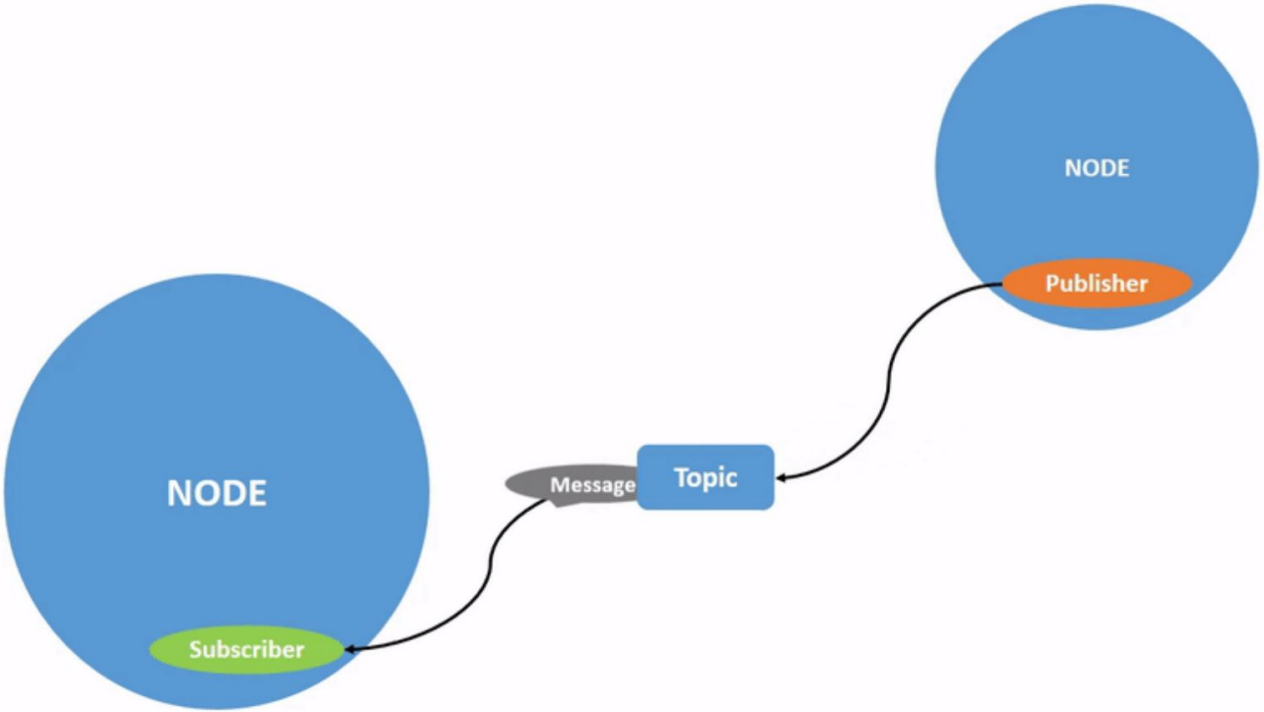
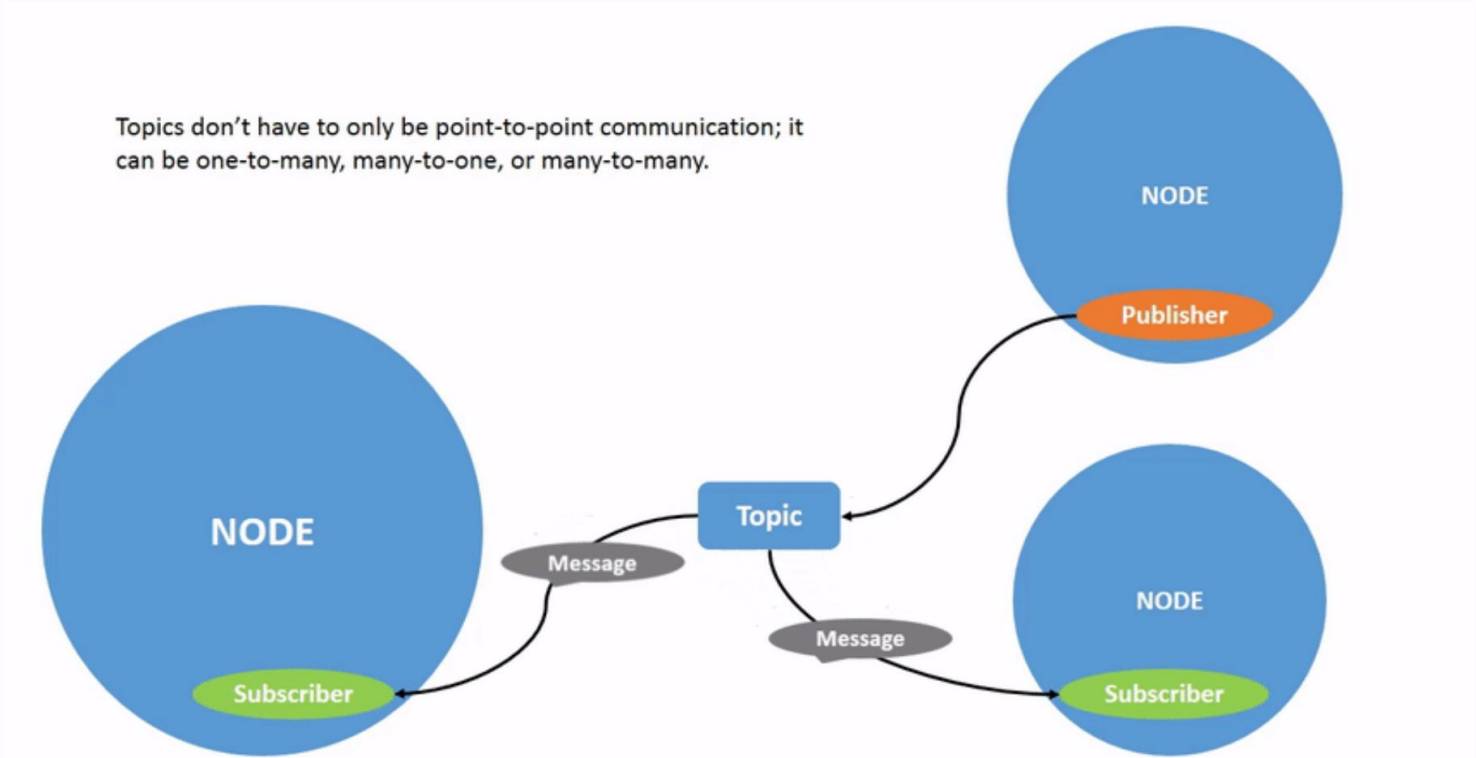
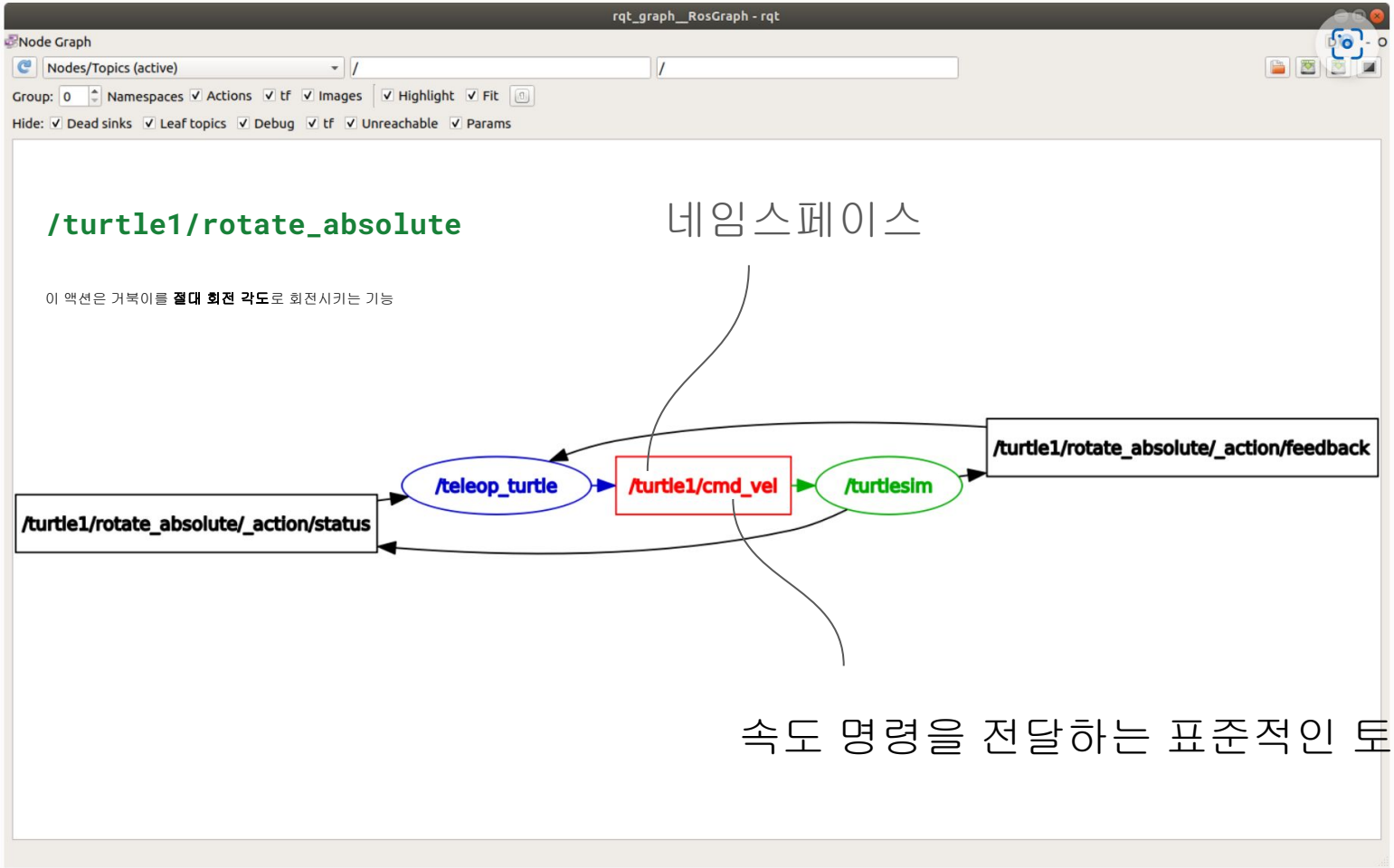
ROS 2 breaks complex systems down into many modular nodes. Topics are a vital element of the ROS graph that act as a bus for nodes to exchange messages.

공유 통신 경로



A node may publish data to any number of topics and simultaneously have subscriptions to any number of topics.

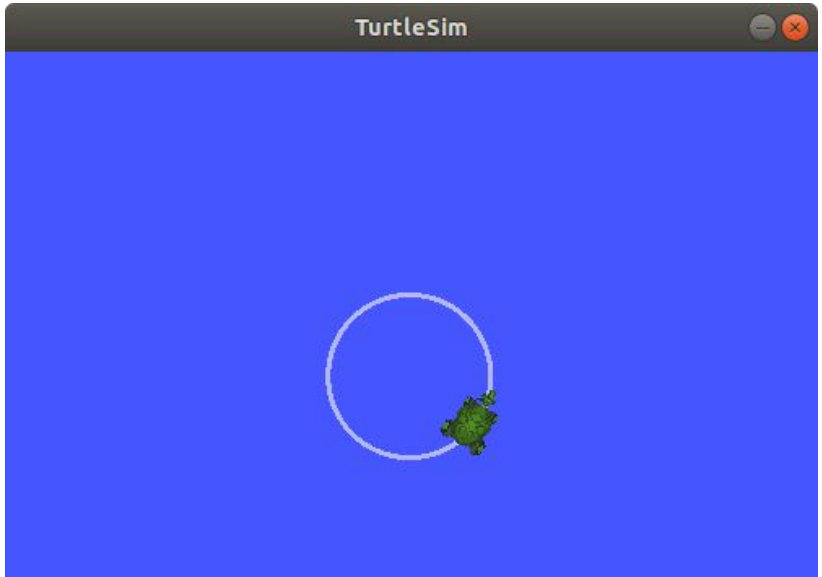


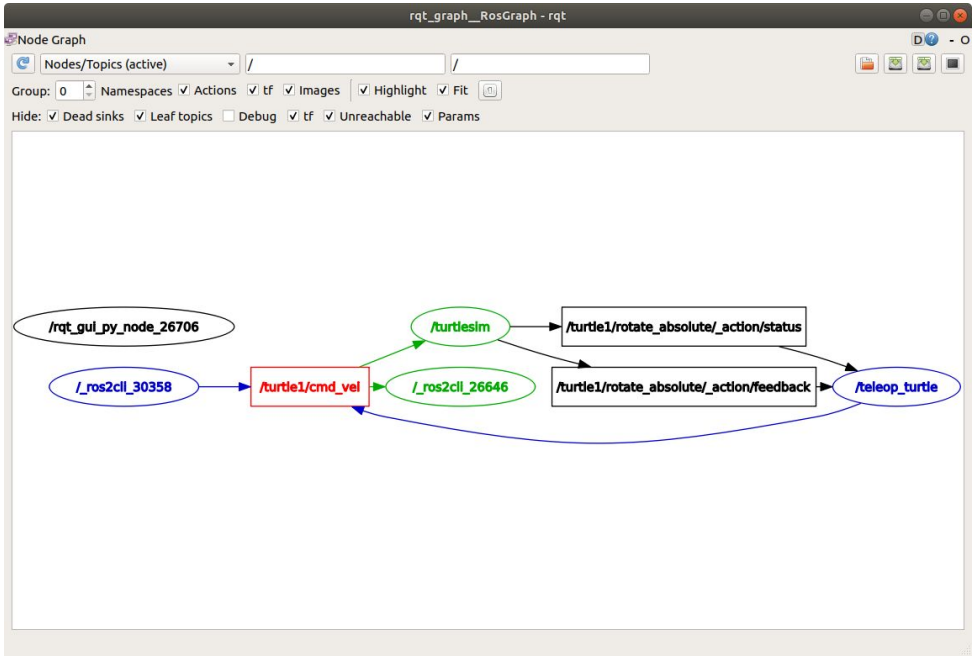


속도 명령을 전달하는 표준적인 토픽 이름입니다.

ros2 topic pub /turtle1/cmd_vel geometry_msgs/msg/Twist "{linear: {x: 2.0, y: 0.0, z: 0.0}, angular: {x: 0.0, y: 0.0, z: 1.8}}"

With no command-line options, publishes the command in a steady stream at 1 Hz.





코드에서 퍼블리셔만 생성하면, 해당 토픽은 자동으로 ROS에 등록됩니다.

ROS에서는 "퍼블리셔 또는 서브스크라이버 중 하나라도 생기면, 해당 토픽은 자동으로 생성"됩니다.

- 즉, **토픽은 미리 선언하거나 만드는 작업이 없습니다.**
- 퍼블리셔를 만들고 특정 이름으로 메시지를 보내면 → 그 이름의 토픽이 생깁니다.

```python
import rclpy
from rclpy.node import Node
from std_msgs.msg import String

class Talker(Node):
    def __init__(self):
        super().__init__('talker')
        self.publisher_ = self.create_publisher(String, 'chatter', 10)
        self.publish_timer = self.create_timer(1.0, self.publish_msg)

    def publish_msg(self):
        msg = String()
        msg.data = "hello"
        self.publisher_.publish(msg)
        self.get_logger().info(f'Published: {msg.data}')

def main():
    rclpy.init()
    node = Talker()
    rclpy.spin(node)
    node.destroy_node()
    rclpy.shutdown()
```

토픽은 "주제 기반" 메시징 모델에서 나온 용어입니다.

Pub/Sub 시스템에서는 메시지를 주고받을 때 **"이 메시지가 어떤 주제(topic)에 해당하는가?"**를 기준으로 분류합니다.

- 퍼블리셔는 **특정 주제에 대해** 메시지를 발행하고
- 서브스크라이버는 **그 주제에 관심이 있는** 경우에만 메시지를 받습니다

즉, **중간 통로이자 메시지를 분류하는 이름(tag)이 바로 Topic**입니다.