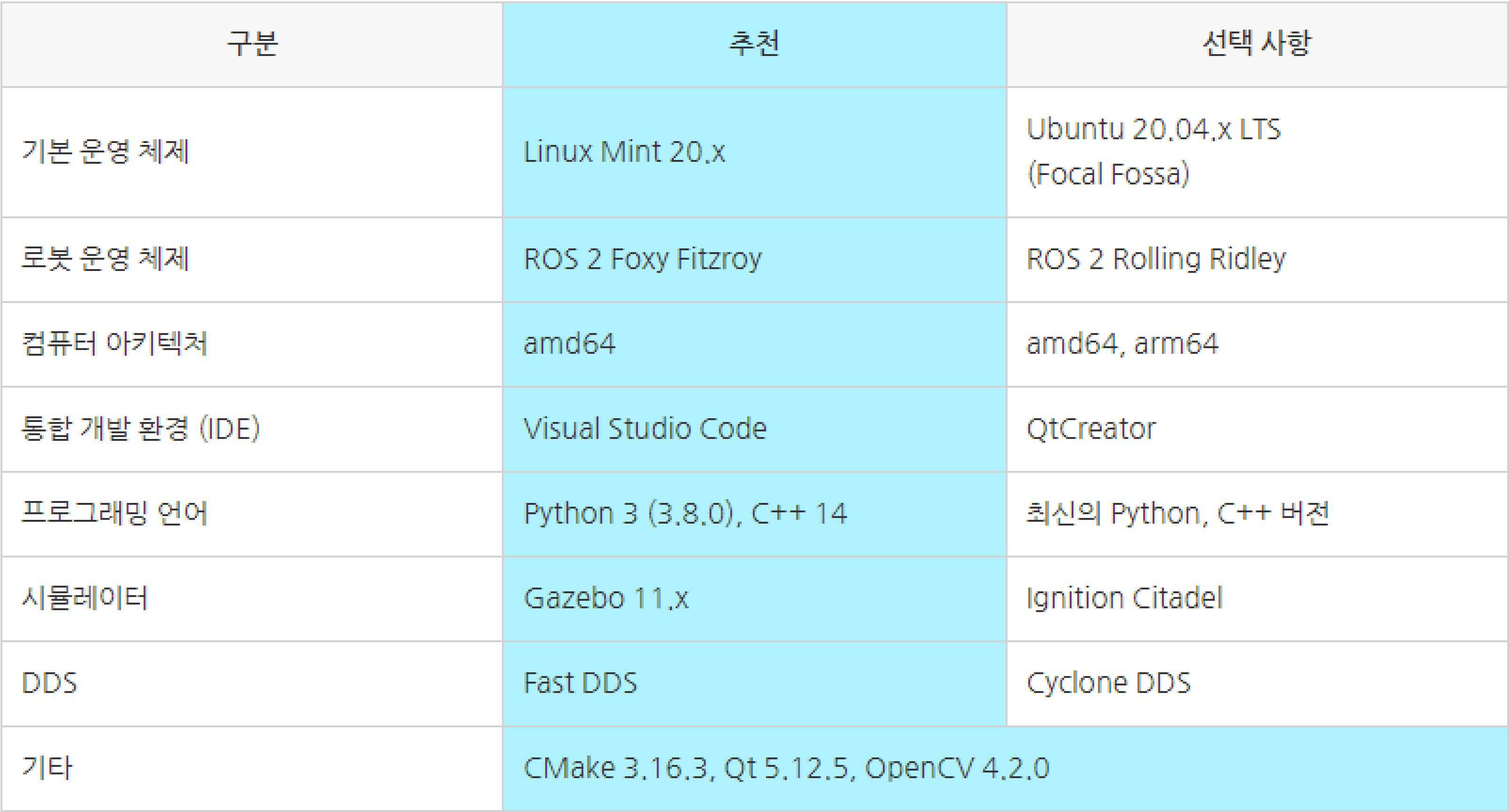
3장 세미나 정리 자료

3장 ROS 2 개발환경 구축

[출처] 001 ROS 2 개발 환경 구축 (오픈소스 소프트웨어 & 하드웨어: 로봇 기술 공유 카페 (오로카)) | 작성자 표윤석

3.1 개발환경



구분	추천	선택 사항
기본 운영 체제	Linux Mint 20.x	Ubuntu 20.04.x LTS (Focal Fossa)
로봇 운영 체제	ROS 2 Foxy Fitzroy	ROS 2 Rolling Ridley
컴퓨터 아키텍처	amd64	amd64, arm64
통합 개발 환경 (IDE)	Visual Studio Code	QtCreator
프로그래밍 언어	Python 3 (3.8.0), C++ 14	최신의 Python, C++ 버전
시뮬레이터	Gazebo 11.x	Ignition Citadel
DDS	Fast DDS	Cyclone DDS
기타	CMake 3.16.3, Qt 5.12.5, OpenCV 4.2.0	

https://cafe.naver.com/openrt/25288

3.2 기본 운영체제 설치

Linux Mint 20.x ?

Ubuntu 20.04(Focal Fossa)!

*windows 사용자 —> virtualbox가 아니라 듀얼 부팅으로 설치 권장

https://www.youtube.com/watch?v=u5QyjHIYwTQ&t=173s

https://www.youtube.com/watch?v=DF_TiZrwPAA

3.3 로봇 운영체제 설치

https://docs.ros.org/en/foxy/Installation/Ubuntu-Install-Debians.html

한줄씩 입력하여 설치할것!

3.3.1 지역 설정

```
locale# check for UTF-8

sudo apt update && sudo apt install localese

sudo locale-gen en_US en_US.UTF-8

sudo update-locale LC_ALL=en_US.UTF-8 LANG=en_US.UTF-8

export LANG=en_US.UTF-8
```

```
locale# verify settings
```

3.3.2 소스 설정

```
sudo apt update && sudo apt install curl gnupg2 lsb-release

sudo curl -sSL https://raw.githubusercontent.com/ros/rosdistro/master/ros.key  -o /usr/share/keyrings/ros-archive-keyring.gpg
```

3.3.3 ROS 2 패키지 설치

```
sudo apt update
```

```
sudo apt install ros-foxy-desktop
```

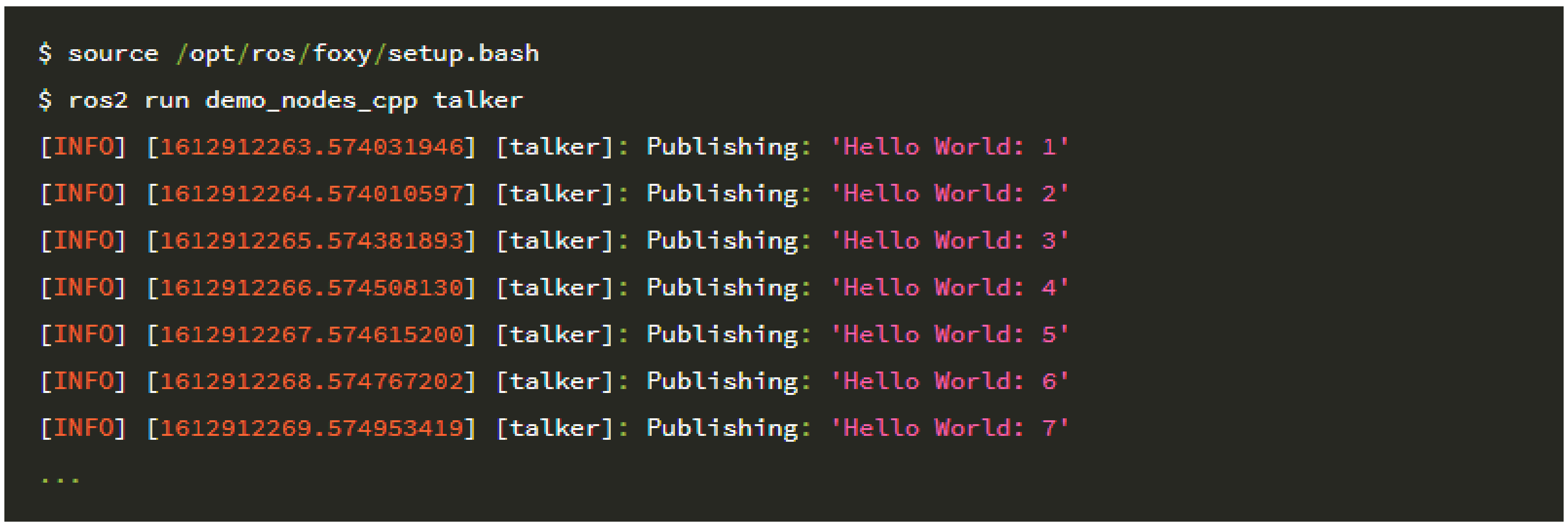
```
sudo apt install ros-foxy-rmw-fastrtps*
```

```
sudo apt install ros-foxy-rmw-cyclonedds*
```

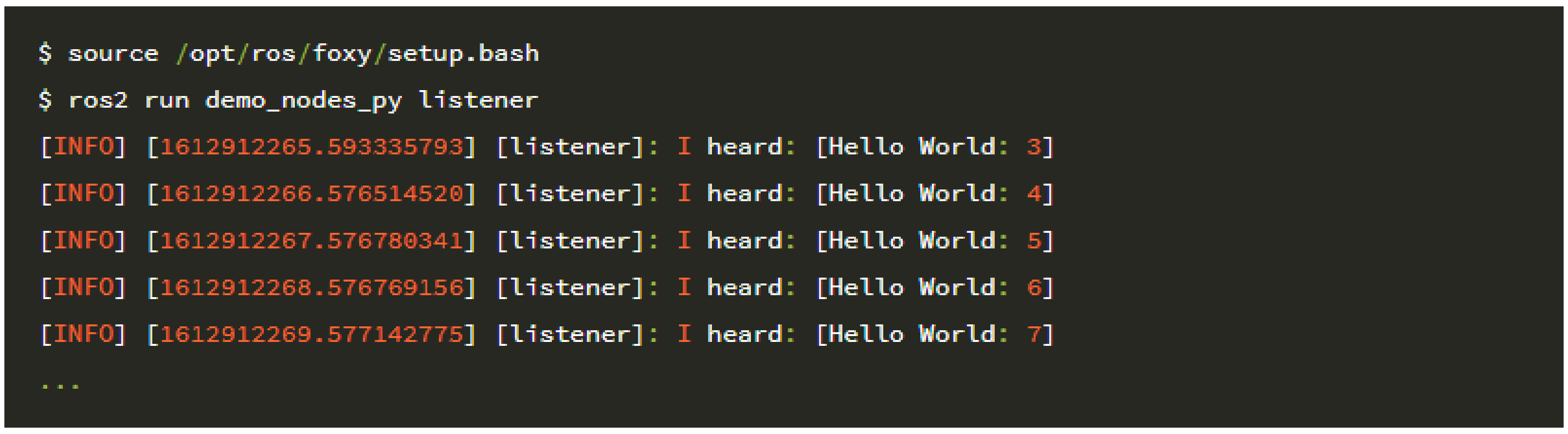
3.3.4 ROS 2 패키지 설치 확인

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

```
source /opt/ros/foxy/setup.bash
ros2 run demo_nodes_py listener
```



```
$ source /opt/ros/foxy/setup.bash
$ ros2 run demo_nodes_cpp talker
[INFO] [1612912263.574031946] [talker]: Publishing: 'Hello World: 1'
[INFO] [1612912264.574010597] [talker]: Publishing: 'Hello World: 2'
[INFO] [1612912265.574381893] [talker]: Publishing: 'Hello World: 3'
[INFO] [1612912266.574508130] [talker]: Publishing: 'Hello World: 4'
[INFO] [1612912267.574615200] [talker]: Publishing: 'Hello World: 5'
[INFO] [1612912268.574767202] [talker]: Publishing: 'Hello World: 6'
[INFO] [1612912269.574953419] [talker]: Publishing: 'Hello World: 7'
...
```



```
$ source /opt/ros/foxy/setup.bash
$ ros2 run demo_nodes_py listener
[INFO] [1612912265.593335793] [listener]: I heard: [Hello World: 3]
[INFO] [1612912266.576514520] [listener]: I heard: [Hello World: 4]
[INFO] [1612912267.576780341] [listener]: I heard: [Hello World: 5]
[INFO] [1612912268.576769156] [listener]: I heard: [Hello World: 6]
[INFO] [1612912269.577142775] [listener]: I heard: [Hello World: 7]
...
```

https://cafe.naver.com/openrt/25288

3.4 ROS 개발 툴 설치

맨 윗줄만 복사 붙여 넣기

```
sudo apt update && sudo apt install -y \
  build-essential \
  cmake \
  git \
  libbullet-dev \
  python3-colcon-common-extensions \
  python3-flake8 \
  python3-pip \
  python3-pytest-cov \
  python3-rosdep \
  python3-setuptools \
  python3-vcstool \
  wget
```

```
python3 -m pip install -U \
 argcomplete \
 flake8-blind-except \
 flake8-builtins \
 flake8-class-newline \
 flake8-comprehensions \
 flake8-deprecated \
 flake8-docstrings \
 flake8-import-order \
 flake8-quotes \
 pytest-repeat \
 pytest-rerunfailures \
 pytest
```

```
sudo apt install --no-install-recommends -y \
 libasio-dev \
 libtinyxml2-dev \
 libcunit1-dev
```

3.5 ROS 2 빌드 테스트

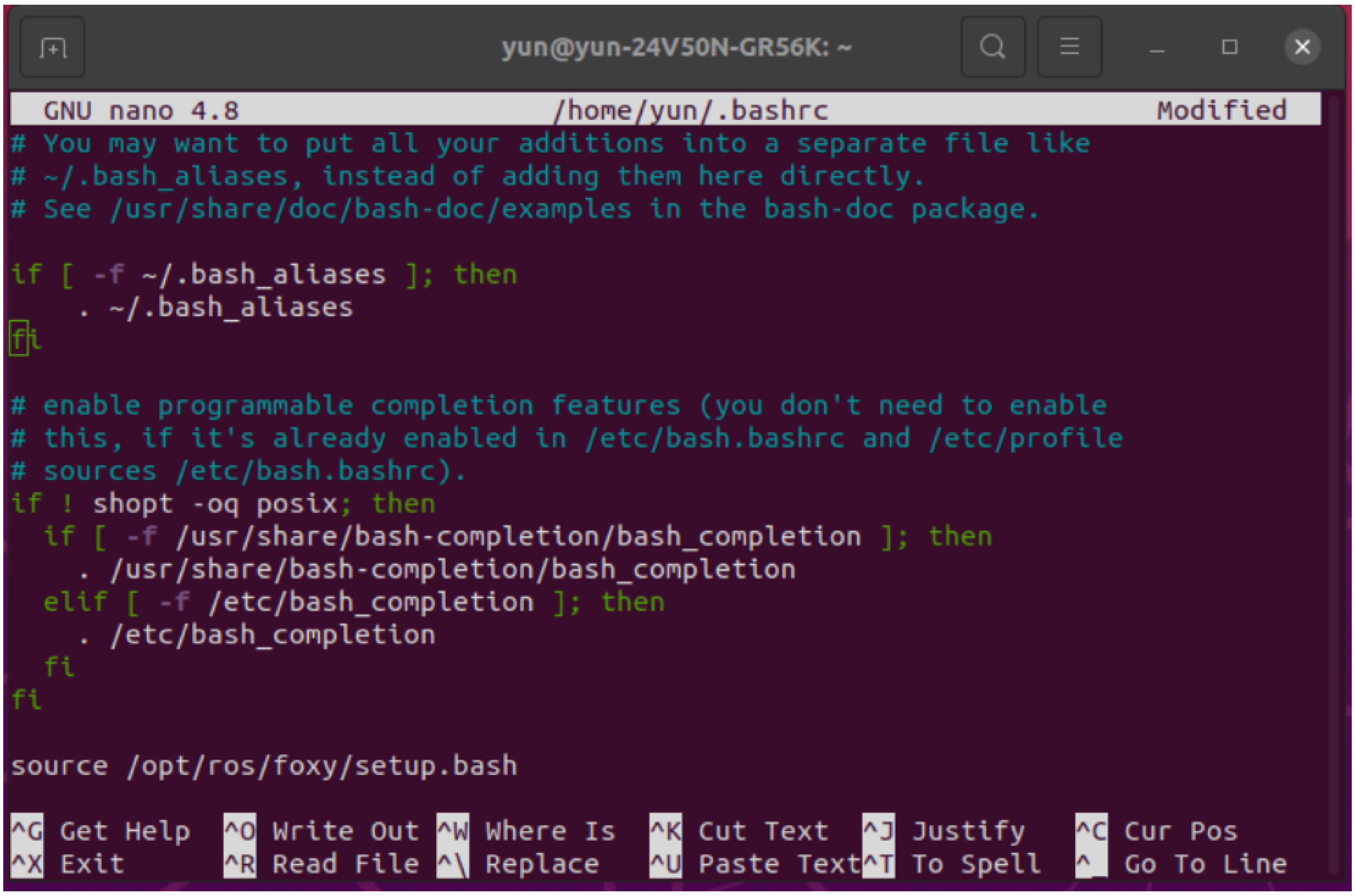
```
source /opt/ros/foxy/setup.bash
mkdir -p ~/robot_ws/src
cd ~/robot_ws/
colcon build --symlink-install
```

home 하위 폴더에 src, build, install, log 폴더 생성 확인 가능

3.6 Run commands 설정

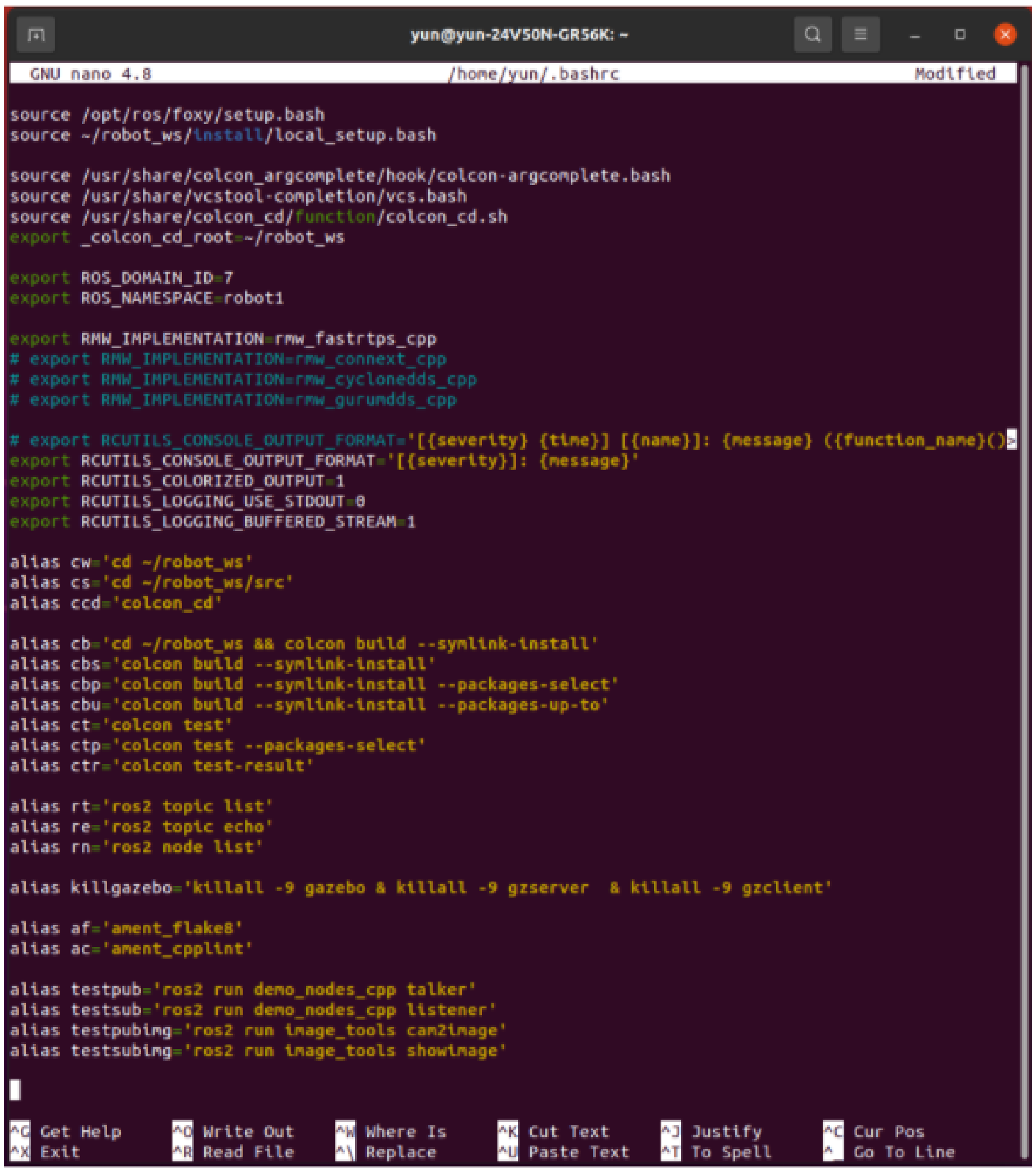
```
nano ~/.bashrc ge
```

(또는 vim ~/.bashrc 또는 xed ~/.bashrc)



```
  GNU nano 4.8                    /home/yun/.bashrc                      Modified
# You may want to put all your additions into a separate file like
# ~/.bash_aliases, instead of adding them here directly.
# See /usr/share/doc/bash-doc/examples in the bash-doc package.

if [ -f ~/.bash_aliases ]; then
    . ~/.bash_aliases
fi

# enable programmable completion features (you don't need to enable
# this, if it's already enabled in /etc/bash.bashrc and /etc/profile
# sources /etc/bash.bashrc).
if ! shopt -oq posix; then
  if [ -f /usr/share/bash-completion/bash_completion ]; then
    . /usr/share/bash-completion/bash_completion
  elif [ -f /etc/bash_completion ]; then
    . /etc/bash_completion
  fi
fi

source /opt/ros/foxy/setup.bash

^G Get Help   ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify    ^C Cur Pos
^X Exit       ^R Read File  ^\ Replace    ^U Paste Text ^T To Spell   ^_ Go To Line
```

```
source /opt/ros/foxy/setup.bash
source ~/robot_ws/install/local_setup.bash

source /usr/share/colcon_argcomplete/hook/colcon-argcomplete.bash
source /usr/share/vcstool-completion/vcs.bash
source /usr/share/colcon_cd/function/colcon_cd.sh
export _colcon_cd_root=~/robot_ws

export ROS_DOMAIN_ID=7
export ROS_NAMESPACE=robot1

export RMW_IMPLEMENTATION=rmw_fastrtps_cpp
# export RMW_IMPLEMENTATION=rmw_connext_cpp
# export RMW_IMPLEMENTATION=rmw_cyclonedds_cpp
# export RMW_IMPLEMENTATION=rmw_gurumdds_cpp

# export RCUTILS_CONSOLE_OUTPUT_FORMAT='[{severity} {time}] [{name}]: {message} ({function_name}() at {file_name}:{line_number})'
export RCUTILS_CONSOLE_OUTPUT_FORMAT='[{severity}]: {message}'
export RCUTILS_COLORIZED_OUTPUT=1
export RCUTILS_LOGGING_USE_STDOUT=0
export RCUTILS_LOGGING_BUFFERED_STREAM=1

alias cw='cd ~/robot_ws'
alias cs='cd ~/robot_ws/src'
alias ccd='colcon_cd'

alias cb='cd ~/robot_ws && colcon build --symlink-install'
alias cbs='colcon build --symlink-install'
alias cbp='colcon build --symlink-install --packages-select'
alias cbu='colcon build --symlink-install --packages-up-to'
alias ct='colcon test'
alias ctp='colcon test --packages-select'
alias ctr='colcon test-result'

alias rt='ros2 topic list'
alias re='ros2 topic echo'
alias rn='ros2 node list'

alias killgazebo='killall -9 gazebo & killall -9 gzserver  & killall -9 gzclient'

alias af='ament_flake8'
alias ac='ament_cpplint'

alias testpub='ros2 run demo_nodes_cpp talker'
alias testsub='ros2 run demo_nodes_cpp listener'
alias testpubimg='ros2 run image_tools cam2image'
alias testsubimg='ros2 run image_tools showimage'
```



```
  GNU nano 4.8                      /home/yun/.bashrc                      Modified

source /opt/ros/foxy/setup.bash
source ~/robot_ws/install/local_setup.bash

source /usr/share/colcon_argcomplete/hook/colcon-argcomplete.bash
source /usr/share/vcstool-completion/vcs.bash
source /usr/share/colcon_cd/function/colcon_cd.sh
export _colcon_cd_root=~/robot_ws

export ROS_DOMAIN_ID=7
export ROS_NAMESPACE=robot1

export RMW_IMPLEMENTATION=rmw_fastrtps_cpp
# export RMW_IMPLEMENTATION=rmw_connext_cpp
# export RMW_IMPLEMENTATION=rmw_cyclonedds_cpp
# export RMW_IMPLEMENTATION=rmw_gurumdds_cpp

# export RCUTILS_CONSOLE_OUTPUT_FORMAT='[{severity} {time}] [{name}]: {message} ({function_name}()>
export RCUTILS_CONSOLE_OUTPUT_FORMAT='[{severity}]: {message}'
export RCUTILS_COLORIZED_OUTPUT=1
export RCUTILS_LOGGING_USE_STDOUT=0
export RCUTILS_LOGGING_BUFFERED_STREAM=1

alias cw='cd ~/robot_ws'
alias cs='cd ~/robot_ws/src'
alias ccd='colcon_cd'

alias cb='cd ~/robot_ws && colcon build --symlink-install'
alias cbs='colcon build --symlink-install'
alias cbp='colcon build --symlink-install --packages-select'
alias cbu='colcon build --symlink-install --packages-up-to'
alias ct='colcon test'
alias ctp='colcon test --packages-select'
alias ctr='colcon test-result'

alias rt='ros2 topic list'
alias re='ros2 topic echo'
alias rn='ros2 node list'

alias killgazebo='killall -9 gazebo & killall -9 gzserver  & killall -9 gzclient'

alias af='ament_flake8'
alias ac='ament_cpplint'

alias testpub='ros2 run demo_nodes_cpp talker'
alias testsub='ros2 run demo_nodes_cpp listener'
alias testpubimg='ros2 run image_tools cam2image'
alias testsubimg='ros2 run image_tools showimage'

^G Get Help    ^O Write Out   ^W Where Is    ^K Cut Text    ^J Justify     ^C Cur Pos
^X Exit        ^R Read File   ^\ Replace     ^U Paste Text  ^T To Spell    ^_ Go To Line
```

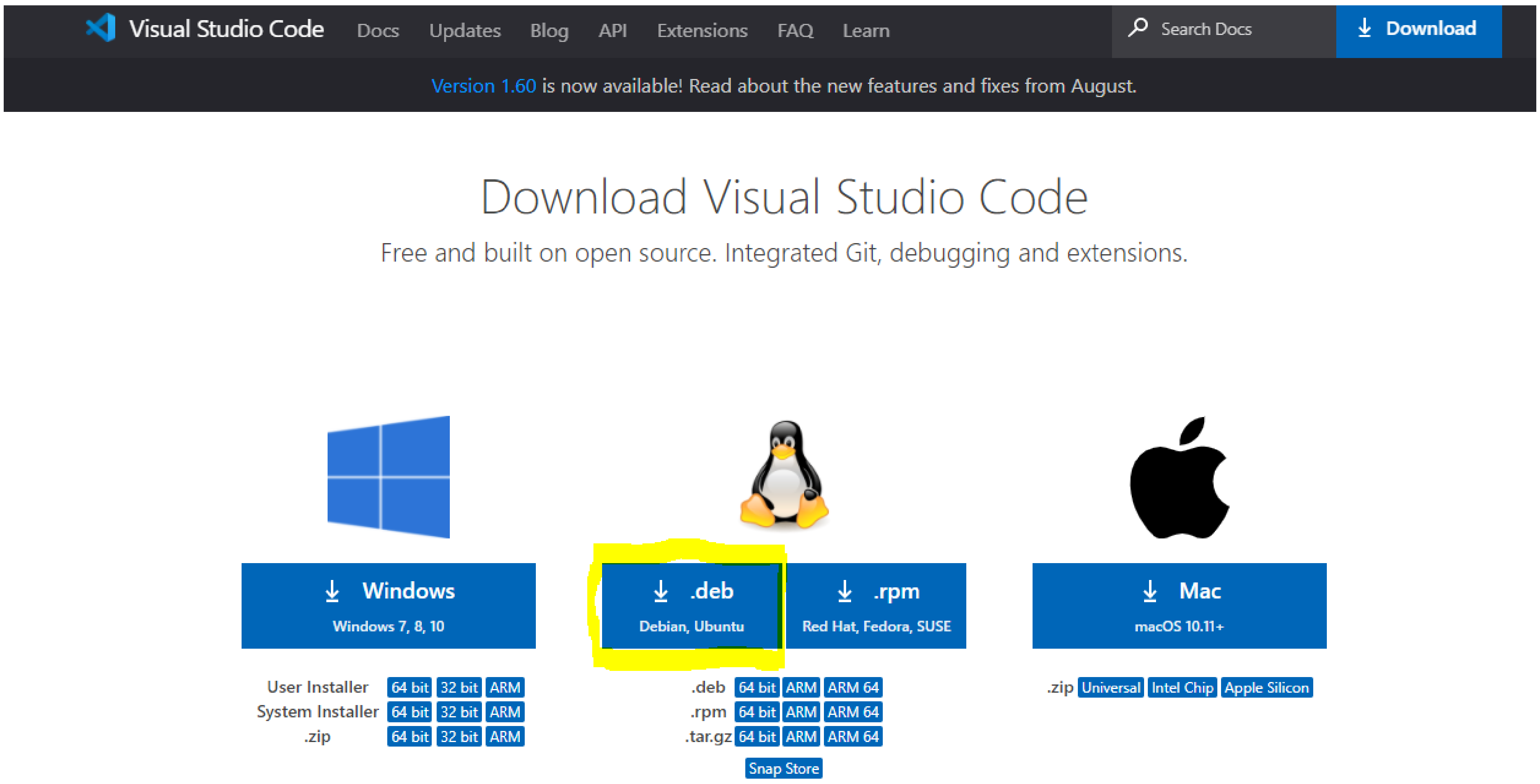
Ctrl X —> (save?) Yes —> Enter —> 터미널 닫기 —>nano ~/.bashrc 입력 후 확인!

3.7 통합 개발환경(IDE) 설치

3.7.1 Visual Studio Code

설치

https://code.visualstudio.com/Download



Version 1.60 is now available! Read about the new features and fixes from August.

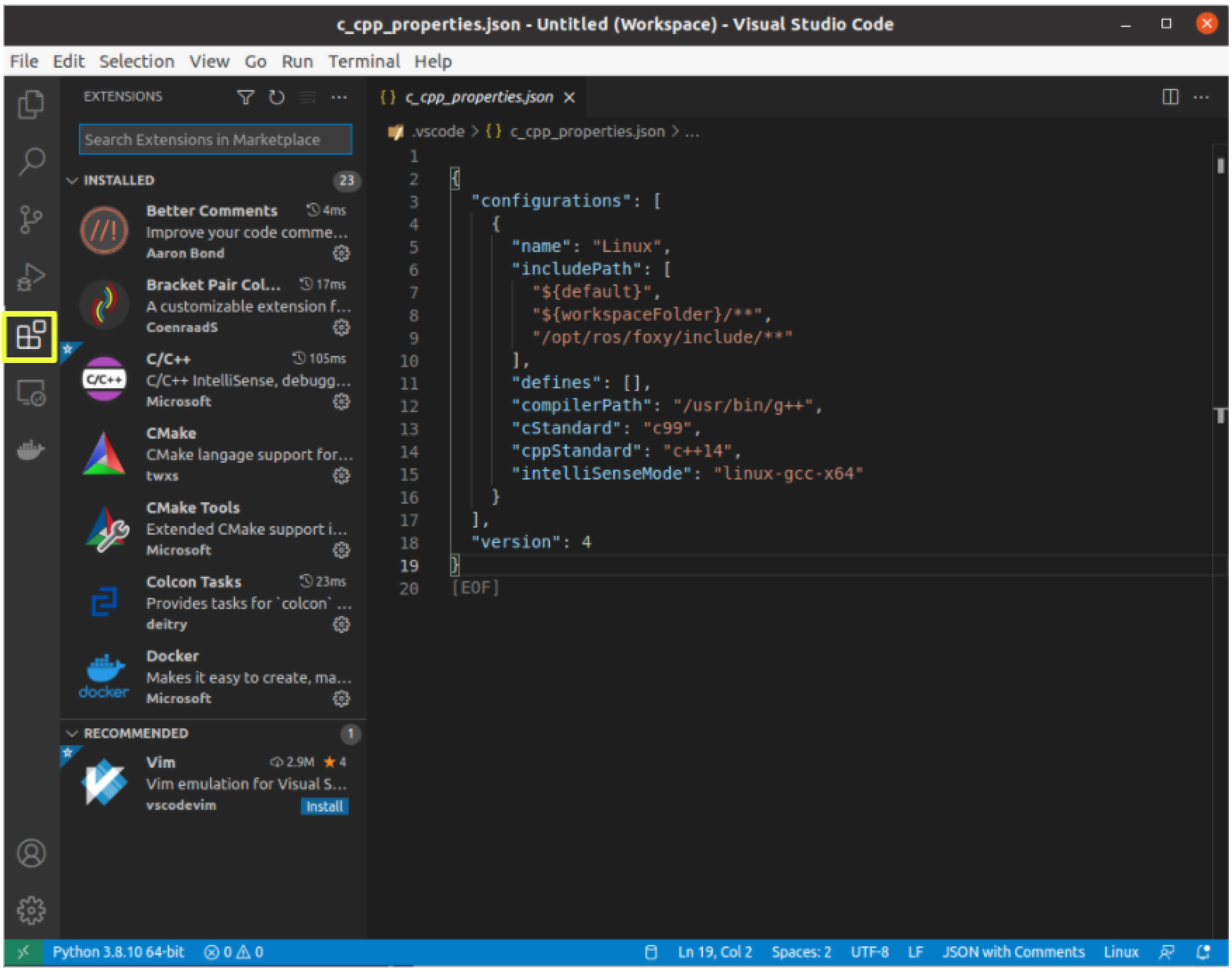
Download Visual Studio Code

Free and built on open source. Integrated Git, debugging and extensions.

실행

```
code
```

확장설치



c_cpp_properties.json - Untitled (Workspace) - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXTENSIONS
Search Extensions in Marketplace
INSTALLED 23
Better Comments 4ms
Improve your code comme...
Aaron Bond
Bracket Pair Col... 17ms
A customizable extension f...
CoenraadS
C/C++ 105ms
C/C++ IntelliSense, debugg...
Microsoft
CMake
CMake langage support for...
twxs
CMake Tools
Extended CMake support i...
Microsoft
Colcon Tasks 23ms
Provides tasks for `colcon` ...
deitry
Docker
Makes it easy to create, ma...
Microsoft
RECOMMENDED 1
Vim 2.9M ★4
Vim emulation for Visual S...
vscodevim
Install
{} c_cpp_properties.json ×
.vscode > {} c_cpp_properties.json > ...
Python 3.8.10 64-bit ⊗ 0 ⚠ 0
Ln 19, Col 2 Spaces: 2 UTF-8 LF JSON with Comments Linux

```

{
  "configurations": [
    {
      "name": "Linux",
      "includePath": [
        "${default}",
        "${workspaceFolder}/**",
        "/opt/ros/foxy/include/**"
      ],
      "defines": [],
      "compilerPath": "/usr/bin/g++",
      "cStandard": "c99",
      "cppStandard": "c++14",
      "intelliSenseMode": "linux-gcc-x64"
    }
  ],
  "version": 4
}
[EOF]
```

- **C/C++/Python Extensions (VS Code Extensions for C++ and Python)**

Aa 이름	≡ 코드명	≡ 설명
C/C++	ms-vscode.cpptools	C/C ++ IntelliSense, 디버깅 및 코드 검색
CMake	twxs.cmake	CMake 언어 지원
CMake Tools	ms-vscode.cmake-tools	CMake 언어 지원 및 다양한 툴
Python	ms-python.python	린팅, 디버깅, Intellisense, 코드 서식 지정, 리팩토링, 단위 테스트 등

- **ROS Extensions (VS Code Extensions for ROS, URDF, Colcon)**

Aa 이름	≡ 코드명	≡ 설명
ROS	ms-iot.vscode-ros	ROS 개발 지원
URDF	smilerobotics.urdf	URDF/xacro 지원
Colcon Tasks	deitry.colcon-helper	Colcon 명령어를 위한 VSCode Task

- **File Format Extensions (VS Code Extensions for XML, YAML, Markdown)**

Aa 이름	≡ 코드명	≡ 설명
XML Tools	dotjoshjohnson.xml	XML, XQuery, XPath 지원
YAML	redhat.vscode-yaml	YAML 지원
Markdown All in One	yzhang.markdown-all-in-one	Markdown 지원

* 유용한 Extensions (VS Code Extensions for Etc.)

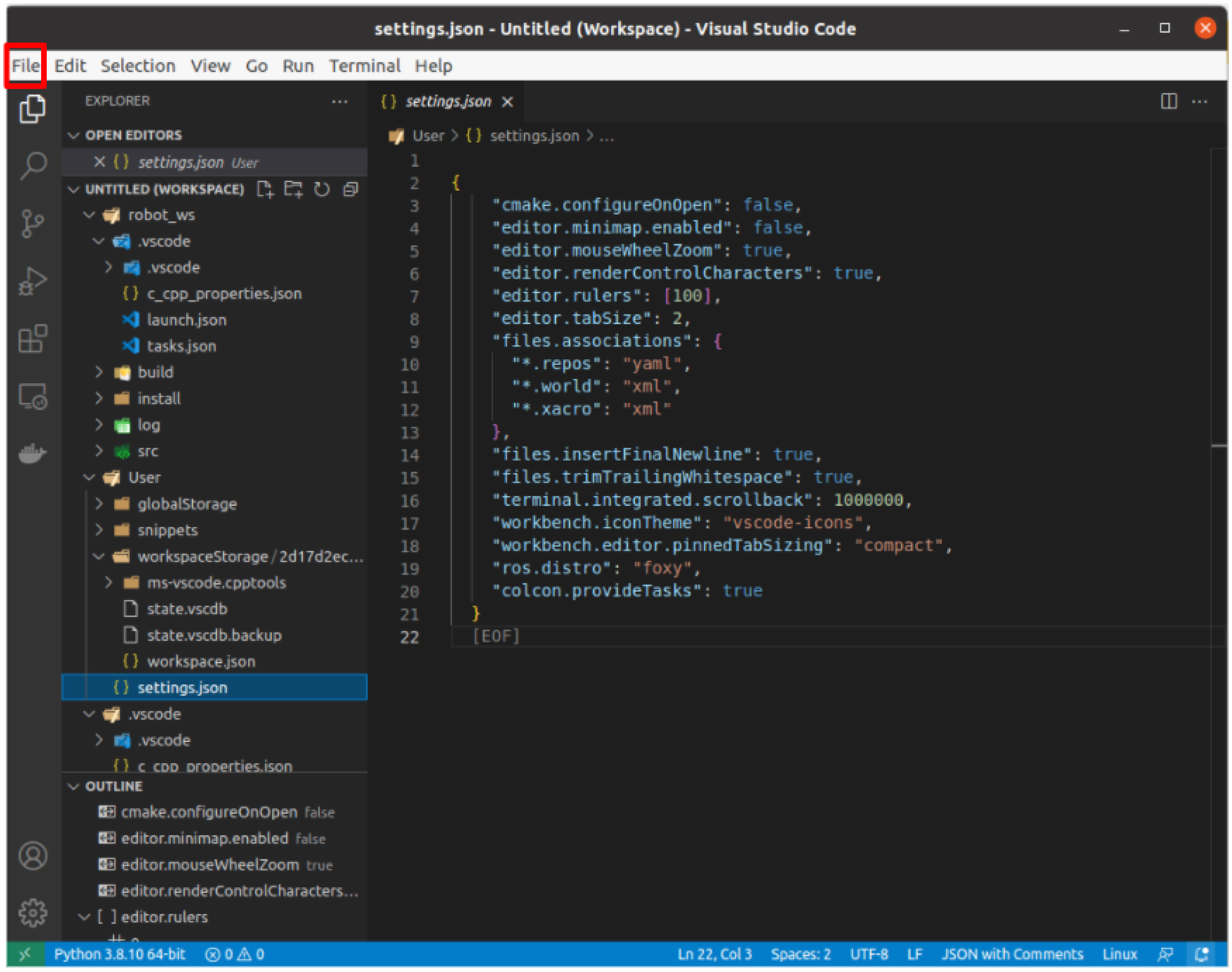
Aa 이름	≡ 코드명	≡ 설명
Highlight Trailing White Spaces	ybaumes.highlight-trailing-white-spaces	의미 없이 사용된 공백의 스페이스 문자 강조
EOF Mark	msfukui.eof-mark	[EOF]이 없이 끝난 파일에 [EOF] 문자 추가
Bracket Pair Colorizer	coenraads.bracket-pair-colorizer	괄호 열기/닫기를 짝을 맞추어 색상화 시킴
Better Comments	aaron-bond.better-comments	alert, informational, TODO 등의 코멘트 강화 기능

- **기타 추천 Extensions**
- ms-azuretools.vscode-docker
- ms-vscode-remote.remote-ssh
- ms-vscode-remote.remote-ssh-edit
- ms-vscode-remote.remote-containers
- ms-python.vscode-pylance
- ms-toolsai.jupyter
- dbaeumer.vscode-eslint
- uctakeoff.vscode-counter
- vscode-icons-team.vscode-icons

워크스페이스 설정

File —> Add Folder to Workspace...

robot_ws 선택



```

{
  "cmake.configureOnOpen": false,
  "editor.minimap.enabled": false,
  "editor.mouseWheelZoom": true,
  "editor.renderControlCharacters": true,
  "editor.rulers": [100],
  "editor.tabSize": 2,
  "files.associations": {
    "*.repos": "yaml",
    "*.world": "xml",
    "*.xacro": "xml"
  },
  "files.insertFinalNewline": true,
  "files.trimTrailingWhitespace": true,
  "terminal.integrated.scrollback": 1000000,
  "workbench.iconTheme": "vscode-icons",
  "workbench.editor.pinnedTabSizing": "compact",
  "ros.distro": "foxy",
  "colcon.provideTasks": true
}
[EOF]
```

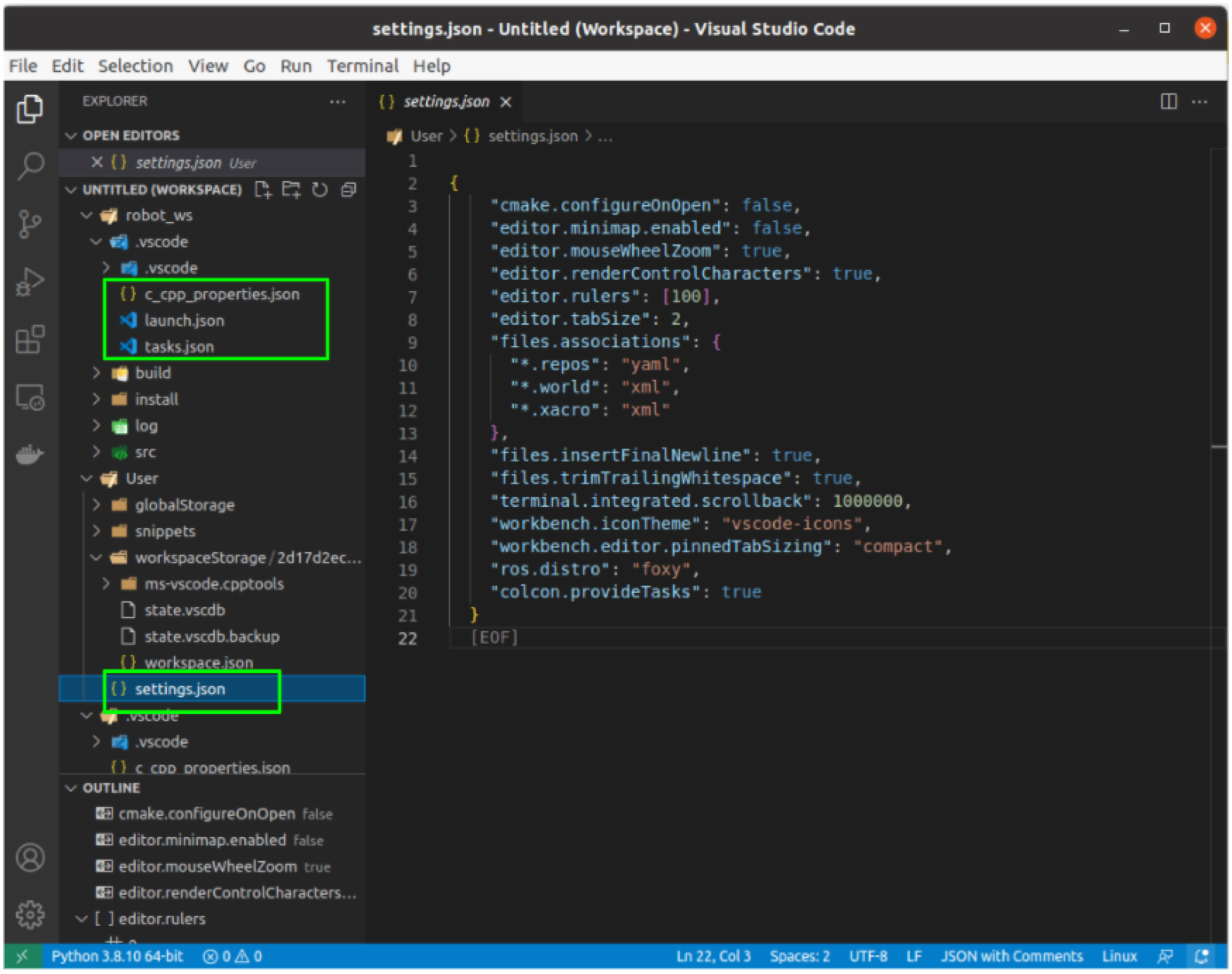
VSCode의 개발환경 설정

~/.config/Code/User/settings.json

~/robot_ws/.vscode/c_cpp_properties.json

~/robot_ws/.vscode/tasks.json

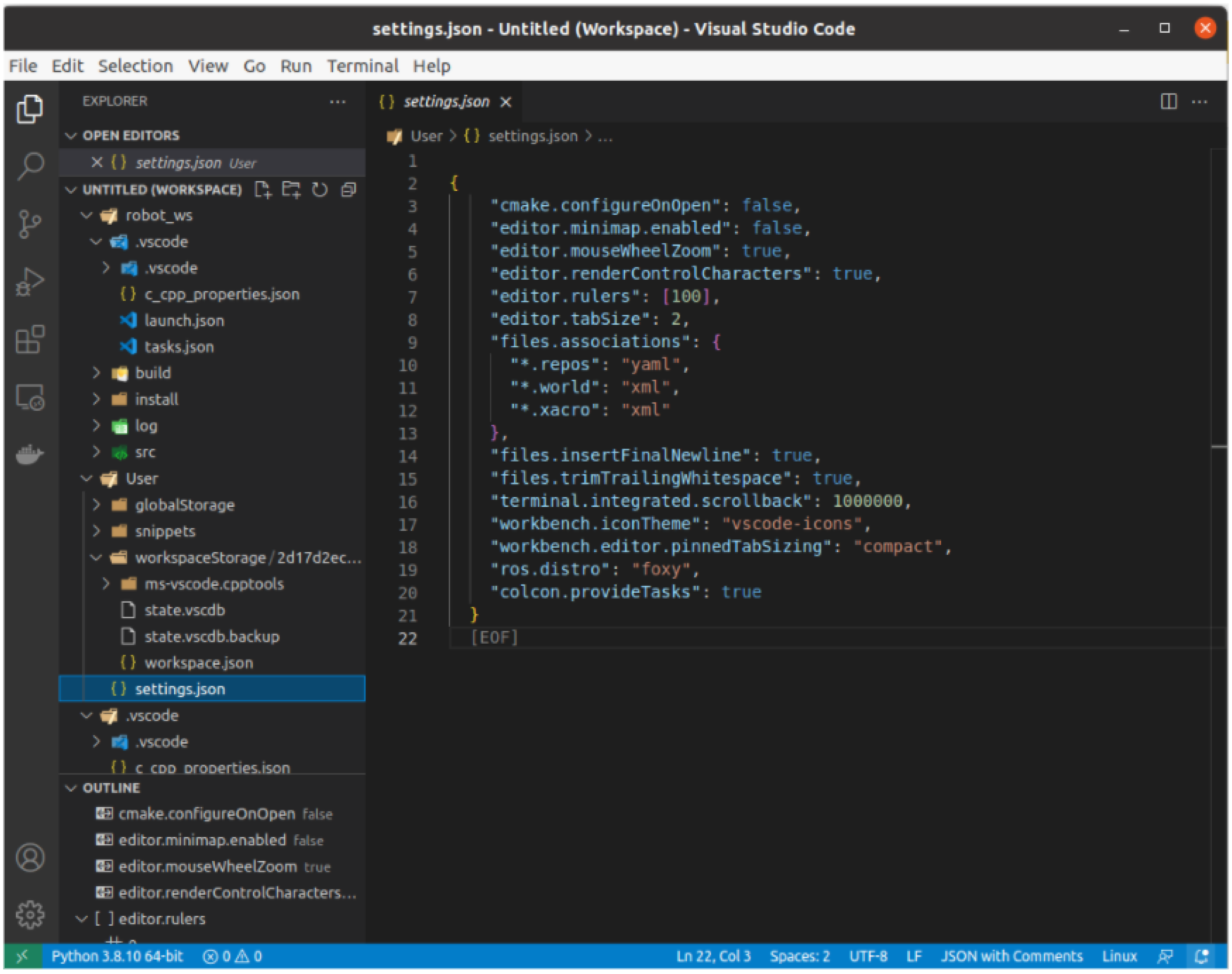
~/robot_ws/.vscode/launch.json



```

{
  "cmake.configureOnOpen": false,
  "editor.minimap.enabled": false,
  "editor.mouseWheelZoom": true,
  "editor.renderControlCharacters": true,
  "editor.rulers": [100],
  "editor.tabSize": 2,
  "files.associations": {
    "*.repos": "yaml",
    "*.world": "xml",
    "*.xacro": "xml"
  },
  "files.insertFinalNewline": true,
  "files.trimTrailingWhitespace": true,
  "terminal.integrated.scrollback": 1000000,
  "workbench.iconTheme": "vscode-icons",
  "workbench.editor.pinnedTabSizing": "compact",
  "ros.distro": "foxy",
  "colcon.provideTasks": true
}
[EOF]
```

User settings 설정

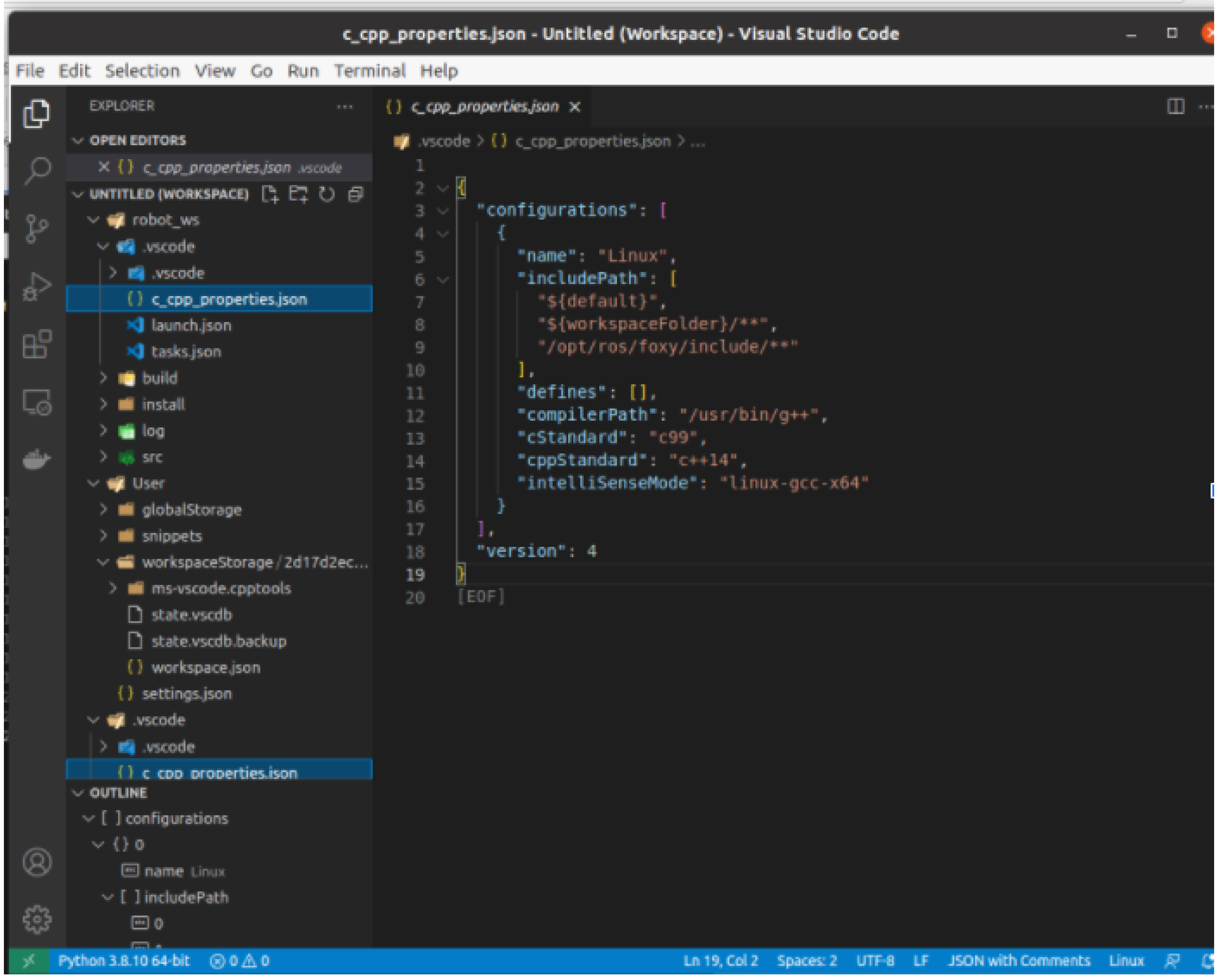
```
{
  "cmake.configureOnOpen": false,
  "editor.minimap.enabled": false,
  "editor.mouseWheelZoom": true,
  "editor.renderControlCharacters": true,
  "editor.rulers": [100],
  "editor.tabSize": 2,
  "files.associations": {
    "*.repos": "yaml",
    "*.world": "xml",
    "*.xacro": "xml"
  },
  "files.insertFinalNewline": true,
  "files.trimTrailingWhitespace": true,
  "terminal.integrated.scrollback": 1000000,
  "workbench.iconTheme": "vscode-icons",
  "workbench.editor.pinnedTabSizing": "compact",
  "ros.distro": "foxy",
  "colcon.provideTasks": true
}
```



```

{
  "cmake.configureOnOpen": false,
  "editor.minimap.enabled": false,
  "editor.mouseWheelZoom": true,
  "editor.renderControlCharacters": true,
  "editor.rulers": [100],
  "editor.tabSize": 2,
  "files.associations": {
    "*.repos": "yaml",
    "*.world": "xml",
    "*.xacro": "xml"
  },
  "files.insertFinalNewline": true,
  "files.trimTrailingWhitespace": true,
  "terminal.integrated.scrollback": 1000000,
  "workbench.iconTheme": "vscode-icons",
  "workbench.editor.pinnedTabSizing": "compact",
  "ros.distro": "foxy",
  "colcon.provideTasks": true
}
[EOF]
```

C/C++ properties 설정

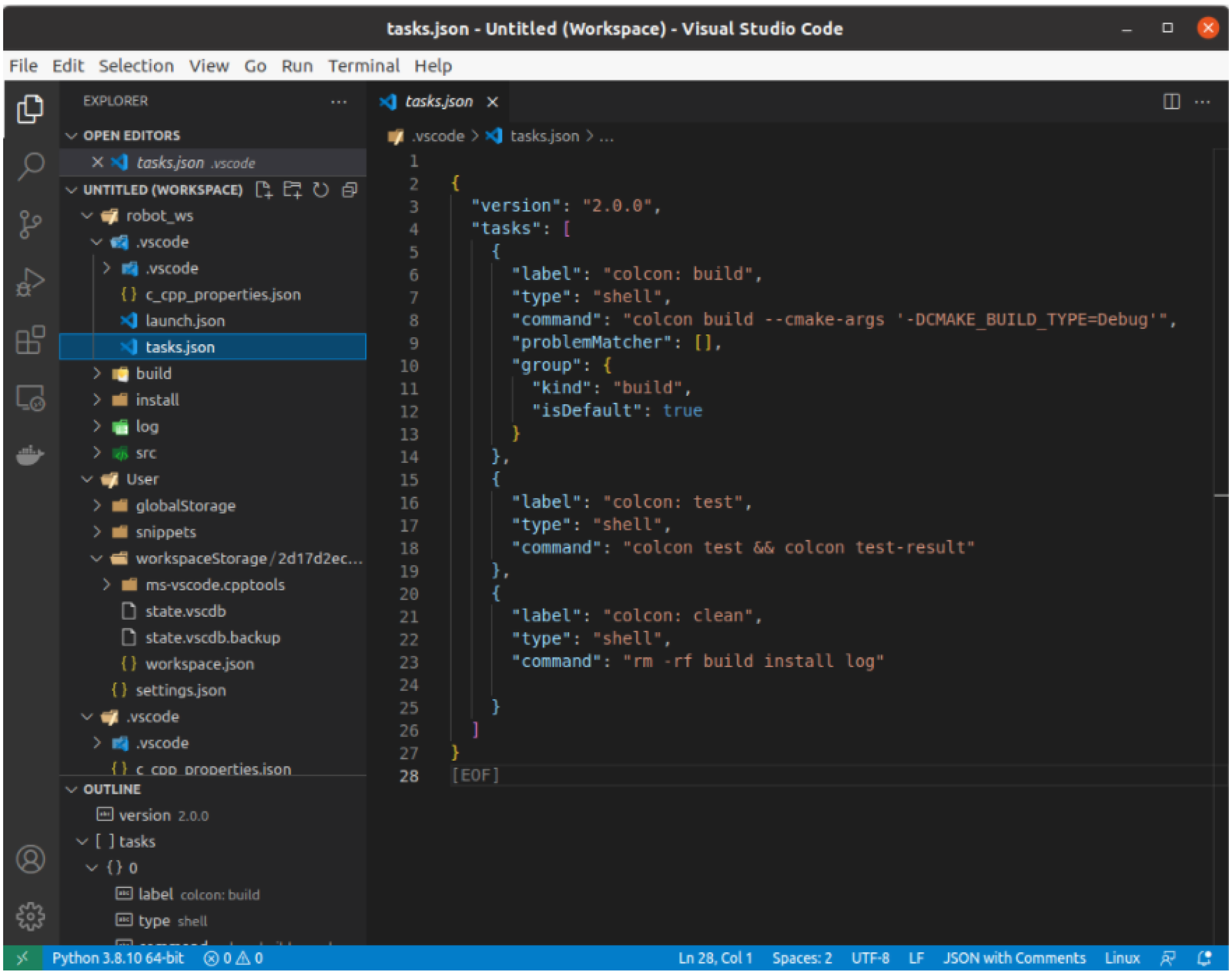
```
{
 "configurations": [
  {
   "name": "Linux",
   "includePath": [
    "${default}",
    "${workspaceFolder}/**",
    "/opt/ros/foxy/include/**"
   ],
   "defines": [],
   "compilerPath": "/usr/bin/g++",
   "cStandard": "c99",
   "cppStandard": "c++14",
   "intelliSenseMode": "linux-gcc-x64"
  }
 ],
 "version": 4
}
```



```

{
  "configurations": [
    {
      "name": "Linux",
      "includePath": [
        "${default}",
        "${workspaceFolder}/**",
        "/opt/ros/foxy/include/**"
      ],
      "defines": [],
      "compilerPath": "/usr/bin/g++",
      "cStandard": "c99",
      "cppStandard": "c++14",
      "intelliSenseMode": "linux-gcc-x64"
    }
  ],
  "version": 4
}
[EOF]
```

Tasks 설정

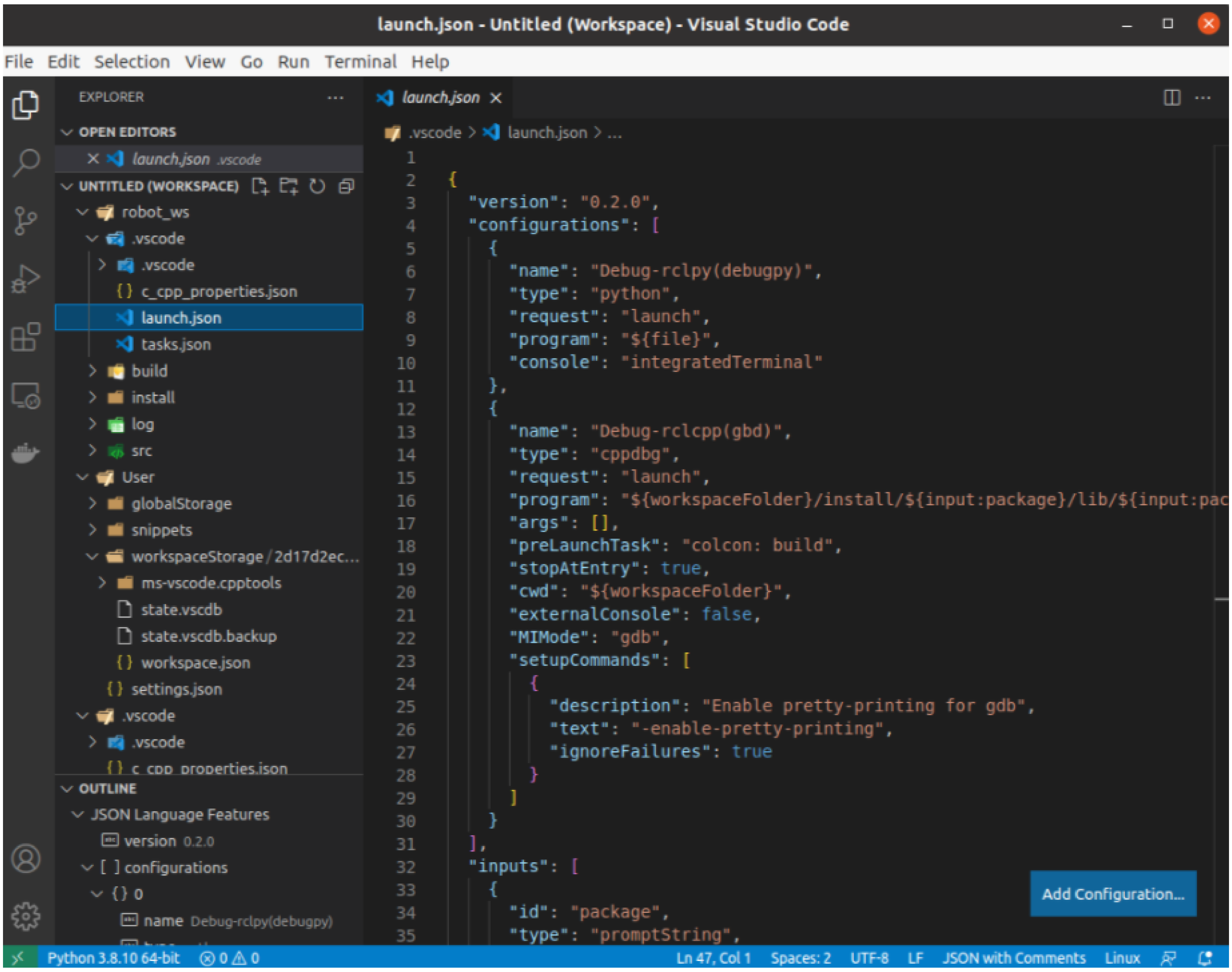
```
{
  "version": "2.0.0",
  "tasks": [
    {
      "label": "colcon: build",
      "type": "shell",
      "command": "colcon build --cmake-args '-DCMAKE_BUILD_TYPE=Debug'",
      "problemMatcher": [],
      "group": {
        "kind": "build",
        "isDefault": true
      }
    },
    {
      "label":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

    }
  ]
}
```



```

{
  "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"
    }
  ]
}
[EOF]
```

Launch 설정

```
{
 "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"
  }
 ]
}
```



```

{
  "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:pac
      "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",
```

저장 Ctrl S

빌드 Ctrl Shift B

디버깅 Ctrl Shift D

rclcpp

- Run and Debug (`Ctrl + Shift + d`)로 이동
- "Debug-rclcpp(gbd)" 선택
- "Package name" 입력 (예: topic_service_action_rclcpp_example)
- "node name" 입력 (예: argument)
- Start Debugging 클릭 (`F5`)

rclpy

- Run and Debug (`Ctrl + Shift + d`)로 이동
- "Debug-rclpy(debugpy)" 선택
- Start Debugging 클릭 (`F5`)

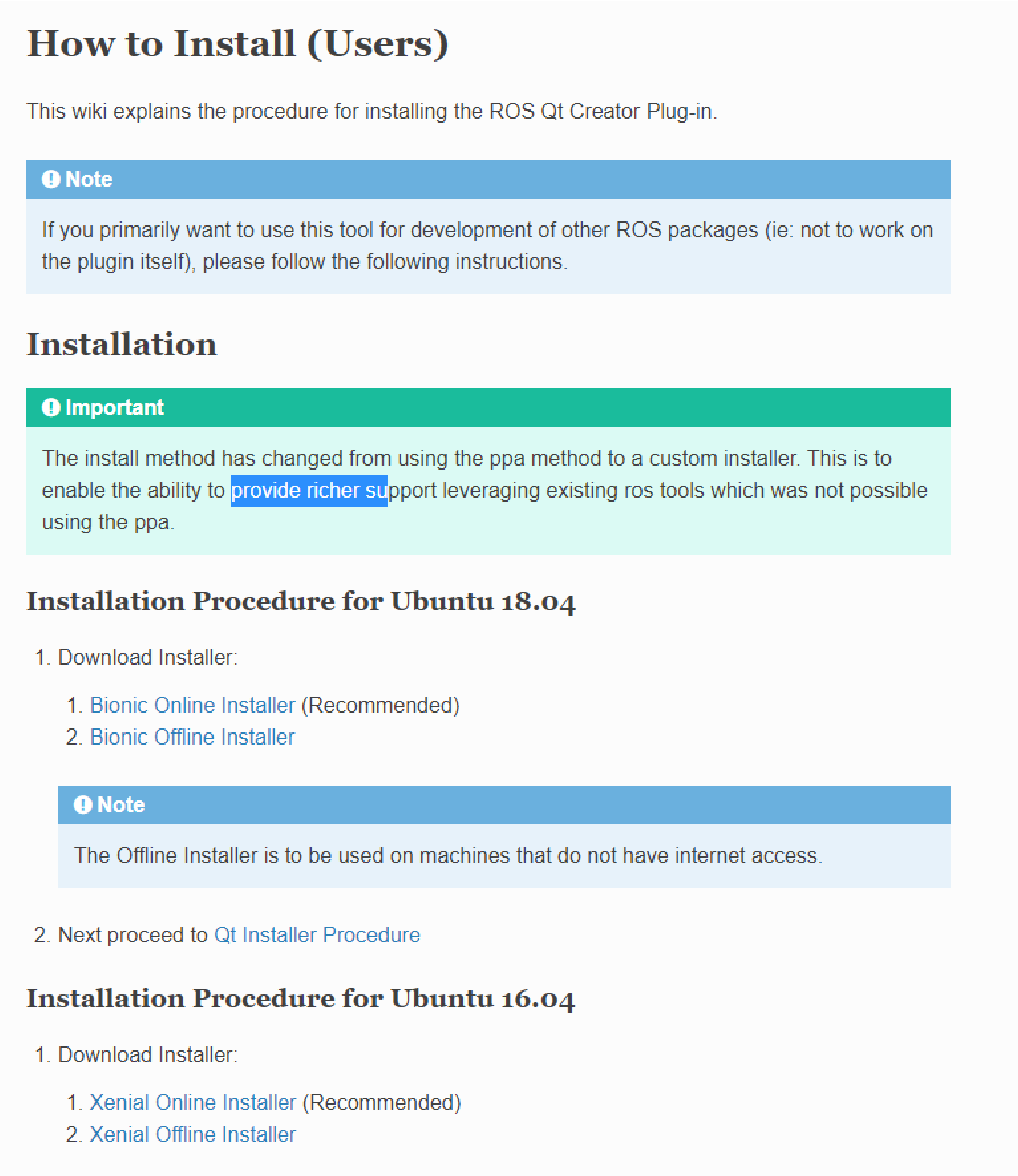
3.7.2 QtCreator설치

```
sudo apt install qtcreator
```

3.7.3 QtCreator Plug-in for ROS

https://ros-qtc-plugin.readthedocs.io/en/latest/index.html

ubuntu 20.04 ?



How to Install (Users)

This wiki explains the procedure for installing the ROS Qt Creator Plug-in.

Note

If you primarily want to use this tool for development of other ROS packages (ie: not to work on the plugin itself), please follow the following instructions.

Installation

Important

The install method has changed from using the ppa method to a custom installer. This is to enable the ability to provide richer support leveraging existing ros tools which was not possible using the ppa.

Installation Procedure for Ubuntu 18.04

1. Download Installer:
    1. Bionic Online Installer (Recommended)
    2. Bionic Offline Installer

    Note

    The Offline Installer is to be used on machines that do not have internet access.

2. Next proceed to Qt Installer Procedure

Installation Procedure for Ubuntu 16.04

1. Download Installer:
    1. Xenial Online Installer (Recommended)
    2. Xenial Offline Installer

3.7.4 기타

http://wiki.ros.org/IDEs

3.8 ROS 2 삭제

```
sudo apt remove ros-foxy-* && sudo apt autoremove
```