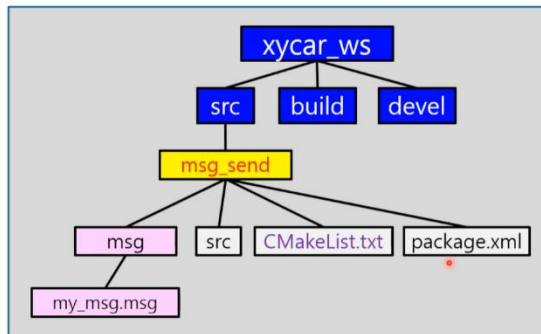


ROS 나만의 메시지 만들기

(예 : 주민번호, 핸드폰 번호 등 다양한 정보 묶어서 보내기)

1. 파일 만들기 (msg)



(1) Package.xml 수정하기

```
<buildtool_depend>catkin</buildtool_depend>
<build_depend>rospy</build_depend>
<build_export_depend>std_msgs</build_export_depend>
<exec_depend>rospy</exec_depend>
<exec_depend>std_msgs</exec_depend>
<build_depend>message_generation</build_depend>
<exec_depend>message_runtime</exec_depend>
```

2줄 새롭게 추가

(2) CMakeList.txt 수정하기

- CMakeLists.txt 수정 (# 코멘트 삭제하고, 추가 삽입하고)

수정 전

```
...
10 find_package(catkin REQUIRED COMPONENTS
11   rospy
12   std_msgs
13 )
...
49 # add_message_files(
50 #   FILES
51 #   Message1.msg
52 #   Message2.msg
53 # )
...
70 # generate_messages(
71 #   DEPENDENCIES
72 #   std_msgs
73 # )
...
104 catkin_package(
105 #   INCLUDE_DIRS include
106 #   LIBRARIES my_pkg1
107 #   CATKIN_DEPENDS rospy std_msgs
108 #   DEPENDS system_lib
```

수정 완료

```
...
10 find_package(catkin REQUIRED COMPONENTS
11   rospy
12   std_msgs
13   message_generation
14 )
...
49 add_message_files(
50   FILES
51   Message1.msg
52   Message2.msg
53 )
...
70 generate_messages(
71   DEPENDENCIES
72   std_msgs
73 )
...
104 catkin_package(
105   CATKIN_DEPENDS message_runtime
106 #   INCLUDE_DIRS include
107 #   LIBRARIES my_pkg1
108 #   CATKIN_DEPENDS rospy std_msgs
109 #   DEPENDS system_lib
```

1줄 추가

코멘트 제거

1줄 추가

코멘트 제거

1줄 추가

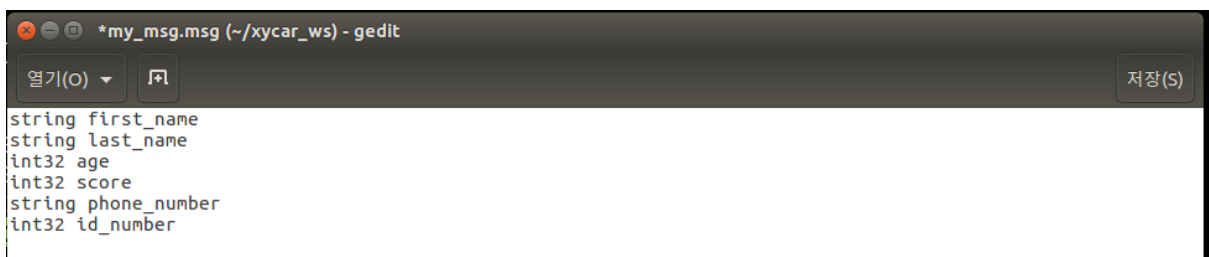
-실습

(1) msg 디렉토리 만들기

```
soorim@soorim-virtual-machine:~$ roscd msg_send/  
soorim@soorim-virtual-machine:~/xycar_ws/src/msg_send$ mkdir msg  
soorim@soorim-virtual-machine:~/xycar_ws/src/msg_send$ ls  
CMakeLists.txt launch msg package.xml src  
soorim@soorim-virtual-machine:~/xycar_ws/src/msg_send$ gedit my_msg.msg
```

(2) My_msg.msg 만들기

soorim@soorim-virtual-machine:~/xycar_ws/src/msg_send/msg\$ gedit my_msg.msg



The screenshot shows a gedit window titled '*my_msg.msg (~/xycar_ws) - gedit'. The window contains the following text:

```
string first_name  
string last_name  
int32 age  
int32 score  
string phone_number  
int32 id_number
```

(3) 파일 수정

<https://wiki.ros.org/ROS/Tutorials/CreatingMsgAndSrv>

- Package.xml파일에 추가

```
<build_depend>message_generation</build_depend>
```

```
<exec_depend>message_runtime</exec_depend>
```

- CMakeList.txt 수정하기

(4) 빌드 : cm

(5) 실행

soorim@soorim-virtual-machine:~/xycar_ws\$ rosmmsg show msg_send/my_msg

```
soorim@soorim-virtual-machine:~/xycar_ws$ rosmmsg show msg_send/my_msg  
string first_name  
string last_name  
int32 age  
int32 score  
string phone_number  
int32 id_number
```

- 실습

(1) publisher 만들기

soorim@soorim-virtual-machine:~/xycar_ws/src/msg_send/src\$ gedit msg_sender.py

```
*msg_sender.py (~/xycar_ws/src/msg_send/src) - gedit
열기(O)  [icon]

#!/usr/bin/env python

import rospy
from msg_send.msg import my_msg

rospy.init_node('msg_sender', anonymous=True)

pub = rospy.Publisher('msg_to_xycar', my_msg, queue_size=1)

msg = my_msg()
msg.first_name = "gildon"
msg.last_name = "Hong"
msg.id_number = 20041003
msg.phone_number = "010-8990-3003"

rate=rospy.Rate(1)

while not rospy.is_shutdown():

    pub.publish(msg)
    print("sending message")
    rate.sleep()
```

(2) subscriber 만들기

soorim@soorim-virtual-machine:~/xycar_ws/src/msg_send/src\$ gedit msg_reciver.py

```
*msg_reciver.py (~/xycar_ws/src/msg_send/src) - gedit
열기(O)  [icon]

#!/usr/bin/env python

import rospy
from msg_send.msg import my_msg

def callback(msg):
    print("1. Name : ", msg.last_name + msg.first_name)
    print("2. ID : ", msg.id_number)
    print("3. Phone Number : ", msg.phone_number)

rospy.init_node('msg_receiver', anonymous=True)

sub = rospy.Subscriber('msg_to_xycar', my_msg, callback)

rospy.spin()
```

(3) 실행 권한 주기

```
soorim@soorim-virtual-machine:~/xycar_ws/src/msg_send/src$ ls
msg_reciver.py msg_sender.py student.py student_int.py teacher.py teacher_int.py
soorim@soorim-virtual-machine:~/xycar_ws/src/msg_send/src$ chmod +x *.py
soorim@soorim-virtual-machine:~/xycar_ws/src/msg_send/src$ ls
msg_reciver.py msg_sender.py student.py student_int.py teacher.py teacher_int.py
```

(4) 빌드 :cm

(5) roscore

-더 좋고 편한 방법은?

-데이터 크기에 따른 전송속도는?

- 파이썬 파일 2개랑 런치파일 1개 만들자
 - ▶ sender_speed.py receiver_speed.py
 - ▶ sr_speed.launch
- 정해진 크기의 데이터를 반복해서 왕창 보내자.
 - ▶ 보내는 쪽은 10분 동안 시간을 정해 놓고 쉴 새 없이 보내자.
 - ▶ 10분 동안 몇 바이트 보냈는지 체크해서 송신속도 계산해 보자.
 - ▶ 받는 쪽도 10분 동안 시간을 정해 놓고, 모두 얼마나 받았는지 체크해서 수신속도를 계산해 보자.
 - ▶ 단위는 300Kbytes/sec 뭐 이렇게.
- 받는 쪽이 없으면 어떻게 될까?
 - ▶ 토픽에 대해서 구독하는 노드가 없으면 송신속도가 더 빨라지나? 아니면 상관 없나?

- 도착하는 데이터를 미처 처리하지 못하면?

- 파이썬 파일 2개랑 런치파일 1개 만들자
 - ▶ sender_overflow.py receiver_overflow.py
 - ▶ sr_overflow.launch
- 받는 쪽이 버벅되게 만들어 놓고 데이터를 왕창 보내자.
 - ▶ 구독자의 콜백함수 안에 시간 많이 걸리는 코드를 넣어서 토픽 처리에 시간이 걸리도록 만들자.
- 콜백함수가 끝나지 않았는데 토픽이 새로 도착하면 어떻게 되는가?
 - ▶ 도착한 토픽은 임시로 어딘가에 쌓이는가? 그걸 나중에 꺼내서 처리할 수 있는가?
 - ▶ 아님 그냥 없어지는가? 한 번 못 받은 토픽은 영영 못 받는 것인가?
 - ▶ 발행자는 이 사실을 아는가? 알려줄 수 있는 방법이 있는가?

-주기적 발송에서 타임슬롯을 오버하면?

- 파이썬 파일 2개랑 런치파일 1개 만들자
 - ▶ sender_timeslot.py receiver_timeslot.py
 - ▶ sr_timeslot.launch
- 1초에 5번 반복하게 하고 작업시간이 0.2초가 넘게 만들어 보자.
 - ▶ Rate(5) 세팅하고 sleep() 앞에 들어간 작업코드에 대해서 수행시간을 늘려보자.
 - ▶ 늘렸다 줄였다 변동성 있게 해보자. 입력값을 받아서 이걸 조정할 수 있게 만들까?
- 1초에 5번 규칙을 지킬 수 없으면 어떻게 할까?
 - ▶ 앞에서부터 쪽 밀리는 식으로 일할까?
 - ▶ 쉬는 시간을 조정할까?
 - ▶ 이번엔 3번만 하고 다음번을 기약할까?

- 협업해야 하는 노드를 순서대로 가동시킬 수 있는가?

- 파이썬 파일 5개랑 런치파일 1개 만들자

- ▶ first.py second.py third.py fourth.py receiver.py
- ▶ sr_order.launch

- 순서대로 receiver에 메시지를 보내도록 만들자.

- ▶ receiver는 도착한 순서대로 출력한다. First - Second - Third - Fourth 이렇게 되어야 한다.
- ▶ 앞에 있는 노드가 움직이기 전에 먼저 움직여서는 안된다. (움직인다 = 토픽을 보내는 걸로 대신하자)

- 어떻게 동기를 맞추고 순서를 맞출 수 있을까?

- ▶ Launch 파일을 이용해서 할 수 있을까?
- ▶ ROS 의 도움으로 할 수 있을까?

-정리



- 1) 누락 없이 모두 잘 도착하는가? 특히 처음과 끝...
- 2) 데이터 크기에 따른 전송속도는 어떻게 되는가?
- 3) 도착하는 데이터를 미처 처리하지 못하면 어떻게 되는가?
- 4) 주기적 발송에서 타임슬롯을 오버하면 어떻게 되는가?
- 5) 협업해야 하는 노드를 순서대로 가동시킬 수 있는가?