

# **개발환경 구축 메뉴얼**

# 목차

---

## 1. 개발환경 구축

- Visual C++ Runtime 설치
- Windows Terminal 설치
- choco 설치
- python 3.7.5 버전 설치
- visual c++ redistributables/ visual studio community 설치
- OPENSSE, OPENCV, CMAKE 설치
- nupkg 설치
- 파이썬 종속패키지 설치
- ROS2 파일 압축 풀기
- Opensplice 설치
- RTI 설치 및 ROS DOMAIN ID 설정
- ROS2 설치 확인

## 2. 작업환경 구축

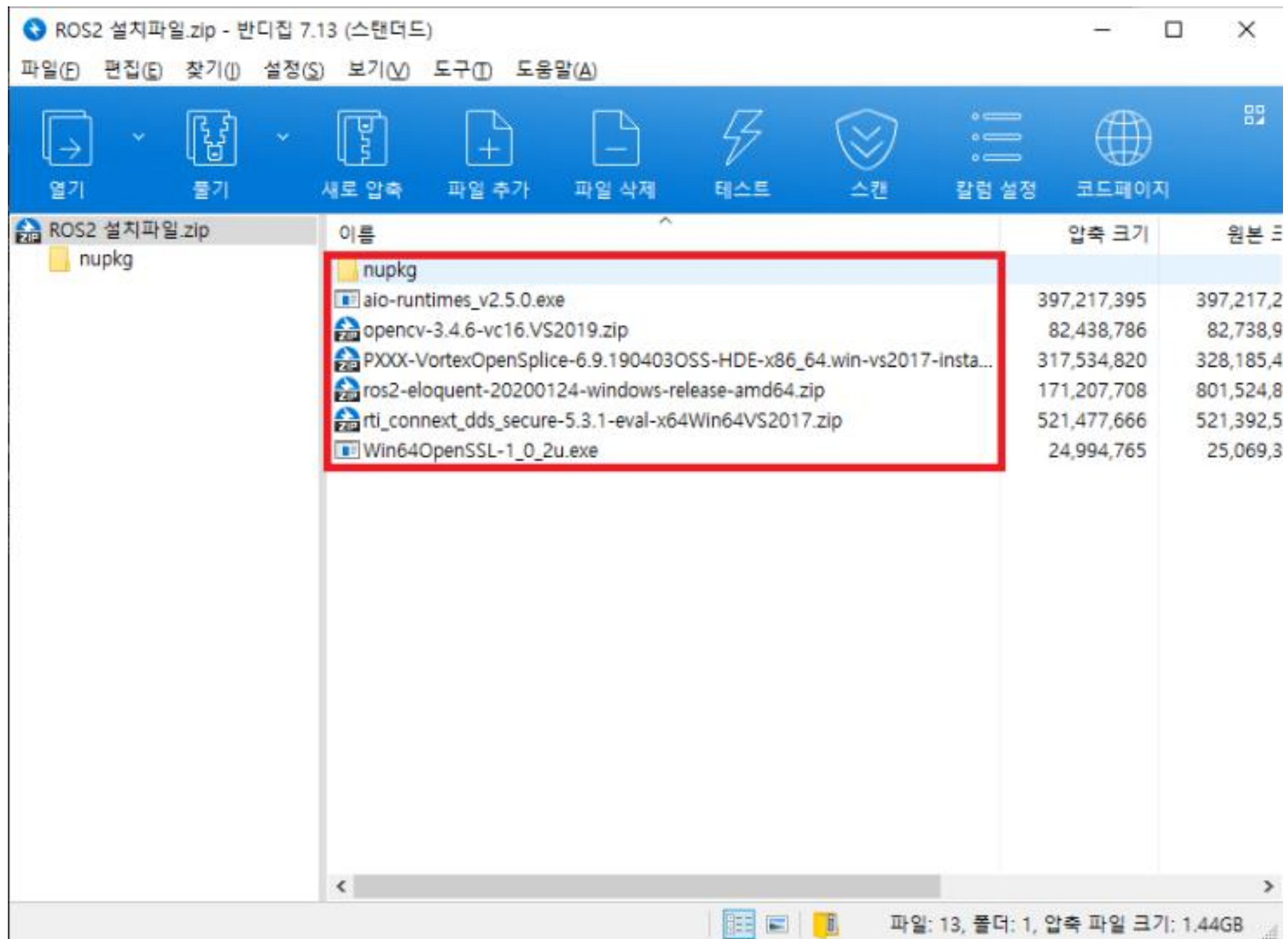
- 작업 폴더인 Workspace 만들기
- 스켈레톤 코드 설치
- 코드 빌드
- ssafy\_bridge 실행
- 네트워크 세팅

	프로그램 및 라이브러리	버전
ROS 관련	ROS	eloquent (20200124 release)
	python	3.7.5
	openssl	1.0.2u
	choco	0.10.15
	opencv	3.4.6
	rti	5.3.1
	opensplice	6.9.190403
tensorflow 관련	tensorflow	1.15
	CUDA Toolkit	10.0
	cuDNN	7.6.4

**개발환경 구축**

# 개발환경 구축

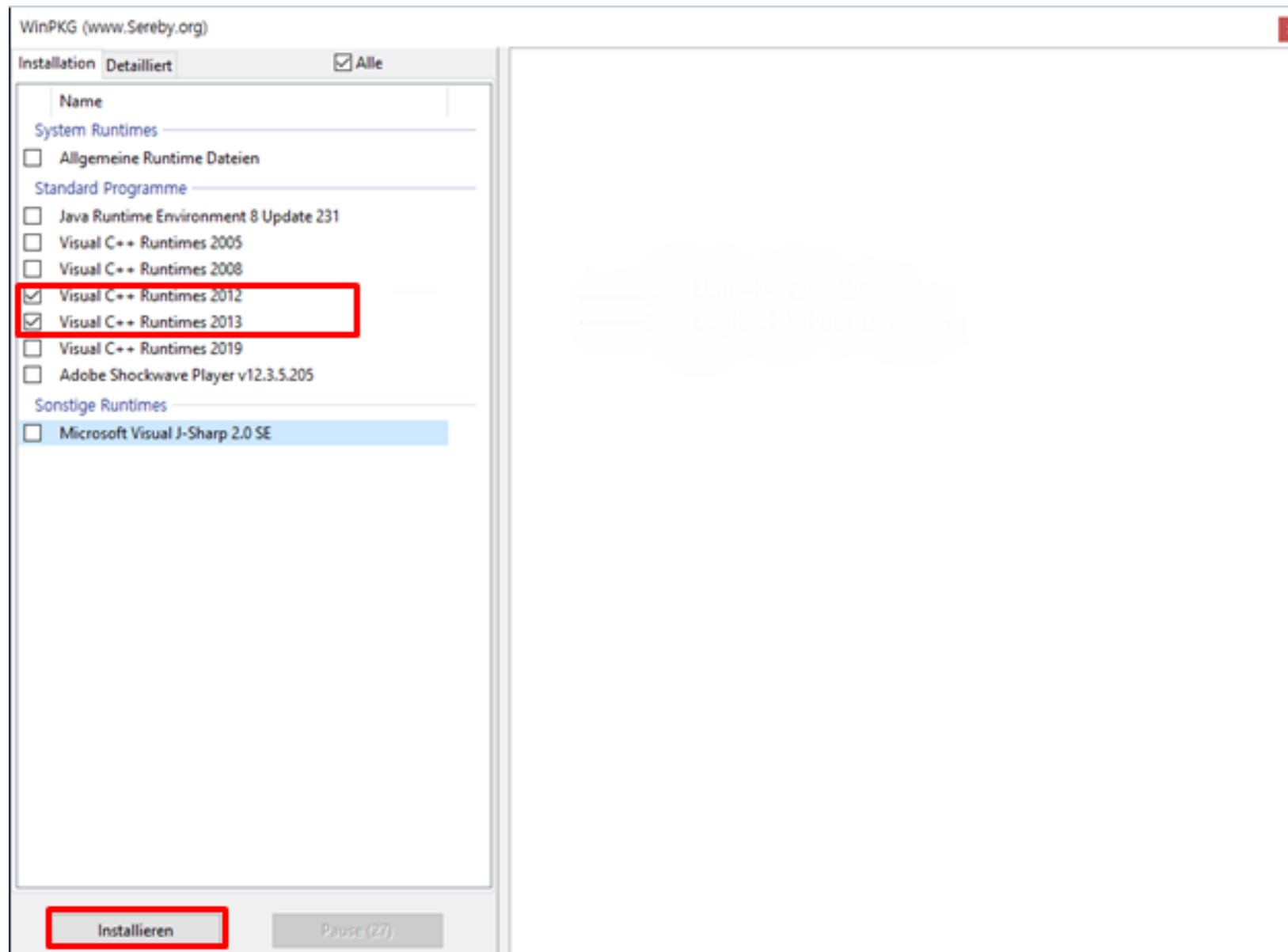
- 개발환경 설치 파일
  - 개발 환경에 필요한 파일은 “ ROS2설치파일.zip ”으로 제공된다.



# 개발환경 구축

- Visual C++ Runtime 설치

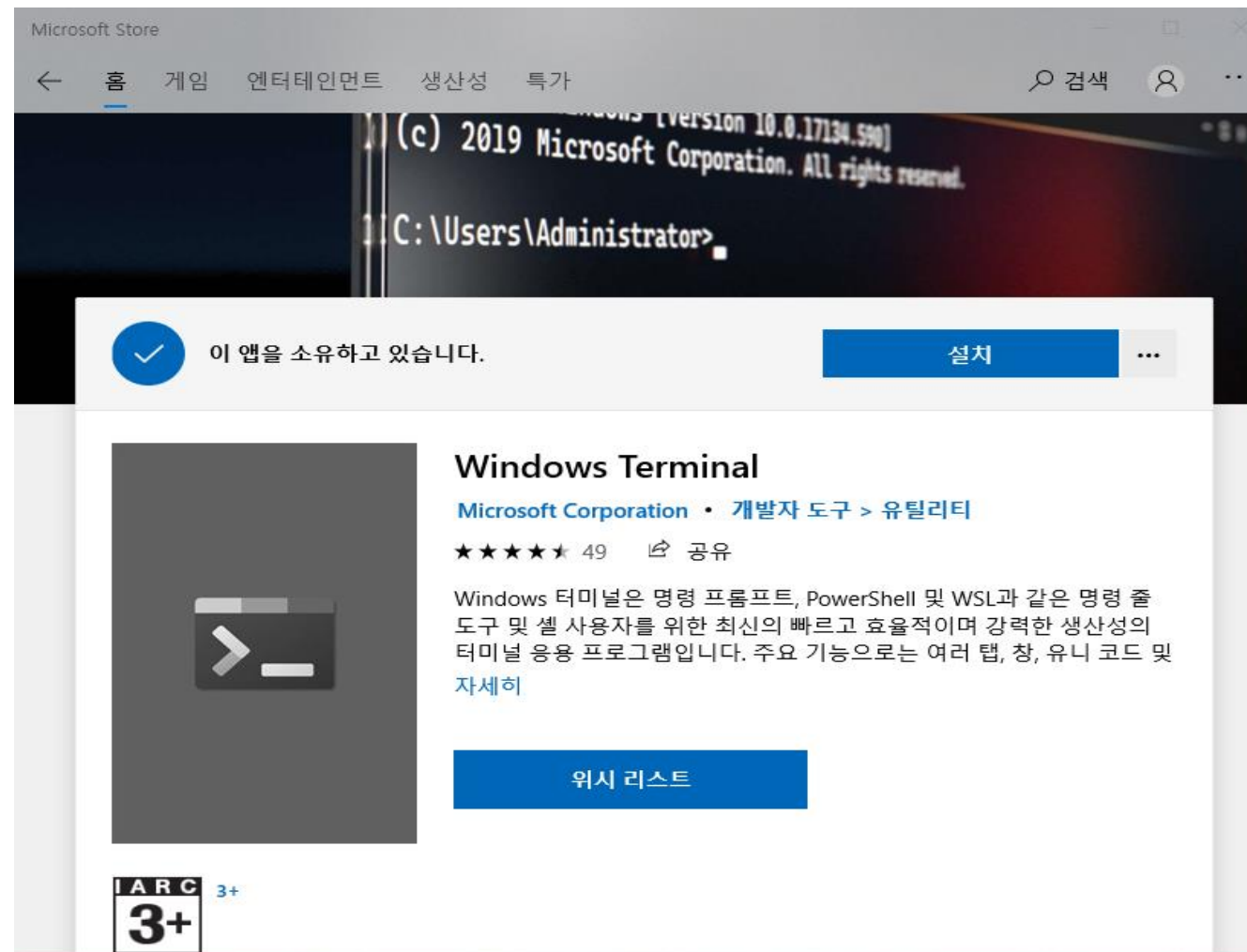
- 적합한 Visual C++ Runtime이 설치 되지 않은 경우 시뮬레이터가 실행 되지 않을 수 있다.
- 제공된 aio-runtime.exe를 실행 시켜 visual c++ Runtime 2012, 2013을 설치 한다.



# 개발환경 구축

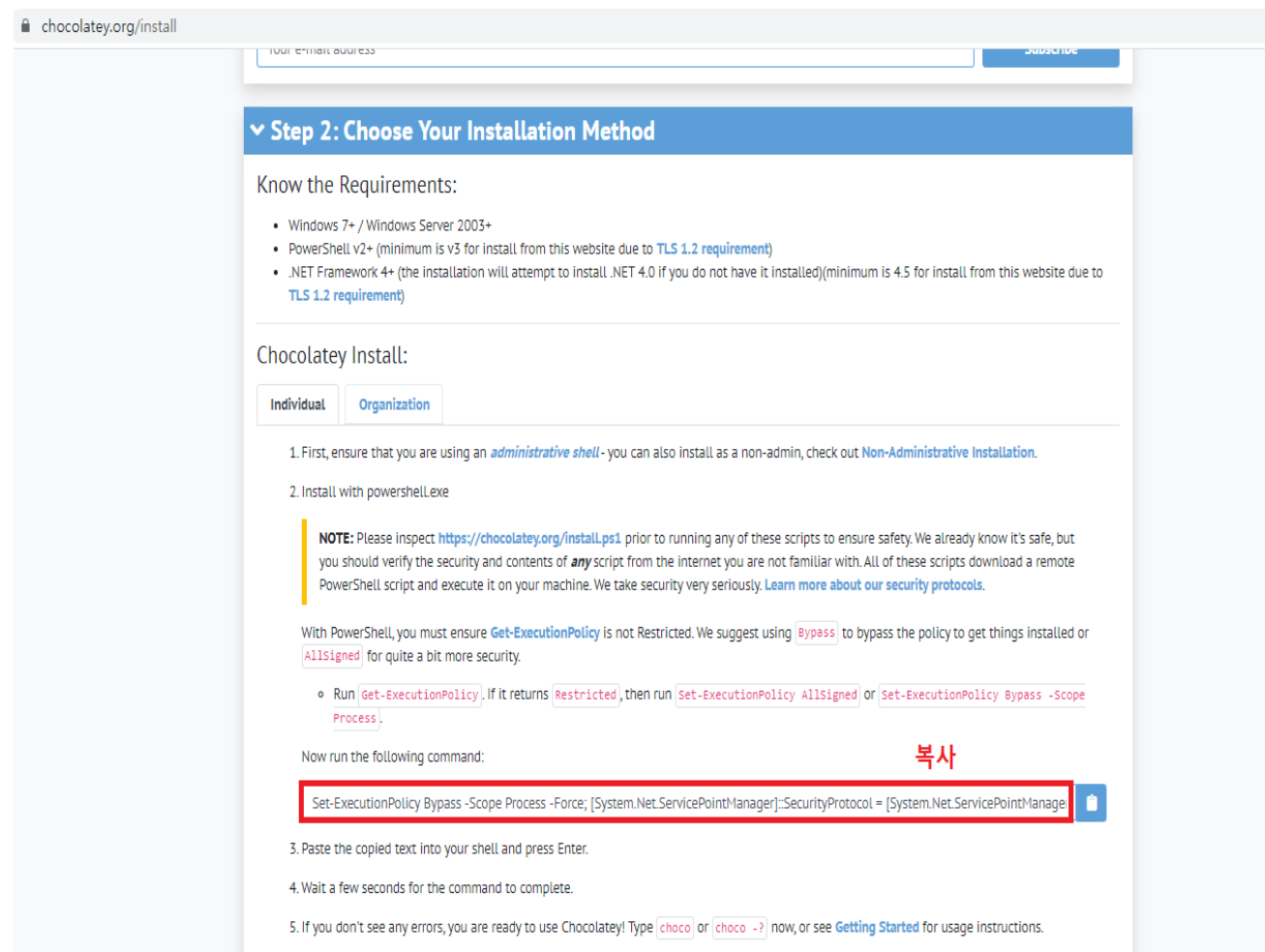
---

- Windows Terminal 설치
  - 마이크로소프트 스토어에서 Windows Terminal 프로그램 설치
  - 프로젝트를 진행하면서 여러개의 터미널을 사용하는데 이 프로그램을 사용하면 터미널을 관리하기 쉽다.



# 개발환경 구축

- choco 설치
  - <https://chocolatey.org/install> 사이트로 들어가서 설치 커맨드 복사
  - powershell을 관리자 모드로 실행 후 커맨드 붙여넣기
  - 설치 후 choco 명령어를 통해 설치 확인



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

새로운 크로스 플랫폼 PowerShell 사용 https://aka.ms/powershell

PS C:\Users\User> choco
Chocolatey v0.10.15
Please run 'choco -?' or 'choco <command> -?' for help menu.
```



# 개발환경 구축

---

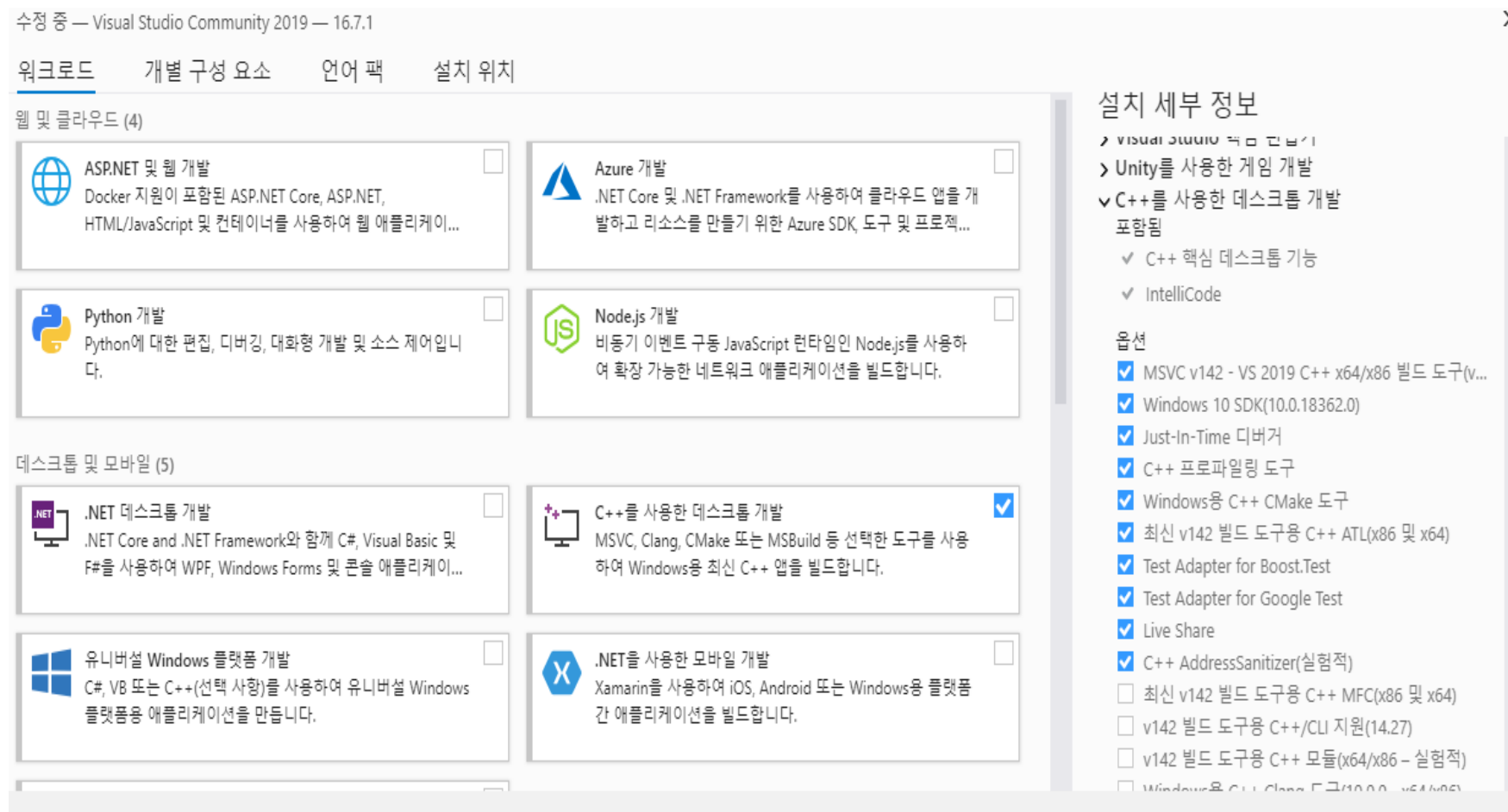
- python 3.7.5 버전 설치
  - choco 명령어를 이용해 파이썬 설치
    - `$ choco install -y python --version 3.7.5`
  - 파이썬 설치후 python 명령어를 이용해 설치 확인
  - 기존에 다른 버전의 파이썬 버전이 설치되어 있다면, 환경변수에서 다른 버전의 파이썬 경로를 삭제

```
C:\Users\user>python --version
Python 3.7.5
```

```
C:\Users\user>python
Python 3.7.5 (tags/v3.7.5:5c02a39a0b, Oct 15 2019, 00:11:34) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> |
```

# 개발환경 구축

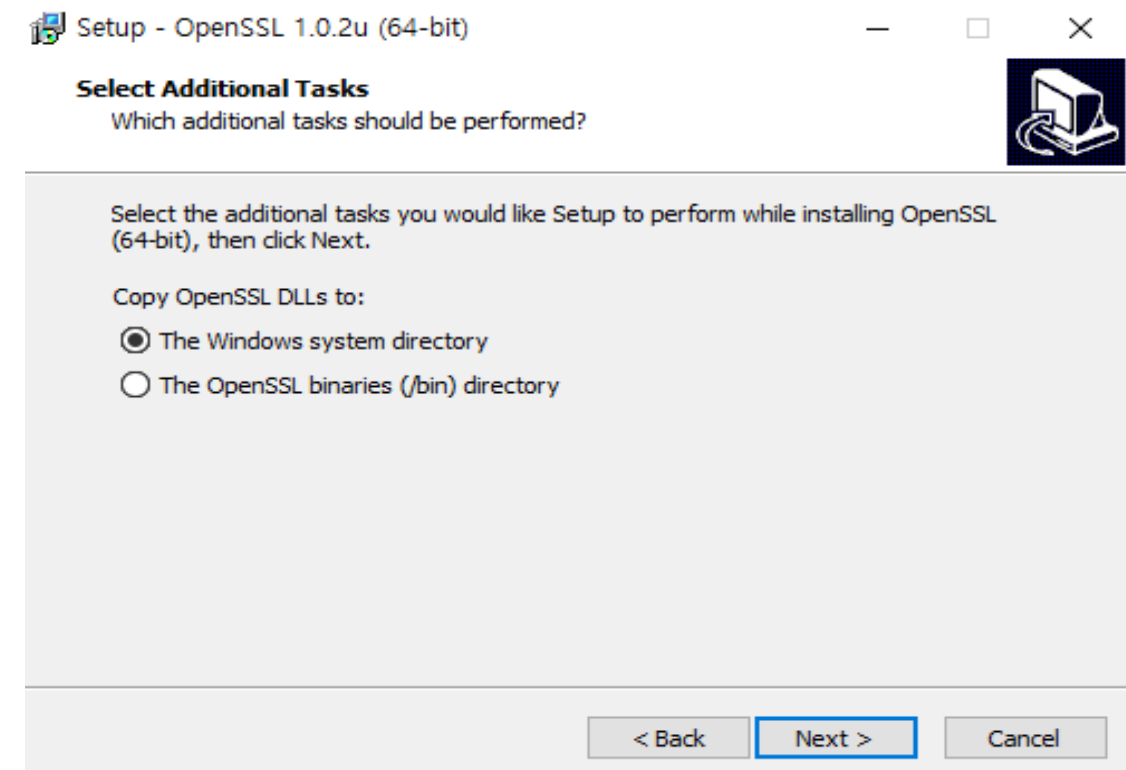
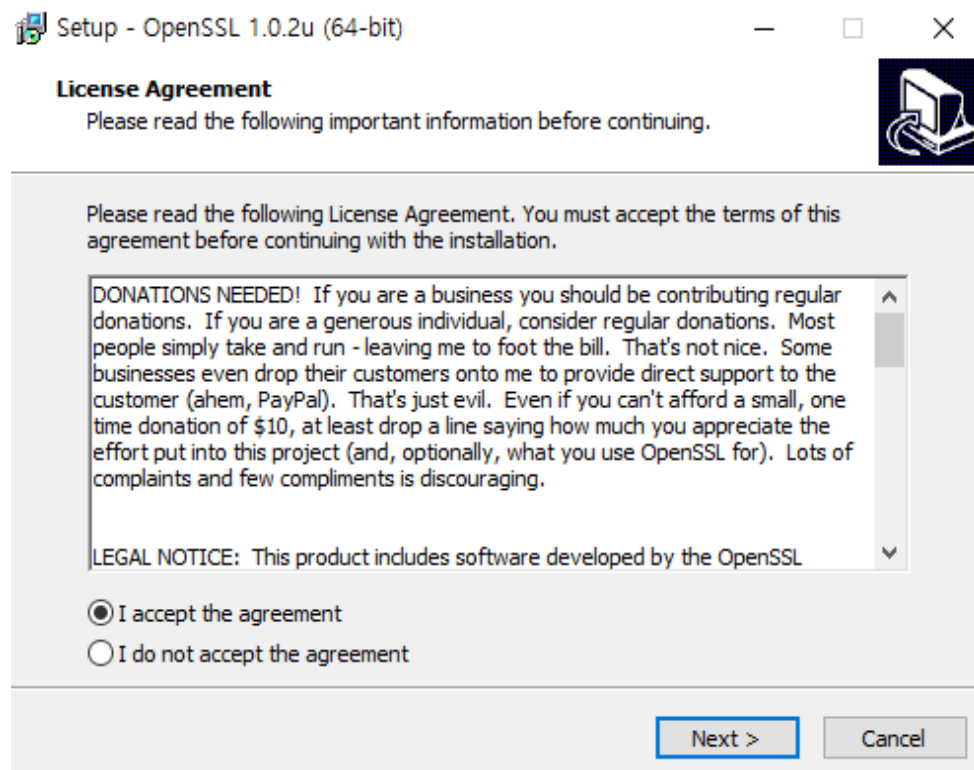
- **visual c++ redistributables 설치**
  - \$ choco install -y vcredist2013 vcredist140
- **visual studio community 2019 설치**
  - visual studio installer에서 C++를 사용한 데스크톱 개발 부분만 설치
  - 네이티브 도구 명령 프롬프트를 사용해 ros2 패키지를 빌드할 예정



# 개발환경 구축

- OPENSSL 설치

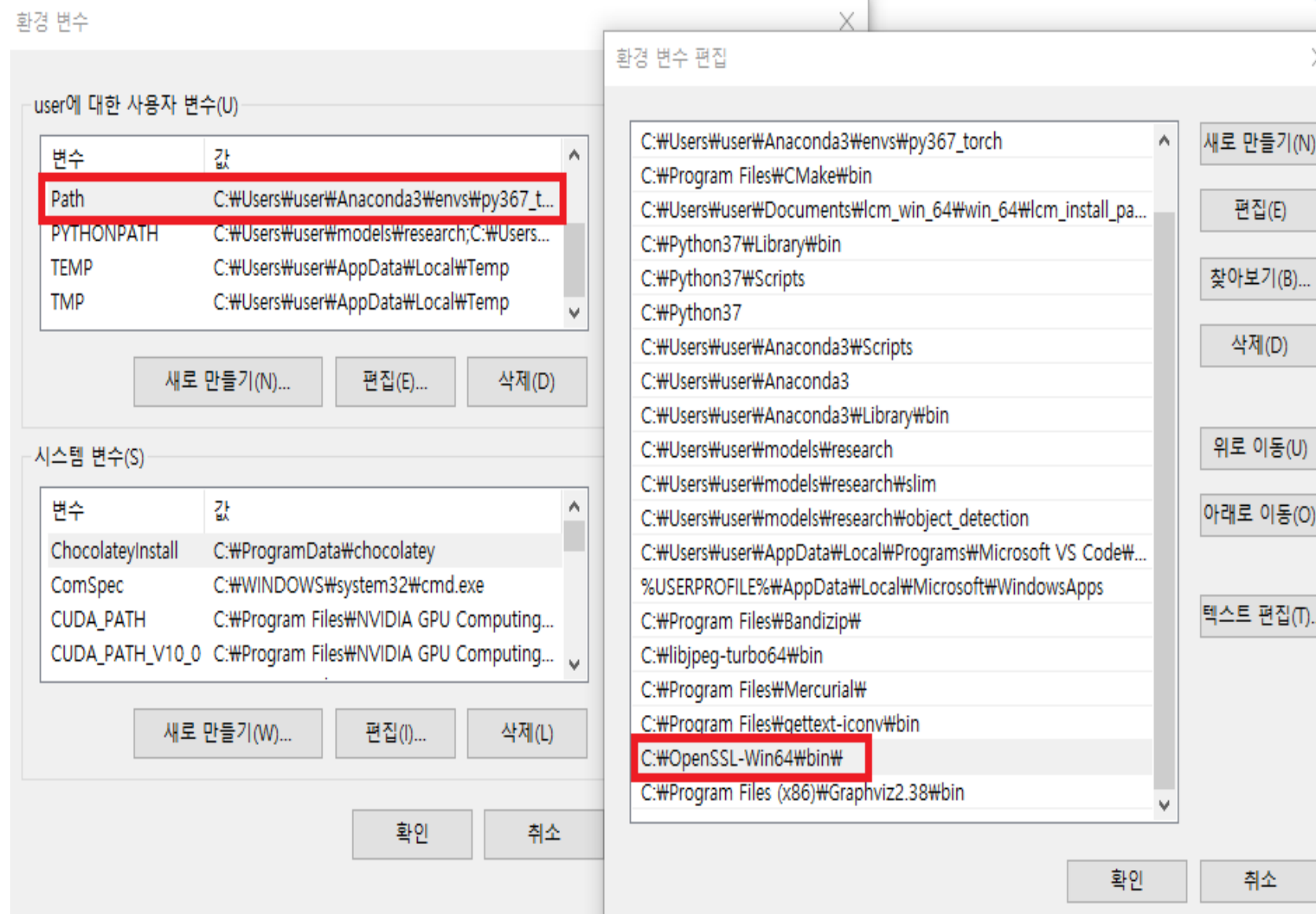
- OPENSSL 이란 : 네트워크를 통한 데이터 통신에 쓰이는 프로토콜인 TLS와 SSL의 오픈 소스 라이브러리. 이전 ROS1과 달리 보안성에 중점을 두기위해 차용됨.
- 제공한 설치 파일안에 Win64OpenSSL-1\_0\_2u.exe 파일을 이용해 설치
- 설치 경로는 C:\WOpenSSL-Win64 로 설정
- 설치 후 powershell에 아래 명령어 입력
  - `$ setx -m OPENSSL_CONF C:\WOpenSSL-Win64\bin\wopenssl.cfg`



# 개발환경 구축

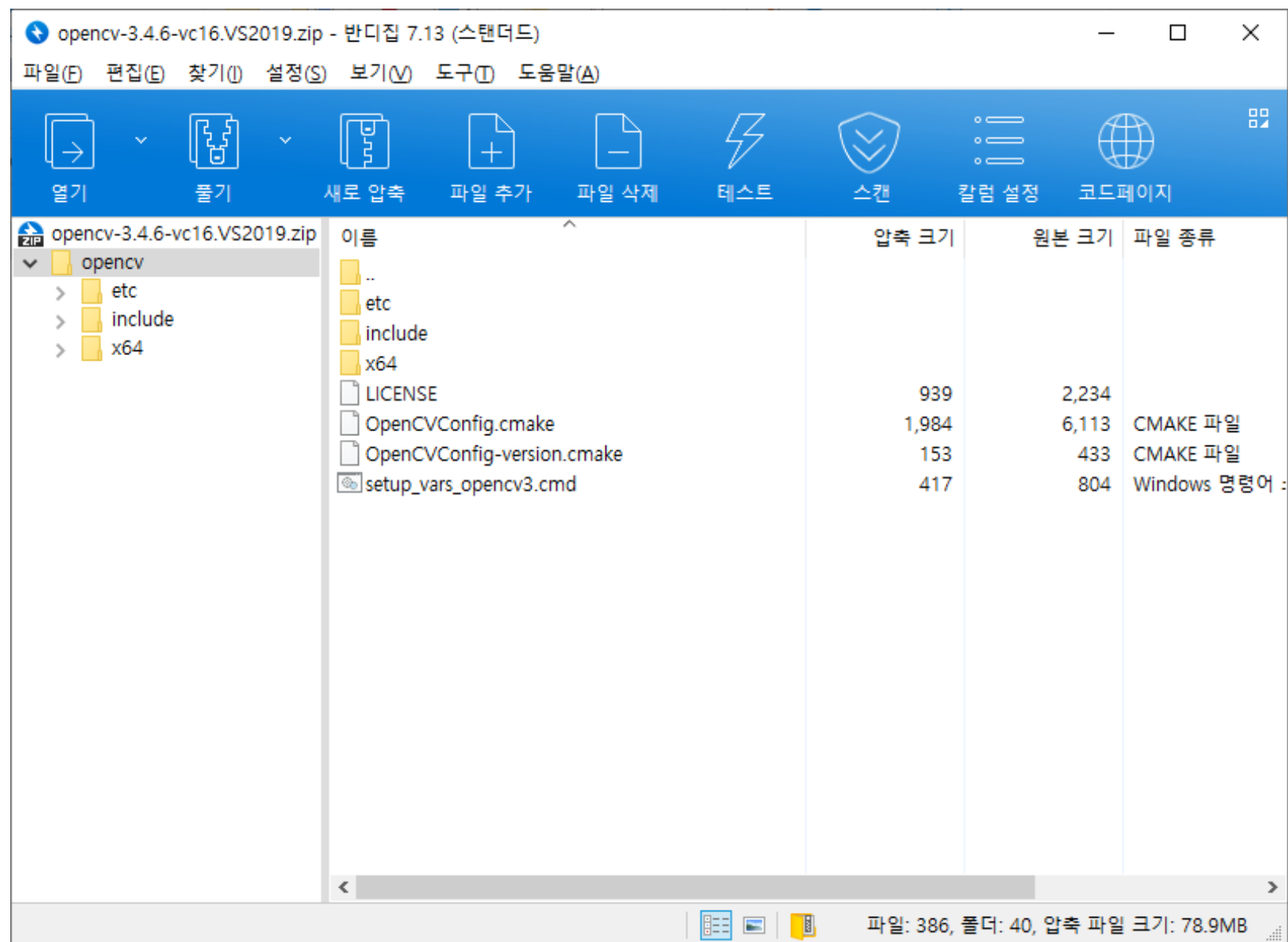
- OPENSSL 설치

- 내 PC의 오른쪽 버튼을 클릭 후 속성에서 [고급 시스템 설정] 클릭, 고급 탭에 [환경 변수] 클릭
- user 환경 변수의 Path 변수에 openssl 경로 추가
- C:\WOpenSSL-Win64\bin



# 개발환경 구축

- OPENCV 설치
  - 제공한 설치파일안에 opencv-3.4.6 버전을 C:\W 에 압축 해제
  - powershell 에 아래 명령어 입력
    - `$ setx -m OpenCV_DIR C:\W\opencv`

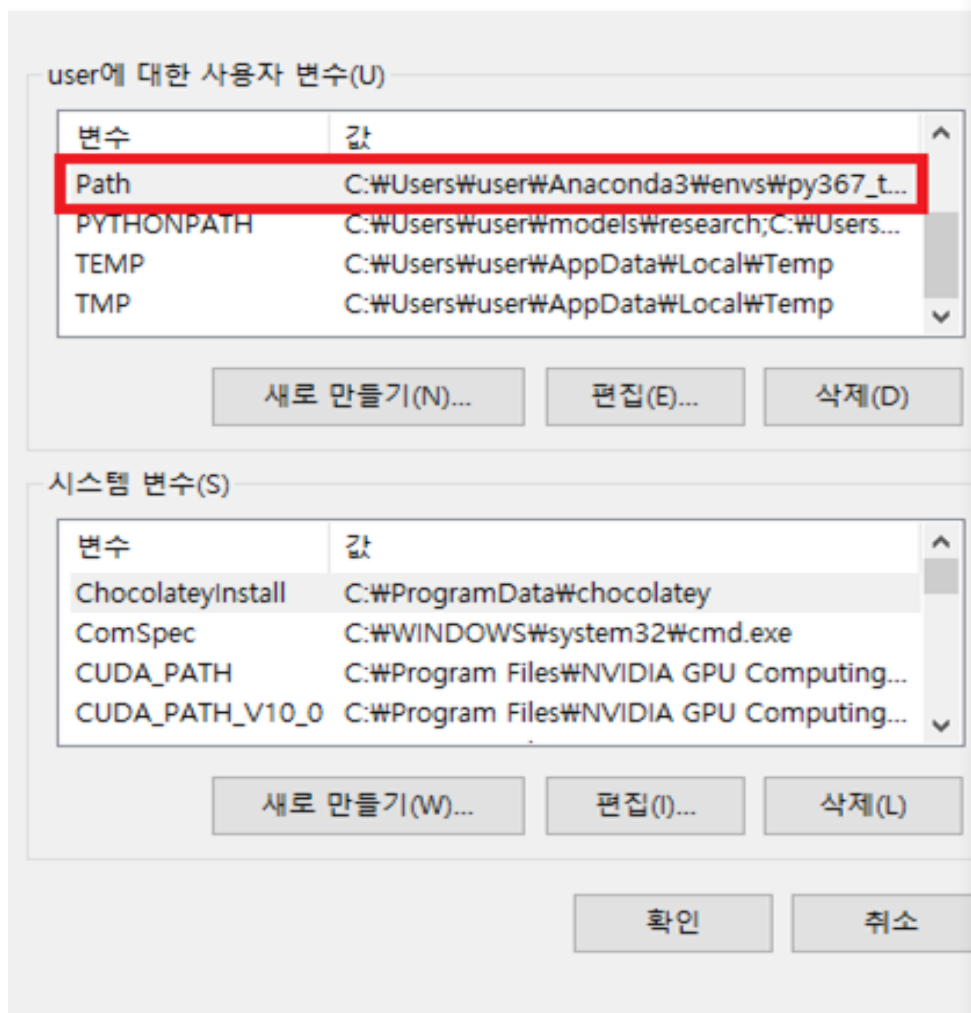


# 개발환경 구축

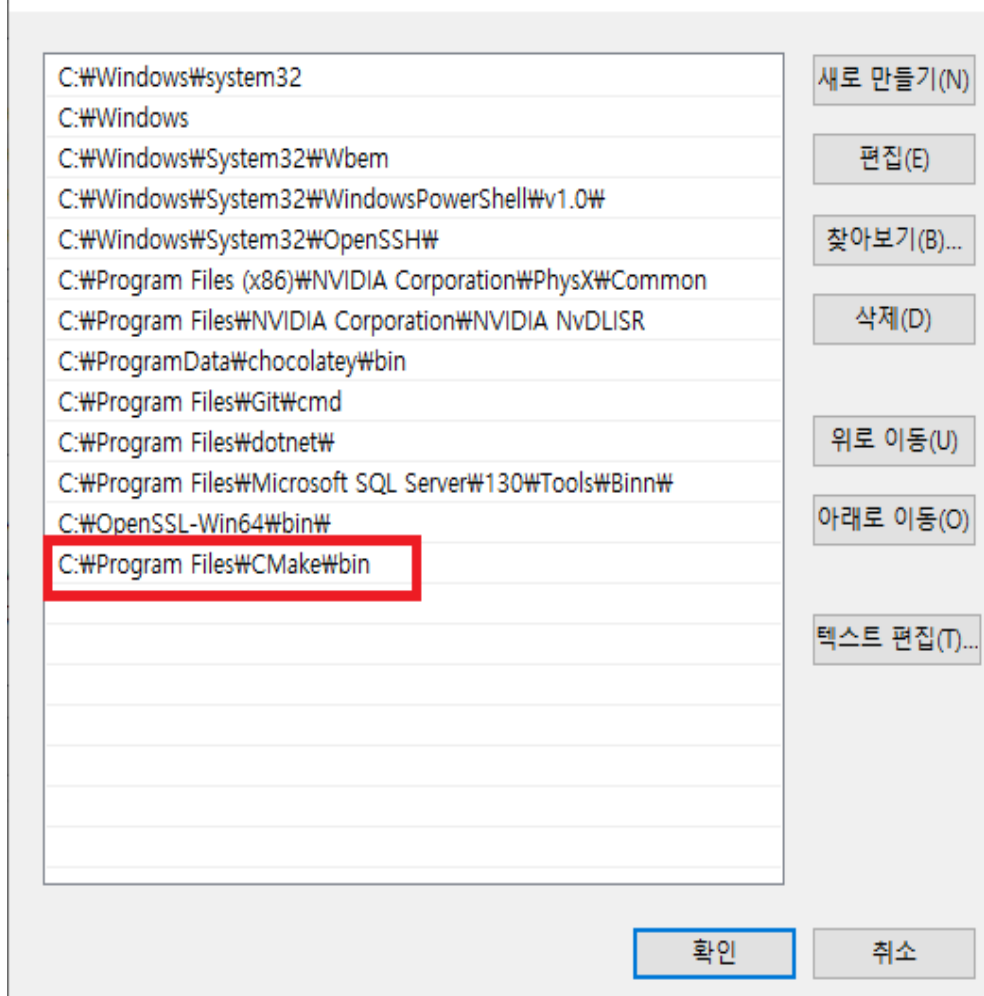
- CMAKE 설치

- 기존 디폴트로 ROS2에 구현된 C++ 노드들을 빌드 시키는데 필요.
- powershell에서 choco 명령어를 이용해 cmake 설치
  - `$ choco install -y cmake.install -version==3.19.3`
- 설치 후 user 환경변수의 Path에 CMAKE 경로 추가
- `C:\Program Files\CMake\bin`

환경 변수

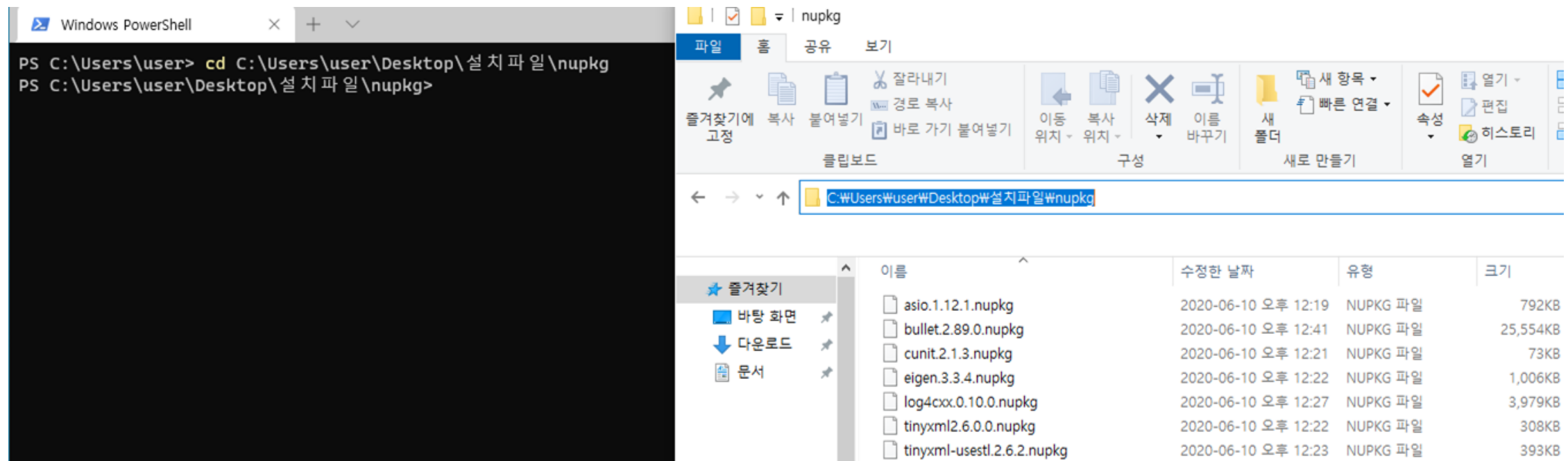


환경 변수 편집



# 개발환경 구축

- nupkg 설치
  - 터미널 경로를 설치파일안에 nupkg 폴더로 이동 후 choco 명령어를 이용해 설치
    - asio : network, low-level I/O programming를 위한
    - cunit : C 기반의 Unit Testing Framework
    - eigen : 매트릭스 연산에 필요한 라이브러리
    - tinyxml-usestl, tinyxml2 : C++ 기반 xml 파서
    - log4cxx : 프로그램 로그 기록 역할
    - bullet : ROS의 충돌, rigid dynamics 모델링 등에 사용하는 패키지
    - \$ choco install -y -s . asio cunit eigen tinyxml-usestl tinyxml2 log4cxx bullet



# 개발환경 구축

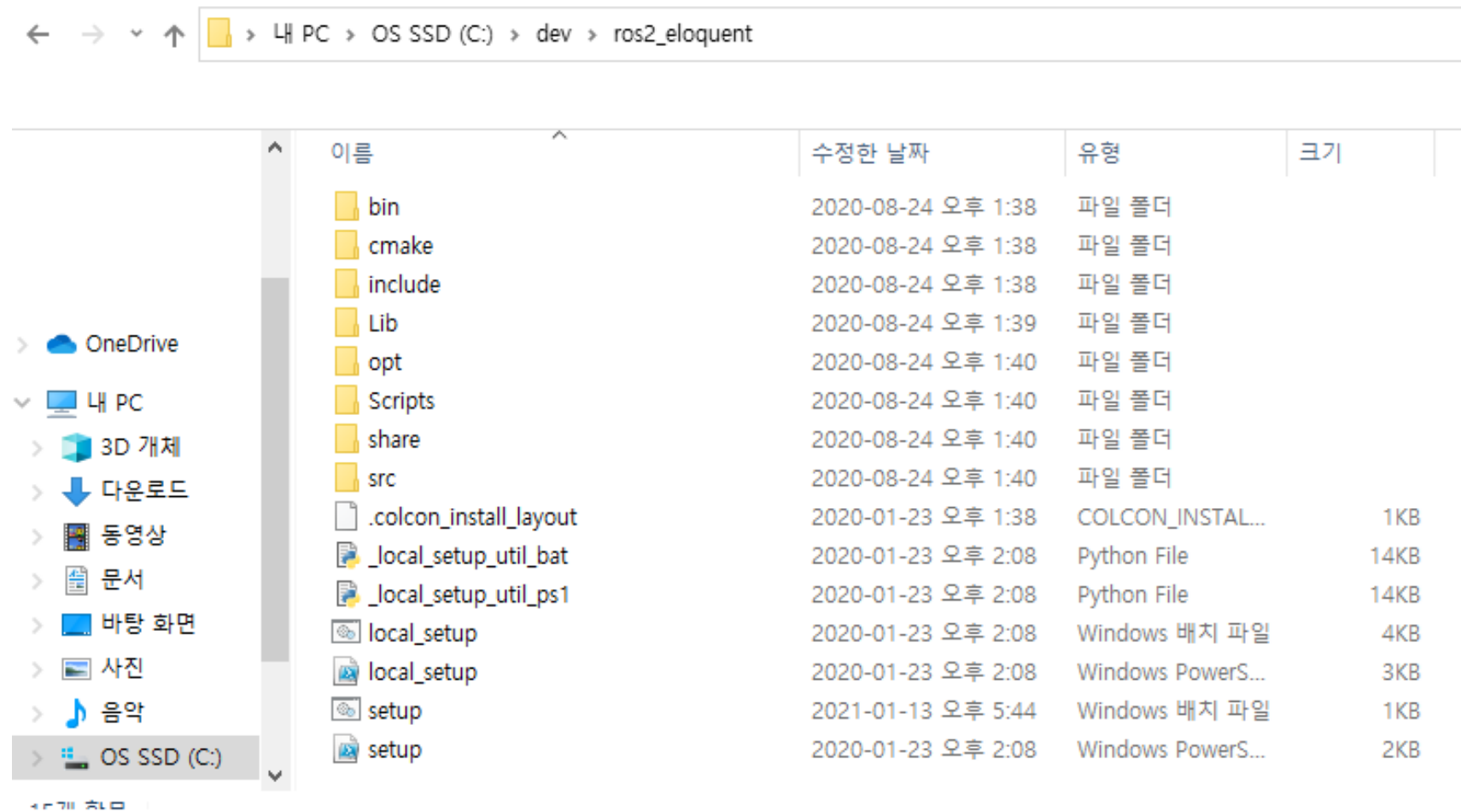
---

- 파이썬 종속패키지 설치
  - ros2를 사용하기 위한 파이썬 종속패키지 설치
  - 공식 홈페이지의 ros2 official install 절차 외에 종속패키지 설치 전 pip 최신화, powershell이 아닌 명령프롬프트 창을 관리자모드로 실행 후 입력
    - `$ pip install --upgrade pip`
  - 업데이트 후 종속 패키지 설치 (제공된 requirements.txt 가 있는 폴더에서 명령어 실행)
    - `$ python -m pip install -r requirements.txt`
  - Colcon build 를 위한 패키지를 설치
    - `$ pip install -U colcon-common-extensions`



# 개발환경 구축

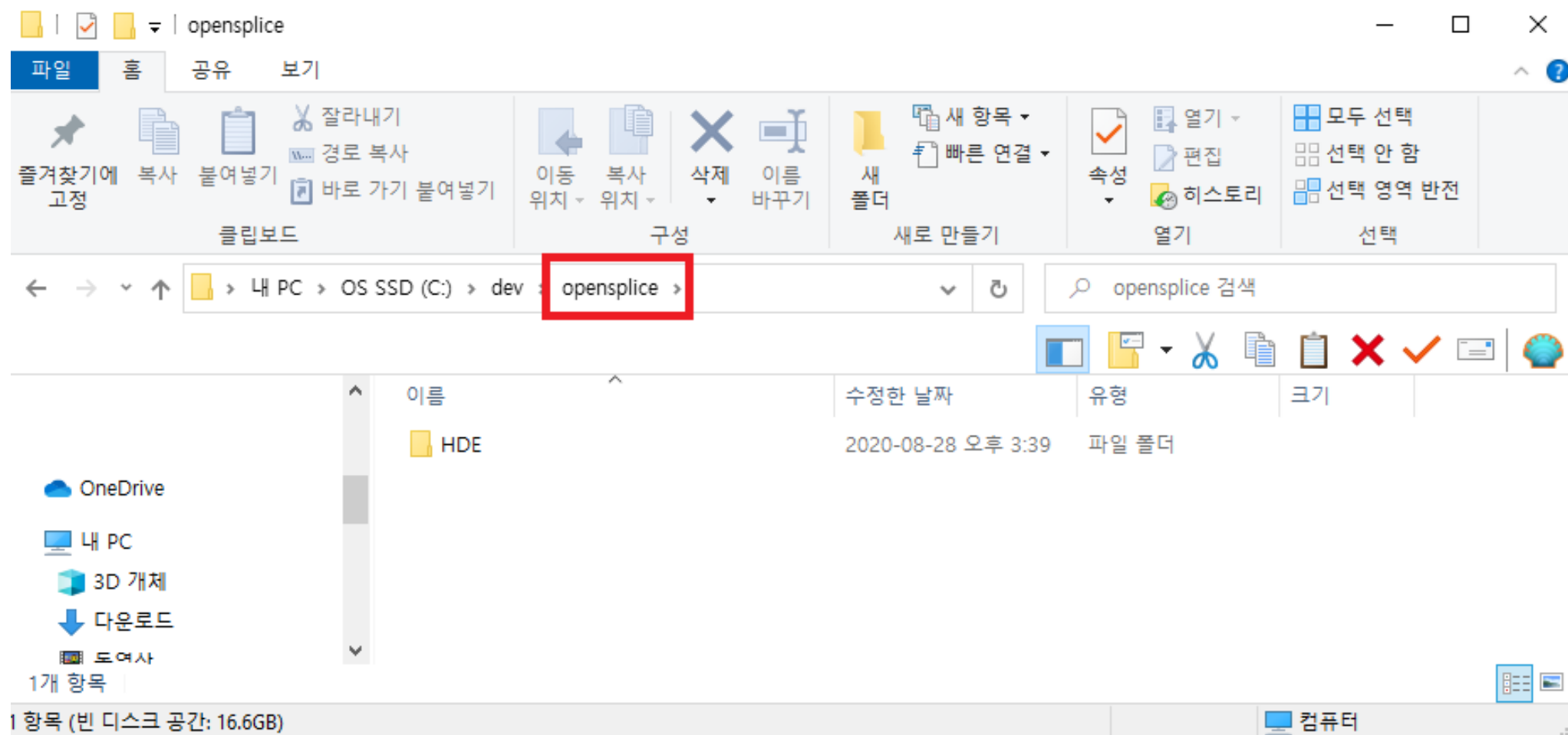
- ROS2 설치
  - c드라이브에 dev 폴더 생성 후
  - 제공한 설치파일 안에 ros2 압축 파일을 해제
  - 압축 해제된 'ros2\_eloquent'폴더가 dev 폴더 안에 위치되면 완료



# 개발환경 구축

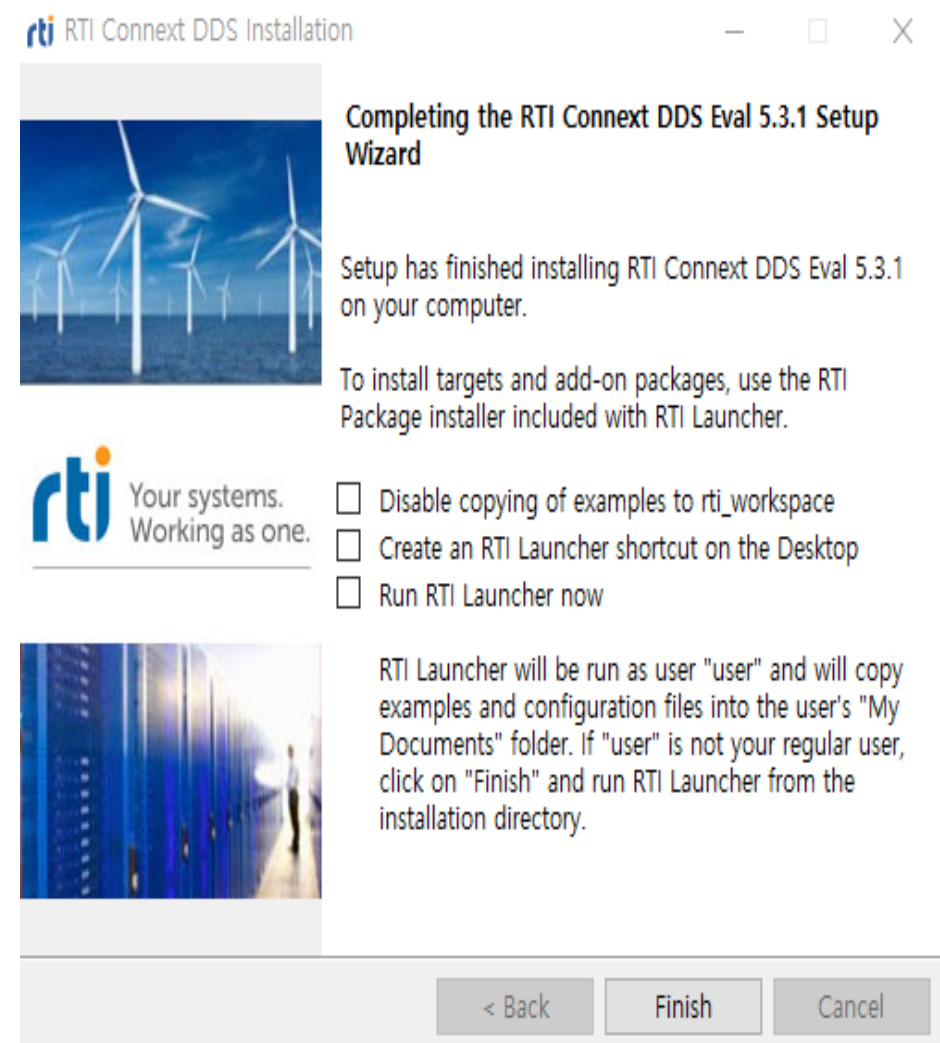
- OpenSplice 설치

- DDS (Data Distribution Service) : 국제 표준을 따르는 실시간 데이터 분배 미들웨어
- OpenSplice : Prismtech가 내놓은 DDS 제품
- C:\Wdev 폴더 안에 opensplice라는 폴더 생성
- 제공한 설치파일 안에 OpenSplice 압축 파일을 opensplice 폴더안에 압축 해제
- 압축해제하면 HDE 폴더가 생성되는 것을 확인



# 개발환경 구축

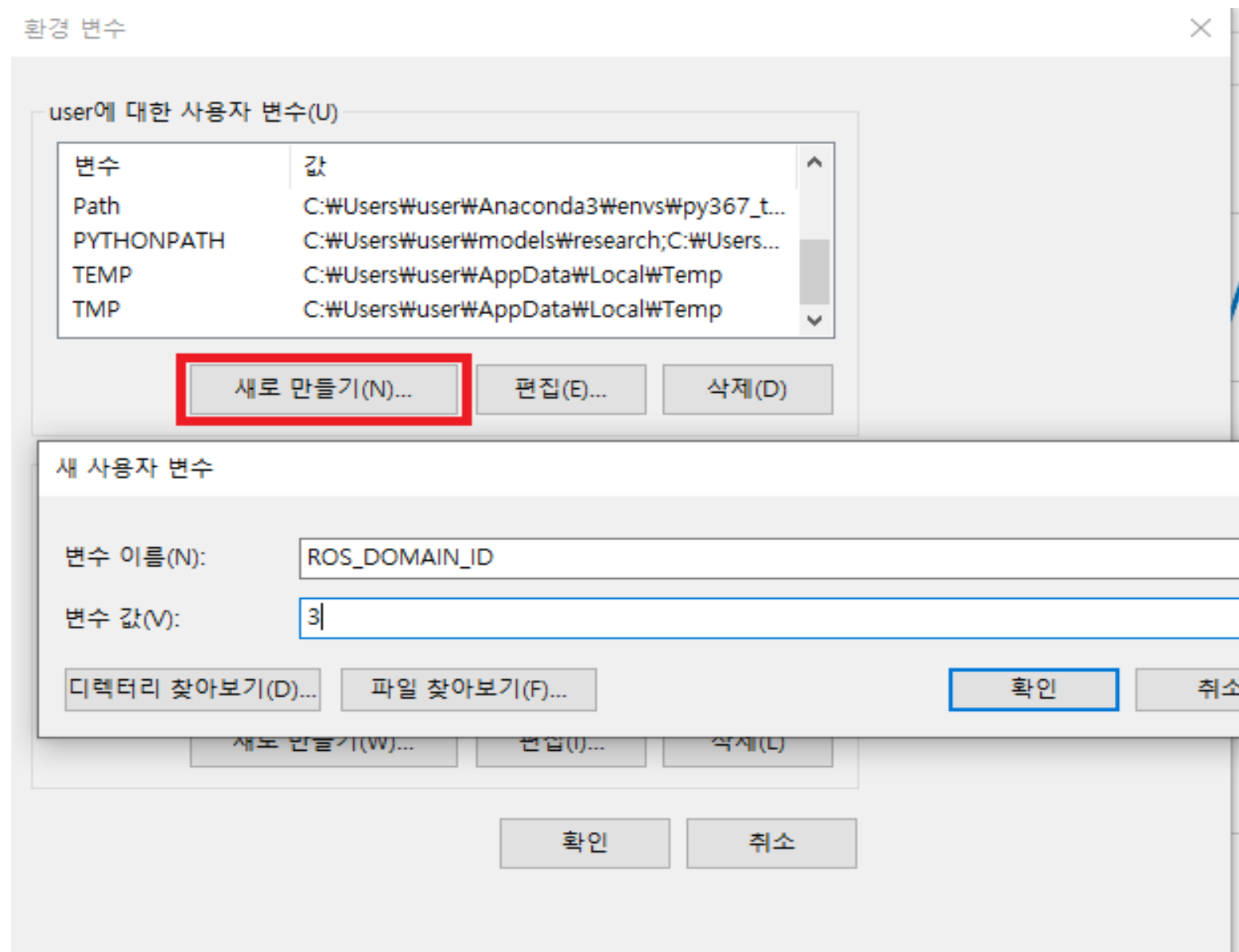
- RTI 설치
  - 제공한 설치파일 안에 rti\_connexdds.exe 실행 후 설치
  - 설정 변경 없이 계속 Next를 눌러서 설치



# 개발환경 구축

- ROS DOMAIN ID 설정

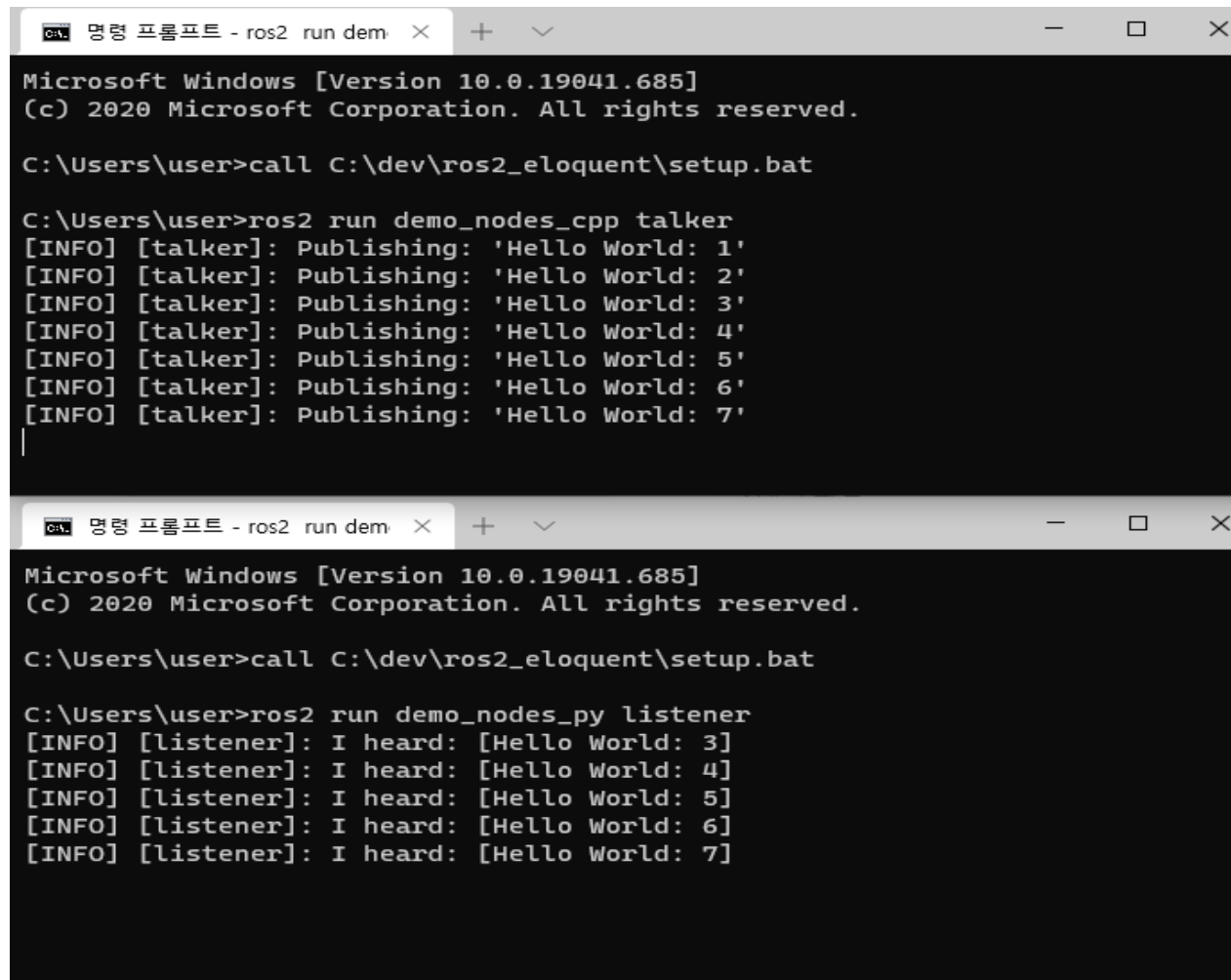
- 환경변수에서 ROS DOMAIN ID를 설정하지 않으면 모두 0으로 설정되어있기 때문에 같은 로컬 네트워크에 묶인 컴퓨터는 ROS2 메시지를 공유
- 각 팀별로 고유의 ID를 정해서 사용해야 다른 팀간의 메시지가 공유되는 문제 방지.
- 같은 네트워크에서 프로젝트를 진행하면 서로 다른 ID를 사용해야 한다.



# 개발환경 구축

---

- ROS2 설치 확인
  - 터미널 창에서 ROS2 기능을 사용하기 위해서는 배치파일을 읽어와야 한다.
  - 새로운 터미널을 사용할 때마다 해당 명령어를 꼭 입력해야한다.
    - `$ call C:\dev\ros2_elloquent\setup.bat`
  - 터미널창을 2개 열어 예제 코드를 실행
    - `$ ros2 run demo_nodes_cpp talker`
    - `$ ros2 run demo_nodes_py listener`



The image shows two terminal windows side-by-side, both titled '명령 프롬프트 - ros2 run dem'. The top window shows the execution of `ros2 run demo_nodes_cpp talker`, which outputs seven lines of 'Hello World' messages. The bottom window shows the execution of `ros2 run demo_nodes_py listener`, which outputs seven lines of 'I heard' messages corresponding to the talker's output.

```
Microsoft Windows [Version 10.0.19041.685]
(c) 2020 Microsoft Corporation. All rights reserved.

C:\Users\user>call C:\dev\ros2_elloquent\setup.bat

C:\Users\user>ros2 run demo_nodes_cpp talker
[INFO] [talker]: Publishing: 'Hello World: 1'
[INFO] [talker]: Publishing: 'Hello World: 2'
[INFO] [talker]: Publishing: 'Hello World: 3'
[INFO] [talker]: Publishing: 'Hello World: 4'
[INFO] [talker]: Publishing: 'Hello World: 5'
[INFO] [talker]: Publishing: 'Hello World: 6'
[INFO] [talker]: Publishing: 'Hello World: 7'

Microsoft Windows [Version 10.0.19041.685]
(c) 2020 Microsoft Corporation. All rights reserved.

C:\Users\user>call C:\dev\ros2_elloquent\setup.bat

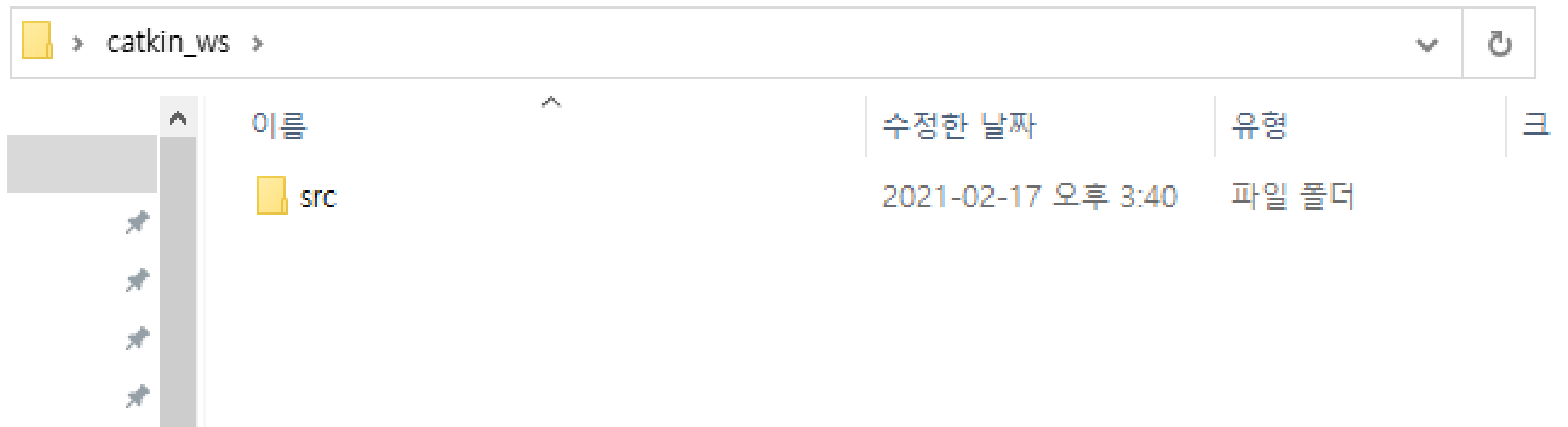
C:\Users\user>ros2 run demo_nodes_py listener
[INFO] [listener]: I heard: [Hello World: 3]
[INFO] [listener]: I heard: [Hello World: 4]
[INFO] [listener]: I heard: [Hello World: 5]
[INFO] [listener]: I heard: [Hello World: 6]
[INFO] [listener]: I heard: [Hello World: 7]
```

## **2. 작업 환경 구축**

# 작업 환경 구축

---

- 작업 폴더 Workspace 만들기
  - catkin\_ws는 ROS2를 이용한 코드를 빌드 하게 될 작업 공간이다
  - 바탕 화면에 catkin\_ws 라는 폴더를 만든다
  - 폴더 안에 src 폴더를 만든다



# 작업 환경 구축

---

- 소스코드 복사하기
  - 제공한 ros2\_smart\_home.zip 파일을 catkin\_ws / src 폴더 안에 압축을 풀어 준다.
  - ros2\_smart\_home 폴더 안에 ssafy\_bridge, ssafy\_msgs, sub1, sub2, sub3 패키지를 확인한다.
  - ssafy\_bridge, ssafy\_msgs는 시뮬레이터와 통신하기 위한 패키지이다.

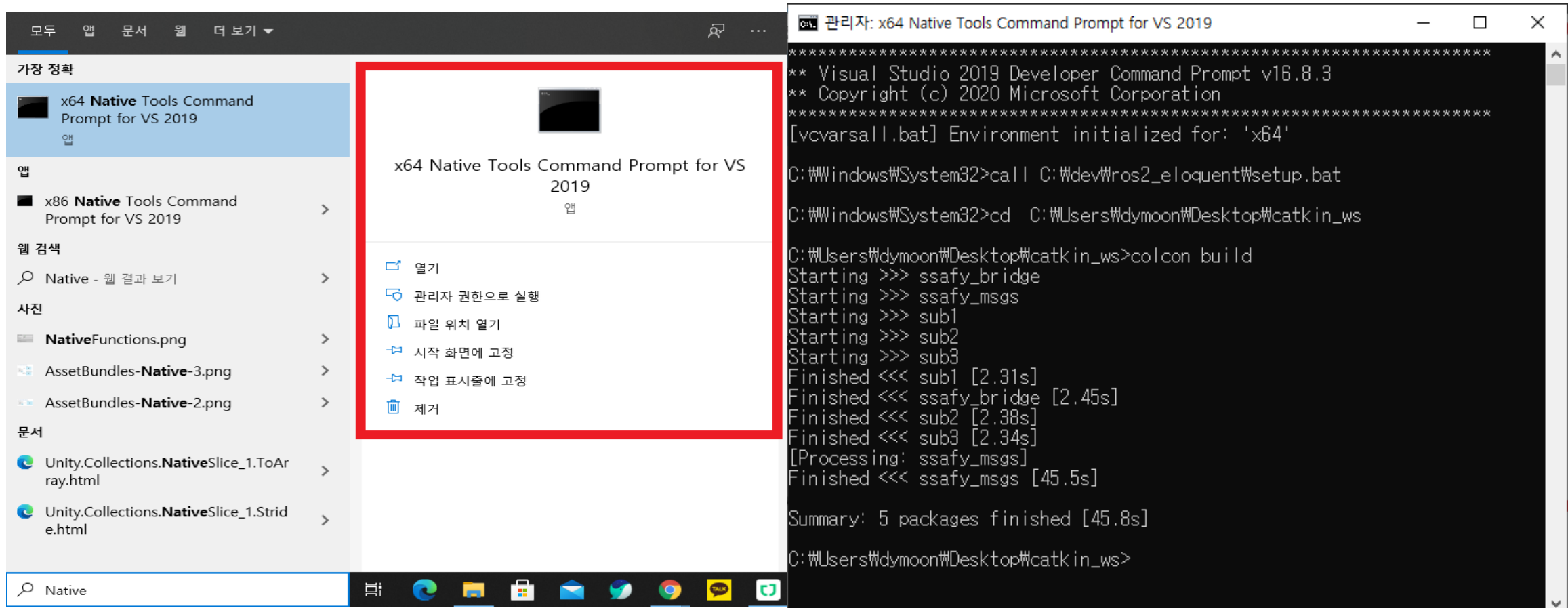
catkin_ws > src > ros2_smart_home					↕	↺
	이름	수정한 날짜	유형	크기		
	ssafy_bridge	2021-02-17 오후 3:41	파일 폴더			
	ssafy_msgs	2021-02-17 오후 3:41	파일 폴더			
	sub1	2021-02-17 오후 3:41	파일 폴더			
	sub2	2021-02-17 오후 3:41	파일 폴더			
	sub3	2021-02-17 오후 3:41	파일 폴더			



# 작업 환경 구축

## • 코드 빌드하기

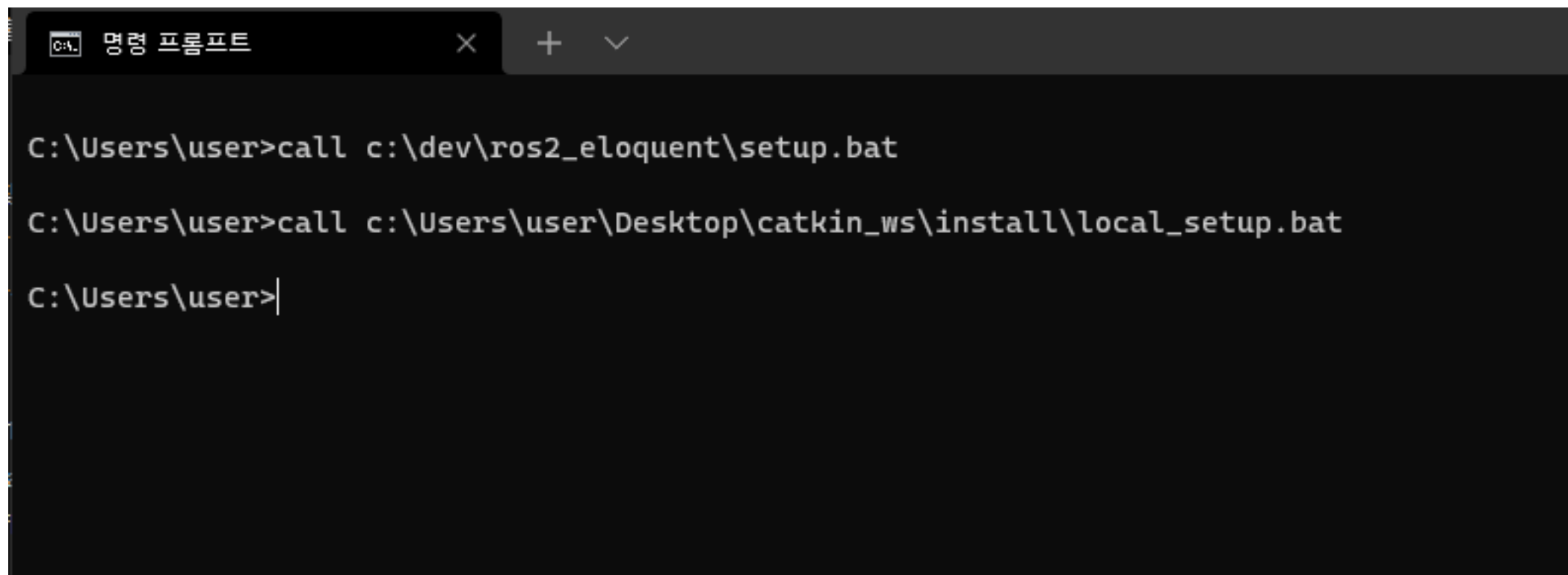
- 빌드는 native Tools Command Prompt for VS 201x 에서 한다.
- 관리자 권한으로 열고 아래와 같이 입력한다
  - `call C:\dev\ros2_eloquent\setup.bat` (ROS2 명령어 실행을 위한 배치 파일 실행)
  - `cd C:\Users\Wuser\Desktop\catkin_ws` (터미널 창의 경로를 catkin\_ws로 이동)
  - `colcon build`
- 아래와 같이 src 폴더에 넣은 패키지가 빌드 되는 것을 확인 가능하다
- 특정 패키지만 빌드하려면 아래와 같이 입력하면 된다
  - `colcon build --packages-select [pkg_name]`
  - EX) `colcon build --packages-select sub1`



# 작업 환경 구축

---

- ssafy\_bridge 실행
  - 터미널에서 ROS2 의 기능과 Workspace 안에 있는 패키지를 사용하기 위해 배치 파일을 실행 해주어야 한다
  - 아래 명령어를 입력한다.
    - `call C:\dev\ros2_elloquent\setup.bat`
    - `call C:\Users\user\Desktop\catkin_ws\install\local_setup.bat`
  - 새로운 터미널 실행 시 항상 위의 명령어를 입력해야 한다



```
C:\Users\user>call c:\dev\ros2_elloquent\setup.bat

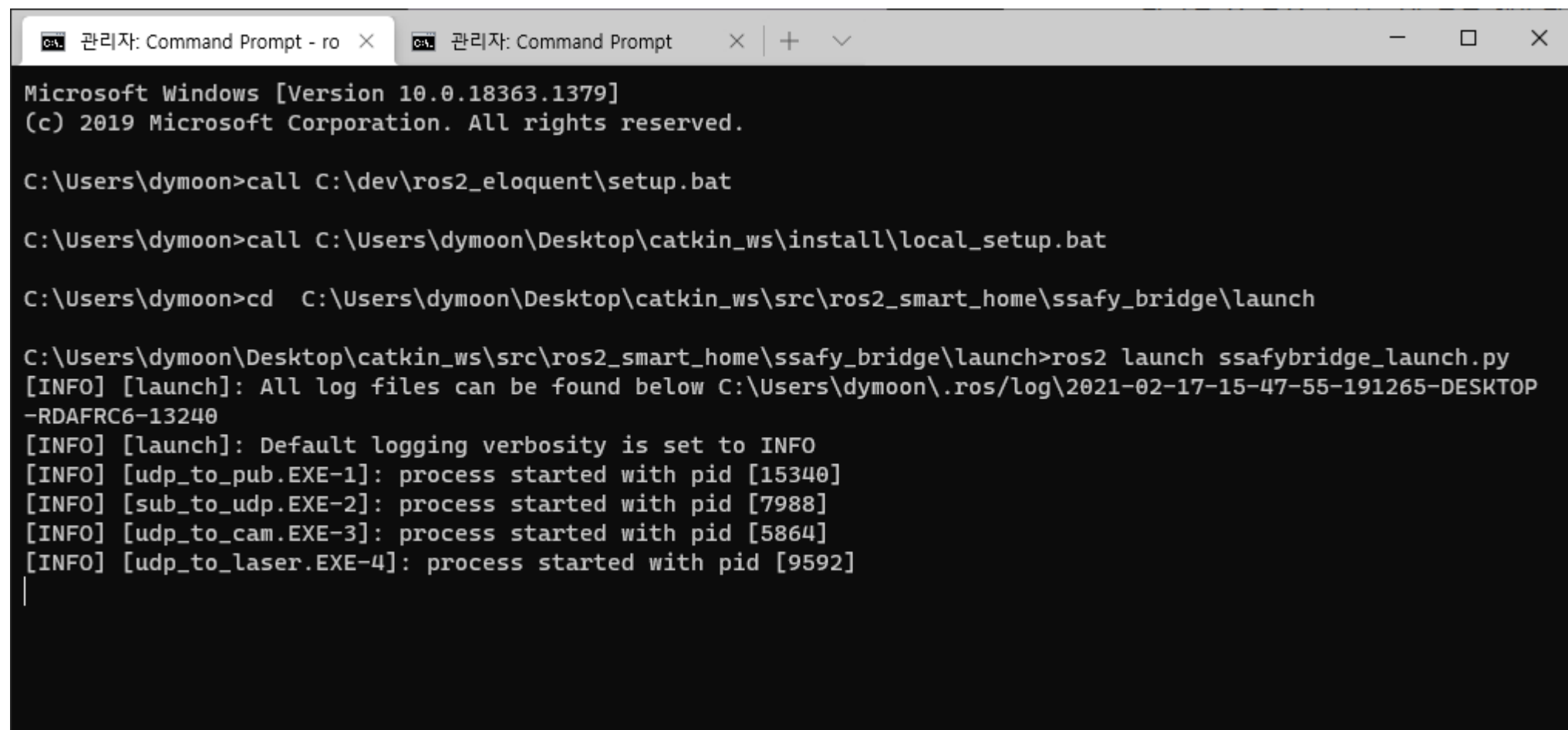
C:\Users\user>call c:\Users\user\Desktop\catkin_ws\install\local_setup.bat

C:\Users\user>
```

# 작업 환경 구축

---

- ssafy\_bridge 실행
  - ssafy\_bridge 실행 명령어
    - `cd C:\Users\Wuser\Desktop\catkin_ws\src\ros2_smart_home\ssafy_bridge\launch`
    - `ros2 launch ssafybridge_launch.py`



```
관리자: Command Prompt - ro × 관리자: Command Prompt × + - □ ×
Microsoft Windows [Version 10.0.18363.1379]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\dymoon>call C:\dev\ros2_elloquent\setup.bat

C:\Users\dymoon>call C:\Users\dymoon\Desktop\catkin_ws\install\local_setup.bat

C:\Users\dymoon>cd C:\Users\dymoon\Desktop\catkin_ws\src\ros2_smart_home\ssafy_bridge\launch

C:\Users\dymoon\Desktop\catkin_ws\src\ros2_smart_home\ssafy_bridge\launch>ros2 launch ssafybridge_launch.py
[INFO] [launch]: All log files can be found below C:\Users\dymoon\.ros\log\2021-02-17-15-47-55-191265-DESKTOP-RDAFRC6-13240
[INFO] [launch]: Default logging verbosity is set to INFO
[INFO] [udp_to_pub.EXE-1]: process started with pid [15340]
[INFO] [sub_to_udp.EXE-2]: process started with pid [7988]
[INFO] [udp_to_cam.EXE-3]: process started with pid [5864]
[INFO] [udp_to_laser.EXE-4]: process started with pid [9592]
```

# 작업 환경 구축

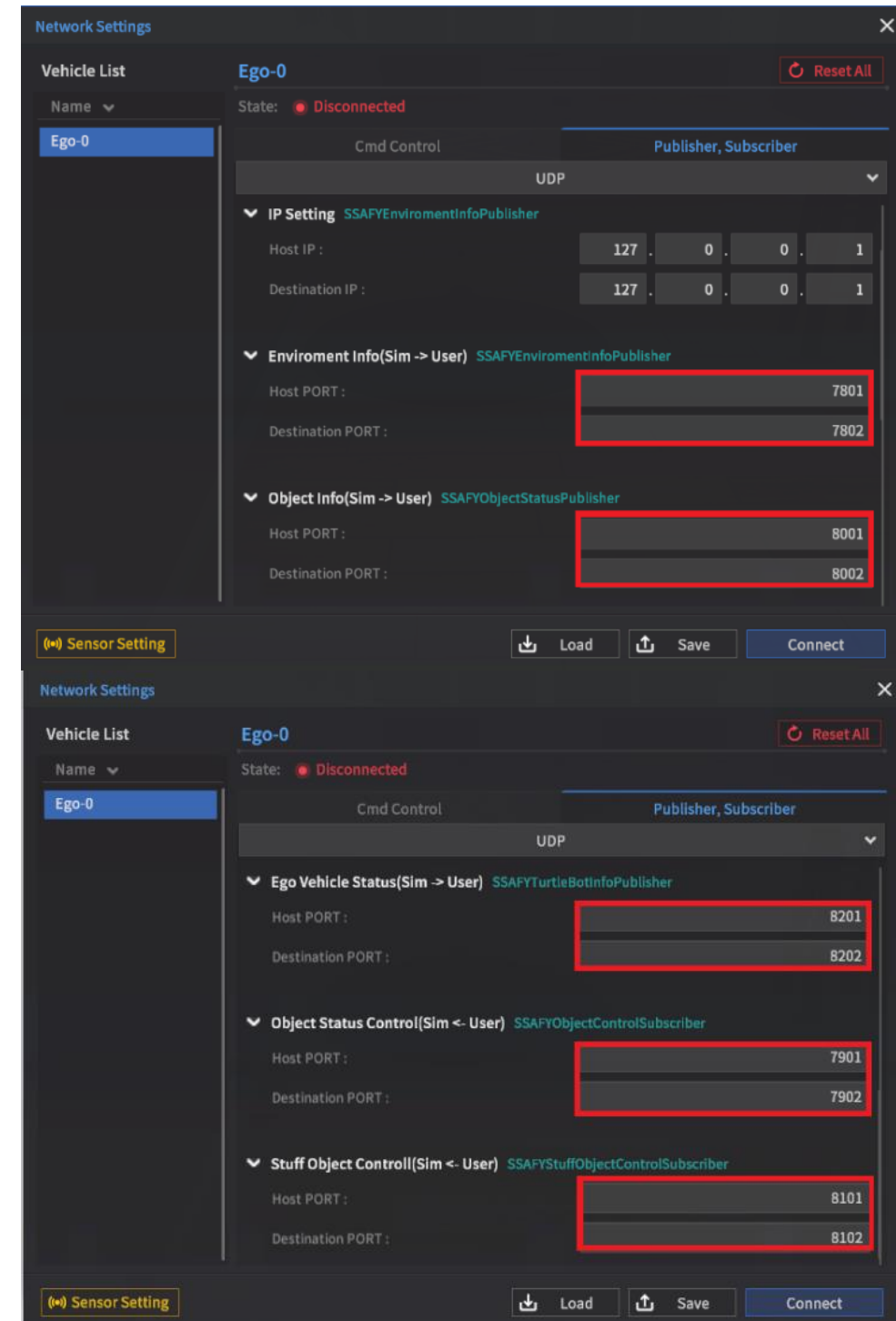
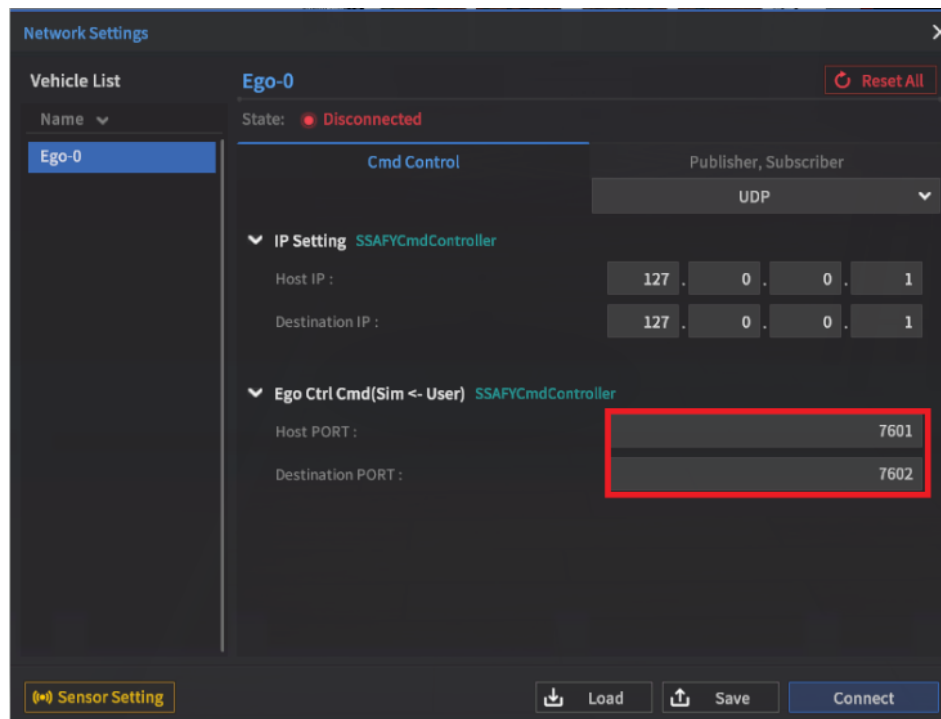
---

- 네트워크 세팅
  - 좌측 상단에 마우스 이동시 메뉴 선택 창이 나온다
  - **Network - Network Settings** 클릭
  - Network Settings 창이 나온다



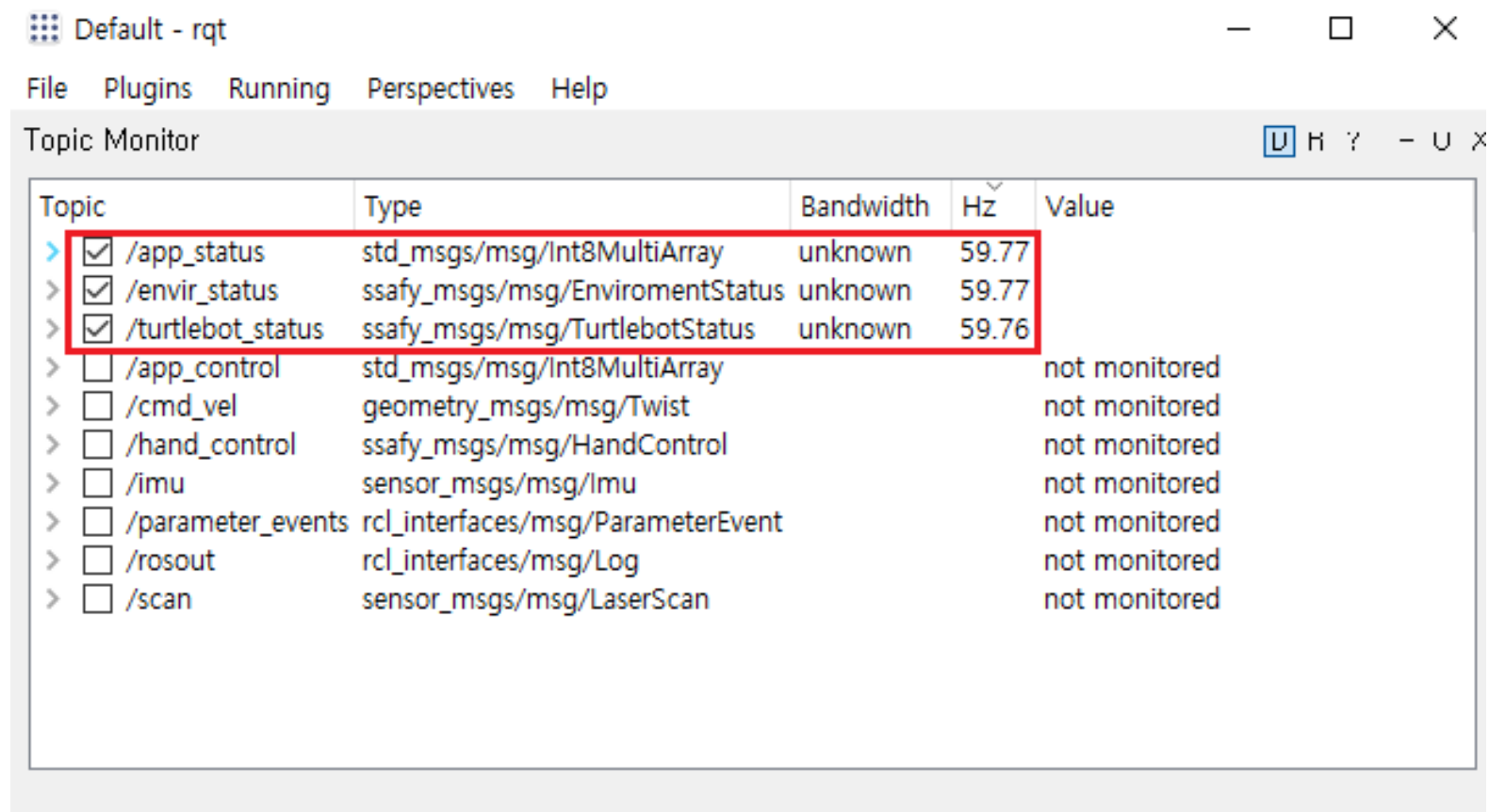
# 작업 환경 구축

- 네트워크 세팅
  - 네트워크 설정 포트 및 아이피 확인 후 Connect 클릭
- IP Setting
- Host IP : 127.0.0.1 / Destination IP : 127.0.0.1
- -----
- cmd controllersub : 7601 / 7602
- enviromentpub : 7801 / 7802
- objectontrolsub : 7901 / 7902
- objectstatepub : 8001 / 8002
- stuffobjectcontrolsub : 8101 / 8102
- turtlebotinfopub : 8201 / 8202



# 작업 환경 구축

- ssafy\_bridge 실행
  - ssafy\_bridge를 통해 나오는 데이터를 rqt 토픽모니터를 통해 확인 할 수 있다
  - 아래와 같이 입력하여 rqt 를 실행 한다
    - call C:\dev\ros2\eloquent\setup.bat
    - call C:\Users\user\Desktop\catkin\_ws\install\local\_setup.bat
    - rqt





# 작업 환경 구축

- ssafy\_bridge 실행 오류 해결
  - ssafy\_bridge 를 종료 후 재시작을 반복하면 다음과 같은 오류가 발생 한다
  - 작업 관리자 - python 클릭 - 작업 끝내기 - 재 시작 을 통해 해결 가능하다

```
C:\Users\dymoon\Desktop\catkin_ws\src\ros2_smart_home\ssafy_bridge\launch>ros2 launch ssafybridge_launch.py
[INFO] [launch]: All log files can be found below C:\Users\dymoon\.ros\log\2021-02-09-14-40-18-123919-DESKTOP-RDAFRC6-12628
[INFO] [launch]: Default logging verbosity is set to INFO
[INFO] [udp_to_pub.EXE-1]: process started with pid [3924]
[INFO] [sub_to_udp.EXE-2]: process started with pid [12044]
[INFO] [udp_to_cam.EXE-3]: process started with pid [6040]
[INFO] [udp_to_laser.EXE-4]: process started with pid [11592]
[udp_to_cam.EXE-3] Traceback (most recent call last):
[udp_to_cam.EXE-3]   File "C:\Users\dymoon\Desktop\catkin_ws\install\ssafy_bridge\lib\ssafy_bridge\udp_to_cam-script.py", line 33, in <module>
[udp_to_cam.EXE-3]     sys.exit(load_entry_point('ssafy-bridge==0.0.0', 'console_scripts', 'udp_to_cam')())
[udp_to_cam.EXE-3]   File "C:\Users\dymoon\Desktop\catkin_ws\install\ssafy_bridge\lib\site-packages\ssafy_bridge\udp_to_cam.py", line 65, in main
[udp_to_cam.EXE-3]     image_parser = IMGPublisher()
[udp_to_cam.EXE-3]   File "C:\Users\dymoon\Desktop\catkin_ws\install\ssafy_bridge\lib\site-packages\ssafy_bridge\udp_to_cam.py", line 34, in __init__
[udp_to_cam.EXE-3]     self.udp_parser = UDP_CAM_Parser(ip=params_cam_0["localIP"], port=params_cam_0["localPort"], params_cam=params_cam_0)
[udp_to_cam.EXE-3]   File "C:\Users\dymoon\Desktop\catkin_ws\install\ssafy_bridge\lib\site-packages\ssafy_bridge\utils.py", line 15, in __init__
[udp_to_cam.EXE-3]     self.sock.bind(recv_address)
[udp_to_cam.EXE-3] OSError: [WinError 10048] ?? ??? ???(????????/?????? ???/???)?? ????? ????? ?? ??????
[udp_to_laser.EXE-4] Traceback (most recent call last):
[udp_to_laser.EXE-4]   File "C:\Users\dymoon\Desktop\catkin_ws\install\ssafy_bridge\lib\ssafy_bridge\udp_to_laser-script.py", line 33, in <module>
[udp_to_laser.EXE-4]     sys.exit(load_entry_point('ssafy-bridge==0.0.0', 'console_scripts', 'udp_to_laser')())
[udp_to_laser.EXE-4]   File "C:\Users\dymoon\Desktop\catkin_ws\install\ssafy_bridge\lib\site-packages\ssafy_bridge\udp_to_laser.py", line 122, in main
[udp_to_laser.EXE-4]     pc_parser = PCPublisher()
[udp_to_laser.EXE-4]   File "C:\Users\dymoon\Desktop\catkin_ws\install\ssafy_bridge\lib\site-packages\ssafy_bridge\udp_to_laser.py", line 27, in __init__
[udp_to_laser.EXE-4]     self.udp_parser = UDP_LIDAR_Parser(ip=params_lidar["localIP"], port=params_lidar["localPort"], params_lidar=params_lidar)
[udp_to_laser.EXE-4]   File "C:\Users\dymoon\Desktop\catkin_ws\install\ssafy_bridge\lib\site-packages\ssafy_bridge\utils.py", line 95, in __init__
[udp_to_laser.EXE-4]     self.sock.bind(recv_address)
[udp_to_laser.EXE-4] OSError: [WinError 10048] ?? ??? ???(????????/?????? ???/???)?? ????? ????? ?? ??????
[ERROR] [udp_to_cam.EXE-3]: process has died [pid 6040, exit code 1, cmd 'C:\Users\dymoon\Desktop\catkin_ws\install\ssafy_bridge\lib\ssafy_bridge\udp_to_cam.
[ERROR] [udp_to_laser.EXE-4]: process has died [pid 11592, exit code 1, cmd 'C:\Users\dymoon\Desktop\catkin_ws\install\ssafy_bridge\lib\ssafy_bridge\udp_to_l
```

작업 관리자

파일(F) 옵션(O) 보기(V)

프로세스	성능	앱 기록	시작프로그램	사용자	세부 정보	서비스
이름	상태	5% CPU	28% 메모리	0% 디스크	0% 네트워크	
Microsoft Windows Search Fil...		0%	1.3MB	0MB/s	0Mbps	
Microsoft Windows Search Pro...		0%	2.8MB	0MB/s	0Mbps	
> Microsoft Windows Search 인...		0%	81.9MB	0MB/s	0Mbps	
NVIDIA Container		0%	29.7MB	0MB/s	0Mbps	
> NVIDIA Container		0%	4.7MB	0MB/s	0Mbps	
oCam Background Task(32비트)		0%	0.3MB	0MB/s	0Mbps	
> Parsec		0%	1.7MB	0MB/s	0Mbps	
Python		0%	30.1MB	0MB/s	0Mbps	
Python		0%	28.6MB	0MB/s	0Mbps	
> Runtime Broker		0%	3.7MB	0MB/s	0Mbps	
> Runtime Broker		0%	12.7MB	0MB/s	0Mbps	
> Runtime Broker		0%	5.1MB	0MB/s	0Mbps	
> Runtime Broker		0%	5.8MB	0MB/s	0Mbps	
> Runtime Broker		0%	4.0MB	0MB/s	0Mbps	

간단히(D) 작업 끝내기(E)

**END**