™

ROS2_day4_HW1

20기 인턴 강은구

목차

1 HW1

1.1 개요

ros2에서 제공되는 qos를 사용하여 통신 과정에서 신뢰성이 확보되고, lifecycle 시스템을 적용하여 원하는 노드를 on/off하여 코드를 구성하였다.

1.2 Qos 및 Lifecycle

- Qos

  qos를 알기 전에, DDS에 대해 설명하자면, 이는 데이터 분산 시스템으로, ros2에서는 데이터 통신을 위한 미들웨이어이다. 운영체제와 사용자 애플리케이션 사이의 소프트 웨어 계층에서 이를 통해 통신하며 데이터를 공유하게 된다. 이때, 노드 간의 데이터 통신을 세부적으로 조정하도록 하는 것이 qos이다. Qos를 통해 tcp처럼 데이터 손실을 방지하여 신뢰도를 높이기도 하며 udp처럼 통신 속도를 우선하여 사용할 수도 있다.

QoS policies	Publisher	Subscription	Compatible
reliability	Best effort	Best effort	Yes
reliability	Best effort	Reliable	No
reliability	Reliable	Best effort	Yes
reliability	Reliable	Reliable	Yes

  이와 같은 명령으를 발행자와 구독자 노드에 적용하여 통신을 허용 및 차단할 수 있다.

- Lifecycle

  lifecycle이란, 시스템의 상태를 관리하고 제어하는 기능을 제공하는ros2의 도구이다. 해당 시스템을 사용하여, 필요한 노드를 끄거나 킬 수 있으며, rqt를 사용하여 노드의 on/off 제어 및 상태 확인을 시각적으로 할 수 있다.

1.3 코드 전체 흐름

해당 코드는 기본적으로 lifecycle헤더파일을 호출하였고, 형식에 맞춰 클래스를 구성하였다. Publisher 클레스는 rclcpp_lifecycle::LifecycNode로 선언하여 lifecycle시스템을 사용할 수 있도록 하였다. 처음 코드를 실행시키게 되면 아무것도 동작되지 않은 상태의 unconfigure상태로 시작하며, rqt를 사용하여 1~5의 숫자를 입력해 노드의 상태를 바꿔줄 수 있다. 순서는 1을 입력할 경우 on_configure함수가 호출되어 동작에 사용되는 각 변수 및 메모리를 초기화하고, 3을 누르게 되면 on_avtivate 상태가 되어 저당된 메모리 및 데이터 전송과 같은 기능을 실질적으로 수행하게된다. 4를 입력하면 on_deactiavte 상태가 되어 수행중이던 작업들이 모두 정지하게 되고, 2와 5를 순서대로 누르면 메모리

초기화 및 해제가 순차적으로 이루어진다. 각 노드 내에서는 publisher_->~과 같은 명령어를 사용하여 노드의 상태와 기능을 변경해주었다. 예를 들어 on_activate함수가 호출된 경우, publisher_->activate();를 해줌으로써 activate기능인 메모리 전송 등의 실질적인 동작 수행 기능을 활성화해주었다 또한 각 상태에 대한 표시 메시지는 publisher가 실행된 터미널에 상태 변환 시 나타내도록 하였고, on activate상태일 경우 메시지 전송 유무를 확인하기 위해 on activat라는 문구를 토픽으로 발행하도록 하였다. 해당 문구는 동일한 토픽을 구독하고 있는 subscriber노드가 읽어 해당 터미널에 나타내도록 하였다. qos통신 방식을 적용하였기 때문에, 통신 시 양 측 reliable을 모두 reliablity로 통일하여 두 노드 간의 통신을 허용해주었다.

1.4 코드 상세 설명-lifecycle_talker

- 코드 맨 위에 "rclcpp_lifecycle/lifecycle_node.hpp"헤더를 선언하여 lifecycle 기능을 사용할 수 있도록 하였다.
- 생성자
  1) Namespace를 사용하여 노드 간의 충돌을 방지하였다.
  2) 생성자에서 qos통시을 위해 qos_profile과 reliability를 설정하였고, reliability의 경우 subscriber와 통일하여 노드 간 통신이 가능하도록 하였다.
  3) 타미머 변수르 선언하여 노드가 일정한 시간 간격으로 callback함수를 호출하도록 하였다. 이는 lifecycle 흐름에 따라 노드가 특정 상태일 경우에만 함수가 호출될 것이며, callback함수 내용인 메시지 전송 동작이 이루어질 것이다.
- On_configure
  1) RCLCPP_INFUL를 통해 해당 로그를 획득하고, "on_configure"이라는 메시지를 터미널 창에 표시하도록 하였다. 또한 메시지 변숭 ㅔ on_configure이라는 문구를 저장하도록 하였다.
  2) 메시지에 문자열이 저장되었지만 현재 상테는 configure상태이므로 메시지 전송과 같은 실질적인 동작은 하지 않는 상태이다.
  3) 함수 마지막에 lifecycle명령어를 사용하여 success를 반영하도록 하였다.
- On_activate
  1) 함수가 호출되고 처음 하는 동작은 publisher_->on_activate(); 명령어를 사용하여 노드의 상태를 activate상태로 전환해주었다. 따라서 노드는 메시지 전송과 같은 실질적인 동작을 한다.
  2) 또한 다른 함수와 마찬가지로 RCLCPP_INFO명령어를 통해 현재 변환

된 상태를 터미널 창에 표시한다.

3) 마지막으로 on_configure함수와 동일하게 상태 변환이 성공했다는 메시지를 반환한다.

- On_deativate
  1) 함수가 호출되면 노드는 deativate상태가 되며, 해당 상태에서 노드는 activate상태에서 수행중이던 모든 작업을 중단한다. 상태 변환은 동이랗게 publisher_-<on_deactivate(); 명령어를 통해 진행하였다.
  2) 이 또한 현재 상태를 RCLCPP_INFO명령어를 통해 터미널 창에 띄우고 상태 변환이 성공했다는 메시지를 반환한다.
- On_cleaup
  1) 함수가 호출되면 publisher_.reset();함수가 호출되어 사용중이던 메모리와 저장된 변수들이 해제된다.
  2) 타이머 또한 timer_.reset(); 명령어를 통해 초기화 되며, 초기화 한뒤에 RCLCPP_INFO 명령어를 통해 터미널 창에 on_cleanup()이라는 문구를 표시에 현재 상태를 알린다.
  3) 마지막으로 동일하게 상태 변환이 성공했다는 메시지를 반환한다.
- private변수 선언
  1) 전송할 메시지가 저장될 변수msg를 std_msgs타임에 맞춰 선언해주었다
  2) Rclcpp::TiemrBase::SharedPtr imer_;-> 타이머를 선언하여 주기적인 동작을 하기 위한 준비를 하였다.

1.5 코드 상세 설명-lifecycle_listener

- 해당 노드는 구독자 노드로, 등록한 토픽으로부터 데이터를 얻는다.
- 생성자
  1) 노드 명은 lifecycle_listener로 서릊ㅇ하였다.
  2) publihser노드와 마찬가지로 rclcpp::QoS qos_profile(10);명령어를 통해 qos통신방식을 사용하도록 하였고, qos_profil.reliability를 publisher와 동일하게 reliable로 저장하여 두 노드간에 통신이 가능하도록 하였다.
  3) Subscription_=~코드에서는 string형 데이터를 얻을 수 있도록 하였으며, 토픽은 publisher와 동일한 eung, qos_profile을 사용하고 topic에 데이터가 저장되었을 경우 topic)callback함수를 호출하도록 하였다.
- Callback()
  1) 단순히 토픽으로부터 데이터를 얻어 터미널 창에 나타내는 함수이다. 명령어는 RCLCPP_INFO를 사용하였다.

- 변수 선언

  1) Rclcpp::Subscription<std_msgs::msg::String>::SharePtr subscription

- main함수

  1) rclcpp::init

     rclcpp를 활성화하였다.

  2) rclcpp::spin

     함수 내의 동작이 반복되도록 설정하였다.

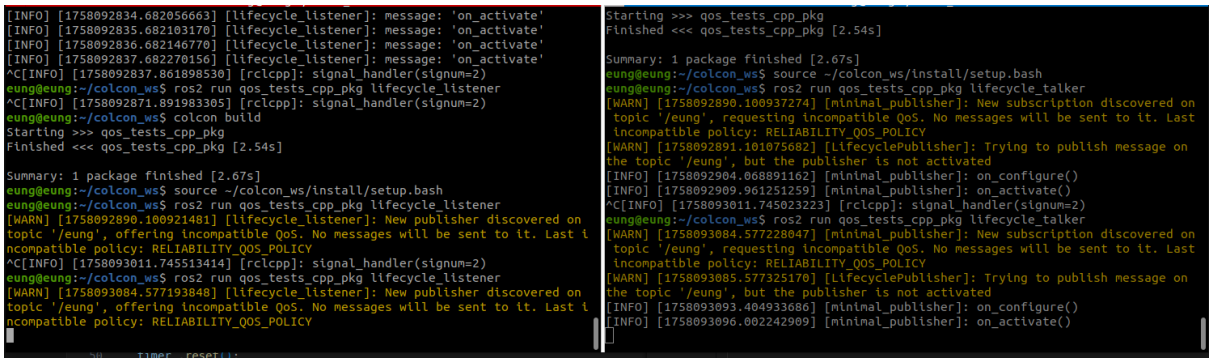
  3) rclcppLshutdown():

     spin을 통해 반복되던 동작들이 ctrl_c가 입력될 경우 종료되게 하였다.

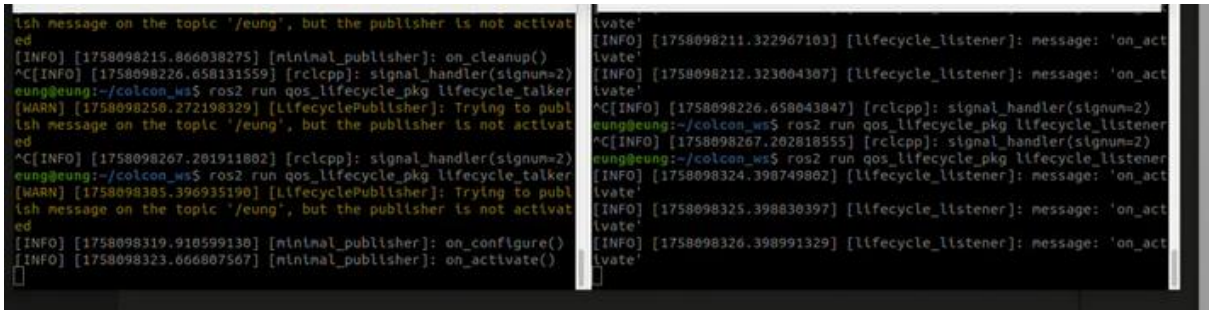
1.6 코드 동작

- qos통신 비활성화

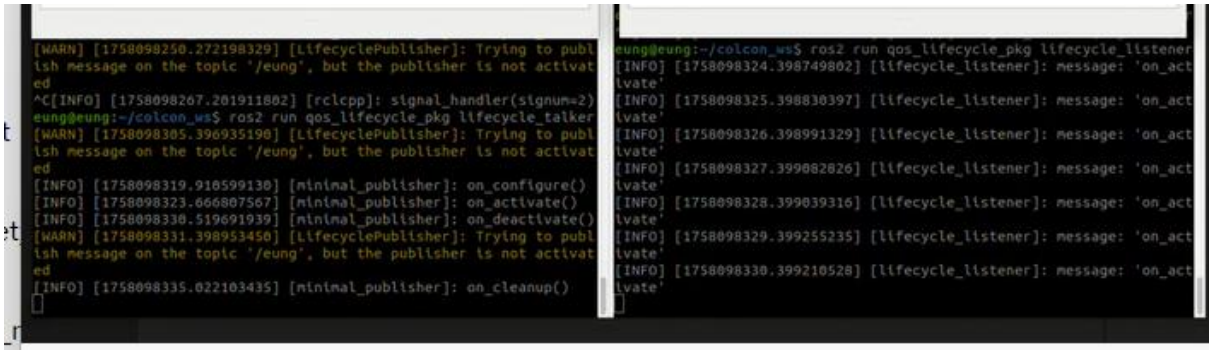
  publisher와 subscriber의 reliability를 각각 best effor, reliability로 설정하였을 경우 노드 간의 통신이 진행되지 않는 모습이다.



- 통신을 활성화 하고 노드의 상태를 activie로 설정하였을 경우 메시지가 정상적으로 전달되는 모습이다.



- Rqt를 사용하여 노드 및 메모리 생성-동작-정지-해제-종료 순으로 실행한 결과이다. 각 실행마다 변환된 상태가 터미널 창에 나타난다.


[WARN] [1758098250.272198329] [LifecyclePublisher]: Trying to publ
ish message on the topic '/eung', but the publisher is not activat
ed
^C[INFO] [1758098267.201911802] [rclcpp]: signal_handler(signum=2)
eung@eung:~/colcon_ws$ ros2 run qos_lifecycle_pkg lifecycle_talker
[WARN] [1758098305.396935190] [LifecyclePublisher]: Trying to publ
ish message on the topic '/eung', but the publisher is not activat
ed
[INFO] [1758098319.910599130] [minimal_publisher]: on_configure()
[INFO] [1758098323.666807567] [minimal_publisher]: on_activate()
[INFO] [1758098330.519691939] [minimal_publisher]: on_deactivate()
[WARN] [1758098331.398953450] [LifecyclePublisher]: Trying to publ
ish message on the topic '/eung', but the publisher is not activat
ed
[INFO] [1758098335.022103435] [minimal_publisher]: on_cleanup()
eung@eung:~/colcon_ws$ ros2 run qos_lifecycle_pkg lifecycle_listener
[INFO] [1758098324.398749802] [lifecycle_listener]: message: 'on_act
ivate'
[INFO] [1758098325.398830397] [lifecycle_listener]: message: 'on_act
ivate'
[INFO] [1758098326.398991329] [lifecycle_listener]: message: 'on_act
ivate'
[INFO] [1758098327.399082826] [lifecycle_listener]: message: 'on_act
ivate'
[INFO] [1758098328.399039316] [lifecycle_listener]: message: 'on_act
ivate'
[INFO] [1758098329.399255235] [lifecycle_listener]: message: 'on_act
ivate'
[INFO] [1758098330.399210528] [lifecycle_listener]: message: 'on_act
ivate'