

두산 Rokey Boot Camp

스터디 주간 활동 보고서

팀명	Robo:Loop	제출자 성명	홍송은
참여 명단	전효재, 홍송은, 김사웅		
모임 일시	2025 년 06 월 24 일 16 시 40 분 ~ 17 시 40 분		
장소	온라인 구글 미팅	출석 인원	3
학습목표	<ul style="list-style-type: none">• ROS 2 의 토픽 기반 통신 구조를 이해한다.• ROS 2 의 서비스 기반 통신 방식(request-response 구조)을 이해한다.• ROS 2 에서 사용자 정의 인터페이스, 메시지(.msg), 서비스(.srv) 를 생성하는 방법을 익힌다.		
학습내용	<ul style="list-style-type: none">• 전효재<ul style="list-style-type: none">○ Publisher / Subscriber<ul style="list-style-type: none">■ 통신 방식 중에 가장 기본이 되며 가장 널리 쓰이는 방법■ 센서 값 전송 및 항시 정보를 주고 받아야 하는 부분에 주로 사용○ ROS 프로그래밍시에 70% 이상이 Topic 으로 사용<ul style="list-style-type: none">■ Publisher : 특정 Topic 으로 데이터를 보냄■ Subscriber : 특정 Topic 를 구독해서 데이터를 받아옴<div><pre>graph LR; A((Node A Publisher)) -- Topic A --> C((Node C Subscriber)); A -- Topic B --> B((Node B Subscriber));</pre></div><ul style="list-style-type: none">■ 자신이 발행한 토픽을 셀프 구독할 수 있게 구성할 수도 있음○ 실습 진행<ul style="list-style-type: none">■ colon build -packages-select my_first_package■ source ./install/setup.bash (# 터미널 열 때마다 해줘야함)		

- `ros2 run my_first_package my_first_node`
- `ros2 run my_first_package my_subscriber`
- `ros2 run py_pubsub listener`

```

ros2 run my_first_package my_first_node
[INFO] [1758734427.833964699] [minimal_publisher]: Publishing: 'hello world: 0'
[INFO] [1758734428.395307111] [minimal_publisher]: Publishing: 'hello world: 1'
[INFO] [1758734428.895713865] [minimal_publisher]: Publishing: 'hello world: 2'
[INFO] [1758734429.395307111] [minimal_publisher]: Publishing: 'hello world: 3'
[INFO] [1758734429.895466694] [minimal_publisher]: Publishing: 'hello world: 4'
[INFO] [1758734430.395307111] [minimal_publisher]: Publishing: 'hello world: 5'
[INFO] [1758734430.895466694] [minimal_publisher]: Publishing: 'hello world: 6'
[INFO] [1758734431.395466694] [minimal_publisher]: Publishing: 'hello world: 7'
[INFO] [1758734431.895466694] [minimal_publisher]: Publishing: 'hello world: 8'
[INFO] [1758734432.395466694] [minimal_publisher]: Publishing: 'hello world: 9'
[INFO] [1758734432.895466694] [minimal_publisher]: Publishing: 'hello world: 10'
[INFO] [1758734433.395466694] [minimal_publisher]: Publishing: 'hello world: 11'
[INFO] [1758734433.895466694] [minimal_publisher]: Publishing: 'hello world: 12'
[INFO] [1758734434.395466694] [minimal_publisher]: Publishing: 'hello world: 13'
[INFO] [1758734434.895466694] [minimal_publisher]: Publishing: 'hello world: 14'
[INFO] [1758734435.395466694] [minimal_publisher]: Publishing: 'hello world: 15'
[INFO] [1758734435.895466694] [minimal_publisher]: Publishing: 'hello world: 16'
[INFO] [1758734436.395466694] [minimal_publisher]: Publishing: 'hello world: 17'

ros2 run py_pubsub listener
[INFO] [1758734429.943362647] [minimal_subscriber]: I heard: 'hello world: 0'
[INFO] [1758734430.398766262] [minimal_subscriber]: I heard: 'hello world: 1'
[INFO] [1758734430.898766262] [minimal_subscriber]: I heard: 'hello world: 2'
[INFO] [1758734431.398766262] [minimal_subscriber]: I heard: 'hello world: 3'
[INFO] [1758734431.898766262] [minimal_subscriber]: I heard: 'hello world: 4'
[INFO] [1758734432.398766262] [minimal_subscriber]: I heard: 'hello world: 5'
[INFO] [1758734432.898766262] [minimal_subscriber]: I heard: 'hello world: 6'
[INFO] [1758734433.398766262] [minimal_subscriber]: I heard: 'hello world: 7'
[INFO] [1758734433.898766262] [minimal_subscriber]: I heard: 'hello world: 8'
[INFO] [1758734434.398766262] [minimal_subscriber]: I heard: 'hello world: 9'
[INFO] [1758734434.898766262] [minimal_subscriber]: I heard: 'hello world: 10'
[INFO] [1758734435.398766262] [minimal_subscriber]: I heard: 'hello world: 11'
[INFO] [1758734435.898766262] [minimal_subscriber]: I heard: 'hello world: 12'
[INFO] [1758734436.398766262] [minimal_subscriber]: I heard: 'hello world: 13'
[INFO] [1758734436.898766262] [minimal_subscriber]: I heard: 'hello world: 14'
[INFO] [1758734437.398766262] [minimal_subscriber]: I heard: 'hello world: 15'
[INFO] [1758734437.898766262] [minimal_subscriber]: I heard: 'hello world: 16'
[INFO] [1758734438.398766262] [minimal_subscriber]: I heard: 'hello world: 17'

```

● 김사웅

- `ros2 pkg create` 명령어로 `cpp_srvcli` 라는 이름의 패키지를 생성
- `--build-type ament_cmake` 은 C++용 빌드 시스템인 `ament_cmake` 를 사용하도록 지정
- `--license Apache-2.0` 은 패키지 라이선스를 Apache 2.0 으로 설정
- `--dependencies rclcpp example_interfaces` 는 C++ 클라이언트 라이브러리와 예제 인터페이스 의존성을 등록.
- `package.xml` 과 `CMakeLists.txt` 에 의존성이 자동으로 추가.
- 서버 코드는 `add_two_ints_server.cpp`, 클라이언트 코드는 `add_two_ints_client.cpp` 로 작성
- CMake에서는 `add_executable` 로 각각 서버와 클라이언트를 빌드 대상으로 지정
- `ament_target_dependencies` 로 필요한 종속 라이브러리를 연결.
- `install()` 명령으로 빌드된 실행파일을 설치 경로에 배치
- `ament_package()`로 전체 패키지를 마무리하고 빌드할 수 있도록 설정.

```

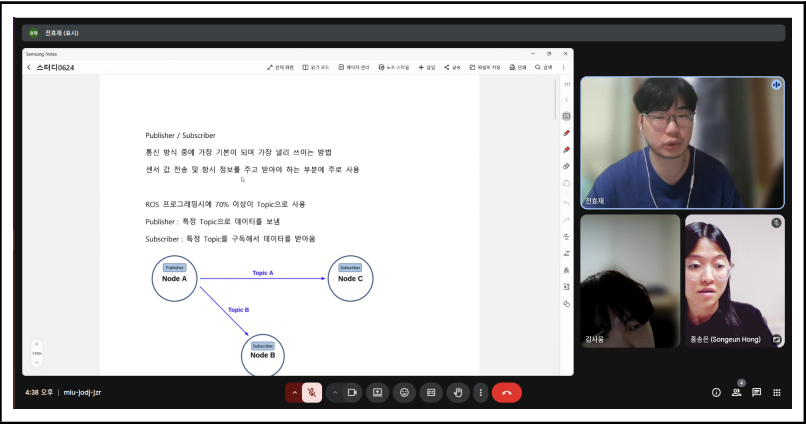
2. tools
  anyxl2_vendor
  anyxl_vendor
  list
  list.cpp
  opic_monitor
  ractools
  trajectory_msgs
  urtlesin
  urtlesin_msgs
  urtlesin_vendor
  unique_identifier_msgs
  urdf
  urdf_parser_plugin
  visualization_msgs
  xmi_cpp_vendor
  std_vendor
saungking@saungking-Vivobook-GO-E1504FA-E1504FA:~/ros2_ws$ ros2 run cpp_srvcli ser
[INFO] [1758732581.441443849] [rclcpp]: Ready to add two ints.
[INFO] [1758732632.041846533] [rclcpp]: Incoming request
2 2 b: 3
[INFO] [1758732632.841136198] [rclcpp]: Sum: 5
saungking@saungking-Vivobook-GO-E1504FA-E1504FA:~/ros2_ws$ source install/setup.ba
sh
saungking@saungking-Vivobook-GO-E1504FA-E1504FA:~/ros2_ws$ ros2 run cpp_srvcli cli
ent 2 3
Package 'cpp_srvcli' not found
saungking@saungking-Vivobook-GO-E1504FA-E1504FA:~/ros2_ws$ source install/setup.ba
sh
saungking@saungking-Vivobook-GO-E1504FA-E1504FA:~/ros2_ws$ ros2 run cpp_srvcli cli
ent 2 3
[INFO] [1758732632.841136198] [rclcpp]: Sum: 5
saungking@saungking-Vivobook-GO-E1504FA-E1504FA:~/ros2_ws$

```

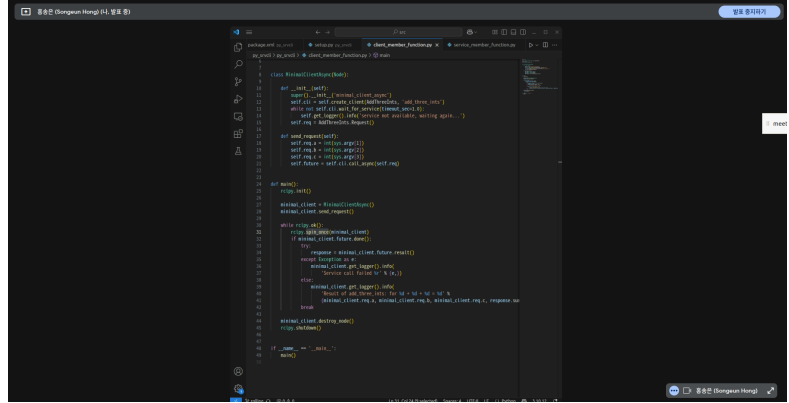
● 홍승은

- ROS 2 Custom Interface 생성
 - `ament_cmake` 패키지로 `tutorial_interfaces` 생성
 - `msg`, `srv` 폴더 내 인터페이스 파일 작성 (.msg, .srv)
- 인터페이스 정의
 - `Num.msg`: `int64 num`
 - `Sphere.msg`: `geometry_msgs/Point` + `float64`
 - `AddThreeInts.srv`: 요청(a, b, c), 응답(sum) 분리 구조
- 설정 파일 수정
 - `CMakeLists.txt`: `rosidl_generate_interfaces()` 호출
 - `package.xml`: 의존성 및 인터페이스 패키지 그룹 선언
- 빌드 및 적용
 - `colcon build` 로 빌드
 - `source install/setup.bash` 실행

	<ul style="list-style-type: none"> ○ Python 패키지 연동 <ul style="list-style-type: none"> ■ py_pubsub: Num.msg 사용 <ul style="list-style-type: none"> ● import 구문 수정 ● setup.py entry_points 등록 ● package.xml 에 exec_depend 추가 ■ py_srvcli: AddThreeInts.srv 사용 <ul style="list-style-type: none"> ● service.py, client.py 에 import ● setup.py, package.xml 수정 동일 ○ 실행 테스트 <ul style="list-style-type: none"> ■ ros2 run py_pubsub talker / listener ■ ros2 run py_srvcli service / client 2 3 4 ○ msg vs .srv 차이 <ul style="list-style-type: none"> ■ msg: 단방향 메시지 (pub/sub) ■ srv: 요청-응답 구조 (client/service) ■ 내부 변환: .idl → .py, .hpp 자동 생성됨 ○ 주의 <ul style="list-style-type: none"> ■ 인터페이스는 ament\ cmake 패키지로 분리 ■ CMakeLists.txt, package.xml 설정 필수 ■ Python 쪽은 import + setup.py + 의존성 명시 필요 ■ 빌드 후 항상 source 설치 적용 필수 						
활동평가	<table> <tr> <td>전효재</td><td>ROS 에서 많이 사용되는 통신 방법인 Publisher / Subscriber 에 대해 정리함. 데이터가 전달되는 방법을 알수 있었고 실습을 진행하며 Publisher / Subscriber 뿐만아니라 Topic 와 Node 가 하는 역할까지 이해하는데 도움이 됨.</td></tr> <tr> <td>홍송은</td><td>ROS 2 에서 사용자 정의 인터페이스를 생성하고 Python 패키지에서 실제로 활용하는 전체 과정을 실습함으로써, ROS 2 의 메시지 통신 구조에 대한 이해를 심화할 수 있었음. 특히, .msg 와 .srv 파일의 차이점, ament_cmake 기반 인터페이스 패키지 구성 방식, 그리고 call_async()와 rclpy.spin_once()를 활용한 비동기 클라이언트 구조 등을 직접 구현하고 테스트해보며, ROS 2 통신 시스템의 실무 적용 역량을 키울 수 있는 유익한 시간이었음.</td></tr> <tr> <td>김사웅</td><td>서비스 통신을 통해 요청-응답 구조가 토픽과는 다르게 동기적이라는 점을 확실히 체감할 수 있음. 특히, 클라이언트가 명시적으로 요청을 보내야 응답이 발생한다는 구조 덕분에 로직 흐름을 보다 명확하게 만들 수 있음. 실습을 통해 동기/비동기 호출의 차이도 체험할 수 있었고, 실제 로봇 시스템에서 어떤 상황에 서비스를 사용할지에 대한 감도 얻을 수 있음.</td></tr> </table>	전효재	ROS 에서 많이 사용되는 통신 방법인 Publisher / Subscriber 에 대해 정리함. 데이터가 전달되는 방법을 알수 있었고 실습을 진행하며 Publisher / Subscriber 뿐만아니라 Topic 와 Node 가 하는 역할까지 이해하는데 도움이 됨.	홍송은	ROS 2 에서 사용자 정의 인터페이스를 생성하고 Python 패키지에서 실제로 활용하는 전체 과정을 실습함으로써, ROS 2 의 메시지 통신 구조에 대한 이해를 심화할 수 있었음. 특히, .msg 와 .srv 파일의 차이점, ament_cmake 기반 인터페이스 패키지 구성 방식, 그리고 call_async()와 rclpy.spin_once()를 활용한 비동기 클라이언트 구조 등을 직접 구현하고 테스트해보며, ROS 2 통신 시스템의 실무 적용 역량을 키울 수 있는 유익한 시간이었음.	김사웅	서비스 통신을 통해 요청-응답 구조가 토픽과는 다르게 동기적이라는 점을 확실히 체감할 수 있음. 특히, 클라이언트가 명시적으로 요청을 보내야 응답이 발생한다는 구조 덕분에 로직 흐름을 보다 명확하게 만들 수 있음. 실습을 통해 동기/비동기 호출의 차이도 체험할 수 있었고, 실제 로봇 시스템에서 어떤 상황에 서비스를 사용할지에 대한 감도 얻을 수 있음.
전효재	ROS 에서 많이 사용되는 통신 방법인 Publisher / Subscriber 에 대해 정리함. 데이터가 전달되는 방법을 알수 있었고 실습을 진행하며 Publisher / Subscriber 뿐만아니라 Topic 와 Node 가 하는 역할까지 이해하는데 도움이 됨.						
홍송은	ROS 2 에서 사용자 정의 인터페이스를 생성하고 Python 패키지에서 실제로 활용하는 전체 과정을 실습함으로써, ROS 2 의 메시지 통신 구조에 대한 이해를 심화할 수 있었음. 특히, .msg 와 .srv 파일의 차이점, ament_cmake 기반 인터페이스 패키지 구성 방식, 그리고 call_async()와 rclpy.spin_once()를 활용한 비동기 클라이언트 구조 등을 직접 구현하고 테스트해보며, ROS 2 통신 시스템의 실무 적용 역량을 키울 수 있는 유익한 시간이었음.						
김사웅	서비스 통신을 통해 요청-응답 구조가 토픽과는 다르게 동기적이라는 점을 확실히 체감할 수 있음. 특히, 클라이언트가 명시적으로 요청을 보내야 응답이 발생한다는 구조 덕분에 로직 흐름을 보다 명확하게 만들 수 있음. 실습을 통해 동기/비동기 호출의 차이도 체험할 수 있었고, 실제 로봇 시스템에서 어떤 상황에 서비스를 사용할지에 대한 감도 얻을 수 있음.						

<p>과제</p>	<ul style="list-style-type: none"> ex_calculator 예제를 통해 ROS 2 서비스 구조, 사용자 정의 인터페이스, 노드 간 통신 흐름, 빌드 및 실행 방법을 실습하며 서비스 구현 능력을 기른다.
<p>향후 계획</p>	<p>ROS 2 패키지 구조 이해</p> <ul style="list-style-type: none"> ex_calculator 패키지의 디렉토리 구조와 역할을 파악한다 (package.xml, setup.py, CMakeLists.txt, src/, srv/ 등). <p>Service 인터페이스 정의 방식 학습</p> <ul style="list-style-type: none"> srv 폴더 내 .srv 파일을 분석하여, 사용자 정의 서비스 메시지 형식을 이해한다. <p>Python 기반 서비스 서버/클라이언트 동작 원리 파악</p> <ul style="list-style-type: none"> server.py, client.py 의 코드 흐름을 분석하고, create_service, call_async 등의 메서드 사용법을 익힌다. <p>서비스 통신 과정 실습 및 검증</p> <ul style="list-style-type: none"> ros2 run, ros2 service call 명령어를 통해 실제로 서비스를 호출해보고, 요청-응답 과정을 확인한다.
<p>첨부 자료</p>	<div data-bbox="367 1435 1367 1856"> <div data-bbox="389 1610 534 1675"> <p>스터디 화면 1</p> </div> <div data-bbox="558 1435 1367 1856">  </div> </div>

스터디 화면 2



결과물

ds_rokey4_study_team2 / codes / week15_ROS_Tutorial /

README.md

docs: add week 15 README

14 minutes ago

ROS_Tutorial_Hyojae.pdf

Rename 2-F1C0624.pdf to ROS_Tutorial_Hyojae.pdf

9 hours ago

ROS_Tutorial_Saung.pdf

upload tutorial_service

10 hours ago

ROS_Tutorial_Songun.pdf

docs: upload week 15 tutorial material - custom interface

1 hour ago

README.md

컴퓨터비전 스터디 기록

15주차 - ROS2 사용자 정의 인터페이스 및 통신 구조 실습

✓ 학습 목표

- ROS 2의 Topic 기반 비동기 통신 구조와 Service 기반 동기 통신 구조 이해
- 사용자 정의 메시지(msg)와 서비스(srv) 생성 및 활용 방법 숙지
- Publisher/Subscriber, Service/Client 구성 실습을 통해 ROS 2 통신 흐름 체험
- Python 및 C++ 패키지를 통한 실제 예제 구현과 빌드/실행 과정 실습
- ROS 2 Humble 공식 문서를 기반으로 실습하며 기본 구조와 사용법 체득

실습 참조 자료:

- [Writing a Simple Python Publisher and Subscriber \(ros2 doc\)](#)
- [Writing a Simple Python Service and Client \(ros2 doc\)](#)
- [Creating Custom ROS 2 Interfaces \(ros2 doc\)](#)

✨ 주요 학습 내용

전호재

- Topic 기반 통신 구조 학습