ROS 2 4주차

13장 ROS 2 액션

13.1 액션

- 비동기식, 동기식 양방향 메시지 송수신 방식
- Action Client : 액션 목표 지정
- Action Server : 액션 피드백(액션 목표를 받아 특정 태스크를 수행),액션 결과(최종 결괏값)을 전송
- 토픽과 서비스의 혼합
- ROS 2에서는 ROS 1과 달리 목표 전달(send_goal), 목표 취소(cancel_goal), 결과 받기(get_result)를 위해 서비스 통신을 사용
- 비동기 방식을 이용하다보면 원하는 타이밍에 적절한 액션을 수행하 어려운데 이를 해결기 위해 목표 상태(goal_state)를 이용
- 목표 상태 : 목푯값을 전달한 후 상태 머신을 구동하여 액션의 프로스를 쫓는 것

13.2 액션 서버 및 클라이언트

```
$ ros2 run turtlesim turtlesim_node
```

```
$ ros2 run turtlesim turtle_teleop_key
Reading from keyboard
---------------------------
Use arrow keys to move the turtle.
Use G|B|V|C|D|E|R|T keys to rotate to absolute orientations. 'F' to cancel a rotation.
'Q' to quit.
```

액션의 결과는 turtlesim_node 터미널 창에 표시

-목표 각도 값에 도달하면

```
[INFO] [1690439955.746218557] [turtlesim]: Rotation goal completed successfully
```

-목표 값에 도달하기 전에 F키를 눌러 액션 목표를 취소하면

```
[INFO] [1690440100.210469519] [turtlesim]: Rotation goal canceled
```

13.3 노드 정보(ros2 node info)

```
$ ros2 node info /turtlesim
/turtlesim
  Subscribers:
    /parameter_events: rcl_interfaces/msg/ParameterEvent
    /turtle1/cmd_vel: geometry_msgs/msg/Twist
  Publishers:
    /parameter_events: rcl_interfaces/msg/ParameterEvent
    /rosout: rcl_interfaces/msg/Log
    /turtle1/color_sensor: turtlesim/msg/Color
    /turtle1/pose: turtlesim/msg/Pose
  Service Servers:
```

```
    /clear: std_srvs/srv/Empty
    /kill: turtlesim/srv/Kill
    /reset: std_srvs/srv/Empty
    /spawn: turtlesim/srv/Spawn
    /turtle1/set_pen: turtlesim/srv/SetPen
    /turtle1/teleport_absolute: turtlesim/srv/TeleportAbsolute
    /turtle1/teleport_relative: turtlesim/srv/TeleportRelative
    /turtlesim/describe_parameters: rcl_interfaces/srv/DescribeParameters
    /turtlesim/get_parameter_types: rcl_interfaces/srv/GetParameterTypes
    /turtlesim/get_parameters: rcl_interfaces/srv/GetParameters
    /turtlesim/list_parameters: rcl_interfaces/srv/ListParameters
    /turtlesim/set_parameters: rcl_interfaces/srv/SetParameters
    /turtlesim/set_parameters_atomically: rcl_interfaces/srv/SetParametersAtomically
  Service Clients:

  Action Servers:
    /turtle1/rotate_absolute: turtlesim/action/RotateAbsolute
  Action Clients:
```

```
$ ros2 node info /teleop_turtle
/teleop_turtle
  Subscribers:
    /parameter_events: rcl_interfaces/msg/ParameterEvent
  Publishers:
    /parameter_events: rcl_interfaces/msg/ParameterEvent
    /rosout: rcl_interfaces/msg/Log
    /turtle1/cmd_vel: geometry_msgs/msg/Twist
  Service Servers:
    /teleop_turtle/describe_parameters: rcl_interfaces/srv/DescribeParameters
    /teleop_turtle/get_parameter_types: rcl_interfaces/srv/GetParameterTypes
    /teleop_turtle/get_parameters: rcl_interfaces/srv/GetParameters
    /teleop_turtle/list_parameters: rcl_interfaces/srv/ListParameters
    /teleop_turtle/set_parameters: rcl_interfaces/srv/SetParameters
    /teleop_turtle/set_parameters_atomically: rcl_interfaces/srv/SetParametersAtomically
  Service Clients:

  Action Servers:

  Action Clients:
    /turtle1/rotate_absolute: turtlesim/action/RotateAbsolute
```

13.4 액션 목록(ros2 action list -t)

현재 개발환경에서 실행 중인 액션 목록 확인

```
$ ros2 action list -t
/turtle1/rotate_absolute [turtlesim/action/RotateAbsolute]
```

13.5 액션 정보(ros2 action info)

검색된 액션 목록을 더 자세히 확인

액션 이름, 해당 액션의 서버 및 클라이언트 노드 이름 및 개수 확인

```
$ ros2 action info /turtle1/rotate_absolute
Action: /turtle1/rotate_absolute
Action clients: 1
    /teleop_turtle
Action servers: 1
    /turtlesim
```

13.6 액션 목표(action goal) 전달

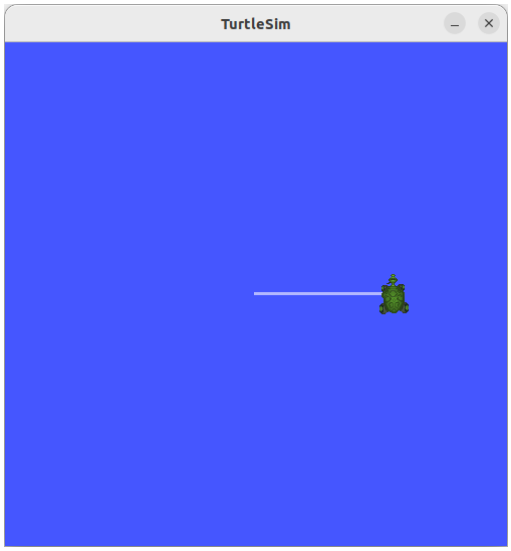
```
ros2 action send_goal <action_name> <action_type> "<values>"
```

```
$ ros2 action send_goal /turtle1/rotate_absolute turtlesim/action/RotateAbsolute "{theta: 1.5708}"
Waiting for an action server to become available...
Sending goal:
     theta: 1.5708

Goal accepted with ID: 4eb147dbe4854b1abd28c770322e0af4

Result:
    delta: 2.175999879837036

Goal finished with status: SUCCEEDED
```



- 피드백 포함

```
$ ros2 action send_goal /turtle1/rotate_absolute turtlesim/action/RotateAbsolute "{theta: 1.5708}" --feedback
Waiting for an action server to become available...
Sending goal:
     theta: 1.5708

Feedback:
    remaining: -0.01957070827484131

Goal accepted with ID: 49c77f8278b34f9aacccefc5a89c13f8

Result:
    delta: 0.0
```

14장 ROS 2 인터페이스

14.1 ROS 2 인터페이스

ROS 노드 간에 데이를 주고 받기 위해 토픽, 서비스, 액션을 사용하는데 이 때 사용되는 데이터 형태를 ROS 2 인터페이스라고 함

ROS 1 : msg, srv, action

ROS 2 : msg, srv, action , IDL(Interface Definition Language)

- 메시지 안에 메시지를 품고 있는 데이터 구조와 메시지들이 나열된 배열 형태는 단순 자료형을 기반으로 함

```
fieldtype1 fieldname1 //메시지 자료형, 메시지 이름
fieldtype2 fieldname2
fieldtype3 fieldname3
```

14.2 메시지 인터페이스(Message interface, msg)

```
$ ros2 interface show geometry_msgs/msg/Twist
# This expresses velocity in free space broken into its linear and angular parts.

Vector3  linear
  float64 x
  float64 y
  float64 z
Vector3  angular
  float64 x
  float64 y
  float64 z
```

```
$ ros2 interface show geometry_msgs/msg/Vector3
# This represents a vector in free space.

# This is semantically different than a point.
# A vector is always anchored at the origin.
# When a transform is applied to a vector, only the rotational component is applied.

float64 x
float64 y
float64 z
```

- list : 현재 개발 환경의 모든 msg, srv, action 메시지를 보여줌

```
$ ros2 interface list
Messages:
    action_msgs/msg/GoalInfo
    action_msgs/msg/GoalStatus
    action_msgs/msg/GoalStatusArray
    actionlib_msgs/msg/GoalID
    actionlib_msgs/msg/GoalStatus
    actionlib_msgs/msg/GoalStatusArray
    builtin_interfaces/msg/Duration
    builtin_interfaces/msg/Time
    control_msgs/msg/AdmittanceControllerState

   (생략)

Services:
    action_msgs/srv/CancelGoal
    composition_interfaces/srv/ListNodes
    composition_interfaces/srv/LoadNode
    composition_interfaces/srv/UnloadNode
    control_msgs/srv/QueryCalibrationState
    control_msgs/srv/QueryTrajectoryState
    controller_manager_msgs/srv/ConfigureController
```

- packages : msg, srv, action 인터페이스를 담고 있는 패키지 목록을 보여줌

```
$ ros2 interface packages
action_msgs
action_tutorials_interfaces
actionlib_msgs
builtin_interfaces
```

- package 옵션에 패키지명을 입력하면 지정한 패키지에 포함된 인터페이스들을 보여줌

```
$ ros2 interface package turtlesim
turtlesim/srv/SetPen
turtlesim/srv/Kill
turtlesim/srv/TeleportAbsolute
turtlesim/msg/Color
turtlesim/srv/Spawn
turtlesim/msg/Pose
turtlesim/srv/TeleportRelative
turtlesim/action/RotateAbsolute
```

- proto에 특정 인터페이스 형태를 입력하면 그 인터페이스의 기본 형태를 표시

```
$ ros2 interface proto geometry_msgs/msg/Twist
"linear:
  x: 0.0
  y: 0.0
  z: 0.0
angular:
  x: 0.0
  y: 0.0
  z: 0.0
"
```

14.3 서비스 인터페이스(Service interface, srv)

```
$ ros2 interface show turtlesim/srv/Spawn
float32 x
float32 y
float32 theta
string name # Optional.  A unique name will be created and returned if this is empty
---
string name
```

*foxy(책 기준)가 아닌 humble 환경에서는 Spawn.srv를 사용하면 오류가 났는데 .srv를 지우고 실행하였더니 해결되었습니다

- 구분자(—)를 기준으로 위는 서비스 요청, 아래는 응답하는 회신 데이터를 나타냄

14.4 액션 인터페이스(Action interface, action)

```
$ ros2 interface show turtlesim/action/RotateAbsolute
# The desired heading in radians
float32 theta //액션 목표
---
# The angular displacement in radians to the starting position
float32 delta  //액션 결과
---
# The remaining rotation in radians
float32 remaining //액션 피드백
```

*srv의 경우와 같이 foxy 기준으로 작성된 책과 다르게 .action을 지우고 실행

15장 ROS 2 토픽/서비스/액션 정리 및 비교

15.1 토픽, 서비스, 액션 비교

- 토픽, 서비스, 액션 비교

	토픽(Topic)	서비스(service)	액션(Action)
연속성	연속성	일회성	복합(토픽+서비스)
방향성	단방향	양방향	양방향
동기성	비동기	동기	동기 + 비동기
다자간 연결	1:1, 1:N. N:1, N:N	1:1	1:1
노드 역할	퍼블리셔, 서브스크라이버	서버, 클라이언트	서버, 클라이언트
동작 트리거	퍼블리셔	클라이언트	클라이언트
인터페이스	msg	srv	action
CLI 명령어	ros2 topic ros2 interface	ros2 tservice ros2 interface	ros2 action ros2 interface
사용 예	센서 데이터, 로봇 상태, 로봇 좌표, 로봇 속도 명령	LED 제어, 모터 토크 On/Off, 이동 경로 계산	목적지로 이동, 물건 파지, 복합 태스크

15.2 인터페이스 비교

	msg 인터페이스	srv 인터페이스	action 인터페이스
확장자	.msg	.srv	.action
데이터	토픽 데이터 (data)	서비스 요청 (request) --- 서비스 응답 (response)	액션 목표 (goal) --- 액션 결과 (result) --- 액션 피드백 (feedback)
형식	fieldtype1 fieldname1 fieldtype2 fieldname2 fieldtype3 fieldname3	fieldtype1 fieldname1 fieldtype2 fieldname2 --- fieldtype3 fieldname3 fieldtype4 fieldname4	fieldtype1fieldname1 fieldtype2 fieldname2 --- fieldtype3 fieldname3 fieldtype4 fieldname4 --- fieldtype5 fieldname5 fieldtype6 fieldname6
사용 예	[geometry_msgs/msg/Twist]	[turtlesim/srv/Spawn.srv]	[turtlesim/action/RotateAbsolute.action]
	Vector3 linear Vector3 angular	float32 x float32 y float32 theta string name --- string name	float32 theta --- float32 delta --- float32 remaining

모든 인터페이스는 msg 인터페이스의 확장형

16장 ROS 2 파라미터

16.1 파라미터

- 서비스 통신 방법을 이용하여 노드 내부 또는 외부에서 노드 내 매개변수를 쉽게 지정하거나 변경하고, 쉽게 가져와서 사용할 수 있게함 (서비스와는 목적이 다름)
- RCL(ROS Client Libraries)의 기본 기능으로 모든 노드가 자신만의 Parameter server를 가지고 있고, 각 노드는 Parameter client를 가질 수도 있어서 자기 자신의 파라미터 및 다른 노드의 파라미터를 읽고 쓸 수 있음 →각 노드의 매개변수를 글로벌 매개변수처럼 사용할 수 있게 되어 추가 프로그래밍이나 컴파일 없이 능동적으로 변화 가능한 프로세스 만들 수 있음
- 노드 실행 시에 파라미터 설정 파일을 불러와 사용할 수 있음

16.2 파라미터 목록 확인(ros2 param list)

실습 전 세팅

```
$ ros2 run turtlesim turtlesim_node
```

```
$ ros2 run turtlesim turtle_teleop_key
```

ros2 param list

```
$ ros2 param list
/teleop_turtle:
  qos_overrides./parameter_events.publisher.depth
  qos_overrides./parameter_events.publisher.durability
  qos_overrides./parameter_events.publisher.history
  qos_overrides./parameter_events.publisher.reliability
  scale_angular
  scale_linear
  use_sim_time
/turtlesim:
  background_b
  background_g
  background_r
  qos_overrides./parameter_events.publisher.depth
  qos_overrides./parameter_events.publisher.durability
  qos_overrides./parameter_events.publisher.history
  qos_overrides./parameter_events.publisher.reliability
  use_sim_time
```

use_sim_time : 모든 노드의 기본 파라미터

16.3 파라미터 내용 확인(ros2 param describe)

파라미터의 형태, 목적, 데이터 형태, 최소, 최댓값

```
$ ros2 param describe /turtlesim background_b
Parameter name: background_b
  Type: integer
  Description: Blue channel of the background color
  Constraints:
    Min value: 0
    Max value: 255
    Step: 1
```

16.4 파라미터 읽기(ros2 param get)

```
ros2 param get <node_name> <parameter_name>
```

```
minseon@minseon-Predator-PH315-52:~$ ros2 param get /turtlesim background_r
Integer value is: 69
minseon@minseon-Predator-PH315-52:~$ ros2 param get /turtlesim background_g
Integer value is: 86
minseon@minseon-Predator-PH315-52:~$ ros2 param get /turtlesim background_b
Integer value is: 255
```

16.5 파라미터 쓰기(ros2 param set)

```
ros2 param set <node_name> <parameter_name> <value>
```

```
minseon@minseon-Predator-PH315-52:~$ ros2 param set /turtlesim background_r 148
Set parameter successful
minseon@minseon-Predator-PH315-52:~$ ros2 param set /turtlesim background_g 0
Set parameter successful
minseon@minseon-Predator-PH315-52:~$ ros2 param set /turtlesim background_b 211
Set parameter successful
```

16.6 파라미터 저장(ros2 param dump)

파라미터 쓰기(ros2 param set)를 이용한 값은 turtlesim 노드를 재실행하면 다시 초깃값으로 설정됨

파라미터 저장을 위해 ros2 param dump 명령어에 노드 이름을 적어주면 현재 경로에 해당 노드 이름으로 설정 파일이 저장됨

```
$ ros2 param dump /turtlesim > turtlesim.yaml
```

```
$ cat ./turtlesim.yaml
/turtlesim:
  ros__parameters:
    background_b: 255
    background_g: 86
    background_r: 69
    qos_overrides:
      /parameter_events:
        publisher:
          depth: 1000
          durability: volatile
          history: keep_last
          reliability: reliable
    use_sim_time: false
```

- 노드 실행 시 저장된 파라미터 값을 사용

```
ros2 run <package_name> <executable_name> --ros-args --params-file <file_name>
```

```
$ ros2 run turtlesim turtlesim_node --ros-args --params-file turtlesim.yaml
```

16.7 파라미터 삭제(ros2 param delete)

ros2 param delete <노드 이름> <파라미터 이름>

```
$ ros2 param delete /turtlesim background_b
```

파라미터의 삭제된 상태 확인

```
$ ros2 param list /turtlesim
```