ROS dashing tutorials
https://docs.ros.org/en/dashing/Tutorials.html

- Writing a simple publisher and subscriber (Python)
- Writing a simple service and client (Python)

```
ros2 pkg create --build-type ament_python py_pubsub
                                                패키지이름
```

rclpy - robot control….

```
import rclpy
from rclpy.node import Node

from std_msgs.msg import String


class MinimalPublisher(Node):

    def __init__(self):
        super().__init__('minimal_publisher')
        self.publisher_ = self.create_publisher(String, 'topic', 10)
        timer_period = 0.5  # seconds
        self.timer = self.create_timer(timer_period, self.timer_callback)
        self.i = 0

    def timer_callback(self):
        msg = String()
        msg.data = 'Hello World: %d' % self.i
        self.publisher_.publish(msg)
        self.get_logger().info('Publishing: "%s"' % msg.data)
        self.i += 1


def main(args=None):
    rclpy.init(args=args)

    minimal_publisher = MinimalPublisher()

    rclpy.spin(minimal_publisher)

    # Destroy the node explicitly
    # (optional - otherwise it will be done automatically
    # when the garbage collector destroys the node object)
    minimal_publisher.destroy_node()
    rclpy.shutdown()


if __name__ == '__main__':
    main()
```

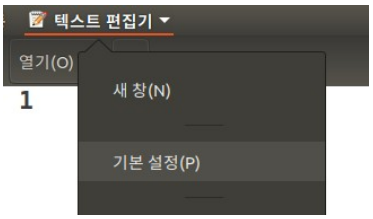
pac

```
    <exec_depend>std_msgs</exec_depend>


entry_points={
        'console_scripts': [
                'talker = py_pubsub.publisher_member_function:main',
                 # 노드이름 = 패키지이름. 소스코드이름(파이썬 메인함수 매핑),
        ],
},
```

```
import rclpy
from rclpy.node import Node

from std_msgs.msg import String


class MinimalSubscriber(Node):

    def __init__(self):
        super().__init__('minimal_subscriber')
        self.subscription = self.create_subscription(
            String,
            'topic',
            self.listener_callback,
            10)
        self.subscription  # prevent unused variable warning

    def listener_callback(self, msg):
        self.get_logger().info('I heard: "%s"' % msg.data)


def main(args=None):
    rclpy.init(args=args)

    minimal_subscriber = MinimalSubscriber()

    rclpy.spin(minimal_subscriber)

    # Destroy the node explicitly
    # (optional - otherwise it will be done automatically
    # when the garbage collector destroys the node object)
    minimal_subscriber.destroy_node()
    rclpy.shutdown()


if __name__ == '__main__':
    main()
```

[터미널열기 ctrl+alt+T]

```
cd colcon_ws/src
ros2 pkg create --build-type ament_python py_pubsub
ls
cd py_pubsub
ls                          # pubsub 안에 pubsub 이 또 있음.. 소스코드가 여기에 들어감
cd py_pubsub
```




```
gedit publisher_member_function.py
```

#기본설정 – 줄번호, 탭간격 4, 자동들여쓰기, 글꼴 적당히

.. 코드작성 ..

콜백함수 - 사용자가 정의함. 운영체제에서 호출

(이벤트핸들러, 인터럽트)

```
cd ..
ls
gedit package.xml setup.cfg setup.py                    # cfg – config
```

[package.xml]

9 번줄에 추가

```
  <exec_depend>rclpy</exec_depend>
  <exec_depend>std_msgs</exec_depend>
```

[setup.cfg] pass

[setup.py]

23 번줄 수정

```
entry_points={
        'console_scripts': [
                'talker = py_pubsub.publisher_member_function:main',
        ],
},
```

```
cd
cd colcon_ws                # cw  엘리어싱

colcon build --packages-select py_pubsub
source ~/.bashrc            # bash 파일 다시 적용 - 터미널창 닫았다가 다시 여는 효과...
ros2 pkg executables py_pubsub
ros2 run py_pubsub talker
```

새 터미널 열고

```
ros2 topic echo “/my_topic”
```

[터미널]
```
cs
cd py_pubsub/py_pubsub
cp publisher_member_function.py subscriber_member_function.py
gedit subscriber_member_function.py
```
.. 코드작성 ..

```
cs
cd py_pubsub
gedit setup.py
```

[setup.py]
23 번째줄
```
    entry_points={
        'console_scripts': [
            'talker = py_pubsub.publisher_member_function:main',
            'listener = py_pubsub.subscriber_member_function:main',
        ],
```

```
cw
colcon build --packages-select py_pubsub
source ~/.bashrc
ros2 pkg executables py_pubsub

ros2 run py_pubsub listener
```

[터미널 새 창]
```
ros2 run py_pubsub talker
```

튜토리얼 - ~~Writing a simple service and client (Python)~~
Writing an action server and client (Python)
https://docs.ros.org/en/dashing/Tutorials/Actions/Writing-a-Py-Action-Server-Client.html

creating an action
https://docs.ros.org/en/dashing/Tutorials/Actions/Creating-an-Action.html#actioncreate

```
int32 order                     리퀘스트
---
int32[] sequence                결과
---
int32[] partial_sequence        피드백
```

[터미널]

```
cs
ros2 pkg create action_tutorials_interfaces
cd action_tutorials_interfaces/

mkdir action
cd action

gedit Fibonacci.action
```

```
int32[] order
---
int32[] sequence
---
int32[] partial_sequence
```

```
cd ..
gedit CMakeLists.txt package.xml
```

[CMakeList.txt]

23 번째줄 추가

```
find_package(rosidl_default_generators REQUIRED)

rosidl_generate_interfaces(${PROJECT_NAME}
  "action/Fibonacci.action"
)
```

[package.xml]

11 번째줄 추가

```
<buildtool_depend>rosidl_default_generators</buildtool_depend>

<depend>action_msgs</depend>

<member_of_group>rosidl_interface_packages</member_of_group>
```

```
cw
colcon build --packages-select action_tutorials_interfaces
source ~/.bashrc
ros2 action show action_tutorials_interfaces/action/Fibonacci

cd
gedit fibonacci_action_server.py
```

.. 코드작성 ..

```
python3 fibonacci_action_server.py
```

[터미널 새창]

```
ros2 action send_goal --feedback fibonacci
action_tutorials_interfaces/action/Fibonacci "{order: 5}"
```

붙여서 한줄로...

```
cd
gedit fibonacci_action_server.py
def execute_callback(self, goal_handle):                # 액션서버가 실행될때 실행되는 함수
        self.get_logger().info('Executing goal...')

        #sequence = [0, 1]
        feedback_msg = Fibonacci.Feedback()
        feedback_msg.partial_sequence = [0, 1]

        for i in range(1, goal_handle.request.order):
            #sequence.append(sequence[i-1] + sequence[i])
            feedback_msg.partial_sequence.append(feedback_msg.partial_sequence[i-1] +
feedback_msg.partial_sequence[i])
            self.get_logger().info('Feedback: {0}'.format(feedback_msg.partial_sequence))
            goal_handle.publish_feedback(feedback_msg)
            time.sleep(1)

        goal_handle.succeed()

        result = Fibonacci.Result()
        #result.sequence = sequence
        result.sequence = feedback_msg.partial_sequence
        return result


python3 fibonacci_action_server.py

[새 터미널]
ros2 action send_goal --feedback fibonacci
action_tutorials_interfaces/action/Fibonacci "{order: 10}"
   # 붙여서 한줄로…
```

```
cd
gedit fibonacci_action_client.py
  ..  코드작성 ..

python3 fibonacci_action_server.py

[새 터미널]
python3 fibonacci_action_client.py
```

매니퓰레이터 매뉴얼
OpenMANIPULATOR-X
13.3 Message List

https://emanual.robotis.com/docs/en/platform/openmanipulator_x/ros2_controller_package/#ros-controller-package

예제폴더(open_manipulator_x_tutorial)을 colcon_ws/src 에 넣기

매니퓰레이터 연결

[터미널]
ls /dev/ttyU*

```
ros2 launch open_manipulator_x_controller open_manipulator_x_controller.launch.py
```

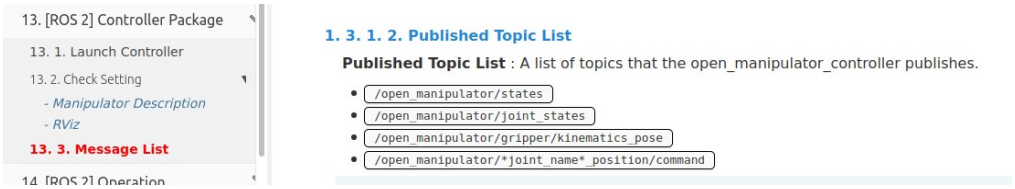
[새 터미널]
ros2 topic list

ros2 topic echo "/joint_states" ctrl+Z
ros2 topic echo "/kinematics_pose" ctrl+Z 좌표 상태 3 차원,,,,,?

rqt
Plugins – Topics – Topic Monitor

Plugins – Services – Service Type Browser
Service : open_manipulator_msgs /

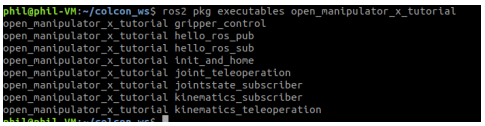
로보티즈 e-매뉴얼 13.3 Message List



Topic List, Service Server List

[터미널]
cs
cd open_manipulator_x_tutorial/open_manipulator_x_tutorial/
gedit get_joint_state_node.py

cw
colcon build --packages-select open_manipulator_x_tutorial
source ~/.bashrc
ros2 pkg executables open_manipulator_x_tutorial



cs
cd open_manipulator_x_tutorial/
gedit package.xml setup.py # 추가되어있음.. 따로 할건 없음..

[package.xml]

```
<depend>rclpy</depend>
<depend>std_msgs</depend>
<depend>sensor_msgs</depend>
<depend>open_manipulator_msgs</depend>
```

[setup.py]

```
entry_points={
        'console_scripts': [
            'hello_ros_pub = open_manipulator_x_tutorial.hello_ros_publisher:main',
            'hello_ros_sub = open_manipulator_x_tutorial.hello_ros_subscriber:main',
            'init_and_home = open_manipulator_x_tutorial.init_and_home_node:main',
            'gripper_control = open_manipulator_x_tutorial.gripper_control_node:main',
            'jointstate_subscriber = open_manipulator_x_tutorial.get_joint_state_node:main',
            'kinematics_subscriber = open_manipulator_x_tutorial.get_kinematics_node:main',
            'joint_teleoperation = open_manipulator_x_tutorial.joint_teleoperation:main',
            'kinematics_teleoperation = open_manipulator_x_tutorial.kinematics_teleoperation:main',
        ],
    },
```

No module named ‘getkey’

```
sudo apt-get install python3-pip
pip3 install --upgrade pip
pip3 install getkey
```

ros2 run open_manipulator_x_tutorial jointstate_subscriber ctrl+Z

cs
cd open_manipulator_x_tutorial/open_manipulator_x_tutorial/
gedit init_and_home_node.py

cw
ros2 run open_manipulator_x_tutorial init_and_home